Tiny Object Detection in Video with Transformers using Temporal Context

Student: Pim Veraar

Student ID: 6409458

Main Supervisor: Dr. ir. Ronald Poppe

Second Supervisor: Dr. Itir Önal Ertuğrul

Department of Information and Computing Sciences
Utrecht University
Artificial Intelligence
December 2024

***Abstract*—Tiny object detection in video footage using computer vision models has a wide range of applications. Following an overview of relevant deep learning methods and related work done in this field, this paper explores the performance of Video Transformer models for this challenging task. More specifically, this work focusses on the effect of early fusion of spatio-temporal features on the performance of Video Transformers. In this paper, we propose a switch to a Video Swin backbone, using spatial-temporal fusion mechanics to resolve the shape mismatch between the backbone output and model input, while attempting to conserve as much temporal information as possible. These fusion mechanics are put between the backbone and the model, making it compatible with any object detector. Using this approach, different fusion mechanics are investigated, of which a 3D convolution resulted in the best performance. The resulting multi-frame model does not show performance improvements compared to state-of-the-art models. When compared to a single frame version of the same model, the overall performance is comparable, although distinct differences can be noted in their respective capabilities to successfully detect specific object classes. The multi-frame model outperforms the single frame model in scenarios with large objects, rare classes and medium speed objects. These results indicate that, while the multi-frame model struggles with interpreting the added temporal information in the context of small objects, it can offer some unique advantages in certain areas. The work provided new insights in how early temporal context can affect a model's performance, thereby increasing the understanding of the remaining challenges of Video Transformers for tiny object detection. Based on these findings, future work can be directed towards improving the method proposed in this paper aiming to achieve performance gains on a broad range of real-world applications.**

***Index Terms*—Tiny Object Detection, Video Transformers**

I. Introduction

Object Detection is a fundamental aspect of computer vision with significant implications for a wide range of practical applications. Recently, there has been a great interest in Transformer models [41], as they have delivered state-of-the-art results on object detection problems. Despite of their impressive results in general, there remains uncertainty on their performance regarding tiny objects [36]. This paper aims to explore the performance of Transformer models, specifically on small objects.

In many applications, there are cases where target objects appear very small in the captured data. For example, aerial surveillance at high altitudes for search and rescue missions [33] or pedestrian detection for autonomous vehicles [6] both have to deal with recognizing objects at a far away distance. With only few pixels representing the target, it is still important that detections happen accurately and reliably.

The domain of Tiny Object Detection (TOD) faces many challenges. Firstly, only limited features can be extracted from tiny objects, in contrast to larger objects with ample feature information. The low pixel quantity of these tiny objects makes them difficult to separate from the background, even for humans. Their representation is easily lost in later layers of popular deep learning methods for computer vision when feature maps are down-sampled to reduce spatial redundancy [37]. Due to the low number of pixels representing tiny objects, they can be easily mistaken for noise in low quality footage.

Secondly, there is low tolerance for error in evaluation of TOD. Bounding boxes are used to enclose the target object, which the model is tasked with predicting. It is common to use the Intersection over Union (IoU) score as the performance metric to determine the quality of the prediction. However, the same absolute translation of the predicted bounding box results in a bigger change of the IoU score for tiny objects compared to medium or large objects. [10]. This can make it difficult for a model to converge during training.

	Small	**Medium**	**Large**
Min Area Size (pixels)	0×0	32×32	96×96
Max Area Size (pixels)	32×32	96×96	$\infty \times \infty$
Object Count (%)	41.43%	34.32%	24.24%
Image Occurence (%)	51.82%	70.07%	82.28%
Total Object Area (%)	1.23%	10.18%	88.59%

TABLE I
MS COCO PROPERTIES OF OBJECT SIZE CATEGORIES. TAKEN FROM [23]

Thirdly, datasets are distributed unevenly with respect to the target object size. As shown in Table I, while small objects make up over 40% of the objects in MS COCO, they appear in only roughly half of the images, far less than medium and large objects. There is a lot less variation in training samples containing small objects, as most will appear within the same image setting [23]. The fact that small objects appear together a lot in the same image has other effects as well. In the object detection domain, all objects need to be annotated with enclosing bounding boxes to be useful for training. This is a resource demanding process which not everyone can afford. Some datasets like UA-DETRAC [44] reduce the number of annotations needed by placing 'ignore regions' over the vanishing point at the horizon, where the model does not have to pay attention to and no objects are annotated, removing a lot of small objects in the process. This is, however, not realistic for real life scenarios. For a more extensive study on challenges in TOD, we refer the reader to [37] [34] [10] [40].

The aforementioned challenges necessitate the development of efficient techniques capable of extracting valuable features from the limited data available for tiny objects. A solution could be to leverage the rich temporal information available in video data to improve the performance of object detection models. Indeed, many state-of-the-art models use multiple frames to enhance the detection performance on the target frame.

The success of the Transformer model [41] has also made it a prime candidate for the extension to video-based object detection. However, as noted in [36], there is a significant research gap in the area of video-based Transformers in TOD compared to other classical or deep learning models. In this paper, we aim to bridge this gap by studying the effect of early fusion of temporal information in the Transformer pipeline.

In particular, a property shared by current Video Transformer object detection models is that the fusion of spatio-temporal features happens only during or after the encoding step. The effectiveness of methods that combine spatial and temporal features early has already been shown for image

classification, with the successful extension of the Swin Transformer [30] to the Video Swin Transformer [31]. Similar to the human brain, where motion processing already starts in the primary visual cortex [19], early fusion appears to be a valid intuition.

However, the application of early fusion methods within the object detection domain have remained unproven. Following the observed improvements in the Video Swin Transformer architecture, we aim to investigate whether this early fusion strategy can be replicated when adopting a similar approach by utilizing it as the backbone for current Video Transformers for object detection. Thus, we intend to answer the main research question of this paper:

- *Does early fusion of spatio-temporal features result in higher mAP scores for Video Transformers on a challenging dataset containing small objects with both moving and static camera footage?*

Following this research question, we formulate a series of sub-questions to help answer the main research question:

1) *What is the performance difference between utilizing the output from a single-frame backbone (Swin Transformer) across multiple frames compared to a dedicated video backbone (Video Swin Transformer)?*
2) *Which temporal fusion strategy (such as 3D convolutions, temporal attention, e.g.) result in the highest performance?*
3) *In which kind of footage does the proposed early fusion strategy excel over current Video Transformer models?*

First, some background concepts will be provided to explain some of the core concepts of the paper. Then, some related works relevant to the current study will be highlighted. Lastly, we share our plan for the study.

II. Background

In this section, we briefly introduce some of the core concepts relevant to the subject of this paper. First, object detection and related domains will be explained. Second, we discuss popular deep learning methods that form the basis of many computer vision applications today. Lastly, we will discuss the attention mechanism and the Transformer model.

A. Computer Vision Domains

As noted in the introduction, computer vision has many applications. However, these applications often require different outputs due to the diverse nature of the problems they aim to address. As a result of these differing requirements, various domains have emerged within the field of computer vision. A few notable ones will be discussed here.

Object detection is a computer vision task that involves the localization and identification of objects within an image or a video frame. This process not only entails predicting the correct class label of an object from a set of predefined class categories, but also defining the spatial extent of the object within the image. The region representing the object is enclosed by a rectangular *bounding box*, where the spatial information is encapsulated in the bounding box corner coordinates. Mathematically, object detection can be defined as a function $f : I \longrightarrow O$, where I is the image input and $O = \{(c_i, b_i)\}$ is a set of tuples for each object i containing a class label c_i and a bounding box $b_i = (x_i, y_i, w_i, h_i)$, representing the top-left coordinates together with the width and height of the bounding box.

The bounding box prediction sets it apart from image segmentation, where each object is outlined by a detailed pixel-wise mask, resulting in a fine-grained representation of its boundaries compared to the rudimentary bounding box. Thus, image segmentation can be defined as a function $f : I \longrightarrow S$, where S is the image with each pixel assigned a class label.

In contrast to image classification, which can be described as a function $f : I \longrightarrow c$ where f considers the entire image I to predict a single class label c, object detection requires a class label for each individual object captured by a bounding box prediction. An object is then 'detected' if there is sufficient overlap between the ground truth bounding box and the predicted bounding box, provided that the predicted class matches the actual class of the object.

Object detection differs from object tracking in that object tracking requires the target to be linked to the object trajectories [24]. When applied to the video domain, object tracking models must establish a relation between a sequence of object appearances across multiple frames and deduce that it is the same object moving through time. The object's path is then the history of its spatial locations in the previous frames, allowing for precise tracking. Given a sequence of frames $F = \{I_1, I_2, ..., I_n\}$, object tracking can be defined as a function $f : F \longrightarrow T_i$ where T_i is a set of positions per frame $T = \{T_1, T_2, ..., T_n\}$ for an object o_i. While it is required that objects are detected in a consistent manner (e.g. remaining the same class overtime) in object detection for video data, this only has to be enforced over a short period and does not require saving the entire object path. Thus, object detection on video data can be seen as a reduced form of object tracking. However, object tracking differs from object detection in that object tracking does not require object classification.

B. Neural Networks

Neural networks are a core concept in the Deep Learning branch of machine learning. They are inspired by the working of the human brain through the use of a network of artificial neurons, enabling machines to learn relations in complex data. The term 'deep' in Deep Learning refers to the fact that these networks consist of many layers of artificial neurons. Deep Learning has shown remarkable success in problems from domains such as Computer Vision and Natural Language Processing.

In Feed-forward Neural Networks (FFNN), information flows in one direction. They are comprised of an input and output layer, with an arbitrary number of hidden layers in between. Thus, the input of the current layer can only come from the previous layer.

Recurrent Neural Networks (RNN) are a class of neural networks that allow the artificial neurons to take their previous

output as its input. This means that cycles exist in this network architecture. It allows the hidden layers to serve as a kind of memory, known as the hidden state, that enables them to learn dependencies over time.

Convolutional Neural Networks (CNN) are specialized neural networks with layers that use the convolution operation to extract information from grid-like data structures like images. This convolutional layer consists of sliding filters (also called kernels) over the input data to identify patterns. These convolutional layers are often applied multiple times in a hierarchical fashion, with higher layers being able to capture more abstract and complex features.

For a comprehensive summary of the topics mentioned above and their role in computer vision problems, we direct the reader to [14] [2].

C. Attention and Transformers

The *attention mechanism* was proposed by Bahdanau et al. [5] to be used in RNN based sequence-to-sequence models. In these early sequence-to-sequence models, an *encoder* captures an input sequence by propagating the input elements through the hidden states of RNN, compressing their information into a single fixed-length context vector. A *decoder* is a RNN that takes the context vector as its starting hidden state to generate the output sequence. Due to the sequential nature of RNN and the fixed size of the context vector, long distance dependencies between input elements cannot be represented accurately in the context vector. Bahdanau et al. resolve this limitation by introducing weighted connections between each current target hidden state and all hidden states of the encoder, possibly bypassing much of the RNN structure. The weights that determine the influence of the hidden state at each encoder stage are learned in conjunction with other parts of the model.

Vaswani et al. [41] departed from the standard RNN architecture and showed that it was possible to build a sequence-to-sequence model with only the attention mechanism. This model, known as the Transformer, still adheres to the standard encoder/decoder structure, applying attention both within these modules separately (self-attention) and between the encoder and decoder (cross-attention). It describes attention as a weighted sum over *values*, where the weights are computed by some sort of compatibility metric between *queries* (what information we are looking for) and *keys* (the information that is offered). This compatibility metric reflects how much attention should be payed to the current key given a query, and is thus also known as the attention score. The attention function used in Transformers is given as

$$\text{Attention}(Q, K, V) = \text{softmax}\left(\frac{QK^T}{\sqrt{d_k}}\right)V \tag{1}$$

with sets of queries, keys and values combined into matrices Q, K and V respectively and d_k the dimensionality of the keys. Equation 1 is referred to as the Scaled Dot Product Attention, as the compatibility metric is the dot product scaled by a factor $\frac{1}{\sqrt{d_k}}$ as a normalization step. To capture different kinds of relations, Multi-Head Attention was introduced by Vaswani et al. [41]. Here, the query, key and value matrices of the same features are divided into h heads, in a way that each query, key and value vector in the matrix is split into h parts. Each head is then given to Equation 1 separately and concatenated again. A visual representation can be found in Figure 1.

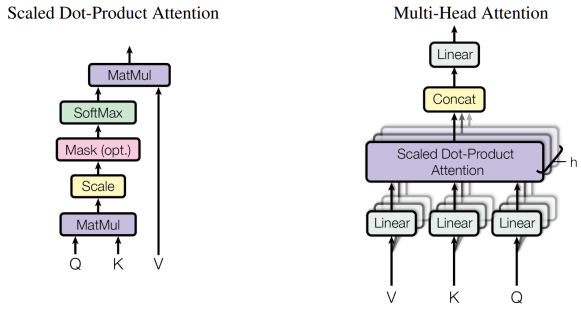

Fig. 1. Attention components in Transformers. Taken from [41]

.

The Multi-Head Attention operation is given as

$$\text{MultiHead}(Q, K, V) = \text{Concat}(\text{head}_1, ..., \text{head}_h)W^O \tag{2}$$

where

$$\text{head}_i = \text{Attention}(QW_i^Q, KW_i^K, VW_i^V) \tag{3}$$

W^O, W_i^Q, W_i^K and W_i^V are weight matrices that are learnable parameters of the model. This makes it possible to simply use a single input vector for the entire Multi-Head Attention operation, allowing the model to adjust the learnable weight matrices to linearly transform that input into their respective query, key and value feature representations. Using the same input to generate the query, key and value feature vectors is called *self-attention*. Self-attention allows the model to capture relationships and dependencies within the same sequence of data. The encoder consists of a Multi-Head Attention operation performing self-attention, followed by a fully connected FFNN. This encoder module can be stacked multiple times if necessary. In the decoder, the first Multi-Head Attention operation is slightly modified to prevent the model from looking ahead in the sequence. The subsequent Multi-Head Attention operation is different as it uses *cross-attention*. Queries are generated by transforming the input (the output of the previous layer in the decoder) through the learnable weights, but the key/value pairs are extracted from the encoder output. Cross-attention allows the model to focus on parts of one sequence (the query) based on the information in another sequence (the keys and values). Similarly to the encoder, the decoder module can be repeated. The output of the last decoder is fed to a FFNN, followed by a linear layer and a softmax layer to do predictions.

Because the encoder and decoder modules are allowed to be stacked, resulting networks can become very deep. This

can lead to the vanishing gradient problem, where gradients become very small and disappear as they are backpropagated throughout the network. To this end, a residual connection is added after every Multi-Head Attention and FFNN layer, following the ResNet approach [20]. In addition, Layer Normalization [4] is applied on the output of the residual connection addition.

A last important feature of the Transformer model are the positional encodings. The attention mechanism computes attention scores between all pairs of queries and keys. This calculation is not influenced by the order of the individual input elements, as it considers these relationships independently. To solve this, a sine or cosine function is added dependent on the position and dimension.

$$PE_{(pos,2i)} = sin(pos/10000^{2i/d_{model}}) \tag{4}$$

$$PE_{(pos,2i+1)} = cos(pos/10000^{2i/d_{model}}) \tag{5}$$

The positional encodings introduce a notion of relative position that the model can identify from features to capture sequential information. A full overview of the Transformer model can be seen in Figure 2.

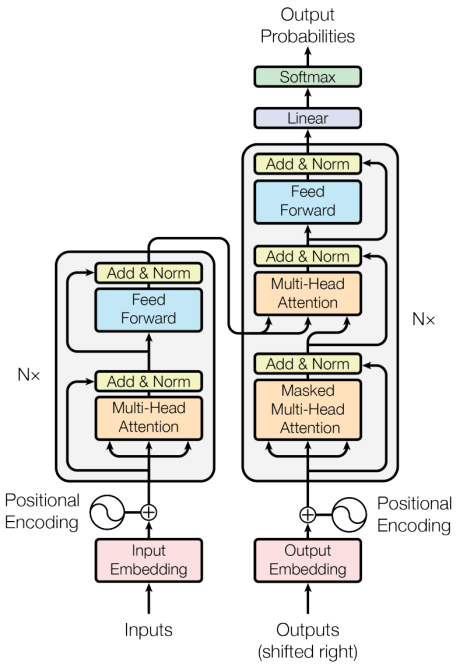

Fig. 2. Overview of the Transformer model. Taken from [41]

D. Vision Transformers

Vision Transfomer (ViT) [15] implemented the Transformer architecture for Image Classification using non-overlapping image patches as the input elements. These patches are flattend and linearly embedded, after which a positional encoding gets added, resembling the standard Transformer model. This is then fed to the Transformer encoder, where the output is directly handled by a FFNN to perform classification. The ViT framework was later adapted to the Object Detection domain as well [46]. A notable ViT variant is the Swin Transformer [30], which introduces a hierarchical design together with a shifted attention window approach to self-attention. Image patches are then merged in later layers. Self-attention is calculated on each patch separately. So in the deeper layers, when all patches are merged, self-attention in the Swin Transformer looks the same as in the original ViT. However, in earlier layers, no interaction is happening between the features of different patches, as the self-attention is applied locally within the patch only. This limits the model's ability to capture global information. To resolve this, the authors shift the attention window diagonally after each layer in a way that each shifted attention window has overlap with multiple attention windows from the last layer. The success of the Swin Transformer has made it a popular backbone for many models today.

The Contextual Transformer Block [26] was inspired by Vision Transformer architecture, extending the process of self-attention to include local spatial information. Instead of every embedded patch generating a key, keys are generated using a 3×3 convolutional operation, thus incorporating local context within each key.

With the success of the Vision Transformer on image tasks, there extension to video analysis was a logical consequence. TimeSformer [7] adapted the ViT model to incorporate temporal information as well. It uniformly samples frames from the video, where each frame is divided into patches similar to ViT, concatenating all patches per frame to obtain the final input of the model. Performing self-attention between all of these patches would be too costly, so they instead propose to perform self-attention on the spatial and temporal dimensions separately. In the spatial dimension, self-attention is calculated within the frame, like in ViT, while self-attention in the temporal dimension is calculated with patches at the same position in different frames. This was empirically found to result in a larger learning capacity as it contains distinct learning parameters for temporal attention and spatial attention [7].

ViViT [3] uses a different procedure to convert video data into the model input. The patches are extended into 3 dimensions, with the extra dimension spanning the temporal axis represented by multiple frames stacked. This *tubelet embedding* strategy allows the model to already fuse spatial-temporal features in the early stages. However, this means that the model could not be fine-tuned starting from strong pre-trained ViT model parameters, as it would cause a shape mismatch in the embedding layer as a result of this change to 3D. To solve this, the embedding weights are initialized by using the pre-trained 2D embedding weight on the position of the central frame, padded with zeroes on both sides along the temporal dimension. Additionally, where self-attention in TimeSformer is performed on spatial and temporal dimensions separately within the encoder, ViViT splits it up further by calculating spatial and temporal self-attention in different encoders. This resembles the late-fusion approach also found

in TransVOD, which first extracts spatial features per frame and only later applies temporal information.

Following the Swin Transformer architecture, Video Swin Transformer [31] extends the concept of shifting attention windows for video data. Using a 3D embedding process similar to ViViT, the embedding mismatch problem encountered in fine-tuning from a pre-trained starting point is resolved by repeating the pre-trained embedding weights along the temporal dimension and averaging them. This way, pre-trained model weights can be used as the remaining architecture remains the same.

III. Related Work

In this section, relevant work for this study will be discussed. Sections III-A and III-B will focus on object detectors using only spatial features. Sections III-C, III-D and III-E will concentrate on methods that benefit from temporal features also.

A. CNN Object Detectors

Object detection networks are methods with a certain class of Neural Networks (see Section II-B) at their core. CNNs have seen great success in the domain of Object Detection. R-CNN [18] introduced a two-stage approach based on the CNN architecture. It first identifies possible object locations using a separate region proposal method before subjecting them to object classification and bounding box regression independently. This is slow, as each region proposal is passed through the CNN sequentially. Fast R-CNN [17] achieves a significant speed-up by feeding the image with all region proposals to the CNN in one forward pass. Classification and bounding box regression is learned jointly during training, combining them into a single, unified network. However, the initial region proposals are still outsourced to a region proposal method, which was found to be a bottleneck in the Fast R-CNN pipeline. Faster R-CNN [38] resolved this by implementing the Region Proposal Network (RPN) to let the model itself predict possible object locations. Feature Pyramid Networks (FPN) [28] was introduced to replace the RPN and address the challenge of handling objects at different scales by using a top-down architecture with lateral connections. FPN helps in producing a feature pyramid, which allows the network to capture multi-scale representations of the input image. This is particularly beneficial for detecting objects of varying sizes. SS R-CNN [22] uses self-supervised learning to gain performance for TOD.

You Only Look Once (YOLO) models [35] are a family of models that eliminate the need for two stages in R-CNN based models by performing them simultaneously. An image is divided into a grid, where each grid predicts a certain number of bounding boxes with a confidence score, together with single conditional class probability for that cell. The confidence score and the conditional class probability are used to determine the final bounding box prediction. Multiple iterations of YOLO exist, with the latest being YOLOv8. TPH-YOLOv5 [50] replaces the head of the YOLO model, which is responsible for the actual object localization and classification, for a Transformer style prediction head. This was found to increase performance on drone footage, which deals with many tiny objects.

B. DETR Transformers

DETR [8], introduced by Carion et al., was the first successful application of the Transformer model to the object detection domain. It views object detection as a set based matching problem. Image features are first extracted using a CNN backbone, which serve as the input of the Transformer encoder. The pipeline of DETR itself consists of 3 important parts. In the encoder, self-attention is performed with key, query and value embeddings generated from the flattened feature vector obtained by the backbone with an added positional encoding. In the decoder, cross-attention is performed between the key-value pair generated from the output of the encoder and the queries generated from N learnable positional encodings. These N *object queries* (usually, $N = 100$) specialize in attending to different parts of the image during training. Lastly, the outputs of the decoder are propagated through N Feed-forward Neural Networks, resulting in N predictions. The N predicted bounding boxes must be mapped to the ground truth bounding boxes in a bijective manner, transforming it into a bipartite matching problem. Usually, N is picked so that it is always higher than the estimated number of object detections in a single image. The excess predictions are then mapped to a no object class as a sort of padding. The loss is then calculated based on the Hungarian algorithm, which determines the correspondence between predicted bounding boxes and the ground truth. The bipartite matching principle forces the model to make a single unique prediction for each object directly, removing the need for extra post-processing components like Non-Maximum Suppression and making it fully trainable end-to-end.

DETR has inspired a whole range of similar architectures aiming to improve the performance regarding one or more parts of the pipeline. Deformable DETR [51] was introduced to address the complexity issues observed in the encoder self-attention and decoder cross-attention of DETR. While attention is initially uniform and dense, the model learns to focus its attention on only a subset of positions in the feature map, resulting in sparse final attention. This transition from dense to sparse attention is slow to converge. Deformable DETR solves this by only attending to a limited set of positions around the reference point. These positions are sampled from learnable offsets to the reference point, inspired by the deformable convolution operation. This *Deformable Attention* can be applied to multi-scale feature maps as well, boosting the performance for TOD. Dynamic DETR [13] approaches the same complexity problem in a different manner by using dynamic attention. They are better able to approximate full self-attention than Deformable DETR by using convolution-based approaches. In addition, the decoder cross-attention is replaced by RoI-based dynamic convolution operation with 1×1 filters to lower the learning difficulty of cross-attention.

Numerous other efforts have been made to improve the decoder cross-attention, as it as was found to mainly be responsible for the slow convergence [39]. Anchor-DETR [43] proposes to use anchor points as input for the decoder cross-attention. In DETR, the object queries that are learnt during training do not have an explicit meaning. When analyzing where these individual object queries attend to, it is empirically found to cover a large region of the image. Anchor points can help the Transformer model to focus on smaller areas where objects are likely to be present. The resulting predicted object queries will be closely concentrated around the initial anchor point, allowing for more precise object localization. DAB-DETR [29] extends on this principle by using 4D coordinates as object queries, representing bounding boxes. These bounding boxes are dynamically updated after every iteration to improve the cross-attention computation, significantly improving the detection of small objects compared to DETR. H-DETR [21] uses a one-to-many matching scheme during training alongside the standard one-to-one matching, allowing for more queries to effectively learn spatial information during the training process. In one-to-one matching, queries that are matches to no object are assigned a 'None' class. The downside of this approach is that only the class-based loss can be used when updating the weights in such an instance. This hybrid matching can be applied to other variants of DETR as well, like Deformable DETR.

DN-DETR [25] takes a different approach by addressing the instability of the Hungarian matching process. Small changes in the matching cost matrix can lead to significantly diverging outcomes, and blocking pairs have been proven to exist in the bipartite matching problem. These difficulties are especially pronounced in the early stages of training. DN-DETR introduces an an auxillary output task where the model must recover the original ground truth bounding box given a version with added noise to the ground truth. This was found to increase training speed significantly.

DINO [47] combines and improves upon ideas proposed in DAB-DETR and DN-DETR, both viewing decoder queries as anchor boxes and implementing a query denoising auxillary task. Where DN-DETR does not allow the model to predict a no object class for the auxiliary tasks when no anchors are nearby, DINO implements a contrastive denoising training, where the model has the ability to reject useless anchors.

To compete with both the impressive speed and accuracy of YOLO models, RT-DETR [32] was introduced in an attempt to make the DETR architecture more efficient. Results indicate that the encoder in Deformable DETR is responsible for 49% of the total processing, while only accounting for a 11% performance increase [27]. So instead, the authors propose to only perform encoding of the high level features of the multi-scale feature map, fusing them later with the lower level features. Additionaly, an IoU-aware Query Selection process is designed to help sort out better queries.

C. CNN Video Object Detectors

Some Object Detection Networks have started to make use of temporal information. Alqaysi et al. [1] introduces the idea of stacking 3 monochrome frames to serve as the input for a YOLOv4 model. This shows the potential to increase performance by making temporal information available to the model early on in the object detection pipeline. T2-YOLOv5 [11] implements a two-stream approach, combining both 3 stacked monochrome frames and motion-only features in a YOLOv5 architecture. This is achieved by using a second backbone which processes the absolute difference of input frames, combining the extracted features with the main YOLOv5 architecture in a late fusion manner. An issue with this method is that it requires a stationary camera, as objects have to move across the frame in a predictable way for the model to learn.

D. Video Transformers

Video Transformers refer to the class of Transformers that incorporate temporal context for object detection. TransVOD [48] was the first method in this new line of Transformer models. It uses Deformable DETR to extract purely spatial information from the feature maps of multiple frames. A given number of reference frames are randomly sampled from frames within a certain window size of the current frame. The object queries outputted by the Deformable DETR decoder (referred to as spatial object queries) are fed to a Feed-forward Neural Network to predict the object classes. A selection of these spatial object queries is made of the top-k confident predictions, on which cross-attention is performed in the Temporal Query Encoder (TQE) with the spatial object queries from the current frame to aggregate the object queries. This results in enhanced object queries that carry temporal information. This procedure can be performed again with these enhanced object queries and a smaller top-k selection of spatial object queries for additional gain, but possibly missing interesting spatial object queries with low confidence. Likewise, the output of the encoder for the current frame is fused with the outputs of the reference frame encoders in the Temporal Deformable Transformer Encoder (TDTE). Deformable Attention is applied between the enhanced object queries and the fused encoder output in the Temporal Deformable Transformer Decoder (TDTD), which is then fed to a Feed-forward Neural Network for final classification and box regression. TransVOD++ and TransVOD Lite are variations of TransVOD. TransVOD++ is focussed on improving the quality of the detections, where TransVOD Lite aims to make Video Object Detection available in real-time.

PTSEFormer [42] proposes a different temporal-spatial feature aggregation method. In this model, feature aggregation happens in between the Image Transformer encoder and decoder. A correlation operation is defined which resembles the standard encoder module, except there is only a residual connection between the queries Q and the output of the Multi-Head Attention. A DETR encoder outputs features of the current frame to be used as the query for the correlation

operation, while the key/value pair are generated from the reference frame features extracted by other DETR encoders. A Gated Correlation operation (STAM) is also introduced, inspired by the gate control design in Gated Recurrent Units. Inversely, the queries are obtained from the output of the respective reference frame, with the key/value pair coming from the current frame. These outputs are then aggregated using another correlation operation and consecutive Gated Correlation operation, followed by a DETR decoder for the final prediction. While TransVOD incorporates temporal information after the decoder, PTSEFormer allows for fusion of spatio-temporal features between the encoder and the decoder.

Feature Aggregated Queries (FAQ) [12] dynamically aggregates queries of reference frames and only employs a single Image Transformer structure, on the current frame. The aggregated query is then used as the query of this Image Transformer decoder. Thus, spatial and temporal features are fused in the decoding step, an earlier stage than in TransVOD but later than PTSEFormer.

Video Sparse Transformer With Attention-Guided Memory (VSTAM) [16] proposes a novel External Memory module that stores features with the highest attention weights, allowing the encoder to access these important frames together with the current adjacent neighbouring frames. VSTAM incorporates temporal information already in the encoder, by allowing self-attention both within each frame, the same position across multiple frames and random positions across multiple frames.

E. Feature Aggregation

Feature aggregation methods are frequently compared to Transformer models, thus playing a crucial role in the ongoing discourse on optimal architectures for video analysis. Flow-Guided Feature Aggregation (FGFA) [52] uses optical flow to capture object motion and improve object detection in video data. Optical flow represents the motion of objects in consecutive frames, which can be utilized to align features of objects over time.

Sequence Level Semantics Aggregation for Video Object Detection (SELSA) [45] aggregates information across frames by considering high-level semantics. Object features in different frames are fused based on their semantic similarity.

Memory Enhanced Global-Local Aggregation for Video Object Detection (MEGA) [9] incorporates a memory mechanism to retain information from previous frames, allowing it to store global semantic context. Unlike SELSA, MEGA can also make use of local information, fusing it with the global information to increase accuracy.

IV. Methodology

A common characteristic shared between the Video Transformers discussed in the background section is that they use temporal information after extracting information of each individual frame in the backbone. TransVOD first handles purely the spatial components of each frame, opting to combine temporal information at a later stage [48]. PTSEFormer fuses the information of multiple frames between the encoder and the decoder [42]. In Feature Aggregated Queries, temporal information is introduced in the decoder [12]. VSTAM incorporates the information of adjacent frames during encoding [16].

Given that small objects provide limited visual information, we hypothesize that introducing temporal information to the model as early as possible in the detection pipeline is beneficial for the performance for detecting small objects. To test this hypothesis, we propose a novel transformer architecture that integrates temporal information at an earlier stage in the detection pipeline. Specifically, our approach modifies the backbone of a detection model, introducing temporal features from the very beginning of the detection pipeline.

A. Model Architecture

Early fusion is achieved by replacing the conventional single-frame Transformer backbone with a Video Swin Transformer backbone for a given detection model. This choice is inspired by Arnab et al. [3], who explored the benefits of the structure of their video classification model ViViT. Instead of sampling the patches per individual frame during embedding, ViViT uses 3D patches extended along the temporal dimension. Multiple frames are stacked, rather than looked at seperately, so that overlapping regions in adjacent frames end up in the same patches. This sampling strategy, known as tubelet embedding, allows for spatio-temporal features to fuse already during embedding [3]. A visual explanation can be found in Figure 3.

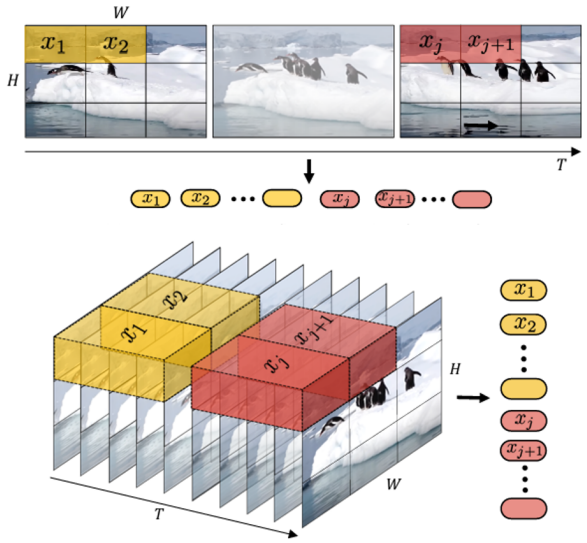

Fig. 3. Top: single-frame sampling. Bottom: Tubelet embedding. Taken from [3].

The Video Swin Transformer [31] combines the tubelet embedding approach with the shifting attention windows of the regular Swin Transformer. As many object detection models use the Swin Transformer as the backbone, Video Swin Transformer is a fitting candidate to introduce temporal information early in the model. This sets it apart from the

Video Transformers discussed in the Section III, as these models only use additional frames further down the object detection pipeline. While other alternative directions are possible, like applying feature aggregation techniques discussed in Section III-E, replacing the Swin Transformer backbone with a Video Swin Transformer offers a unique advantage. This modification incorporates time early into the model without fundamentally changing the architecture. This allows for a precise study of the effects of early temporal context in isolation.

1) Video Transformer Integration: The added temporal dimension results in a shape mismatch when replacing the Swin Transformer backbone with a Video Swin Transformer. Specifically, the Video Swin Transformer output feature maps have an additional temporal dimension alongside the standard spatial dimensions (height and width). Typically, an object detection model expects a feature map with just the spatial and channel dimensions. To resolve this, a fusion mechanism needs to be added which reduces the temporal dimension without losing the valuable information that can be learned from the motion of objects. These fusion mechanics aim to capture temporal dependencies across frames, enhancing the backbone output when compared to the output of a single-frame backbone, while remaining the same shape. This is important, as it enables us to examine the impact of introducing temporal context at the backbone level. By keeping the original detection model's structure intact, any observed improvements can be attributed to the added temporal information.

B. Fusion Mechanics

Multiple fusion mechanics will be explored in this study to determine the most effective way to replace a Swin Transformer backbone with a Video Swin Transformer with only minimal change to the structure of the object detection model itself. Three different fusion mechanics will be tested: First, a **summation** over the temporal dimension is added between the backbone and the detection model to reduce the dimensionality of the backbone output. This parameter-free fusion method deviates the least from the single-frame model, as no extra parts are needed. However, the fact that it's a non learnable method means that it will likely struggle with the newly added temporal information. Second, a learnable approach will be tested with a **Multi-Head Attention** module. We use four attention heads to limit the number of extra parameters that are introduced in the model, while still allowing some different relations to be learned. While the ability to establish connections across multiple frames is a big advantage, the increased complexity might be detrimental to the model performance. Finally, we introduce a **3D Encoder** by adapting the existing encoder of the object detection model into three dimensions to remove the need for an added piece between the backbone and the model, serving as another learnable fusion method. This combines the advantages of previous methods, both staying close to the original model architecture, while also having the capability to adjust to the added temporal information. An overview of the different fusion mechanics can be found in Figure 4.

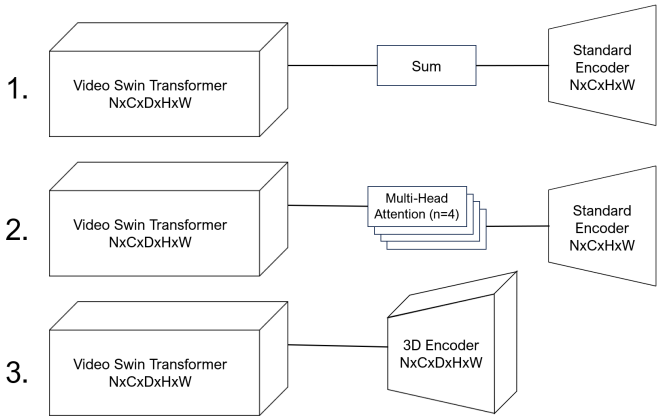

Fig. 4. The different options for fusion mechanics from top to bottom: 1.) Sum, 2.) Multi-Head Attention (n=4), 3.) 3D encoder. Note that each variant is distinct and used separately.

The object detection model we use for experimentation is H-Deformable DETR. The multi-scale deformable attention increases the effectiveness of detecting small objects [51]. Together with the hybrid matching scheme discussed in the Related Work Section, which increases training efficacy [21], it makes H-Deformable DETR a promising model for the purposes of this study. It is important to note that, while we use H-Deformable DETR during this study, our method can be applied to a multitude of other object detection models with a similar structure. We choose to adapt a single-frame Transformer to better isolate the effect of introducing temporal information early on in the model, turning it into a Video Transformer. However, the alterations discussed in this section can also be applied to Video Transformers discussed in Section III-D.

C. Initialization

Another advantage is that H-Deformable DETR has been pretrained with a Swin Transformer backbone. This allows for easy experimentation with different weight initialization methods when switching to a Video Swin Transformer backbone. Two approaches will be tested in this study. One approach consists of taking H-Deformable DETR weights and replacing the backbone weights with the pretrained weights of the Video Swin Transformer, combining the models into one end-to-end model. The other approach involves only the pretrained H-Deformable DETR weights and inflating them to fit the dimensions of the Video Swin Transformer backbone, according to the following procedure.

First, the weights in the patch embedding layer are inflated by inserting a new axis and repeating the weights along this axis. If the original weight tensor has a shape (N, C, H, W), the inflated tensor will have the shape (N, C, D, H, W), where D is the number of frames. The weights relating to the relative position bias tables are then extended using bicubic interpolation to adjust the dimensions. Lastly, some parameters

specific to the 2D Swin Transformer are deleted, such as the attention mask and relative position index . The attention mask controls which tokens can attend to each other, whereas the relative position index helps the model to understand the relative positions of those tokens. Moving to a 3D structure, the relations these tokens have among themselves changes significantly. Thus, these parameters will be deleted from the pretrained weights, as there is no way to appropriately extend these tensors into three dimensions. They will be initialized in the Video Swin Transformer model upon construction, as if no pretrained weights are being used.

D. Temporal Coverage

To determine the optimal temporal fusion method, we train the three variants with different configurations of the number of frames and the distance between them. We introduce the definition of *temporal coverage*, which is defined as the total span of frames used for detection. This value consists of the number of frames (including the current frame) and the stride between them. Different combinations of stride and number of frames can result in the same temporal coverage. This allows for selecting interesting values for the number of frames and stride, which can be easily compared in a meaningful way. We can compare specific models on the basis of how many frames they use (number of frames), how much an object moves across a timed sequence (stride), and how much of the timed sequence the model is able to see (temporal coverage). Figure 5 gives a visual explanation of how this sampling strategy is employed. The notation we use to refer to a certain temporal coverage is generalized as:

$$\text{TEMPORAL COVERAGE: (FRAMES} \times \text{STRIDE)}$$

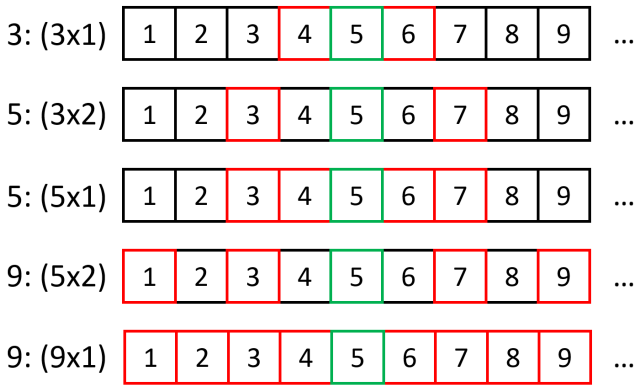

Fig. 5. Visualization of temporal coverage, with a sequence of frames represented as squares with example image ids. The current frame of the sequence used for object detection is highlighted in green, while the sampled support frames are highlighted in red.

V. Experiment

This section will detail the experimental setup used to produce results. Important parameters will be discussed, and the data for training and testing will be covered. Following this, training specifications are provided and evaluation metrics are given.

A. Experimental Setup

For the single-frame model, a Swin-Tiny backbone is used in all experiments. For every fusion method, the Video Swin-Tiny backbone is used. The different fusion mechanics will be tested with various settings of temporal coverages.

Three temporal coverages will be considered. A temporal coverage of three will naturally only consist of three frames with a stride of one. Next, a temporal coverage of five includes three frames with a stride of two and five frames with stride one. Finally, a temporal coverage of nine includes five frames with stride two and nine frames with stride one. An overview of all variations can be found in Table II.

Frames are sampled with the detection frame at the center of the sequence. In cases where this is not feasible, such as when a frame is near the beginning or the end of a video, we adjust sampling by using as many frames as possible from the restricted side of the current frame, and fill up the sequence with frames from the opposite end to meet the temporal coverage requirement. This is necessary, as the backbone requires a fixed number of frames as input, meaning that we must ensure the sequence of frames is always fully populated. Another solution would be to copy the furthest possible frame from the current frame to fill up the sequence. However, this would remove translation of objects between frames, and hereby loosing vital temporal information.

B. Data

The data used in this study is the Visdrone-VID dataset by Zhu et al.[49]. The dataset consists of a training set with 24,198 frames over 56 videos and a test set with 6,322 frames over 16 videos filmed at 24 frames per second. Footage is obtained with a moving drone, recording traffic and pedestrian situations in urban areas. While the videos show substantial overlap in scenery, they can be broadly categorized in either being centered around vehicles in traffic or focused on pedestrians walking. The dataset consists of 11 classes. A breakdown of the class distribution can be found in Figure 13 in the appendix. While this study is concerned with small object detection, we aim to create a model which does not harm its ability to detect larger objects. This makes Visdrone-VID a suited candidate for experimentation, as it contains a variety of different object sizes.

To reduce the redundancy in video data and speed up training, validation and testing, we select frames from Visdrone-VID at an interval of five. Because footage is filmed at a relatively high frame rate, subsequent frames will have a lot of information in common. We decide to filter out some of this redundant data, especially since the multi-frame models would have a lot of overlap when selecting neighbouring frames for detection on the current frame. While we leave out some frames for detection, the multi-frame models still have access to those frames.

C. Training Specifications

The training parameters of the standard H-Deformable DETR implementation are applied in this study as well. The models are trained over 10 epochs with a learning rate of .0002, starting from the pretrained H-Deformable DETR weights. The backbone weights of the pretrained H-Deformable DETR weights are used to initialize the Video Swin Transformer backbone following the procedure detailed in Section IV-A. All models are trained on a GPU-cluster consisting of 4 A16 GPUs. Each GPU uses the maximum possible batch size of one, making the effective batch size during training four. After each epoch, the model is evaluated and after training is finished, the best result is be reported from among those evaluations.

D. Evaluation Metrics

To evaluate the model performances, we employ a variety of quantitative metrics to analyze the different fusion methods. To determine the best performing multi-frame method, we use mean average precision at an IoU overlap of .5. The single frame and the best performing multi-frame method are then compared with a precision-confidence curve, a recall-confidence curve, a precision-recall curve, a F1-confidence curve and finally a confusion matrix to see the mistakes the models are making. These metrics give a good overview of the differences in the models. To give a further comprehensive overview, we will analyze the model based on different scene characteristics to observe how the model performs in different settings. Multiple object/scene properties will be taken into account to categorize a frame so that we can understand the advantages and disadvantages of each model. These results are then contrasted against current state-of-the-art models mentioned in the original Visdrone-VID paper [49].

VI. Results

This section will provide an overview of the results of the experiments detailed in the previous section. First, the different fusion mechanics are compared to determine the most effective way to incorporate temporal information into a model. Then, the best multi-frame method is tested against the single frame method on multiple metrics. These results are then compared to state-of-the-art models previously trained on the Visdrone-VID dataset. Finally, a quantitative analysis is made on the multi-frame and single frame method, which dives deeper into the performance on different characteristics in scenes between these models.

A. Temporal Fusion Methods

In Table II, results of each fusion mechanic are given for each temporal coverage configuration. Model performances are listed as the mean average precision on an IoU-overlap of .5 after their best validation score during training. The results for the Multi-Head Attention fusion method on nine frames with stride 1 are not available due to hardware memory limitations.

TABLE II
Results for temporal fusion variants, trained on different temporal coverages. Temporal coverage is listed as 'Temporal coverage: (Number of frames x Stride)'. All values reported are mAP@50

	Fusion Type		
Temporal Coverage	Sum Fusion	Multi-Head Attention Fusion	3D Encoder Fusion (Enc5x1)
3: (3x1)	32.5	31.2	32.8
5: (3x2)	31.8	29.3	32.0
5: (5x1)	32.1	29.6	33.5
9: (5x2)	27.7	25.7	31.3
9: (9x1)	30.0	-	32.2
Single-frame baseline H-Deformable DETR			36.2

From these results, it is clear that the 3D Encoder variant is the best performing fusion method when using a temporal coverage of five with five frames and stride one. Five frames seems to strike a good balance in the number of frames used for detection, as both three frames and nine frames results in lower precision. A stride of one is optimal not only for the 3D Encoder variant with five frames, but across all other fusion methods and temporal coverage configurations, as a stride of two decreases performance for all models.

B. Single-frame Comparison

We provide a quantitative analysis between the single-frame method and the multi-frame method, using the 5 frames with stride 1 3D Encoder variant, which achieved the best performance as shown in Table II. From this point on, we will refer to this multi-frame variant as Enc5x1, indicating that it uses the 3D Encoder with five frames and stride one, and we will refer to the single-frame baseline model as SF. We visualize the differences between the models by examining some key metrics for each class: the precision-confidence curve, recall-confidence curve, precision-recall curve, F1-confidence curve, and the confusion matrix. The results can be found in Appendix A.

C. State of the Art Comparison

The single-frame and multi-frame models are measured against state-of-the-art models evaluated on the Visdrone-VID dataset [49]. We report the mean average precision at the IoU tresholds of .5 and .75, as well as the mean average recall for 100 detections per frame over all IoU tresholds. We only list the models that were among the top three performers in any category on the dataset. For an extended report on model performances on Visdrone-VID, we refer the reader to the original paper [49].

TABLE III
COMPARISON AGAINST STATE-OF-THE-ART MODELS REPORTED BY ZHU ET AL.[49]. SF IS THE SINGLE-FRAME BASELINE MODEL. ENC5X1 IS OUR 3D ENCODER VARIANT WITH FIVE FRAMES AND STRIDE ONE.

	Metrics		
Model	mAP@50	mAP@75	mAR_{100}
AFSRNet	52.5	19.4	45.1
CN-DhVaSa	48.1	16.8	39.6
DBAI-Det	58.0	25.3	50.8
HRDet+	51.8	16.8	39.0
VCL-CRCNN	43.9	18.3	33.5
CFE-SSDv2	44.8	18.0	41.9
H-Deform. DETR (SF)	36.2	14.3	50.6
H-Deform. DETR (Enc5x1)	33.5	14.2	46.4

D. Quantitative Analysis

This quantitative analysis aims to give a deeper insight into the model performances on different scene/object characteristics, to determine the advantages of each model. We first provide the precision values per object size in Figure 6, and a detailed overview of the precision and recall per class in Table IV.

TABLE IV
COMPARISON ON PRECISION OF SINGLE FRAME BASELINE (SF) AND MULTI-FRAME MODEL (ENC5X1) FOR EACH CLASS ON VISDRONE-VID, IOU.5.

	Precision		Recall	
Class	SF	Enc5x1	SF	Enc5x1
Pedestrian	49.6	41.9	56.8	59.7
People	22.9	19.8	34.7	36.4
Bicycle	6.0	8.6	36.4	26.7
Car	58.5	52.8	71.9	76.0
Van	16.6	22.8	31.1	32.1
Truck	24.5	33.7	41.9	42.9
Tricycle	19.5	19.5	33.5	37.4
Awning-tricycle	14.5	15.8	34.8	18.4
Bus	14.7	14.7	15.7	26.8
Motor	29.2	32.4	36.2	35.7
Others	0.0	4.6	0.0	2.5

An interesting next step is to split the Visdrone-VID dataset into static and moving camera scenes, since some video object detectors are known to struggle with moving camera scenes, as discussed in the Section III. Videos are classified as either static or moving based on significant motion. Static videos are thus not strictly static, since all video data is recorded with a drone, some light movement is present in all videos. We also allow videos with some camera turns to be classified as static. The difference between videos shot by drones recording over large distances and videos shot by a relatively stationary drone is so pronounced that they can be unambiguously categorized into one of the two groups. Four videos are labeled as stationary, while 12 videos are labeled as moving. While this split is heavily imbalanced, it still provides a valuable insight. Using the precision reported in Table IV as a reference for the performance on all videos, Table V offers a comprehensive breakdown of the precision for all classes in each scene, and Table VI does the same for recall.

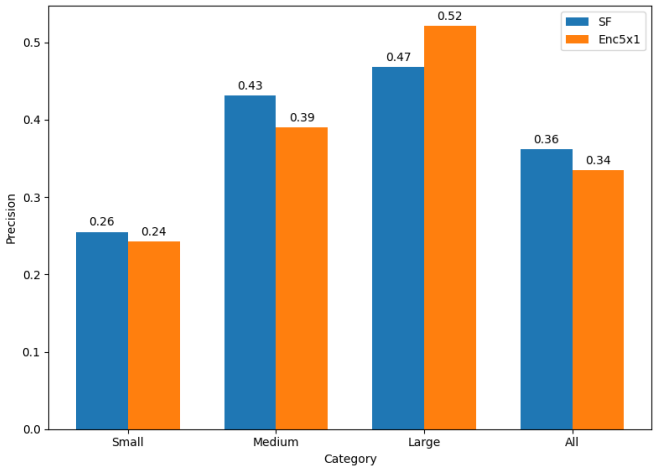

Fig. 6. Average Precision for COCO standardized object sizes.

TABLE V
COMPARISON ON PRECISION OF SINGLE FRAME BASELINE (SF) AND MULTI-FRAME MODEL (ENC5X1) ON STATIC AND MOVING CAMERA FOOTAGE, IOU.5.

	Precision			
Class	SF, Static	Enc5x1, Static	SF, Moving	Enc5x1, Moving
Pedestrian	48.1	41.7	50.3	42.0
People	35.0	27.8	16.7	14.7
Bicycle	10.7	18.4	3.3	3.3
Car	60.3	51.6	57.0	53.7
Van	17.4	20.2	16.1	24.5
Truck	15.9	20.9	30.3	40.9
Tricycle	25.8	24.8	16.4	16.5
Awning-tricycle	30.7	20.3	5.6	10.7
Bus	15.1	15.9	14.6	13.9
Motor	32.8	37.8	27.0	29.1
Others	0.0	12.4	0.0	0.0

TABLE VI
COMPARISON ON RECALL OF SINGLE FRAME BASELINE (SF) AND MULTI-FRAME MODEL (ENC5X1) ON STATIC AND MOVING CAMERA FOOTAGE, IOU.5.

	Recall			
Class	SF, Static	Enc5x1, Static	SF, Moving	Enc5x1, Moving
Pedestrian	57.8	57.2	56.5	60.6
People	36.8	41.3	32.7	32.0
Bicycle	39.4	31.9	31.8	18.1
Car	73.6	75.4	70.6	76.4
Van	31.5	34.5	30.9	30.9
Truck	36.7	27.7	44.1	51.1
Tricycle	38.1	48.0	30.6	31.6
Awning-tricycle	42.2	21.3	23.0	14.2
Bus	9.0	15.6	28.0	53.4
Motor	40.6	43.7	33.6	31.1
Others	0.0	2.5	0.0	0.0

A different perspective would be to look at the speed of objects themselves. By grouping objects based on their movement across frames, we can better understand how the multi-frame model handles objects at different speeds. Since it is able to use temporal context, it might be possible for the multi-frame model to learn something about the object from its speed. We define the speed of an object by measuring the total translation of the center of an object bounding box in pixels between consecutive frames. This gives a new speed to an object for each frame. This could allow the model to discern between different patterns of motions, such as the different accelerations of busses and cars. The speed bins are chosen in such a way that all of them contain a sufficient number of objects, while keeping the ranges of each speed bin meaningful. The first bin contains 251595 (almost) stationary objects, the second bin contains 146890 slow moving objects, the third bin contains 133637 medium-paced objects, and the fourth bin contains 131378 fast objects.

VII. DISCUSSION

The following discussion will seek to further understand the main findings given in the previous section. We will highlight the important patterns in the results and discuss how we can interpret them within the framework of small object detection for Video Transformers. Additionally, we will touch on some of the limitations of the current study

A. Temporal Fusion Methods Results

The results of the comparison of temporal fusion methods reveal an interesting trend when trained on different temporal coverages. Most notably, increasing the stride negatively impacts the performance of all tested fusion methods. When comparing each fusion method with a fixed number of frames while only varying the stride, a consistent drop is observed

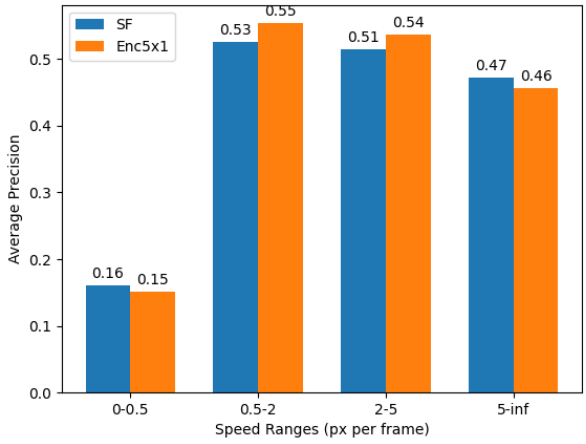

Fig. 7. Average precision grouped on object speed measured in pixels per frame.

across all models. For example, the performance for the 3D Encoder fusion variant decreases from 32.8 to 32.0 when switching from three frames with stride one to three frames with stride two. Similarly, performance drops from 33.5 to 31.3 when comparing five frames with stride one to five frames with stride two, respectively. While an increased stride allows the model to observe a broader temporal range, it appears that the model struggles with the larger jumps between frames.

Analyzing the optimal number of frames gives a more conflicting picture. While the 3D Encoder Fusion method achieves the highest performance with five frames, both the Sum Fusion and Multi-Head Attention Fusion methods perform better with three frames. This difference may be attributed to the complexity of each fusion method. The Sum fusion method, being a simple parameter-free implementation, is perhaps unable to effectively process a larger number of frames. Without learnable parameters, temporal information might be lost when summing the backbone output feature maps. Conversely, the Multi-Head Attention fusion method might become too complex with more added frames leading to difficulties with learning effectively. The 3D Encoder fusion method is potentially a middle road, being able to use more frames without overwhelming the model. We can also observe from Table II that adding more frames beyond this point will not result in an increase in performance.

When compared against the single-frame baseline implementation, all configurations of temporal coverage and fusion mechanics are outperformed by the single-frame baseline. The best-performing multi-frame model, which uses the 3D Encoder Fusion approach, falls short of the baseline by 3 percentage points. This suggests that the model has difficulty adapting to the added temporal context.

B. Single-frame Comparison Results

When comparing the precision-confidence curve of the single-frame model to the multi-frame model using the 3D Encoder variant with five frames and stride one, we see a sharper increase in the precision for the single-frame model.

This indicates that the single-frame model is more conservative in assigning high confidence scores, being able to reach perfect precision for all classes at a confidence of 0.836 as opposed to the multi-frame variant, which only reaches perfect precision at a confidence of 0.920.

Ideally, precision should be high across all confidences, even at the low end. At this lower spectrum of confidence values, we can see some key differences between the models on various classes. The multi-frame model has a lower performance for the *pedestrian*, *car* and *people* classes, but has a better performance on the *truck*, *motor* and *van* classes at lower confidences. An interesting observation here is that the multi-frame model performs noticeably better at distinguishing between the closely aligned classes *car*, *van* and *truck*, while it seems to struggle with *pedestrian* class. We can observe this by looking at the range of lower confidences in Figures 14 and 15, noting a big shift in performance between the models on these classes. This is further backed up by analyzing the confusion matrices in Figures 22 and 23. They show increased off-diagonal numbers for the classes *car*, *van*, *truck*. From the confusion matrix of the multi-frame method in Figure 15, we can also observe that the reason for the performance drop in the *pedestrian* class is due to the model making much more false positive predictions of that class.

The anomalies in the precision-confidence graph where there occurs a drastic drop to zero at higher confidence scores signify that the model is unable to make any predictions exceeding this confidence threshold. The single-frame model is not able to make highly confident predictions for the class *tricycle*. This can most likely be attributed to the relatively low class count of *tricycle* (as can be seen in Figure 13), and a strong resemblance to the class *bicycle*. In the case of the multi-frame model, the precision for the *people* class drops to zero at higher confidence thresholds. This is unexpected, since *people* is one of the larger classes in the dataset with a significant number of training samples. As seen earlier, this is the result too many false positives for the *people* class, as shown by Figure 23. A similar phenomenon happens for the class *car* in the multi-frame model. Figure 13 shows that these are among the most frequent classes in the training set. This indicates that the multi-frame model is biased towards frequent classes. The overconfidence of the multi-frame model then leads it to assign a high confidence score to a *people* prediction that is incorrect, resulting in the drastic drop in precision.

A notable difference between the single-frame and multi-frame models in the recall-confidence curve is the *awning-tricycle* class. In the single-frame model, the *awning-tricycle* curve is close to the *all classes* average, whereas in the multi-frame variant, it is one of the lowest ranking classes. If we look at the confusion matrices in Figures 22 and 23, we see that the multi-frame model misses the *awning-tricycle* class entirely (meaning no other prediction is made for that object) almost twice as much as the single-frame model.

This trend can be also be found in the precision-recall and F1-confidence curves, where closely related classes result in the biggest performance gaps. Another universal trend in all graphs is the ability of the multi-frame model to correctly handle the *other* class, although in a very limited way, with performance consistency the lowest of all classes. However, as can be seen from the precision-recall curve of the single-frame model, the baseline makes no *other* predictions at all. This once again shows the conservative detection capability of the single-frame model, which is a disadvantage in this case.

C. State of the Art Comparison Results

It is apparent from Table III that both the single-frame H-Deformable DETR baseline and our own implemented H-Deformable DETR (Enc5x1) do not outperform the best reported models of the Visdrone-VID dataset when compared on mean average precision with IoU .5 and .75. However, the recall of both models are comparable to the reported values on Visdrone-VID. This means that both models are making too many false positive predictions. This is supported by the confusion matrices in Figure 22 and Figure 23. Both models predict objects where in reality there are none.

It is important to note that the focus of this study is on adapting a generic object detection Transformer to incorporate temporal information. This means that no further optimization techniques have been applied. In contrast, many state-of-the-art models trained on Visdrone-VID may have used specialized techniques for the specifics of the dataset. This could also be an explanation for the performance difference between the models.

D. Quantitative Analysis Results

The results of Table IV reaffirm the findings of Section VII-B. For certain classes, like *van* and *truck*, the multi-frame model achieves a better performance on both precision and recall metrics. On frequently occurring classes like *car*, *pedestrian* and *people*, we can observe that the multi-frame model makes more predictions of these classes, resulting in a higher recall but a lower precision.

Examining Figure 6, which compares the precision of both models across objects grouped by COCO-standardized size categories, reveals some interesting details about the multi-frame model. Although its design was intended to improve the performance on small objects, the only performance gain of the multi-frame model is achieved on large objects. This could be attributed to the inherent nature of large bounding boxes, which are more likely to overlap with themselves in neighboring frames, relative to smaller objects. This might make it easier for the multi-frame model to keep track of the object as it moves across time.

Tables V and VI provide the differences in precision and recall on both static camera footage as well as moving camera footage. It is difficult to establish a trend from these results. On average, the single-frame model performs better in terms of precision on static camera footage, particularly for frequently observed classes like *car*, *pedestrian* and *people*. However, the results on these classes lie closer together when focusing on moving camera footage. Here, the multi-frame model even

outperforms the single-frame model by a larger margin on the classes *van*, *truck* and *awning tricycle* compared to the static camera footage. Differences in recall between static and moving camera footage seem more arbitrary. While the *bus* class improves by a huge margin for the multi-frame model in moving camera footage (especially when comparing against the single-frame model), the opposite is true for the *bicycle* class. Overall, the multi-frame model appears more effective at handling moving camera footage when evaluated on precision, being able to leverage the temporal context to make consistent predictions, while recall remains more variable.

Finally, when observing the results of Figure 7, we see that the multi-frame model achieves higher performances on slow to medium-paced objects. This indicates that, while the model can benefit from motion information of objects, objects that are too fast prove challenging for the multi-frame model. This may be linked to the previously discussed theory that sufficient bounding box overlap between consecutive frames is needed for the model to excel, as both small and fast objects prove to be difficult for the multi-frame model.

E. Limitations

Although the results of this study provide a meaningful insight into the differences between models, there are several limitations that were encountered during experimentation worth addressing. The availability of high quality data containing both moving and static camera footage is very limited. In this regard, the Visdrone-VID dataset is suited well for this study, as it features both small objects and moving camera footage, both of which were identified as challenging for previous methods discussed in SectionIII. However, the dataset does have its shortcomings. It suffers from inconsistent labeling issues, which negatively impact model performance. For example, in Figure 8 we can see that the car in this frame is wrongly labeled as an *awning-tricycle*. Since *awning-tricycle* is already a very rare class in this dataset, these errors can have great effect on the model's ability to understand that class. Furthermore, since the model correctly predicts the object as *car*, it is considered an error during evaluation, lowering the performance scores for the model.

Fig. 8. Example of annotation error in Visdrone-VID. Cyan is the annotated label, while blue is the model prediction.

More nuanced errors also appear in the dataset. In Figure 9, we see an object annotated as *bicycle*, while the exhaust pipe on its side suggests that it is actually a motor. Indeed, the model predicts the object as *motor*.

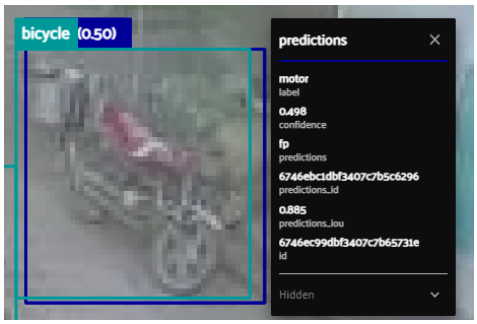

Fig. 9. Confusion between bicycle and motor in Visdrone-VID. Cyan is the annotated label, while blue is the model prediction.

A similar issue is found in the distinction between the classes *people* and *pedestrian*. The authors of the Visdrone-VID papers define the classes as follows: "If a human maintains standing pose or walking, we classify it as a pedestrian; otherwise, it is classified as a person." [49]. However, in Figure 10 it is obvious that both humans maintain a standing pose, but are labeled as *people*. Since they maintain a standing pose, the model correctly classifies both humans as *pedestrian*.

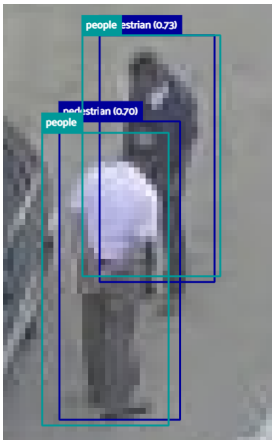

Fig. 10. Confusion between people and pedestrian in Visdrone-VID. Cyan is the annotated label, while blue is the model prediction.

Even humans who are in a sitting position, and thus should be classified as *people*, are sometimes annotated incorrectly. In Figure 11, three people sit on a motor, yet the human in the middle is labeled *pedestrian*. This shows the inconsistency of the annotations in the Visdrone-VID dataset. This is a challenging issue, as the problem is most pressing in difficult edge cases, such as closely related classes. We can see from the confusion matrices in Figures 22 and 23 that the classes in the given examples are those that the model struggles to differentiate.

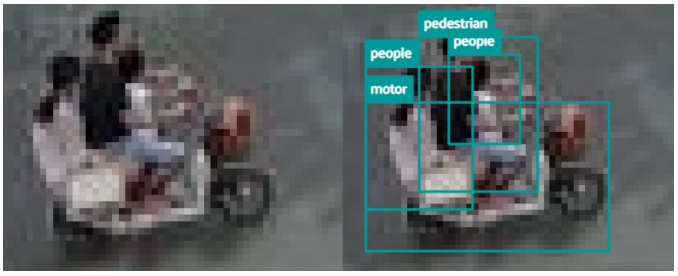

Fig. 11. Confusion between people and pedestrian on a motor in Visdrone-VID. Cyan is the annotated label.

Another edge case that leads to lower performance scores is the lenient approach to the annotations of objects leaving or entering the frame. In most occurrences of objects leaving a scene, the model is able to detect the object longer than they are annotated. Conversely, the model often picks up on objects entering a scene before they are even annotated. While this behaviour should be encouraged in the model, it currently results in a lower precision. An example of this problem is displayed in Figure 12, where the parked car in the bottom left is about to leave the scene as the camera moves forward.

Fig. 12. Early cut-off of annotations in Visdrone-VID. Cyan is the annotated label, while blue is the model prediction.

Lastly, due to time constraints, we were not able to experiment with the model's full capability after training from scratch. Instead, we relied on the pretrained weights of the H-Deformable DETR model. While this approach is common and gave a clear picture of differences between models, it might mean that the potential of the multi-frame model was not fully reached. Ultimately, the weights are derived from a 2D framework following the procedure detailed in Section IV-C. Some of the learned patterns might not carry over well to 3D, and the model might struggle to relearn new connections.

F. Future Work

Following the discussion on the limitations of the current study, several new interesting research opportunities arise. As shown by the results of our experiments, the adapted H-Deformable DETR model struggles with interpreting the added temporal context introduced early in the object detection pipeline. A deeper analysis of the model mechanics could reveal the exact difficulties the model faces when dealing with multiple frames. Future studies could identify the current bottlenecks of early temporal information.

For example, the current Video Swin backbone used in the model may struggle to extract meaningful features across frames due to the original weights being focused on single-frame object detection. A potential solution could be to experiment with different training techniques that allow the model to better capture temporal dependencies. A study on the effect of pretraining on large-scale video datasets and the possible increased ability of the model to extract motion-related features more effectively could be worth considering.

Additionally, more refined fusion mechanics could be explored in future work. Currently, we only consider three fusion mechanics covering a broad spectrum of possible complexity, ranging from a simple summation to Multi-Head Attention with many learnable parameters. Our results show that a compromise between simplicity and complexity is most optimal. However, there are potential fusion mechanics other than the 3D Encoder that could be developed. A separate study solely focused on finding the optimal way of 'collapsing' the temporal dimension in the feature map while preserving the information would be a great supplement to our findings.

Further research could also focus on using our approach with various other object detection models. While the goal of this study was to specifically adapt a Transformer to include temporal information, other architectures might deal with early temporal information better. A comparison of a wide variety of models could further highlight the current shortcomings of our method.

Lastly, the lack of sufficient high-quality video data containing small objects from both a static and moving camera perspective could be addressed in future work. While Visdrone-VID was a great baseline for the comparison done in this study, the increasing demands on model performances require datasets with higher quality annotations. Exploring meaningful partitions within the dataset could be considered, such as a balanced split in static and moving camera footage, among other characteristics found in various real-life scenarios.

VIII. Conclusion

The purpose of this paper was to examine the applicability of Transformers in tiny object detection under challenging circumstances. Specifically, we aimed to address the effectiveness of early spatial-temporal feature fusion.

Our findings are that **early fusion of spatial-temporal features does currently not result in improved mAP scores for Video Transformers** on datasets containing small objects with both moving and static camera footage.

When comparing the performance of single-frame backbone to the multi-frame backbone **distinct trade-offs** were highlighted in the results. While the multi-frame model showed a bias towards making false positive predictions for the most frequent classes, it was better equipped to distinguish rarer classes in the dataset.

Among the tested temporal fusion strategies, **adapting the encoder of the detection model into a 3D convolution** proved to deliver the best results. It introduced a learnable way of combining the temporal information of multiple frames, while staying close to the original model architecture.

Interestingly, the proposed early temporal fusion strategy did not achieve better results on small objects, as it was originally intended and designed for. However, our method did outperform the single-frame model in scenarios with **large objects, rare classes and medium speed objects**. These results indicate that, while the multi-frame model struggles with interpreting the added temporal information in the context of small objects, it can offer some unique advantages in certain areas.

In conclusion, this study explored the effects of introducing temporal information early on in the object detection pipeline. By experimenting with different fusion strategies, new insights were gained in how early temporal context can affect a model's performance. By testing the model under different constraints regarding scene/object characteristics, we were able to better understand the current challenges facing Transformers in the domain of tiny object detection. These findings can steer future work in this field by identifying and resolving the bottlenecks in our proposed method, such that a performance gain can be achieved on a broader range of real-world applications.

References

[1] Hiba Alqaysi, Igor Fedorov, Faisal Z. Qureshi, and Mattias O'Nils. "A Temporal Boosted YOLO-Based Model for Birds Detection around Wind Farms". In: *Journal of Imaging* 7.11 (2021). ISSN: 2313-433X. DOI: 10.3390/jimaging7110227. URL: https://www.mdpi.com/2313-433X/7/11/227.

[2] Laith Alzubaidi, Jinglan Zhang, Amjad J. Humaidi, Ayad Al-Dujaili, Ye Duan, Omran Al-Shamma, J. Santamaría, Mohammed A. Fadhel, Muthana Al-Amidie, and Laith Farhan. "Review of deep learning: concepts, CNN architectures, challenges, applications, future directions". In: *Journal of Big Data* 8.1 (Mar. 2021), p. 53. ISSN: 2196-1115. DOI: 10.1186/s40537-021-00444-8. URL: https://doi.org/10.1186/s40537-021-00444-8.

[3] Anurag Arnab, Mostafa Dehghani, Georg Heigold, Chen Sun, Mario Lučić, and Cordelia Schmid. *ViViT: A Video Vision Transformer*. 2021. arXiv: 2103.15691 [cs.CV].

[4] Jimmy Lei Ba, Jamie Ryan Kiros, and Geoffrey E. Hinton. *Layer Normalization*. 2016. arXiv: 1607.06450 [stat.ML].

[5] Dzmitry Bahdanau, Kyunghyun Cho, and Yoshua Bengio. *Neural Machine Translation by Jointly Learning to Align and Translate*. 2016. arXiv: 1409.0473 [cs.CL].

[6] Aduen Benjumea, Izzeddin Teeti, Fabio Cuzzolin, and Andrew Bradley. *YOLO-Z: Improving small object detection in YOLOv5 for autonomous vehicles*. 2023. arXiv: 2112.11798 [cs.CV].

[7] Gedas Bertasius, Heng Wang, and Lorenzo Torresani. *Is Space-Time Attention All You Need for Video Understanding?* 2021. arXiv: 2102.05095 [cs.CV].

[8] Nicolas Carion, Francisco Massa, Gabriel Synnaeve, Nicolas Usunier, Alexander Kirillov, and Sergey Zagoruyko. *End-to-End Object Detection with Transformers*. arXiv:2005.12872 [cs]. May 2020. URL: http://arxiv.org/abs/2005.12872 (visited on 08/10/2023).

[9] Yihong Chen, Yue Cao, Han Hu, and Liwei Wang. *Memory Enhanced Global-Local Aggregation for Video Object Detection*. 2020. arXiv: 2003.12063 [cs.CV].

[10] Gong Cheng, Xiang Yuan, Xiwen Yao, Kebing Yan, Qinghua Zeng, Xingxing Xie, and Junwei Han. "Towards Large-Scale Small Object Detection: Survey and Benchmarks". In: *IEEE Transactions on Pattern Analysis and Machine Intelligence* (2023). arXiv:2207.14096 [cs], pp. 1–20. ISSN: 0162-8828, 2160-9292, 1939-3539. DOI: 10.1109/TPAMI.2023.3290594. URL: http://arxiv.org/abs/2207.14096 (visited on 08/25/2023).

[11] Christof W Corsel, Michel van Lier, Leo Kampmeijer, Nicolas Boehrer, and Erwin M Bakker. "Exploiting Temporal Context for Tiny Object Detection". In: *Proceedings of the IEEE/CVF Winter Conference on Applications of Computer Vision*. 2023, pp. 79–89.

[12] Yiming Cui and Linjie Yang. *FAQ: Feature Aggregated Queries for Transformer-based Video Object Detectors*. 2023. arXiv: 2303.08319 [cs.CV].

[13] Xiyang Dai, Yinpeng Chen, Jianwei Yang, Pengchuan Zhang, Lu Yuan, and Lei Zhang. "Dynamic DETR: End-to-End Object Detection with Dynamic Attention". en. In: *2021 IEEE/CVF International Conference on Computer Vision (ICCV)*. Montreal, QC, Canada: IEEE, Oct. 2021, pp. 2968–2977. ISBN: 978-1-66542-812-5. DOI: 10.1109/ICCV48922.2021.00298. URL: https://ieeexplore.ieee.org/document/9709981/ (visited on 08/11/2023).

[14] Patel Dhruv and Subham Naskar. "Image Classification Using Convolutional Neural Network (CNN) and Recurrent Neural Network (RNN): A Review". In: *Machine Learning and Information Processing*. Ed. by Debabala Swain, Prasant Kumar Pattnaik, and Pradeep K. Gupta. Singapore: Springer Singapore, 2020, pp. 367–381. ISBN: 978-981-15-1884-3.

[15] Alexey Dosovitskiy et al. *An Image is Worth 16x16 Words: Transformers for Image Recognition at Scale*. 2021. arXiv: 2010.11929 [cs.CV].

[16] Masato Fujitake and Akihiro Sugimoto. "Video Sparse Transformer With Attention-Guided Memory for Video

Object Detection". In: *IEEE Access* 10 (2022), pp. 65886–65900. DOI: 10.1109/ACCESS.2022.3184031.

[17] Ross Girshick. *Fast R-CNN*. 2015. arXiv: 1504.08083 [cs.CV].

[18] Ross Girshick, Jeff Donahue, Trevor Darrell, and Jitendra Malik. *Rich feature hierarchies for accurate object detection and semantic segmentation*. 2014. arXiv: 1311.2524 [cs.CV].

[19] Anila Gundavarapu, V. Srinivasa Chakravarthy, and Karthik Soman. "A Model of Motion Processing in the Visual Cortex Using Neural Field With Asymmetric Hebbian Learning". In: *Frontiers in Neuroscience* 13 (2019). ISSN: 1662-453X. URL: https://www.frontiersin.org/articles/10.3389/fnins.2019.00067 (visited on 11/02/2023).

[20] Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. *Deep Residual Learning for Image Recognition*. 2015. arXiv: 1512.03385 [cs.CV].

[21] Ding Jia, Yuhui Yuan, Haodi He, Xiaopei Wu, Haojun Yu, Weihong Lin, Lei Sun, Chao Zhang, and Han Hu. *DETRs with Hybrid Matching*. 2023. arXiv: 2207.13080 [cs.CV]. URL: https://arxiv.org/abs/2207.13080.

[22] Ling Jian, Zhiqi Pu, Lili Zhu, Tiancan Yao, and Xijun Liang. "SS R-CNN: Self-Supervised Learning Improving Mask R-CNN for Ship Detection in Remote Sensing Images". In: *Remote Sensing* 14.17 (2022). ISSN: 2072-4292. DOI: 10.3390/rs14174383. URL: https://www.mdpi.com/2072-4292/14/17/4383.

[23] Mate Kisantal, Zbigniew Wojna, Jakub Murawski, Jacek Naruniec, and Kyunghyun Cho. *Augmentation for small object detection*. 2019. arXiv: 1902.07296 [cs.CV].

[24] B Y Lee, L H Liew, W S Cheah, and Y C Wang. "Occlusion handling in videos object tracking: A survey". In: *IOP Conference Series: Earth and Environmental Science* 18.1 (Feb. 2014), p. 012020. DOI: 10.1088/1755-1315/18/1/012020. URL: https://dx.doi.org/10.1088/1755-1315/18/1/012020.

[25] Feng Li, Hao Zhang, Shilong Liu, Jian Guo, Lionel M. Ni, and Lei Zhang. *DN-DETR: Accelerate DETR Training by Introducing Query DeNoising*. arXiv:2203.01305 [cs]. Dec. 2022. URL: http://arxiv.org/abs/2203.01305 (visited on 08/14/2023).

[26] Yehao Li, Ting Yao, Yingwei Pan, and Tao Mei. *Contextual Transformer Networks for Visual Recognition*. 2021. arXiv: 2107.12292 [cs.CV].

[27] Junyu Lin, Xiaofeng Mao, Yuefeng Chen, Lei Xu, Yuan He, and Hui Xue. $D^2ETR : Decoder-OnlyDETRwithComputationallyEfficientCross-ScaleAttention$. 2022. arXiv: 2203.00860 [cs.CV].

[28] Tsung-Yi Lin, Piotr Dollár, Ross Girshick, Kaiming He, Bharath Hariharan, and Serge Belongie. *Feature Pyramid Networks for Object Detection*. 2017. arXiv: 1612.03144 [cs.CV].

[29] Shilong Liu, Feng Li, Hao Zhang, Xiao Yang, Xianbiao Qi, Hang Su, Jun Zhu, and Lei Zhang. *DAB-DETR: Dynamic Anchor Boxes are Better Queries for DETR*. arXiv:2201.12329 [cs]. Mar. 2022. URL: http://arxiv.org/abs/2201.12329 (visited on 08/14/2023).

[30] Ze Liu, Yutong Lin, Yue Cao, Han Hu, Yixuan Wei, Zheng Zhang, Stephen Lin, and Baining Guo. *Swin Transformer: Hierarchical Vision Transformer using Shifted Windows*. 2021. arXiv: 2103.14030 [cs.CV].

[31] Ze Liu, Jia Ning, Yue Cao, Yixuan Wei, Zheng Zhang, Stephen Lin, and Han Hu. *Video Swin Transformer*. 2021. arXiv: 2106.13230 [cs.CV].

[32] Wenyu Lv, Yian Zhao, Shangliang Xu, Jinman Wei, Guanzhong Wang, Cheng Cui, Yuning Du, Qingqing Dang, and Yi Liu. *DETRs Beat YOLOs on Real-time Object Detection*. 2023. arXiv: 2304.08069 [cs.CV].

[33] Ignacio Martinez-Alpiste, Gelayol Golcarenarenji, Qi Wang, and Jose Alcaraz Calero. "Search and Rescue Operation Using UAVs: A Case Study". In: *Expert Systems with Applications* 178 (Apr. 2021), p. 114937. DOI: 10.1016/j.eswa.2021.114937.

[34] Muhammed Muzammul and Xi Li. *A Survey on Deep Domain Adaptation and Tiny Object Detection Challenges, Techniques and Datasets*. arXiv:2107.07927 [cs]. July 2021. DOI: 10.48550/arXiv.2107.07927. URL: http://arxiv.org/abs/2107.07927 (visited on 08/10/2023).

[35] Joseph Redmon, Santosh Divvala, Ross Girshick, and Ali Farhadi. *You Only Look Once: Unified, Real-Time Object Detection*. 2016. arXiv: 1506.02640 [cs.CV].

[36] Aref Miri Rekavandi, Shima Rashidi, Farid Boussaid, Stephen Hoefs, Emre Akbas, and Mohammed bennamoun. *Transformers in Small Object Detection: A Benchmark and Survey of State-of-the-Art*. 2023. arXiv: 2309.04902 [cs.CV].

[37] Aref Miri Rekavandi, Lian Xu, Farid Boussaid, Abd-Krim Seghouane, Stephen Hoefs, and Mohammed Bennamoun. *A Guide to Image and Video based Small Object Detection using Deep Learning : Case Study of Maritime Surveillance*. arXiv:2207.12926 [cs] version: 1. July 2022. URL: http://arxiv.org/abs/2207.12926 (visited on 08/08/2023).

[38] Shaoqing Ren, Kaiming He, Ross Girshick, and Jian Sun. *Faster R-CNN: Towards Real-Time Object Detection with Region Proposal Networks*. 2016. arXiv: 1506.01497 [cs.CV].

[39] Zhiqing Sun, Shengcao Cao, Yiming Yang, and Kris Kitani. *Rethinking Transformer-based Set Prediction for Object Detection*. 2021. arXiv: 2011.10881 [cs.CV].

[40] Kang Tong and Yiquan Wu. "Deep learning-based detection from the perspective of small or tiny objects: A survey". In: *Image and Vision Computing* 123 (2022), p. 104471. ISSN: 0262-8856. DOI: https://doi.org/10.1016/j.imavis.2022.104471. URL: https://www.sciencedirect.com/science/article/pii/S0262885622001007.

[41] Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N. Gomez, Lukasz

Kaiser, and Illia Polosukhin. *Attention Is All You Need*. 2023. arXiv: 1706.03762 [cs.CL].

[42] Han Wang, Jun Tang, Xiaodong Liu, Shanyan Guan, Rong Xie, and Li Song. *PTSEFormer: Progressive Temporal-Spatial Enhanced TransFormer Towards Video Object Detection*. arXiv:2209.02242 [cs]. Sept. 2022. URL: http://arxiv.org/abs/2209.02242 (visited on 09/07/2023).

[43] Yingming Wang, Xiangyu Zhang, Tong Yang, and Jian Sun. *Anchor DETR: Query Design for Transformer-Based Object Detection*. arXiv:2109.07107 [cs]. Jan. 2022. URL: http://arxiv.org/abs/2109.07107 (visited on 09/07/2023).

[44] Longyin Wen, Dawei Du, Zhaowei Cai, Zhen Lei, Ming-Ching Chang, Honggang Qi, Jongwoo Lim, Ming-Hsuan Yang, and Siwei Lyu. "UA-DETRAC: A New Benchmark and Protocol for Multi-Object Detection and Tracking". In: *Computer Vision and Image Understanding* (2020).

[45] Haiping Wu, Yuntao Chen, Naiyan Wang, and Zhaoxiang Zhang. *Sequence Level Semantics Aggregation for Video Object Detection*. 2019. arXiv: 1907.06390 [cs.CV].

[46] Michael Yang. *Visual Transformer for Object Detection*. 2022. arXiv: 2206.06323 [cs.CV].

[47] Hao Zhang, Feng Li, Shilong Liu, Lei Zhang, Hang Su, Jun Zhu, Lionel M. Ni, and Heung-Yeung Shum. *DINO: DETR with Improved DeNoising Anchor Boxes for End-to-End Object Detection*. arXiv:2203.03605 [cs]. July 2022. URL: http://arxiv.org/abs/2203.03605 (visited on 08/14/2023).

[48] Qianyu Zhou, Xiangtai Li, Lu He, Yibo Yang, Guangliang Cheng, Yunhai Tong, Lizhuang Ma, and Dacheng Tao. "TransVOD: End-to-End Video Object Detection with Spatial-Temporal Transformers". In: *IEEE Transactions on Pattern Analysis and Machine Intelligence* 45.6 (June 2023). arXiv:2201.05047 [cs], pp. 7853–7869. ISSN: 0162-8828, 2160-9292, 1939-3539. DOI: 10.1109/TPAMI.2022.3223955. URL: http://arxiv.org/abs/2201.05047 (visited on 08/07/2023).

[49] Pengfei Zhu, Longyin Wen, Dawei Du, Xiao Bian, Heng Fan, Qinghua Hu, and Haibin Ling. "Detection and tracking meet drones challenge". In: *IEEE Transactions on Pattern Analysis and Machine Intelligence* 44.11 (2021), pp. 7380–7399.

[50] Xingkui Zhu, Shuchang Lyu, Xu Wang, and Qi Zhao. *TPH-YOLOv5: Improved YOLOv5 Based on Transformer Prediction Head for Object Detection on Drone-captured Scenarios*. 2021. arXiv: 2108.11539 [cs.CV].

[51] Xizhou Zhu, Weijie Su, Lewei Lu, Bin Li, Xiaogang Wang, and Jifeng Dai. *Deformable DETR: Deformable Transformers for End-to-End Object Detection*. arXiv:2010.04159 [cs]. Mar. 2021. URL: http://arxiv.org/abs/2010.04159 (visited on 08/10/2023).

[52] Xizhou Zhu, Yujie Wang, Jifeng Dai, Lu Yuan, and Yichen Wei. *Flow-Guided Feature Aggregation for Video Object Detection*. 2017. arXiv: 1703.10025 [cs.CV].

APPENDIX

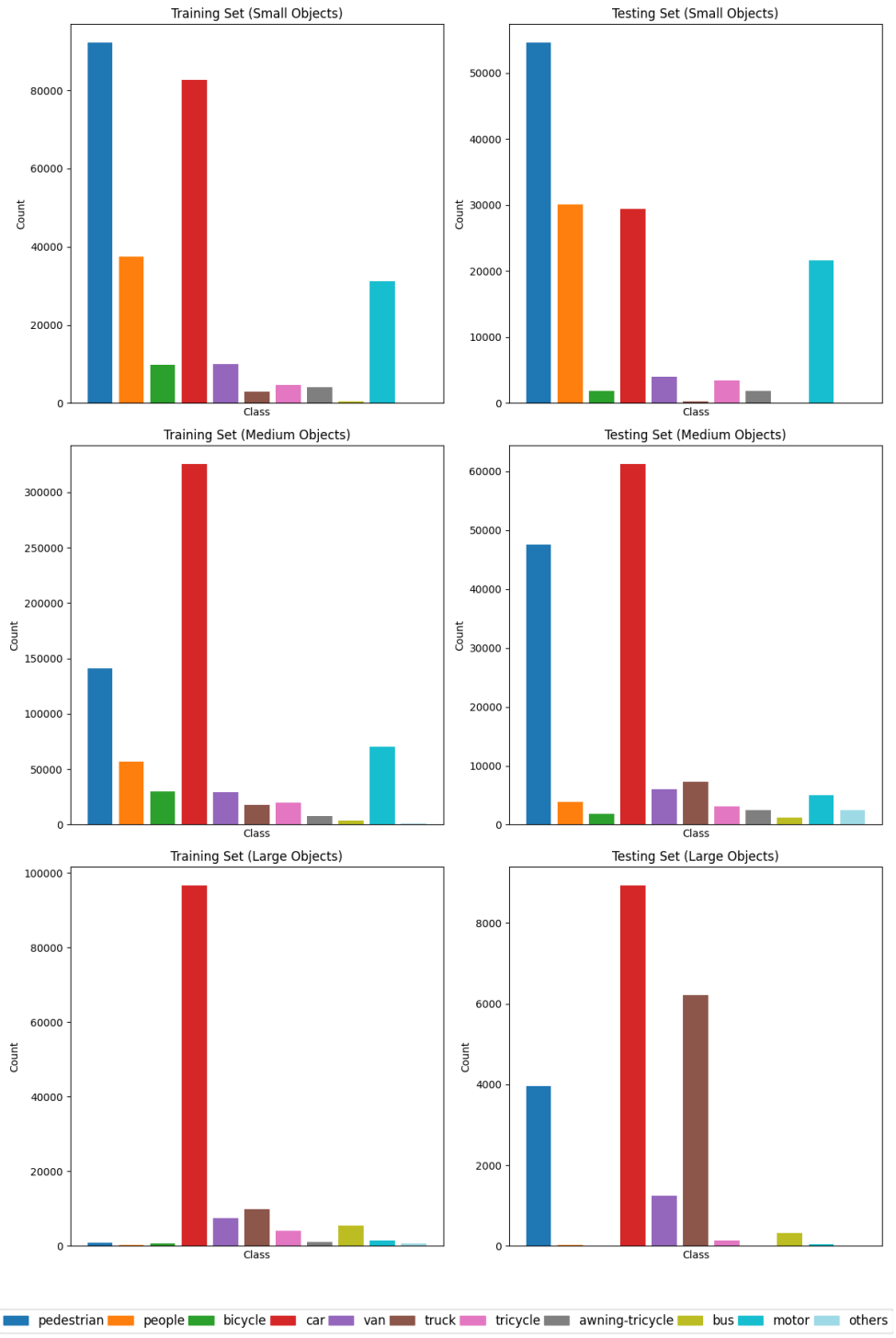

Fig. 13. Class distribution of training and testing set Visdrone-VID [49], grouped along COCO standardized sizes. *Small* objects have a bounding box area less than or equal to 32×32 pixels, *medium* objects have an area between 32×32 and 96×96 pixels, and *large* objects have an area greater than 96×96 pixels.

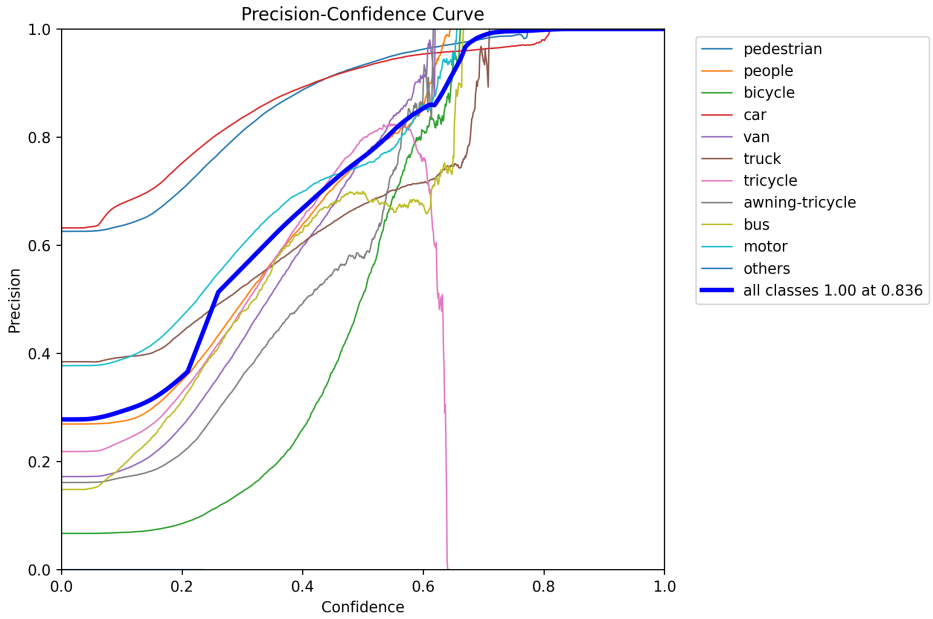

Fig. 14. Precision-confidence curve for H-Deformable DETR with a single-frame Swin Transformer backbone.

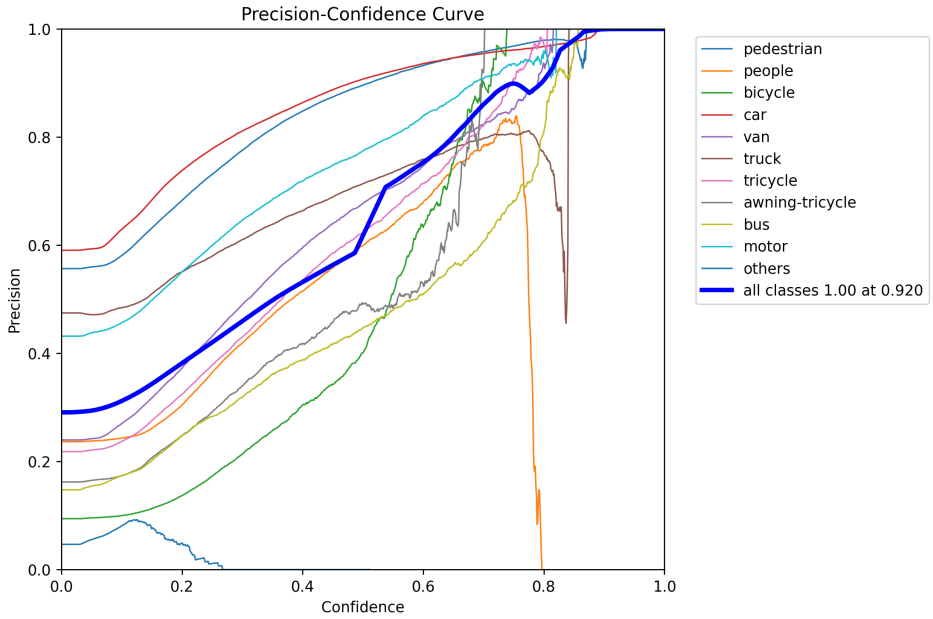

Fig. 15. Precision-confidence curve for H-Deformable DETR with a multi-frame Video Swin Transformer backbone using the Enc5x1 fusion mechanic.

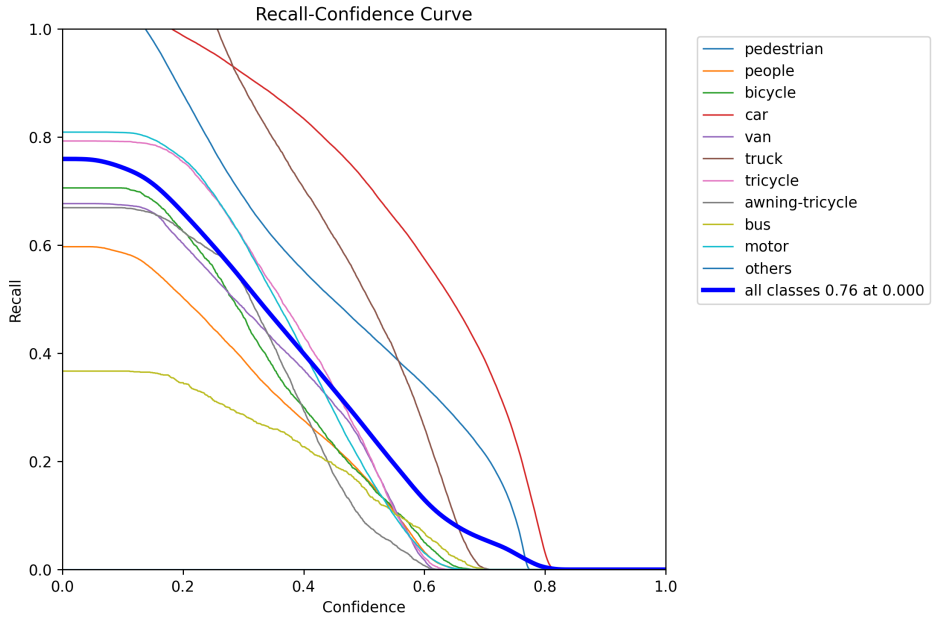

Fig. 16. Recall-confidence curve for H-Deformable DETR with a single-frame Swin Transformer backbone.

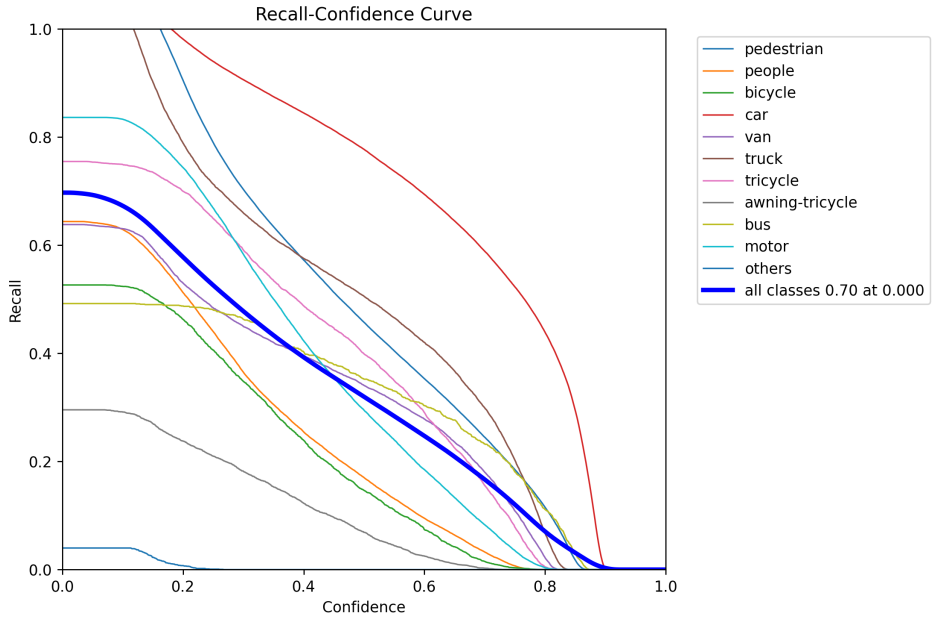

Fig. 17. Recall-confidence curve for H-Deformable DETR with a multi-frame Video Swin Transformer backbone using the Enc5x1 fusion mechanic.

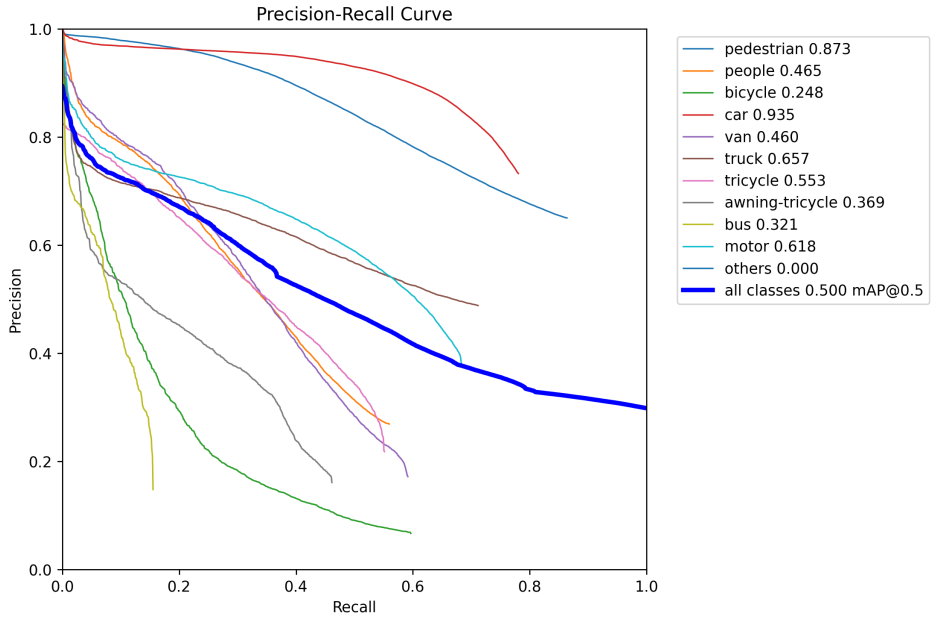

Fig. 18. Precision-Recall curve for H-Deformable DETR with a single-frame Swin Transformer backbone.

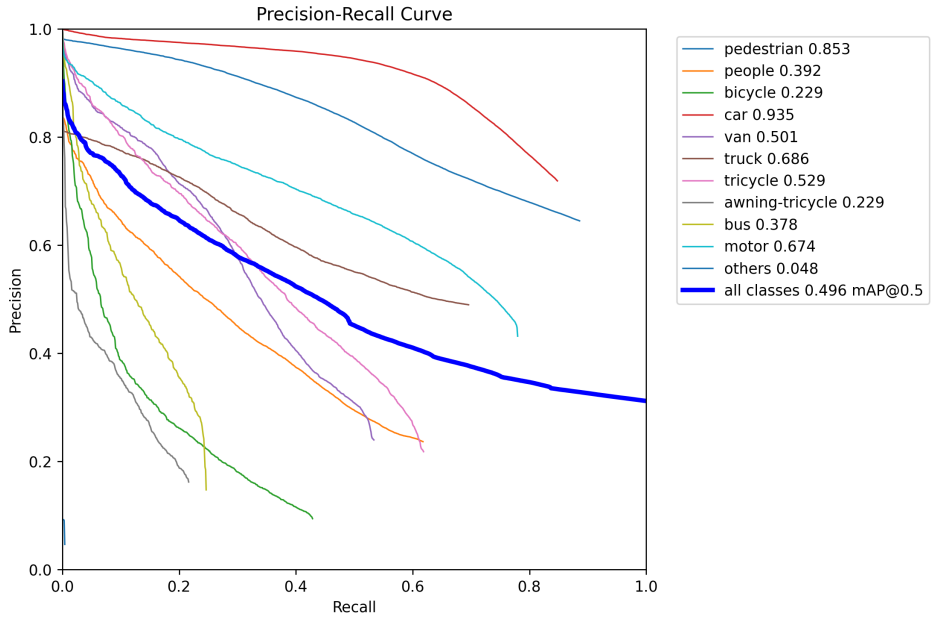

Fig. 19. Precision-Recall curve for H-Deformable DETR with a multi-frame Video Swin Transformer backbone using the Enc5x1 fusion mechanic.

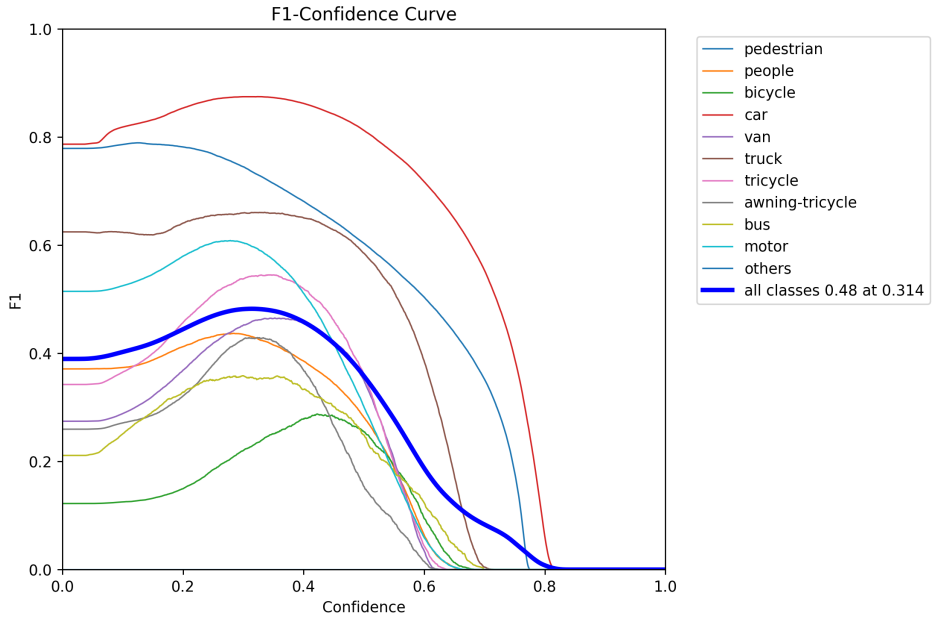

Fig. 20. F1-confidence curve for H-Deformable DETR with a single-frame Swin Transformer backbone.

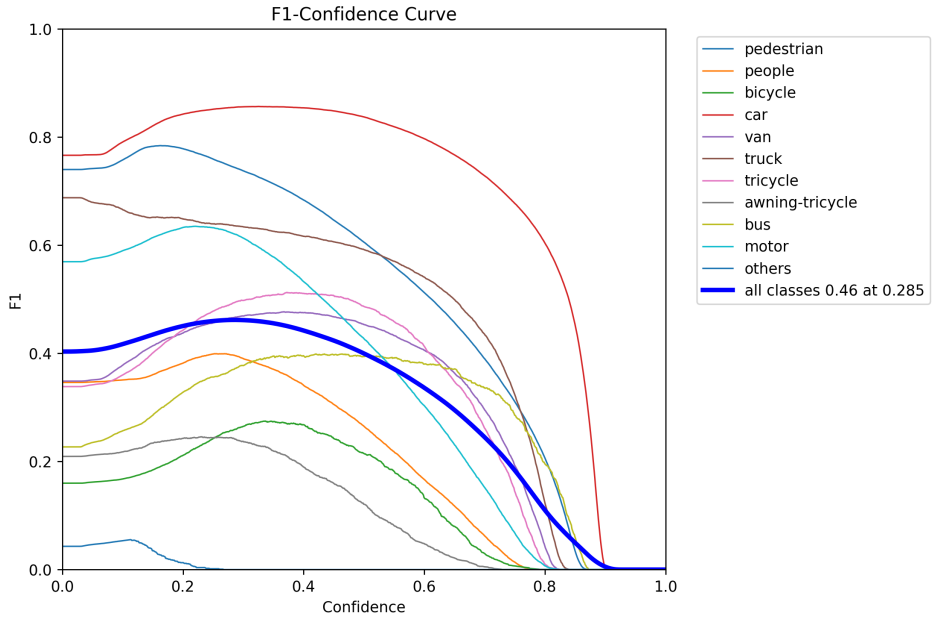

Fig. 21. F1-confidence curve for H-Deformable DETR with a multi-frame Video Swin Transformer backbone using the Enc5x1 fusion mechanic.

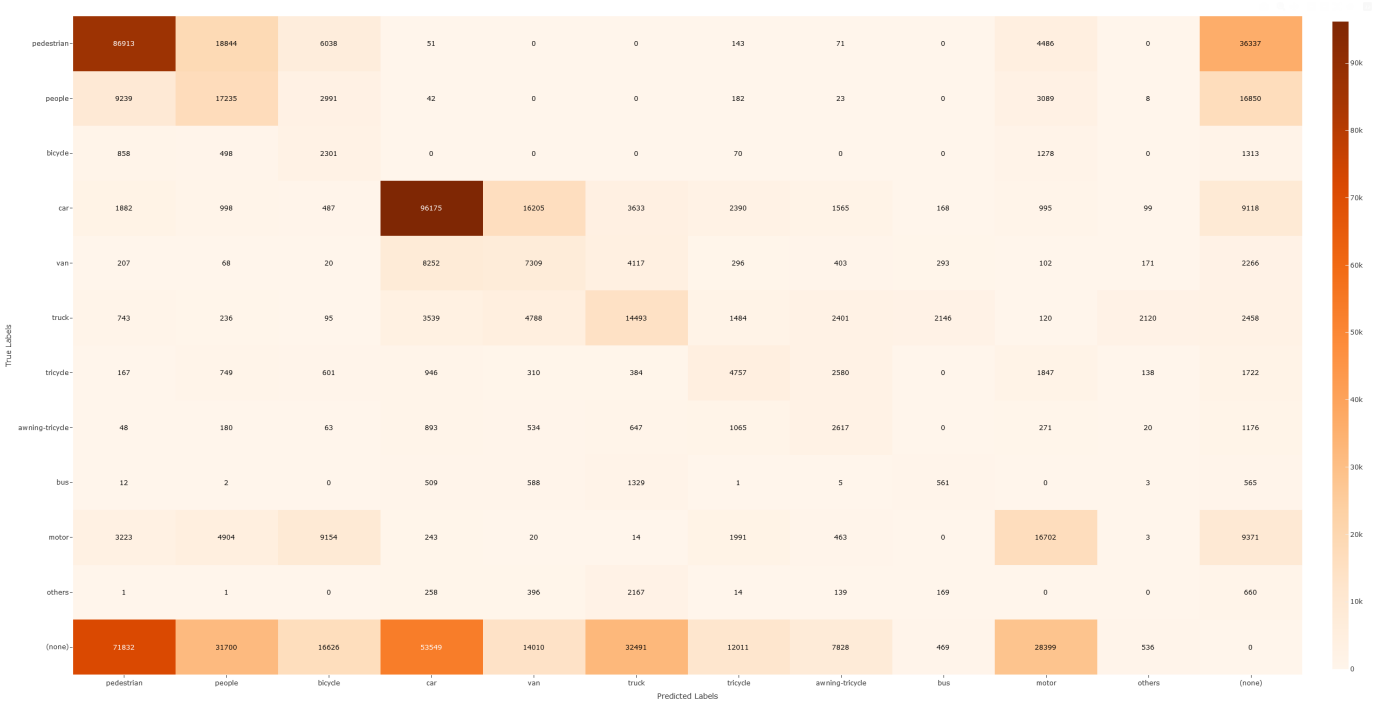

Fig. 22. Confusion matrix for H-Deformable DETR with a single-frame Swin Transformer backbone.

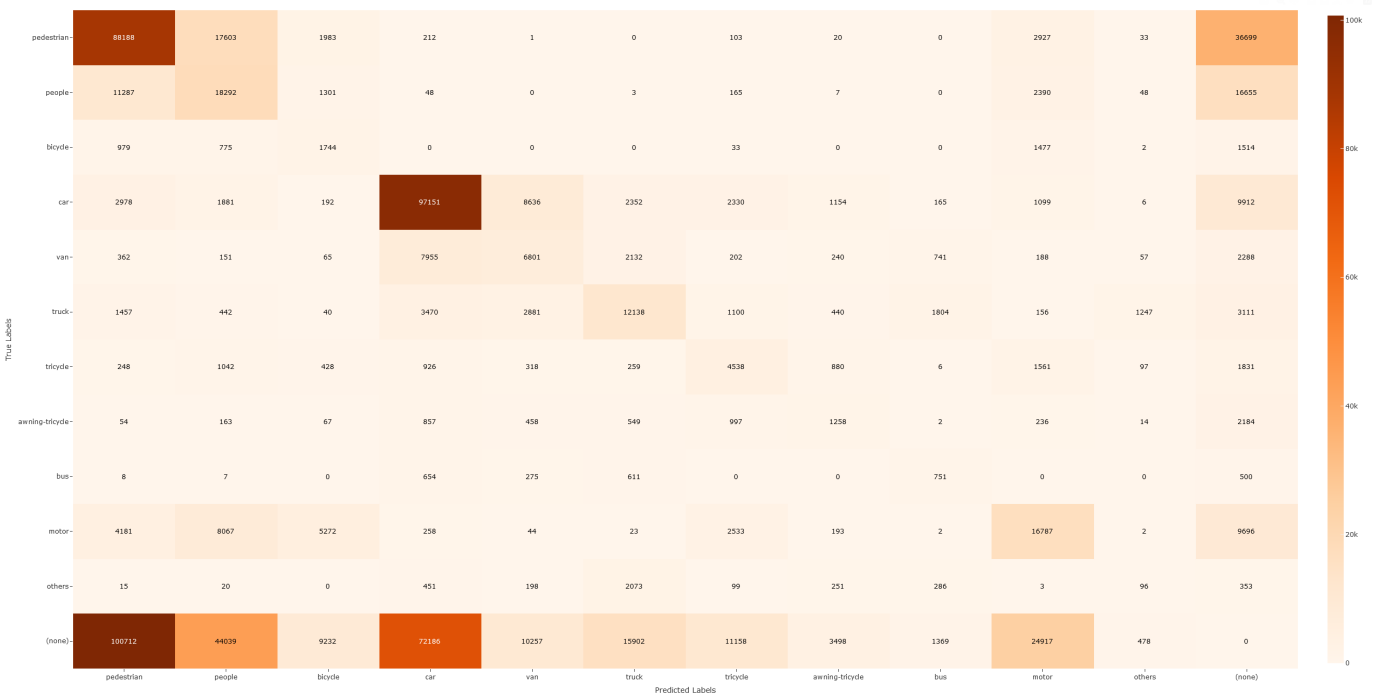

Fig. 23. Confusion matrix for H-Deformable DETR with a multi-frame Video Swin Transformer backbone using the Enc5x1 fusion mechanic.