Exploring Attention in Video Instance Segmentation

Aditya Mehrotra [1] **Rahul Allam** [1] **Gaurav Bhosale** [1] **Akash Thorat** [1]

Abstract

This work focuses on exploring attention mechanisms to improve performance of the baseline architecture, ObjProp, for Video Instance Segmentation. Two new modifications are introduced, an attention neck module for region proposal and weighing the inter-frame affinity for mask propagation. Moreover, the techniques of sampling reference frames for mask propagation are experimented on. Multiple trials were conducted with these variables and the resulting metrics were investigated. Although the approaches did not improve the performance of the state-of-the-art, they provide future directions which can lead to better attention-based architecture with refined performance. Code has been made available in the respective branches at: https://github.com/aditya9710/ObjectMaskPropWAttn.git

1. Introduction

This work tackles the problem for the Video Instance Segmentation. The problem is an extension of Image Instance Segmentation from the image domain to video domain. Instance segmentation is the task of detecting and masking each distinct object of interest appearing in an image. The new problem aims at simultaneous detection, segmentation and tracking of object instances in videos. The application of Instance Segmentation is finer image understanding than just segmentation which does not provide distinct object understanding. This is helpful for a spectrum types of downstream tasks.

Formally, in video instance segmentation (Yang et al., 2019), the predefined category label set is

$$C = \{\text{person, cat, dog, } \cdots\}$$

Given a sequence of image frames $I_1, ..., I_T \in \{0,1\}^{h \times w}$, the algorithm should produce n video instance hypotheses. For each hypothesis j, the following are predicted:

- a category label $cls^j \in C$
- a confidence score $conf^j \in [0, 1]$, and
- a sequence of predicted binary masks $M_1^j, \cdots, M_T^j \in \{0,1\}^{h \times w}$, where T is the number of frames.

Current approaches to this problem are solved by mask propagation and track-by-detect approaches. The current state-of-the-art is Object Propagation using attention (ObjProp) (Chakravarthy et al., 2021) which is built upon Yang et al. (2019) MaskTrack R-CNN framework. The ObjProp architecture focuses on improving temporal consistency. This work thus, builds furthermore on this method by trying to include spatial attention in the framework through an attention neck and weighted inter-frame affinity.

1.1. Research contributions

This paper contributes and builds on MaskTrack R-CNN and ObjProp in the following ways :

- Introducing an additional attention neck in ObjProp. This module is a spatio-temporal attention layer placed between the Feature Proposed Network (FPN) and the Region Proposal Network (RPN).
- This paper introduces weighted inter-frame affinity matrix between the current and reference frame which is learnable.
- The sampling of frames for generating attention maps is experimented on compared to the previous method of random sampling.

2. Related Work

Image instance segmentation was extended to the video domain in 2019. Yang et al. (2019) introduced a novel approach titled "MaskTrack R-CNN" which builds upon the previous state-of-the-art, Mask R-CNN. They propose a tracking head over the existing Mask R-CNN framework, which enables tracking of various instances across video

[1]Worcester Polytechnic Institute. Correspondence to: Aditya Mehrotra <amehrotra@wpi.edu>.

CS/DS 541 Deep Learning Final Project, Worcester Polytechnic Institute.

frames. The primary objective of this method was to compute the the probability of assigning an instance label to a candidate box based on pictorial cues such as appearance similarity, detection confidence, semantic consistency and spatial correlation, across the video frames. Though the network performed very well, it failed to classify and assign different instance masks to very similar and occluded object categories across frames, due to lack of sufficient temporal context. Thus also resulting in temporally inconsistent masks and missing detections.

This problem of temporal inconsistency was addressed by Chakravarthy et al. (2021), who proposed a propagation head on MaskTrack R-CNN pipeline. This approach leverages the temporal dimension by using inter-frame attention across the time domain. The missing detections, due to object deformation or occlusion were subjugated by this approach. Therefore making use of temporal and as well as spatial context has resulted in higher quality output masks for instances.

Literature survey reveals that there is a substantial scope for improvement in mask generation and tracking. To overcome the temporal instability bottleneck, which arises due to several reasons such as missing proposals from RPN, object misclassification, or aliasing from small visual displacements. The authors focus on the missing proposals from the RPN by leveraging attention mechanism and further explore these in the pipeline.

3. Proposed Methods

This work proposes exploration on four fronts in the architecture:

3.1. Powerful Baseline

Baseline Tuning 1 The intuition for this approach was that improving the backbone may lead to better instance segmentation of frames through better feature extraction. To this end, the feature extractor modified from ResNet-50 to ResNet-101. The difference between the two is that ResNet-101 has 101 deep layers instead of 50. Furthermore, the number of fully connected layers for the tracking head and bounding box head was increased from 2 to 4. The number of convolution layers in the masking head was increased from 4 to 6. The intention was to improve the performance of these heads individually and as a form of hyperparameter tuning.

Baseline Tuning 2 Further parameter tuning was proposed after analyzing the inference results of the baseline. In most failure cases, the object classification was incorrect, which may be a result of a weak bounding box classifier loss. The losses are weighed in the overall loss, hence the weights for the bounding box classifier loss was increased from 1 to 1.5 to put more emphasis on the classifier. Moreover, in some cases, multiple detections were identified for the a single occurrence of the object. To reduce overlapping proposals, the minimum IoU threshold was increased by a factor of 0.1.

3.2. Attention Neck

The attention neck framework is presented in Figure 1. Liu et al. (2019)'s work served as an inspiration which was implemented on MaskTrack R-CNN. This module is a Spatio-temporal attention layer placed between the FPN and the RPN. The authors hypothesize that introducing an attention map as an input to the RPN should improve the overall score. The shape of input has to be maintained in this module to avoid any other changes in the overall architecture.

The attention neck takes in two inputs (Figure 2), a_t and a_{t-1} which are feature maps of the current and the reference frames from the FPN, respectively. A feature map has the dimensions $C \times W \times H$ where C is the number of channels, W is the width of the image, and H is the height of the image. After performing convolution *Conv1*, the two feature maps are multiplied following a softmax to form a correlation map. The current frame is reshaped to $WH \times C$ to account for this. The resulting correlation map has the dimensions $WH \times WH$.

The attention map is calculated with the correlation map by performing another matrix multiplication with the result of a convolution on the current frame. The output from *Conv2* gives $C \times W \times H$, so its multiplication with the correlation map results in $C \times WH$. This is reshaped again to $C \times W \times H$, multiplied with a learnable parameter (γ) and added again to the current frame. Each convolution layer is followed by ReLU activation.

3.3. Weighted Inter-Frame Affinity

This is a novel approach as addition to the propagation head where multiple inter-frame affinity matrices (Figure 3) are weighed to produce a single output. The aim is to produce weights that can improve attention on particular classes if present in the frame to improve mask propagation.

The input to this module is the features of the current and reference frame. Each has the dimension $B \times F \times H \times W$ where B is the batch size and F is the number of features in the frame. The reference frame is reshaped to the dimension $B \times HW \times F$. The weights for inter-frame affinity are initialized (different for each batch) with the dimension $B \times F_t \times F_{t-\delta}$. However, the number of features in both frames is equal ($F_t = F_{t-\delta}$). A batch matrix multiplication is performed first between the reference frame and the weights, then the result and the current frame. The output

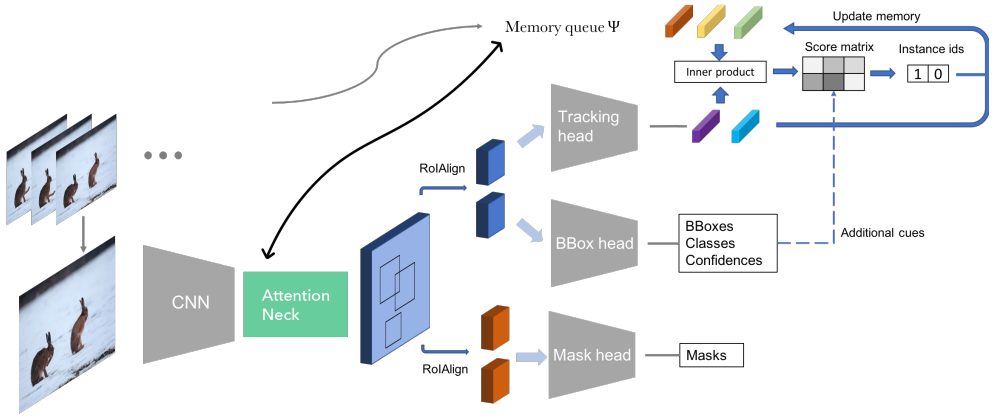

Figure 1. Attention-based Neck Module added between the FPN and RPN

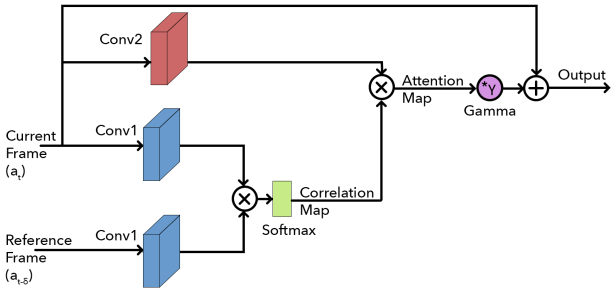

Figure 2. Neck Module Implementation

has the dimensions $B \times F_t \times F_{t-\delta}$. This does not change the rest of the architecture of the propagation head.

3.4. Reference Frame Sampling Technique

In the baseline architecture, a reference frame is randomly sampled from the video sequence. This method of sampling was compared to using the previous frame as reference. If the immediately preceding frame does not have annotations, the next best frame is chosen. This approach was proposed to test if there are any changes in performance by taking chronology into account. Henceforth, ordered pairs is referred to sampling the previous frame with respect to the current frame.

4. Experiment

4.1. Dataset

The YouTube-Video Instance Segmentation dataset (2019 version) (Yang et al., 2019) was used for evaluating the approaches. It is a large-scale dataset based on the YouTube-Video Object Segmentation dataset, specifically for video instance segmentation. Following are the statistics of the dataset:

- 3,859 high-resolution YouTube videos, 2,985 training videos, 421 validation videos and 453 test videos.
- An improved 40-category label set by merging eagle and owl into bird, ape into monkey, deleting hands, and adding flying disc, squirrel and whale
- 8,171 unique video instances
- 232k high-quality manual annotations

4.2. Evaluation Metrics

The test dataset does not have annotations. The test annotations are hidden by Yang et al. (2019) for unbiased testing on the dataset on CodaLab[1].

The predicted annotations can be uploaded which return a score (mean Average Precision, mAP). The portal also produces metrics comprising AP50, AP75, AR10 and AR1 that are calculated according to the evaluation criteria[2].

4.3. Training

The architecture changes were developed using the modular MMDetection library (Chen et al., 2019). The full model is trained end-to-end in 12 epochs. The initial learning rate is set using Stochastic Gradient Descent optimizer with a learning rate of 0.005. It is decayed by a factor of 10 at 8 and 11 epochs. The model is trained on 4 Tesla V100 GPUs. A batch size of 16, with 4 images on each GPU.

[1] https://competitions.codalab.org/competitions/28988#results

[2] https://competitions.codalab.org/competitions/28988#learn_the_details-evaluation

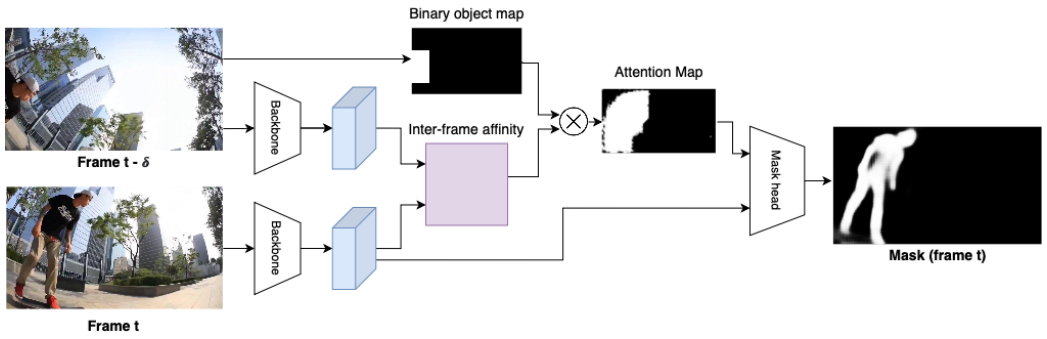

Figure 3. Inter-frame affinity in Chakravarthy et al. (2021)'s ObjProp

For a comparative analysis, ObjProp was set as a baseline. Although Chakravarthy et al. (2021) reported an mAP of 36.0, the authors were not able to reproduce this result. Two approaches were attempted: inference using end-to-end trained ObjProp model; inference using the pretrained model made available by Chakravarthy et al. (2021). Both of these approaches resulted in the mAP of 35.5 which was then considered the baseline's score.

5. Results

5.1. Training observations

The loss trend can be seen in Figure 4, where the following losses have been plotted:

- RPN Classification Loss: Provides information in regards to the model's ability to confidently predict the existence of an object.
- RPN B-Box Loss: Provides information in regards to the model's ability to confidently predict the dimensions of the region proposal.
- Match Loss: Provides information in regards to the model's ability to effectively track objects across frames.
- Classification Loss: Provides information in regards to the model's ability to classify an object as one of the 41 classes.
- B-Box Loss: Provides information in regards to the model's ability to predict the dimensions of an object
- Total Loss: Sum of all the above losses

5.2. Comparative Analysis

In this section, the associated empirical performance are discussed based on performance metrics and inferred for a comparative study (Table 1).

In **Baseline Tuning 1**, the performance degrades across all validation metrics. Perhaps, the increased backbone complexity ((∼ 27 million) additional parameters) caused the model to learn information specific to the training data and not generalize well enough for the validation set and consequently, the real world.

In **Baseline Tuning 2**, the performance degrades across all validation metrics. Perhaps, increasing the parameter values of MaxIoUAssigner results in missed detections. For example, if the region proposals just cover a part of the object but not the whole object, the computed IoU with the ground truth bounding box would be less even though the object is present around that area. Additionally, increasing the loss weight of the bounding box classifier, should have ideally penalized the model more for any misclassifications, which forces the model to further tweak the weights to get the associated loss under control. However, since there is a degrade in performance, the authors believe the threshold change to MaxIoUAssigner may have countered the impact of to the loss weighing.

In **Baseline with Ordered Pairs**, the performance degrades across all validation metrics. In general, the movement of an object from the previous frame to the current frame is not significant, unless the object is moving too fast. Based on the latter statement, if the previous frame is picked as a reference frame, the model fails to learn weights capable of generalizing to cases where there is a significant change from the previous frame to the current frame. Hence, this could be the reason why the model fails to improve. Instead, if the reference frame were random (baseline), there would be a significant difference between the reference and current frames, which forces the weights to adapt to be able to handle these situations.

In **Additional Training**, metrics AP75 and mAP slightly improve over the baseline model. Perhaps, the increased training fine-tuned the model parameters further, which as a consequence bumped the AP75 and mAP scores.

In **Attention Neck**, the performance degrades across all

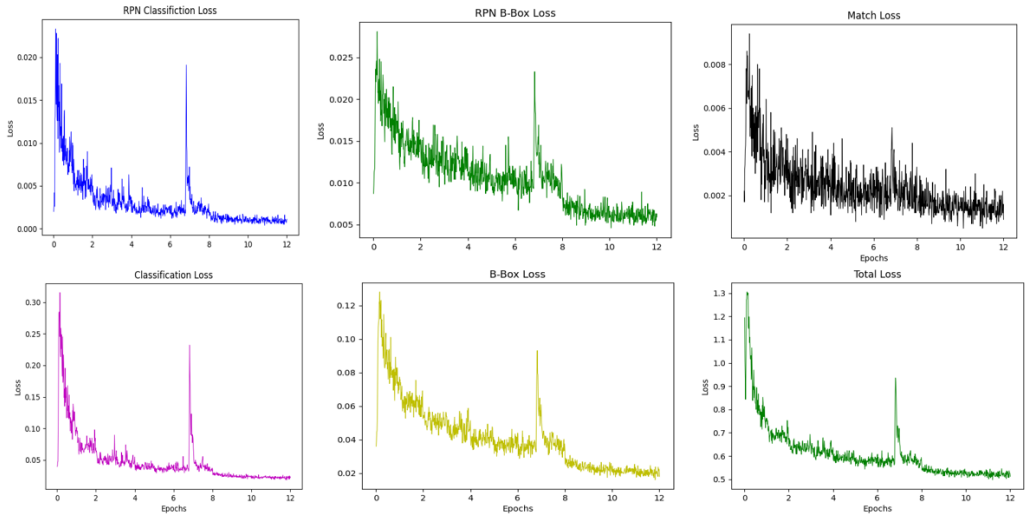

Figure 4. Training loss convergence across epochs for Attention Neck

Category	Trials	AP50	AP75	AR1	AR10	mAP
Baseline		**0.5734**	0.3907	**0.375**	**0.4519**	0.3551
Architecture Tweaking	Baseline Tuning 1	0.5056	0.3458	0.344	0.392	0.3139
	Baseline Tuning 2	0.5362	0.3666	0.3532	0.4191	0.3358
	Baseline with Ordered Pairs	0.5389	0.3754	0.3541	0.4189	0.335
	Additional Training	0.5722	**0.4123**	0.3748	0.4429	**0.3603**
Neck Modules	Attention Neck	0.5223	0.365	0.3624	0.428	0.3291
	Attention Neck with Ordered Pairs	0.5643	0.3675	0.3721	0.4318	0.3437
Weighted Inter-frame Affinity		0.5119	0.3312	0.3402	0.3998	0.3082

Table 1. Trial results

validation metrics. Both convolutions feature a single layer with 256 3×3 filters with a ReLU activation. Perhaps, the *Conv1* and *Conv2* layers are not deep enough to learn anything meaningful from the current frame and reference frame feature maps, which as a consequence is affecting the attention map generated by the Neck Attention module.

In **Attention Neck with Ordered Pairs**, the performance although degrades in comparison to baseline, it at least performs better than the **Attention Neck** trial. Perhaps, The attention neck estimates the global correlation map between the successive frames and transfers it to the attention map. Added with the attention information, the new features may enhance the response of the instance for predefined categories.

In **Weighted Inter-frame Affinity**, the proposed approach had to be tweaked during execution. The weight size was fixed in implementation for training. However, this was problematic in testing since the batch size is 1. Thus, the layers in the weighted inter-frame affinity approach were pruned during testing. This may have compromised the result. This, as a consequence, degraded the model by far the largest.

5.3. Visual Analysis

In this section, scenarios where the model performs well are discussed. At the same time, the scenarios where the model fails to either segment, track or detect objects in a frame sequence are also discussed.

In Figure 5, in the first sequence of images, there are two human instances present and the instance color consistency confirms is maintained throughout the sequence of frames. In the second sequence of images, the human instances and the motorcycle instances are tracked accurately. The instance color consistency confirms this. The same applies to the third sequence of images as well.

However, there are a multitude of scenarios where the model does not perform well. For instance, in Figure 6, in the first sequence of images, the model misclassifies a bear as an ape but also creates a new ape instance after the ape gets occluded by the tree in frame t. In the second sequence of images, even though the model detects two human in-

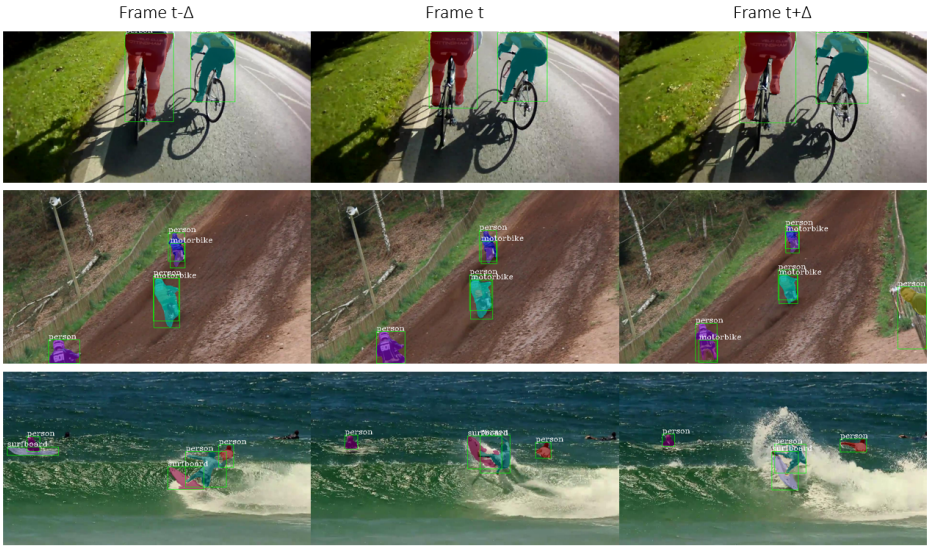

Figure 5. Decent performance of the model in complex segmentation scenarios

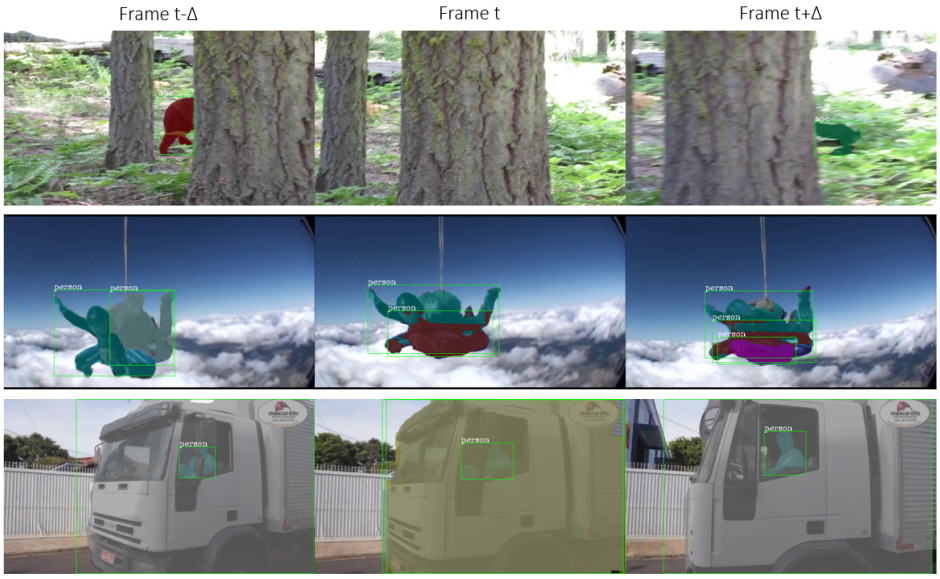

Figure 6. Model failure in complex segmentation scenarios

stances at frame $t - \Delta$, the associated masks are slightly misplaced. At frame t, the model incorrectly loses track of the previous instance and creates a new instance even though the instance itself is unchanged. That being said, it at least gets the mask for each instance right. At frame $t + \Delta$, the model predicts a new instance, which is not the case. There are two human instances but the model incorrectly detects three. Finally, in the third image sequence, the model accurately detects the object instances and their associated masks. But in frame t, the model incorrectly detects a new truck instance but recovers back at frame $t + \Delta$, by maintaining the instance color consistency for the truck instance between frame $t - \Delta$ and $t + \Delta$.

Additionally in Figure 7, in the first sequence of frames, the model incorrectly detects an inanimate non-object as an object in frame t and misclassifies it as a frog. In the second sequence of frames, the model incorrectly identifies a dog as a giant panda and a mouse at frames $t - \Delta$ and t. It corrects for this at frame $t + \Delta$. Finally, in the third sequence of images, the model correctly predicts the instances (lizard and hand) at frame $t - \Delta$. However, at frame t, the model fails completely. The model detects two more instances (cat and mouse), which are not present in the im-

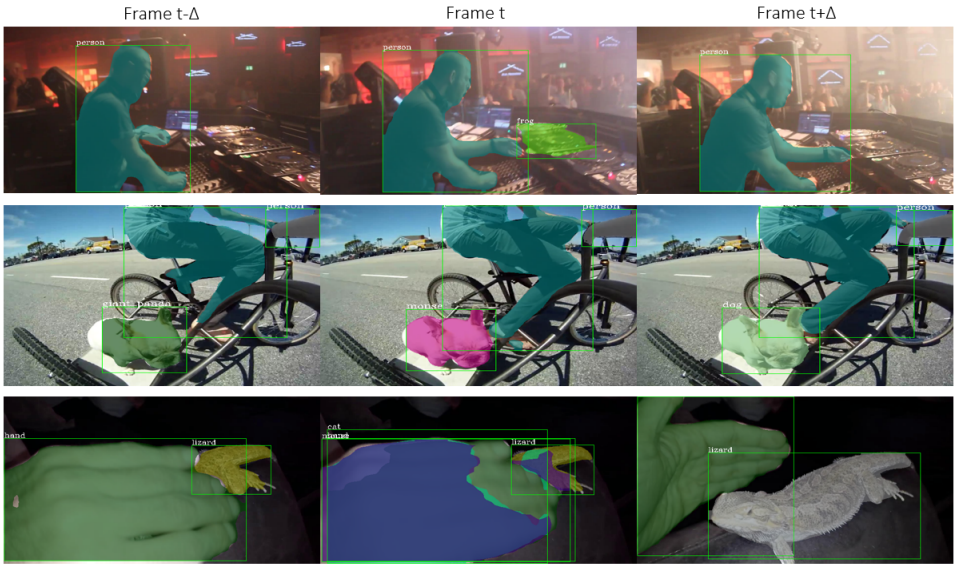

Figure 7. Model failure with more complex segmentation scenarios

age. On top of this, the instance masks for both cat and mouse are sparse in nature. In essence, the model thinks there is a cat and a mouse instance and the hand is occluding them, which resulted in the sparse instance masks, which is the wrong conclusion. Finally, at frame $t + \Delta$, the model still identifies the hand as the same instance but incorrectly identifies the lizard as a new instance.

6. Conclusions and Future Work

In this work, although the authors find that adding Attention Neck with Ordered Pairs module in the baseline network degrades its performance compared to baseline, it at least performs better than the Attention Neck trial. It is speculated that the attention neck estimates the global correlation map between the successive frames and transfers it to the attention map. This could be a research direction in future. Weighted Inter-Frame Affinity also decreases the overall performance of the model. Adding weights may reduces the learning capacity of the base network and on the run-time does not actually lead to any better performance. However, this implementation can be changed to having separate weight for each class rather than basing its dimension on batch. Random sampling of frames may perform better due to the fact that random sampling generalizes very well since some objects maybe occluded in preceding and subsequent frames. Another interesting observation noted is that additional training improves AP75 values leading to the increase in overall mAP.

This paper advances the field by introducing and implementing two different modules and its effect on the ObjProp architecture. A similar attention neck implemented on MaskTrack R-CNN improved the performance (Liu et al., 2019). Though, the goal of this work was to implement it on state-of-the-art in the domain which did not achieve a better performance. However, it should also be acknowledged the research could not be refined due to time constraints of the course, CS541. Yet, the results of the present work may prove useful to guide future work in this field of research.

References

Chakravarthy, Anirudh S, Jang, Won-Dong, Lin, Zudi, Wei, Donglai, Bai, Song, and Pfister, Hanspeter. Object propagation via inter-frame attentions for temporally stable video instance segmentation. *arXiv preprint arXiv:2111.07529*, 2021.

Chen, Kai, Wang, Jiaqi, Pang, Jiangmiao, Cao, Yuhang, Xiong, Yu, Li, Xiaoxiao, Sun, Shuyang, Feng, Wansen, Liu, Ziwei, Xu, Jiarui, et al. Mmdetection: Open mmlab detection toolbox and benchmark. *arXiv preprint arXiv:1906.07155*, 2019.

Liu, Xiaoyu, Ren, Haibing, and Ye, Tingmeng. Spatio-temporal attention network for video instance segmentation. In *Proceedings of the IEEE/CVF International Conference on Computer Vision Workshops*, pp. 0–0, 2019.

Yang, Linjie, Fan, Yuchen, and Xu, Ning. Video instance segmentation. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pp. 5188–5197, 2019.