Towards Real-Time Open-Vocabulary Video Instance Segmentation

Bin Yan[1]*, Martin Sundermeyer[2], David Joseph Tan[2], Huchuan Lu[1], Federico Tombari[2,3]
[1] Dalian University of Technology [2] Google [3] TU Munich

Abstract

In this paper, we address the challenge of performing open-vocabulary video instance segmentation (OV-VIS) in real-time. We analyze the computational bottlenecks of state-of-the-art foundation models that performs OV-VIS, and propose a new method, TROY-VIS, that significantly improves processing speed while maintaining high accuracy. We introduce three key techniques: (1) Decoupled Attention Feature Enhancer to speed up information interaction between different modalities and scales; (2) Flash Embedding Memory for obtaining fast text embeddings of object categories; and, (3) Kernel Interpolation for exploiting the temporal continuity in videos. Our experiments demonstrate that TROY-VIS achieves the best trade-off between accuracy and speed on two large-scale OV-VIS benchmarks, BURST and LV-VIS, running 20× faster than GLEE-Lite (25 FPS v.s. 1.25 FPS) with comparable or even better accuracy. These results demonstrate TROY-VIS's potential for real-time applications in dynamic environments such as mobile robotics and augmented reality. Code and model will be released at https://github.com/google-research/troyvis.

1. Introduction

In recent years, open-vocabulary video instance segmentation (OV-VIS) has gained increasing attention in mobile robotics, autonomous driving, and augmented reality where diverse objects need to be tracked in dynamic environments. Different from traditional video instance segmentation [30, 43, 46] which only focuses on a limited set of pre-defined object categories, OV-VIS [3, 38] requires the methods to identify, segment, and track objects of unlimited categories outside the supervised training set. This presents significant challenges for traditional models in computer vision. However, the rapid development of foundation models [18, 31, 32, 40] in recent years has brought new opportunities. Trained on vast amounts of data, these models exhibit unprecedented zero-shot understanding capabilities. For example, the recent object-level foundation model GLEE [40] has achieved state-of-the-art performance on several large-scale OV-VIS benchmarks, including LV-VIS [38] and BURST [3].

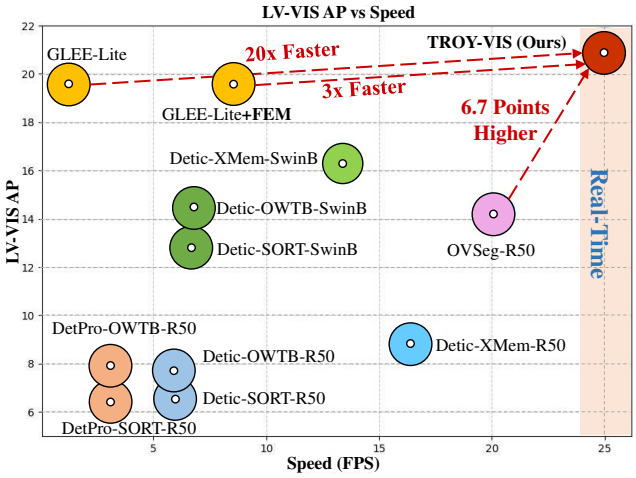

Figure 1. Performance and speed comparison on the LV-VIS [38] benchmark. TROY-VIS is the only method that runs in real-time. Compared with GLEE-Lite [40] TROY-VIS runs 20× faster while achieving better results. TROY-VIS surpasses OVSeg-R50 [38], the previously fastest model, by 6.7 AP.

Despite the progress made, existing approaches for OV-VIS still face significant limitations. The most notable issue is the computational inefficiency of state-of-the-art models like GLEE [40], which achieve impressive accuracy but are unable to process sensor streams in real-time. GLEE-Lite, the most lightweight variant of GLEE, processes video frames at only 1.25 frames per second (FPS) on an A100 GPU, far below the threshold (24 FPS) required for real-time applications as shown in Fig. 1. This poor efficiency restricts the deployment of these models in time-sensitive real-world scenarios, where both accuracy and speed are equally crucial. Consequently, to make OV-VIS methods applicable to real-world scenarios, the greatest challenge is reducing their substantial computational burden.

In this paper, we propose a novel architecture and inference strategy **T**owards **R**eal-time **O**pen-vocabular**Y** **V**ideo

*This work was performed while Bin Yan worked as an intern at Google.

Instance **S**egmentation (TROY-VIS). First, we analyze every component of GLEE, identifying the text encoder, feature enhancer, and instance decoder as the most important computational bottlenecks. Based on this analysis, we then introduce three core techniques to improve processing efficiency: (1) Decoupled Attention Feature Enhancer, which replaces heavy modality-scale hybrid attention with efficient modality attention and scale attention, reducing computational and memory costs; (2) Flash Embedding Memory, which allows for fast retrieval of text embeddings without redundant recalculations during training and inference; and, (3) Kernel Interpolation, which leverages the temporal continuity in videos to obtain proxy instance kernels by interpolating accurate kernels from key frames. These innovations allow TROY-VIS to achieve significant speed improvements up to 20× without sacrificing accuracy.

Our experimental results demonstrate that TROY-VIS outperforms other efficient methods in both speed and accuracy on large-scale OV-VIS benchmarks, BURST [3] and LV-VIS [38]. Specifically, TROY-VIS runs at 25 FPS (20× faster than GLEE-Lite), while achieving state-of-the-art performance on these datasets. To the best of our knowledge, we propose the first real-time open-vocabulary video instance segmentation model, paving the way for the application of OV-VIS to real-world scenarios.

2. Related Works

2.1. Open-Vocabulary Perception

Compared with traditional closed-set perception operating within a predefined and unchanging set of categories, open-vocabulary perception [3, 38, 49] requires models to have strong generalization ability so as to recognize categories unseen during training. Since this ability allows models to operate more effectively in complex, dynamic, and unpredictable environments, open-vocabulary perception has drawn more and more attention in recent years.

In this field, the primary focus has been on image-level open-vocabulary detection (OVD) [49]. Inspired by the great success of large-scale image-text pretraining [31], recent studies redefine open-vocabulary object detection as a region-text matching task and utilize large-scale image-text datasets to expand the vocabulary. GLIP [22] first unifies object detection and phrase grounding, introducing a grounded pre-training framework that shows superior capabilities in a zero-shot setting. OWL-ViT [26] extends vision transformers to simple, yet effective, open-vocabulary detectors by fine-tuning them with detection and grounding datasets. Later works like Grounding DINO [33] and YOLO-World [8] replace the original ATSS [54] architecture in GLIP [22] with the DINO [51] and the YOLO [13] framework respectively.

Compared with OVD, open-vocabulary video instance segmentation [3, 38] (OV-VIS) raises additional challenges because it requires models to detect, segment, and track object simultaneously in a zero-shot setting. BURST [3] and LV-VIS [38] are two challenging OV-VIS benchmarks, which cover 482 and 1196 object categories, respectively. Mainstream methods [3, 38] combine existing open-vocabulary detectors [58] with off-the-shelf association methods [4, 7] to accomplish OV-VIS. However, the separation of detection and association models may lead to suboptimal performance. Recently, GLEE [40] achieves state-of-the-art performance on both OV-VIS benchmarks by training a powerful Transformer-based [21] detector and associating objects based on object queries.

2.2. Generalist Models for Vision Perception

We recently witnessed a paradigm shift from specialist to generalist models [2, 36, 40, 44, 45, 52, 60], which can solve multiple tasks using a unified model and condense knowledge from various domains into one suite of parameters. In object tracking, Unicorn [44] and TarVIS [2] are the pioneers in this field. On one hand, Unicorn [44] unifies the temporal correspondence required by different tracking tasks and detects objects of various properties with the help of target priors. On the other, TarVIS [2] unifies multiple pixel-level tracking tasks into a query-based temporal segmentation architecture and uses different target queries to accomplish different tracking tasks.

Later, several multi-modal perception generalist models [36, 40, 45, 52, 60] were developed by incorporating advanced text encoders [11, 31]. With the powerful vision-language understanding abilities, they solved more complicated perception tasks. To be more specific, X-Decoder [60] is a unified framework for image-level referring and open-vocabulary segmentation. OpenSeeD [52] proposes a simple unified open-vocabulary detection and segmentation framework. APE [36] is a universal image perception model, which can perform detection, segmentation, and grounding with a single model. Although achieving success in image-level perception tasks, the effectiveness of these methods on videos is unexplored. In contrast, UNINEXT [45] reformulates 10 instance perception tasks on both images and videos into a prompt-guided object discovery and retrieval paradigm. Building on UNINEXT, GLEE [40] further scales-up the training data and enhances its open-vocabulary ability, which has shown superior performance on open-vocabulary video instance segmentation [3, 38].

2.3. Efficient Vision Models

In recent years, as model sizes have consistently grown [17], there has been a qualitative leap in model performance [1, 12, 28], with the emergence of previously unforeseen capabilities [18, 31]. However, the surge in

the number of model parameters and computational requirements has also hindered the practical application of these models, particularly in deploying them on resource-constrained edge platforms. To address this challenge, many recent works have attempted to train efficient vision models that significantly reduce computational costs while maintaining model performance as much as possible.

Based on powerful vision-language or pure vision foundation models like CLIP [31] and SAM [18], there are a series of works attempting to build their efficient variants. For example, TinyCLIP [41] proposes two techniques, affinity mimicking and weight inheritance, largely reducing the model size. MobileCLIP [37] further designs more lightweight image and text encoder, optimizing them with multi-modal reinforced training. Besides, most efficient variants of SAM [18] try transferring knowledge from the original heavy ViT-H backbone to more lightweight ones. Specifically, MobileSAM [50] and EfficientViT-SAM [55] adopt ViT-Tiny and EfficientViT [5] as the backbone respectively, training them via knowledge distillation using MSE Loss. In contrast, EfficientSAM [42] trains the lightweight encoder with more complex masked image pretraining [15]. Although achieving great success, these techniques are specifically designed for architectures like CLIP [31] and SAM [18], being difficult to apply on other frameworks like DETR [6]-style models.

In the field of object-level perception, DETR-style methods [40, 45, 51, 57] are currently the most popular solutions. There are also some works trying to design more lightweight models of this style. For example, [33, 53, 56] design more lightweight feature enhancers and instance decoders to speed-up models. Moreover, MobileInst [53] proposes a technique called kernel reuse to save computations of the instance decoder. Our kernel interpolation uses similar principles but has two key differences. First, while MobileInst only applies kernel reuse on closed-vocabulary VIS benchmarks [43, 46] with tens of object categories, we introduce kernel interpolation that can be scaled to the more challenging open-vocabulary VIS task [3, 38]. Furthermore, the kernel reuse in MobileInst requires temporal training, restricting the training data to annotated videos. In contrast, we find that solely applying kernel interpolation during the inference can already achieve good results, allowing us to train on larger scale image datasets. To optimize the text encoder, we introduce Flash Embedding Memory, which adopts a similar motivation as the language cache in the object detection method OmniDet-Turbo [56]. This technique enables us to use larger text encoders without causing extra computational costs.

3. Methodology

Existing OV-VIS methods [38, 40] have achieved state-of-the-art accuracy. However, their inference time is far from real-time, consequently limiting their real world application. Taking the recent object-level foundation model GLEE [40] as an example, it took GLEE-Lite 805ms to process a single frame on the LV-VIS [38] dataset which is equivalent to 1.25FPS only (see Tab. 1). Note that this is the fastest version of GLEE running on a powerful A100 GPU. To overcome this hurdle and improve the speed, we analyzed component-wise latency in Tab. 1 and discovered that the text encoder, feature enhancer, and instance decoder are the most time-consuming bottlenecks.

Text Encoder. The initial step in OV-VIS includes having the desired object categories pass through the text encoder to obtain the text embeddings. As the number of categories increases, the latency of the text encoder grows drastically. For instance, to process 1196 categories of LV-VIS [38], it takes the text encoder more than 800ms as shown in Tab. 1. What's worse is that the original implementation of GLEE repeats this computation on each frame, which causes serious computational redundancy.

Feature Enhancer. To correlate the visual and textual features, an early fusion module, which is a cross-attention between two modalities, is introduced. Specifically, the input visual features are hierarchical representations with strides of $\{8, 16, 32, 64\}$. Furthermore, when processing a video frame of 480p, the total number of visual tokens reaches 8K. This amount of visual tokens leads to heavy computational costs and high memory demands during training.

Instance Decoder. The instance decoder of the previous works follows the design of Mask DINO [21] but replaces the fixed-size classifier with vision-language alignment [47]. As for mask prediction, first, N object queries Q are passed through the Transformer decoder, obtaining N instance kernels K. Then, the N instance masks m are predicted by convolving N instance kernels K with a 1/4 downsampled pixel embedding map M, written as

$$m = K \star M;\ K = \text{Dec}(Q)\ . \tag{1}$$

The instance decoder consists of 9 decoder layers and processes $N = 300$ queries. This setting makes the instance decoder computationally expensive to run on each frame.

TROY-VIS. To address the aforementioned bottlenecks, we introduce three key techniques: (1) **Decoupled Attention Feature Enhancer** to speed up information interaction between different modalities and scales; (2) **Flash Embedding Memory** for fast retrieval of the text embeddings of object categories; and, (3) **Kernel Interpolation** for exploiting the temporal continuity in videos and making predictions efficiently.

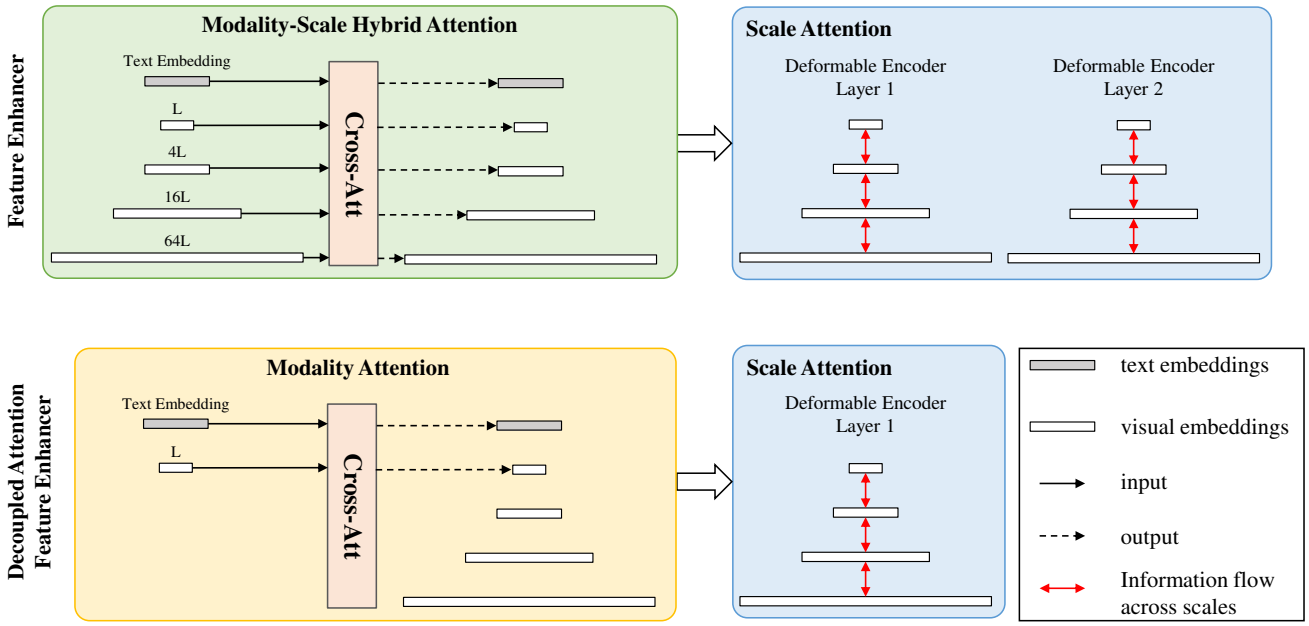

Figure 2. Architecture comparison between the original feature enhancer and our decoupled attention feature enhancer. In our design, a fast modality attention and an efficient scale attention are used to replace the heavy modality-scale hybrid attention.

3.1. Decoupled Attention Feature Enhancer

To alleviate the large memory and computational costs in the feature enhancer, we propose a lightweight decoupled attention feature enhancer. Our method is inspired by depth-wise separable convolution [9], which decomposes a dense 3x3 convolution into a depth-wise 3x3 convolution in spatial dimension and a point-wise 1x1 convolution in channel dimension. Similarly, we decouple the original modality-scale hybrid attention into a modality attention and a scale attention as shown in Fig. 2. In this section, we introduce this design in more detail and compare it with the original feature enhancer in terms of time and space complexity.

Before computing the specific complexity of the two architectures, we first derive the general formula for the time and space complexity of the cross-attention operation. Suppose there are two input sequences with length of L_1 and L_2, both with a feature dimension of d. Cross-attention consists of three steps: (1) projection of query, key, and value; (2) attention matrix computation between query and key; and, (3) weighted sum on value using the attention matrix. The time complexity Com_T and space complexity Com_S are denoted as

$$Com_T = 2 \times L_1 \times L_2 \times d + (L_1 + L_2) \times d^2, \quad (2)$$

$$Com_S = L_1 \times L_2 + (L_1 + L_2) \times d\,. \quad (3)$$

Usually, $L_1 >> d$ and $L_2 >> d$ especially when dealing with high-resolution images and a large vocabulary set. Thus, both Com_T and Com_S are nearly proportional to $L_1 \times L_2$.

In our case, the length of the text sequence is L_t and there are L_v tokens in the visual features at the smallest scale (stride of 64). Then the total number of visual tokens from all scales is $(1 + 4 + 16 + 64)L_v = 85L_v$. Thus, the time and space complexity of modality-scale hybrid attention are about $170L_t \times L_v \times d$ and $85L_t \times L_v$, respectively. In contrast, in our proposed modality attention, only the visual features with the lowest resolution are used to compute the cross-attention with the text tokens. Thus, its time and space complexities are $2L_t \times L_v \times d$ and $L_t \times L_v$, respectively, being $85\times$ more efficient than modality-scale hybrid attention.

However, the modality attention itself is not equivalent to the original hybrid attention since it lacks scale-level information interaction. To compensate for this drawback, we combine our modality attention with a scale attention module. Specifically, we find that the pixel decoder (*i.e.* deformable encoder [59]) can naturally serve as the scale attention module because it enhances the visual features by modelling the relationship among points from different scales. Therefore, we reuse the pixel encoder as the scale attention module instead of introducing additional networks. To make the scale attention more efficient, we reduce the number of deformable encoder layers from 6 to 3.

3.2. Flash Embedding Memory

When dealing with a large number of vocabularies, the processing time of the language encoder can become the

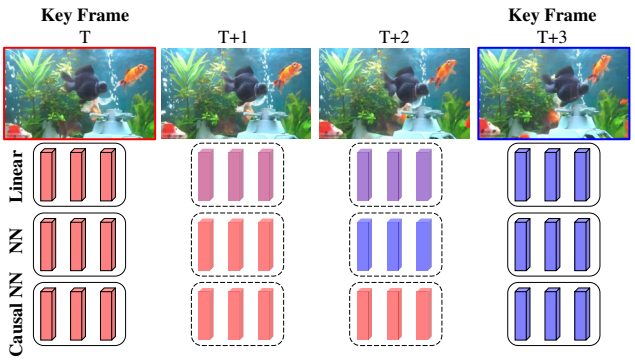

Figure 3. Illustration of kernel interpolation. Cuboids represent instance kernels and kernels from the same frame are in the same color. Besides, accurate kernels on key frames and proxy kernels from non-key frames are circled in solid and dashed lines respectively. We explore three types of interpolation methods: linear, nearest neighbor (NN), and causal NN interpolation.

bottleneck of the entire network. To address this, we propose Flash Embedding Memory. Instead of repeatedly running the text encoder to get the text embeddings of object categories in the current batch on-the-fly, we store the embeddings of all seen categories in a memory, with category name as the key and text embedding as the value. In this way, each category is only processed once without repeated computations. For seen categories, the embeddings can be obtained in a flash by retrieving the keys and picking out the corresponding values in $O(1)$ time complexity.

Notably, Flash Embedding Memory can be used to speed up both inference and training (if the text encoder is frozen during training). Before training, we can first summarize all object categories from diverse training data and store them into a set without duplication. Then, we send them into the text encoder in parallel, saving the category-embedding pairs in the Flash Embedding Memory. With this powerful memory, during training, there is no need to run even to keep the text encoder because all the required text embeddings can be retrieved in a flash.

During inference, the following two scenarios are considered: evaluating on benchmarks (*i.e.*, categories are fixed); and, testing in the wild (*i.e.*, categories are not fixed and can be infinite). In the former, we can construct Flash Embedding Memory of all categories in advance, then directly use the embeddings during the inference. Furthermore, in the latter case, since the possible categories are infinite, it is not possible to cover all of them in the memory beforehand. However, Flash Embedding Memory strategy can still work with some small modifications. When coming across unseen categories, we can choose K nearest neighbours of the new categories from the existing ones in the memory based on their semantic similarities. The embeddings of new category can then be obtained by averaging the embeddings of its K nearest neighbours. Finally, the new key-value pair is saved to the memory.

Moreover, Flash Embedding Memory allows us to use embeddings from more powerful text encoders during inference without incurring additional computational costs. In this work, we replace the original CLIP-B with more advanced EVA-02-CLIP-L [12].

3.3. Kernel Interpolation

Videos are temporally consistent, which means that adjacent frames usually have strong correlations. This motivates us to apply a key frame propagation strategy. Specifically, we only run the whole network on sparse key frames and retrieve the results on other frames by propagating predictions of the key frames. In the context of OV-VIS, we propose a technique called kernel interpolation for fast inference. Suppose we uniformly sample one key frame every F frames ($F = 3$ in this work). On these key frames, we run the instance decoder as described in Sec. 3 to extract accurate instance kernels for mask prediction. On other non-key frames, instead of running the heavy instance decoder, we interpolate kernels from adjacent key frames to form proxy kernels. Finally, on every frame, the proxy or accurate kernels are convolved with the current pixel embedding map, obtaining the instance masks of the current frame.

For interpolation, we try three different approaches, namely linear, nearest neighbor (NN) and causal nearest neighbor as shown in Fig. 3. In the linear interpolation, the instance kernels on a non-key frame are the weighted sum of kernels from the adjacent two key frames. Taking frame $T + 1$ as an example, the proxy kernels can be computed as $\hat{K}(T + 1) = \frac{2}{3}K(T) + \frac{1}{3}K(T + 3)$. A simpler way of interpolation is nearest neighbor, which directly copies the kernels from the nearest key frame to the proxy kernels.

Although these two approaches exploit bi-directional information, it is impossible to access information from the future in real-time applications like autonomous driving. To solve this causality issue, we propose a new interpolation method called causal nearest neighbor. Given a non-key frame, we only consider its causal neighbours, *i.e.* previous key frames. In this way, this approach can not only speed-up the inference but can also be used in online applications. Please note that this technique is training-free, *i.e.* it can still be trained on image-level annotations and does not cause extra complexity.

To account for the large number of kernels in OV-VIS, we further increase the efficiency of our method by reducing the number of decoder layers from 9 to 3.

3.4. Training Strategy

Overall, the training process consists of two stages: **pre-training** on large-scale image datasets; and, **joint tuning** on many diverse image and video datasets. In the first stage,

Figure 4. Qualitative results of TROY-VIS on challenging indoor and outdoor scenarios. Best viewed in color with zoom-in.

our model is pre-trained on two large-scale object detection datasets, Objects365 [35] and OpenImages [20], which contain 1.8M and 1.7M images respectively. The goal of this stage is to learn the general concept of objects, enabling our model to detect and recognize common instances in the environment.

Our model in the second stage is fine-tuned on many diverse image and video datasets jointly. Specifically, we additionally introduce image instance segmentation datasets, *e.g.* COCO [23], LVIS [14], VisualGenome [19] with rich object-level description and object noun phrases; and, video instance segmentation datasets, *e.g.* YouTube-VIS 2019 [46], YouTube-VIS 2021 [43], and OVIS [30]. Consequently, this step gives our model the ability to classify, segment, and track diverse objects in an open environment.

There are some notable differences between our training strategy and that of GLEE [40]. In stage 2, we don't use the datasets without category name [18, 39], and the data for referring expression comprehension&segmentation [27, 34, 48]. Moreover, our model is trained for 300K iterations rather than 500K iterations in [40] in both stages. Unlike GLEE [40], there is no stage 3 of scaling up data [18, 29], making our method more data-efficient.

4. Experiments

4.1. Implementation Details

We choose EfficientViT-L2 [5] as the vision backbone due to its good balance between accuracy and efficiency. The model is trained on 64 A100 GPUs (40GB) with global batch size of 128 using AdamW [24] optimizer. The base learning rate is set as $1e^{-4}$. To stabilize training in the beginning, the initial learning is set as $1e^{-8}$ and is then warm-up for 1,000 iterations. Similar to GLEE, the text encoder is frozen in the first stage and fine-tuned in the second stage. During joint tuning, the learning rate of the text encoder is $1/10$ of the base learning rate and an auxiliary distillation loss is introduced to prevent it from drifting too far from the original text embedding space.

4.2. Qualitative Results

In this section, we demonstrate some qualitative results of TROY-VIS on some challenging scenarios. As shown in Fig. 4, TROY-VIS can accurately recognize, segment, and track almost all objects in the test scenarios. Even for some uncommon categories like *handle*, *watch*, *pole* and *mitten*, TROY-VIS is able to detect and track the instance masks. These results illustrate that TROY-VIS can serve as a good open-vocabulary video instance perceiver, which can help robots to better understand dynamic scenes, make decisions and take actions in complex environments.

4.3. Numerical Evaluation

To evaluate the open-vocabulary perception ability (including classification, segmentation, and tracking) and the running efficiency of our method, we test it on two large-vocabulary video instance segmentation benchmarks, BURST [3] and LV-VIS [38]. Following previous works [40], we evaluate TROY-VIS and compare it with other methods in a zero-shot manner, *i.e.* without learning on the training split of these two datasets.

BURST [3]. BURST builds upon the previous large-scale multiple object tracking benchmark TAO [10], extending its original box-level annotations to pixel-precise mask annotations. BURST contains 2,914 videos, including 500, 993, and 1,421 videos in training, validation, and test set, respectively. There are 482 object categories totally, consisting of 425 common categories and 57 uncommon categories. The core ranking metric of BURST [3] is Higher Order Tracking Accuracy (HOTA) [25], which is the geometric mean of Detection Accuracy (DetA) and the Association Accuracy (AssA). Another metric is mAP, which is the average track-level mask IoU among different categories.

Based on the measured HOTA, TROY-VIS achieves the best results across all, common, and uncommon object categories as shown in Tab. 2. Especially on common and all categories, TROY-VIS outperforms the previous best method GLEE-Lite by HOTA of 5.9% and 1.3%, respectively. Besides, in terms of the segmentation metric mAP, TROY-VIS also achieves the best or the second best performance. These results demonstrate TROY-VIS's superior video perception ability in challenging scenarios.

LV-VIS [38]. LV-VIS is a more recent benchmark, which includes more videos and covers more diverse object categories. Specifically, the whole dataset includes 4828 videos and there are 3083, 837, and 908 videos in the training, validation, and test set respectively. Besides, LV-VIS covers 1196 object categories, consisting of 641 base categories and 555 novel categories. The final ranking metric is the mean Average Precision (mAP) on all categories, which can also be divided into $\mathrm{AP_b}$ on base categories and $\mathrm{AP_n}$ on novel categories.

TROY-VIS sets the new state-of-the-art performance among efficient OV-VIS methods, surpassing previous best method GLEE-Lite by 1.3%, 1.3% and 1.4% on AP of all, base and novel object categories, respectively.

	GLEE-Lite LV-VIS (ms)	**GLEE-Lite+FEM** LV-VIS (ms)	**TROY-VIS** LV-VIS (ms)
Total	**805**	**125**	**40**
Text Encoder	680	< 1	< 1
Feature Enhancer	75	75	20
Instance Decoder	41	41	4
Vision Encoder	9	9	16

Table 1. Efficiency comparison between the components of GLEE and TROY-VIS. GLEE-Lite+FEM means applying Flash Embed Memory (FEM) to GLEE-Lite. The latency is measured on a A100 GPU. The resolution of the input frame is 480p.

Efficiency. To demonstrate the efficiency of our method, we compare the overall and component-wise latency of TROY-VIS and GLEE-Lite on LV-VIS [38] dataset. First, the short side of the input frame is resized to 480 pixels then the resized frame is sent to the network for forward pass. The latency is measured using image batchsize 1 on a A100 GPU. As shown in Tab. 1, it takes GLEE-Lite more than 800ms to process one frame and the corresponding text embeddings. However, TROY-VIS only needs 40ms, being $\mathbf{20}\times$ faster than GLEE-Lite. We also compare to a variant where we add the Flash Embedding Memory to GLEE-lite, shown as **GLEE-Lite+FEM** in Fig. 1. Compared with GLEE-Lite+FEM, TROY-VIS still improves the inference speed by $3\times$ thanks to the other introduced methods.

Each component, the Flash Embedding Memory, decoupled attention, instance decoder and kernel interpolation cumulatively reduce the inference time by a large margin. Notably, TROY-VIS spends slightly more compute on the vision encoder where the original ResNet-50 [16] backbone is replaced by a stronger EfficientViT-L2 [5] backbone. Our ablations in the next section show that although EfficientViT-L2 is slightly slower, the corresponding performance gain is justifying the compute.

4.4. Ablation Study

This section demonstrates the effectiveness of each change to the baseline using the AP on the LV-VIS benchmark as our metric. Since the complete training process described in Sec. 3.4 consumes lots of time and resources, we adopt a more lightweight training setting in the whole ablations. The models in this setting are only trained for 100K iterations on 16 GPUs in both stages rather than 300K iterations on 64 GPUs.

Method	*BURST [3]*						*LV-VIS [38]*			
	ALL		Common		Uncommon		AP	AP_b	AP_n	FPS
	HOTA	mAP	HOTA	mAP	HOTA	mAP				
STCN Tracker [3]	5.5	0.9	17.5	0.7	2.5	0.6	-	-	-	-
Box Tracker [3]	8.2	1.4	27.0	3.0	3.6	0.9	-	-	-	-
Detic [58]-SORT [4]	-	-	-	-	-	-	12.8	21.1	6.6	6.7
Detic [58]-XMem [7]	-	-	-	-	-	-	16.3	24.1	10.6	13.4
OV2Seg [38]	-	3.7	-	-	-	-	14.2	17.2	11.9	20.1
GLEE-Lite [40]	22.6	**12.6**	36.4	18.9	19.1	**11.0**	19.6	22.1	17.7	1.3
TROY-VIS	**23.9**	12.4	**42.3**	**19.6**	**19.3**	10.7	**20.9**	**23.4**	**19.1**	**20.9**

Table 2. Comparison of TROY-VIS to recent specialist and generalist models on BURST [3] and LV-VIS [38] in a zero-shot manner. Evaluation metrics of BURST are reported separately for 'common', 'uncommon' and 'all' classes. The mAP computes mask IoU at the track level, HOTA is a balance of per-frame detection accuracy (DetA) and temporal association accuracy (AssA). The AP, AP_b, and AP_n in LV-VIS mean the average precision of overall categories, base categories, and novel categories.

	Decoupled Attention	Flash Embed Memory+EVA	Enc3+Dec3	Kernel Interpolation	EfficientViT-L2	Latency (ms)	AP
#1	-	-	-	-	-	805	13.2
#2	✓	-	-	-	-	760	13.0
#3	✓	✓	-	-	-	80	16.3
#4	✓	✓	✓	-	-	45	15.1
#5	✓	✓	✓	✓	-	**32**	14.7
#6	✓	✓	✓	✓	✓	40	**15.7**

Table 3. Ablations for the introduced changes. ✓means that the corresponding technique is used while dash means not used. The latency and AP performance is tested on LV-VIS benchmark.

As shown in Tab. 3, we start with the baseline labeled as #1, which uses the same architecture as GLEE-Lite but trained with our lightweight training setting. The following #2 to #6 incrementally adds new components to highlight their effects on the overall performance. In #2, we replace the original modality-scale hybrid attention with the more efficient decoupled attention, reducing the latency by 45ms without an obvious performance drop. Then, we introduce Flash Embedding Memory and replace the text encoder with stronger EVA-02-CLIP-L [12] in #3. This change significantly speeds up our model from 1.3FPS to 12.5FPS and brings an AP improvement of 3.3%. To further speed up our model, we reduce the number of encoder and decoder layers from {6,9} to {3,3}. As shown in #4, although causing a performance drop of 1.2% AP, this change reduces the latency from 80ms to 45ms (12.5FPS vs 22.2FPS). Furthermore, in #5, kernel interpolation speeds up our method from 22.2FPS to 31.3FPS, while only causing an AP drop of 0.4%. Finally, in #6, by replacing the original vision encoder with stronger EfficientViT-L2 backbone, our final method not only achieves better AP performance but also runs 20× faster than the baseline.

5. Conclusion

This work presents TROY-VIS, a real-time algorithm for open-vocabulary video instance segmentation, solving the problem of heavy computational costs of current methods. By carefully redesigning the key components of open-vocabulary perception models such as GLEE [40], we significantly reduce the processing time per frame while improving or maintaining accuracy across challenging benchmarks. The proposed decoupled attention feature enhancer, Flash Embedding Memory and kernel interpolation techniques are particularly effective in reducing redundant computations and leveraging temporal consistency in video data. Our experiments demonstrate that TROY-VIS outperforms existing methods not only in terms of speed but also in accuracy of perceiving objects of diverse categories. These advancements pave the way for broader applications of OV-VIS in fields requiring quick, accurate visual understanding in dynamic, complex scenarios.

References

[1] Josh Achiam, Steven Adler, Sandhini Agarwal, Lama Ahmad, Ilge Akkaya, Florencia Leoni Aleman, Diogo Almeida, Janko Altenschmidt, Sam Altman, Shyamal Anadkat, et al. Gpt-4 technical report. *arXiv preprint arXiv:2303.08774*, 2023. 2

[2] Ali Athar, Alexander Hermans, Jonathon Luiten, Deva Ramanan, and Bastian Leibe. Tarvis: A unified approach for target-based video segmentation. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 18738–18748, 2023. 2

[3] Ali Athar, Jonathon Luiten, Paul Voigtlaender, Tarasha Khurana, Achal Dave, Bastian Leibe, and Deva Ramanan. Burst: A benchmark for unifying object recognition, segmentation and tracking in video. In *Proceedings of the IEEE/CVF Winter Conference on Applications of Computer Vision*, pages 1674–1683, 2023. 1, 2, 3, 7, 8

[4] Alex Bewley, Zongyuan Ge, Lionel Ott, Fabio Ramos, and Ben Upcroft. Simple online and realtime tracking. In *2016 IEEE international conference on image processing (ICIP)*, pages 3464–3468. IEEE, 2016. 2, 8

[5] Han Cai, Junyan Li, Muyan Hu, Chuang Gan, and Song Han. Efficientvit: Lightweight multi-scale attention for high-resolution dense prediction. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pages 17302–17313, 2023. 3, 6, 7

[6] Nicolas Carion, Francisco Massa, Gabriel Synnaeve, Nicolas Usunier, Alexander Kirillov, and Sergey Zagoruyko. End-to-end object detection with transformers. In *ECCV*, 2020. 3

[7] Ho Kei Cheng and Alexander G Schwing. Xmem: Long-term video object segmentation with an atkinson-shiffrin memory model. In *European Conference on Computer Vision*, pages 640–658. Springer, 2022. 2, 8

[8] Tianheng Cheng, Lin Song, Yixiao Ge, Wenyu Liu, Xinggang Wang, and Ying Shan. Yolo-world: Real-time open-vocabulary object detection. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 16901–16911, 2024. 2

[9] François Chollet. Xception: Deep learning with depthwise separable convolutions. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 1251–1258, 2017. 4

[10] Achal Dave, Tarasha Khurana, Pavel Tokmakov, Cordelia Schmid, and Deva Ramanan. Tao: A large-scale benchmark for tracking any object. In *ECCV*, 2020. 7

[11] Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. Bert: Pre-training of deep bidirectional transformers for language understanding. *arXiv preprint arXiv:1810.04805*, 2018. 2

[12] Yuxin Fang, Quan Sun, Xinggang Wang, Tiejun Huang, Xinlong Wang, and Yue Cao. Eva-02: A visual representation for neon genesis. *arXiv preprint arXiv:2303.11331*, 2023. 2, 5, 8

[13] Ayush Chaurasia Glenn Jocher and Jing Qiu. Ultralytics yolov8. https://github.com/ultralytics/ultralytics, 2023. 2

[14] Agrim Gupta, Piotr Dollar, and Ross Girshick. Lvis: A dataset for large vocabulary instance segmentation. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, pages 5356–5364, 2019. 6

[15] Kaiming He, Xinlei Chen, Saining Xie, Yanghao Li, Piotr Dollár, and Ross Girshick. Masked autoencoders are scalable vision learners. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, pages 16000–16009, 2022. 3

[16] Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. Deep residual learning for image recognition. In *CVPR*, 2016. 7

[17] Jared Kaplan, Sam McCandlish, Tom Henighan, Tom B Brown, Benjamin Chess, Rewon Child, Scott Gray, Alec Radford, Jeffrey Wu, and Dario Amodei. Scaling laws for neural language models. *arXiv preprint arXiv:2001.08361*, 2020. 2

[18] Alexander Kirillov, Eric Mintun, Nikhila Ravi, Hanzi Mao, Chloe Rolland, Laura Gustafson, Tete Xiao, Spencer Whitehead, Alexander C Berg, Wan-Yen Lo, et al. Segment anything. In *ICCV*, 2023. 1, 2, 3, 6

[19] Ranjay Krishna, Yuke Zhu, Oliver Groth, Justin Johnson, Kenji Hata, Joshua Kravitz, Stephanie Chen, Yannis Kalantidis, Li-Jia Li, David A Shamma, et al. Visual genome: Connecting language and vision using crowdsourced dense image annotations. *International journal of computer vision*, 123:32–73, 2017. 6

[20] Alina Kuznetsova, Hassan Rom, Neil Alldrin, Jasper Uijlings, Ivan Krasin, Jordi Pont-Tuset, Shahab Kamali, Stefan Popov, Matteo Malloci, Alexander Kolesnikov, Tom Duerig, and Vittorio Ferrari. The open images dataset v4: Unified image classification, object detection, and visual relationship detection at scale. *IJCV*, 2020. 6

[21] Feng Li, Hao Zhang, Huaizhe Xu, Shilong Liu, Lei Zhang, Lionel M Ni, and Heung-Yeung Shum. Mask dino: Towards a unified transformer-based framework for object detection and segmentation. In *CVPR*, 2023. 2, 3

[22] Liunian Harold Li, Pengchuan Zhang, Haotian Zhang, Jianwei Yang, Chunyuan Li, Yiwu Zhong, Lijuan Wang, Lu Yuan, Lei Zhang, Jenq-Neng Hwang, et al. Grounded language-image pre-training. In *CVPR*, pages 10965–10975, 2022. 2

[23] Tsung-Yi Lin, Michael Maire, Serge Belongie, James Hays, Pietro Perona, Deva Ramanan, Piotr Dollár, and C Lawrence Zitnick. Microsoft coco: Common objects in context. In *ECCV*, 2014. 6

[24] Ilya Loshchilov and Frank Hutter. Decoupled weight decay regularization. In *ICLR*, 2019. 6

[25] Jonathon Luiten, Aljosa Osep, Patrick Dendorfer, Philip Torr, Andreas Geiger, Laura Leal-Taixé, and Bastian Leibe. Hota: A higher order metric for evaluating multi-object tracking. *International journal of computer vision*, 129:548–578, 2021. 7

[26] Matthias Minderer, Alexey Gritsenko, Austin Stone, Maxim Neumann, Dirk Weissenborn, Alexey Dosovitskiy, Aravindh Mahendran, Anurag Arnab, Mostafa Dehghani, Zhuoran Shen, et al. Simple open-vocabulary object detection with

vision transformers. In *European Conference on Computer Vision*, pages 728–755. Springer, 2022. 2

[27] Varun K Nagaraja, Vlad I Morariu, and Larry S Davis. Modeling context between objects for referring expression understanding. In *ECCV*, 2016. 6

[28] OpenAI. Sora. https://openai.com/index/video-generation-models-as-world-simulators/, 2024. 2

[29] Zhiliang Peng, Wenhui Wang, Li Dong, Yaru Hao, Shaohan Huang, Shuming Ma, and Furu Wei. Kosmos-2: Grounding multimodal large language models to the world. *arXiv preprint arXiv:2306.14824*, 2023. 6

[30] Jiyang Qi, Yan Gao, Yao Hu, Xinggang Wang, Xiaoyu Liu, Xiang Bai, Serge Belongie, Alan Yuille, Philip HS Torr, and Song Bai. Occluded video instance segmentation: A benchmark. *International Journal of Computer Vision*, pages 1–18, 2022. 1, 6

[31] Alec Radford, Jong Wook Kim, Chris Hallacy, Aditya Ramesh, Gabriel Goh, Sandhini Agarwal, Girish Sastry, Amanda Askell, Pamela Mishkin, Jack Clark, et al. Learning transferable visual models from natural language supervision. In *International conference on machine learning*, pages 8748–8763. PMLR, 2021. 1, 2, 3

[32] Nikhila Ravi, Valentin Gabeur, Yuan-Ting Hu, Ronghang Hu, Chaitanya Ryali, Tengyu Ma, Haitham Khedr, Roman Rädle, Chloe Rolland, Laura Gustafson, et al. Sam 2: Segment anything in images and videos. *arXiv preprint arXiv:2408.00714*, 2024. 1

[33] Tianhe Ren, Qing Jiang, Shilong Liu, Zhaoyang Zeng, Wenlong Liu, Han Gao, Hongjie Huang, Zhengyu Ma, Xiaoke Jiang, Yihao Chen, et al. Grounding dino 1.5: Advance the" edge" of open-set object detection. *arXiv preprint arXiv:2405.10300*, 2024. 2, 3

[34] Seonguk Seo, Joon-Young Lee, and Bohyung Han. Urvos: Unified referring video object segmentation network with a large-scale benchmark. In *Computer Vision–ECCV 2020: 16th European Conference, Glasgow, UK, August 23–28, 2020, Proceedings, Part XV 16*, pages 208–223. Springer, 2020. 6

[35] Shuai Shao, Zeming Li, Tianyuan Zhang, Chao Peng, Gang Yu, Xiangyu Zhang, Jing Li, and Jian Sun. Objects365: A large-scale, high-quality dataset for object detection. In *Proceedings of the IEEE/CVF international conference on computer vision*, pages 8430–8439, 2019. 6

[36] Yunhang Shen, Chaoyou Fu, Peixian Chen, Mengdan Zhang, Ke Li, Xing Sun, Yunsheng Wu, Shaohui Lin, and Rongrong Ji. Aligning and prompting everything all at once for universal visual perception. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 13193–13203, 2024. 2

[37] Pavan Kumar Anasosalu Vasu, Hadi Pouransari, Fartash Faghri, Raviteja Vemulapalli, and Oncel Tuzel. Mobileclip: Fast image-text models through multi-modal reinforced training. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 15963–15974, 2024. 3

[38] Haochen Wang, Cilin Yan, Shuai Wang, Xiaolong Jiang, Xu Tang, Yao Hu, Weidi Xie, and Efstratios Gavves. Towards open-vocabulary video instance segmentation. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pages 4057–4066, 2023. 1, 2, 3, 7, 8

[39] Weiyao Wang, Matt Feiszli, Heng Wang, and Du Tran. Unidentified video objects: A benchmark for dense, open-world segmentation. In *ICCV*, 2021. 6

[40] Junfeng Wu, Yi Jiang, Qihao Liu, Zehuan Yuan, Xiang Bai, and Song Bai. General object foundation model for images and videos at scale. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 3783–3795, 2024. 1, 2, 3, 6, 7, 8

[41] Kan Wu, Houwen Peng, Zhenghong Zhou, Bin Xiao, Mengchen Liu, Lu Yuan, Hong Xuan, Michael Valenzuela, Xi Stephen Chen, Xinggang Wang, et al. Tinyclip: Clip distillation via affinity mimicking and weight inheritance. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pages 21970–21980, 2023. 3

[42] Yunyang Xiong, Bala Varadarajan, Lemeng Wu, Xiaoyu Xiang, Fanyi Xiao, Chenchen Zhu, Xiaoliang Dai, Dilin Wang, Fei Sun, Forrest Iandola, et al. Efficientsam: Leveraged masked image pretraining for efficient segment anything. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 16111–16121, 2024. 3

[43] Ning Xu, Linjie Yang, Jianchao Yang, Dingcheng Yue, Yuchen Fan, Yuchen Liang, and Thomas S. Huang. Youtube-vis dataset 2021 version. https://youtube-vos.org/dataset/vis/. 1, 3, 6

[44] Bin Yan, Yi Jiang, Peize Sun, Dong Wang, Zehuan Yuan, Ping Luo, and Huchuan Lu. Towards grand unification of object tracking. In *ECCV*, 2022. 2

[45] Bin Yan, Yi Jiang, Jiannan Wu, Dong Wang, Ping Luo, Zehuan Yuan, and Huchuan Lu. Universal instance perception as object discovery and retrieval. In *CVPR*, 2023. 2, 3

[46] Linjie Yang, Yuchen Fan, and Ning Xu. Video instance segmentation. In *ICCV*, 2019. 1, 3, 6

[47] Lewei Yao, Jianhua Han, Youpeng Wen, Xiaodan Liang, Dan Xu, Wei Zhang, Zhenguo Li, Chunjing Xu, and Hang Xu. Detclip: Dictionary-enriched visual-concept paralleled pre-training for open-world detection. In *NeurIPS*, 2022. 3

[48] Licheng Yu, Patrick Poirson, Shan Yang, Alexander C Berg, and Tamara L Berg. Modeling context in referring expressions. In *ECCV*, 2016. 6

[49] Alireza Zareian, Kevin Dela Rosa, Derek Hao Hu, and Shih-Fu Chang. Open-vocabulary object detection using captions. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 14393–14402, 2021. 2

[50] Chaoning Zhang, Dongshen Han, Yu Qiao, Jung Uk Kim, Sung-Ho Bae, Seungkyu Lee, and Choong Seon Hong. Faster segment anything: Towards lightweight sam for mobile applications. *arXiv preprint arXiv:2306.14289*, 2023. 3

[51] 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. 2023. 2, 3

[52] Hao Zhang, Feng Li, Xueyan Zou, Shilong Liu, Chunyuan Li, Jianwei Yang, and Lei Zhang. A simple framework for

open-vocabulary segmentation and detection. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pages 1020–1031, 2023. 2

[53] Renhong Zhang, Tianheng Cheng, Shusheng Yang, Haoyi Jiang, Shuai Zhang, Jiancheng Lyu, Xin Li, Xiaowen Ying, Dashan Gao, Wenyu Liu, et al. Mobileinst: Video instance segmentation on the mobile. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 38, pages 7260–7268, 2024. 3

[54] Shifeng Zhang, Cheng Chi, Yongqiang Yao, Zhen Lei, and Stan Z Li. Bridging the gap between anchor-based and anchor-free detection via adaptive training sample selection. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, pages 9759–9768, 2020. 2

[55] Zhuoyang Zhang, Han Cai, and Song Han. Efficientvit-sam: Accelerated segment anything model without performance loss. *arXiv preprint arXiv:2402.05008*, 2024. 3

[56] Tiancheng Zhao, Peng Liu, Xuan He, Lu Zhang, and Kyusong Lee. Real-time transformer-based open-vocabulary detection with efficient fusion head. *arXiv preprint arXiv:2403.06892*, 2024. 3

[57] Yian Zhao, Wenyu Lv, Shangliang Xu, Jinman Wei, Guanzhong Wang, Qingqing Dang, Yi Liu, and Jie Chen. Detrs beat yolos on real-time object detection. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 16965–16974, 2024. 3

[58] Xingyi Zhou, Rohit Girdhar, Armand Joulin, Philipp Krähenbühl, and Ishan Misra. Detecting twenty-thousand classes using image-level supervision. In *ECCV*, pages 350–368. Springer, 2022. 2, 8

[59] Xizhou Zhu, Weijie Su, Lewei Lu, Bin Li, Xiaogang Wang, and Jifeng Dai. Deformable detr: Deformable transformers for end-to-end object detection. In *ICLR*, 2021. 4

[60] Xueyan Zou, Zi-Yi Dou, Jianwei Yang, Zhe Gan, Linjie Li, Chunyuan Li, Xiyang Dai, Harkirat Behl, Jianfeng Wang, Lu Yuan, et al. Generalized decoding for pixel, image, and language. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 15116–15127, 2023. 2