Autoregressive Universal Video Segmentation Model

Miran Heo[*,†,1,2] Sukjun Hwang[*,3] Min-Hung Chen[1] Yu-Chiang Frank Wang[1,4]
Albert Gu[3] Seon Joo Kim[‡,2] Ryo Hachiuma[‡,1]

[1]NVIDIA [2]Yonsei University [3]Carnegie Mellon University [4]National Taiwan University

Abstract

Recent video foundation models such as SAM2 excel at prompted video segmentation by treating masks as a general-purpose primitive. However, many real-world settings require unprompted segmentation that aims to detect and track all objects in a video without external cues, leaving today's landscape fragmented across task-specific models and pipelines. We recast streaming video segmentation as sequential mask prediction, analogous to language modeling, and introduce the Autoregressive Universal Video Segmentation Model (AUSM), a single architecture that unifies both prompted and unprompted video segmentation. Built on recent state-space models, AUSM maintains a fixed-size spatial state and scales to video streams of arbitrary length. Furthermore, all components of AUSM are designed for parallel training across frames, yielding substantial speedups over iterative training. On standard benchmarks (DAVIS17, YouTube-VOS 2018 & 2019, MOSE, YouTube-VIS 2019 & 2021, and OVIS) AUSM outperforms prior universal streaming video segmentation methods and achieves up to 2.5× faster training on 16-frame sequences.

1. Introduction

Language and video are both sequential modalities that arrive as streams, yet their modeling trajectories have diverged. In language, decoder-only large language models (LLMs) show that a single scalable architecture trained on massive corpora can subsume diverse tasks. Video perception would benefit from the same unification: annotations are fragmented across tasks while video data is expensive to curate, so a unified model could amortize supervision and deployment. We focus on streaming video segmentation, and an ideal universal model should: (i) accommodate a broad set of tasks, (ii) preserve fine-grained spatio-temporal details from past inputs, (iii) support inference over long videos, and (iv) enable training that scales efficiently with sequence length. Despite the structural parallels to language, current practice in video remains partitioned into task-specific architectures and training protocols, and no existing approach simultaneously meets all four criteria.

We categorize streaming video segmentation into two regimes: prompted and unprompted. Prompted video segmentation, exemplified by video object segmentation (VOS), takes an initial human cue (*e.g.,* mask, box, point, or text) and propagates specified targets over time; it is effective for user-interactive editing, but does not naturally handle the emergence of novel instances without re-prompting (*e.g.,* in autonomous driving). Unprompted video segmentation – comprising video instance and panoptic segmentation (VIS/VPS) – aims to detect and track all instances of predefined categories throughout a video. Many detect-then-track pipelines [18, 46, 52] process frames independently and associate *post hoc*; even approaches that leverage past information [22, 46, 55] compress each instance into a few vectors, preserving history mainly for identity association and discarding fine-grained spatio-temporal details needed for detection. Recent attempts at universal models [2, 24, 51] retrofit VOS into VIS-style architectures by encoding an instance as a heavily compressed token, which yields noticeable performance drops on VOS. Furthermore, to our knowledge, existing training frameworks for video segmentation lack LLM-style parallelized training, limiting scalability with respect to sequence length.

In this paper, we present an autoregressive universal video segmentation model, **AUSM**, that unifies both prompted and unprompted video segmentation tasks while scaling to long video settings.

Unified Structure. First, we identify a key connection between the autoregressive pipeline of decoder-only LLMs and streaming video perception: **next-word prediction as next-frame mask prediction**. Based on this perspective, we present a unification of prompted and unprompted video segmentation and design a universal model that achieves state-of-the-art performance among online universal methods on seven benchmarks comprising both VOS and VIS

[*]Equal contribution.
[†]Work partially done during an internship at NVIDIA.
[‡]Corresponding authors.

using shared weights.

Architecture. AUSM is designed with components specialized for streaming videos, **History Marker** and **History Compressor**. History Marker removes abstraction of instances by leveraging Token Mark [16] and dissolves segmentation masks into frame features for retrieval of instance-wise information. This effectively preserves fine-grained information, demonstrating a nearly 10% improvement in VOS performance compared to previous unified online architectures. History Compressor then takes the output and compresses spatio-temporal information of all past frames into a single spatial state [8, 11]. While video segmentation models typically store fewer than ten frame features [31, 37], our design makes processing arbitrarily long streams feasible.

Training Acceleration. AUSM supports **parallel training**, a critical property of the building blocks [11, 41] used in decoder-only LLMs for extending to long sequences. Compared to existing frameworks that recurrently process frames during training [17, 37, 54], our training pipeline shows significant speedups.

Finally, we evaluate AUSM across a diverse set of benchmarks spanning both prompted and unprompted video segmentation: DAVIS 2017 [32], YouTube-VOS 18&19 [50], MOSE [9], YouTube-VIS 19&21 [52], and OVIS [34]. AUSM delivers strong performance across all benchmarks, outperforming previous online universal video segmentation models [24, 51]. Importantly, these results are achieved *without relying on FIFO memory buffers* [31, 37], highlighting the efficiency and scalability of our autoregressive design. Furthermore, parallel training becomes faster compared to recurrent training frameworks as the sequence length increases, achieving up to $2.5\times$ faster training than iterative baselines with 16-frame sequences.

2. Related Work

Unprompted Video Segmentation. A key distinction among existing models is whether they condition on past predictions $\hat{y}_{<t}$ to generate the current output during inference. Conventional online tracking-by-detection approaches completely exclude $\hat{y}_{<t}$ in their model design, treating each visual feature independently from previous predictions [18, 22, 46, 55]. While these methods benefit from parallelizable computation during training, they often require external memory banks to maintain temporal consistency at inference time.

In contrast, models such as GenVIS [15] adopt query propagation by conditioning on prior predictions $\hat{y}_{<t}$ through object-level vector representations. Although this formulation enhances temporal coherence, object vectorization significantly degrades the granularity of mask predictions [22]. RoCoVIS [17] addresses this by introducing instance mask propagation, which greatly improves mask quality. However, this design inherently breaks parallelism, as predictions must be generated sequentially, leading to reduced training efficiency.

Another line of work follows an offline paradigm, where the entire video $\mathcal{I}_{1:T}$ is available in advance during inference [14, 19, 44]. While this setting enables models to access long-range context, such models are typically not conditioned on intermediate outputs $\hat{y}$ and therefore cannot refine predictions based on prior object states. As a result, despite high computational cost, these models often underperform recurrent methods [46] that explicitly leverage temporal feedback.

Prompted Video Segmentation. Prompted Video Segmentation focuses on segmenting and tracking an object throughout a video, given a specified target in the first frame, without requiring any class labels. Unlike Unprompted Segmentation tasks that involve discovering and classifying all objects, this task aims to track a user-specified target. Prompted Video Segmentation is highly practical, particularly for interactive and general-purpose applications.

One of the most influential paradigms is the Space-Time Memory Network (STM) [31], which inspired a wide range of follow-up methods. STM maintains a memory bank consisting of past frames and their corresponding masks, and performs dense matching between the current frame and the stored memory to retrieve relevant information for accurate mask propagation. Building on this, XMem [6] introduces enhanced memory mechanisms for improved long-term tracking. More recently, SAM2 [37], building upon SAM [23], supports more flexible input representations such as points and boxes, and also introduces large-scale datasets to enable broader evaluation and training.

Another line of research explores hierarchical propagation using transformer-based architectures [53, 54]. These methods gradually propagate identity information from past frames to the current frame through a hierarchical structure. It supports multi-object processing in a unified manner, unlike STM-based approaches that typically handle each object separately.

Universal Video Segmentation. The emergence of transformer-based detection and segmentation architectures [3, 5, 43] has facilitated early attempts to unify multiple tasks within unprompted video segmentation (*e.g.* VIS, VPS, and VSS) under a single framework [21, 25]. In parallel, similar efforts have been made in the prompted setting, aiming to jointly handle VOS and referring VOS within a shared architecture [47, 48].

Recent research has progressed toward bridging the gap between Unprompted and Prompted Video Segmentation. These efforts aim to develop universal models that can support both interactive and fully automatic scenarios within a unified framework, thereby reducing reliance on task-specific designs and enabling broader applicability across

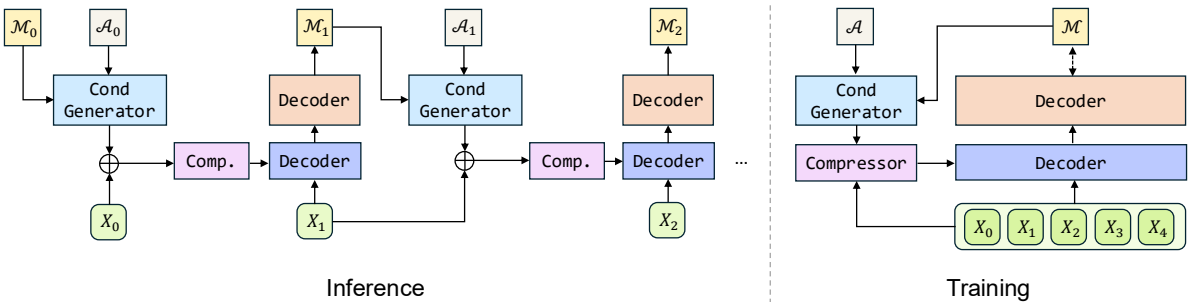

Figure 1. High-level overview of AUSM during training and inference. **Left:** At inference time, AUSM processes frames in a recurrent manner with constant decoding time per frame. Rather than maintaining an explicit memory buffer, temporal history is compressed via the Mamba layer within the `Compressor` module and passed through time. **Right:** During training, a parallel formulation is used to jointly optimize over multiple frames, enabling efficient and scalable learning.

diverse video understanding tasks. An offline method, TarViS [2], represents the first attempt to jointly model both settings by encoding task-specific targets as a set of queries. UNINEXT [51] further introduces a prompt-guided object discovery and retrieval paradigm. UniVS [24] leverages prompts as queries by treating predicted masks from previous frames as visual prompts for the current frame.

State Space Models. State Space Models (SSMs) [11–13] have recently emerged as a promising alternative to Transformer [41] for sequence modeling. While Transformer requires storing all tokens in key-value caches, SSMs compress all history into a single state. This property gives SSMs constant computational complexity and memory requirements during inference, providing significant advantages for modeling long sequences compared to Transformers' linearly growing complexity and memory.

3. Autoregressive Universal Video Segmentation Model (AUSM)

In Sec. 3.1, we first formalize video segmentation within an autoregressive framework inspired by LLMs, which provides a seamless unification of prompted and unprompted video segmentation tasks. In Sec. 3.2, we then introduce the architectural components and overall algorithm of AUSM using the recurrent form (Fig. 1-Left). Finally, in Sec. 3.3, we demonstrate how each component of AUSM is designed for parallel training (Fig. 1-Right).

3.1. Video Segmentation as Language Modeling

Video segmentation encompasses a family of tasks [7, 20, 32, 42, 52] with varying objectives but a shared core definition. Given a video consisting of T frames, $(\mathcal{I}_1, \dots, \mathcal{I}_T)$, where each frame $\mathcal{I}_t \in \mathbb{R}^{H \times W \times 3}$ is an RGB image with spatial resolution $H \times W$, the goal is to produce $(\hat{y}_1, \dots, \hat{y}_T)$ that predicts a sequence of segmentation ground truth $Y = (y_1, \dots, y_T)$. Each y_t represents ground truth at time t and is defined as a set $\{(c_t^i, m_t^i)\}_{i=1}^{N^{\text{gt}}}$, where $c_t^i \in \{1, \dots, K\}$ denotes the class label of the i-th object, $m_t^i \in \{0,1\}^{H \times W}$ is its segmentation mask, and N^{gt} is the number of foreground objects. Similarly, each $\hat{y}_t = \{(\hat{c}_t^i, \hat{m}_t^i)\}_{i=1}^{N^{\text{det}}}$ represents predictions at time t, where $\hat{c} \in \{1, \dots, K, \texttt{bg}\}$ denotes the class label (`bg` indicates background), $\hat{m} \in \{0,1\}^{H \times W}$ is the predicted mask, and N^{det} is the number of object queries.

We observe that video segmentation can be reformulated within an autoregressive framework used in modern LLMs [10, 29]. Language models generate sequences by conditioning each token on all previous tokens:

$$P(y_{1:T}) = \prod_{t=1}^{T} P(y_t \mid y_{<t}). \tag{1}$$

The sequential nature of language naturally aligns with this autoregressive formulation, which provides an elegant unification of different tasks in language into a single universal architecture [35]. Similarly, the segmentation of a streaming video can be expressed as

$$P(y_{1:T} \mid \mathcal{I}_{1:T}) = \prod_{t=1}^{T} P(y_t \mid y_0, y_{<t}, \mathcal{I}_{\leq t}), \tag{2}$$

which explicitly models the dependence of each frame's segmentation y_t on the current frame $\mathcal{I}_t$, all previous frames $\mathcal{I}_{<t}$, previous segmentations $y_{<t}$, and potentially an initial prompt y_0 [32]. This formulation covers all video segmentation tasks: prompted video segmentation begins with an initial human prompt $y_0 = m_0$, while no initial prompt is given for unprompted video segmentation ($y_0 = \varnothing$) [20, 52].

3.2. AUSM Architecture

The design of AUSM allows sophisticated conditioning on past history while maintaining constant memory to process arbitrarily long video sequences. As shown in Alg. 1, AUSM uses a pool of object queries $\mathcal{V} \in \mathbb{R}^{N^{\text{det}} \times D}$ and keeps a set of buffer ID vectors $\mathcal{B} \in \mathbb{R}^{N^{\text{id}} \times D}$ that are essential for tracking identified instances, where N^{id} is the total

number of available ID vectors. Additionally, AUSM maintains two sets during inference: $\mathcal{A}$ for storing allocated ID vectors and $\mathcal{M}$ for storing previous mask predictions. These sets maintain a one-to-one mapping, ensuring that $\mathcal{A}_t^i$ (i-th element of $\mathcal{A}_t$) corresponds to $\mathcal{M}_t^i$ (i-th element of $\mathcal{M}_t$), thus $|\mathcal{A}_t| = |\mathcal{M}_t|$ at all times. We assume each frame is encoded by a frame-independent backbone, yielding features $\{X_1, \ldots, X_T\}$ where $X_t = \texttt{backbone}(\mathcal{I}_t) \in \mathbb{R}^{H \times W \times D}$ for $t \in \{1, \ldots, T\}$, and D is the channel dimension. Additionally, we define $X_0 \in \mathbb{R}^{H \times W \times D}$ by spatially repeating a D-dimensional trainable vector.

Algorithm 1 Recurrent Form for Inference.

1: Initialize $\mathcal{B} \in \mathbb{R}^{N^{\text{id}} \times D}$, $\mathcal{V} \in \mathbb{R}^{N^{\text{det}} \times D}$
2: **if** $y_0 = \varnothing$ **then**
3: $\quad \mathcal{A}_0, \mathcal{M}_0 \leftarrow \varnothing, \varnothing$
4: $\quad \mathcal{B}_0 \leftarrow \mathcal{B}$
5: **else**
6: $\quad \mathcal{A}_0 \leftarrow \texttt{Sampler}(\mathcal{B}, |y_0|)$
7: $\quad \mathcal{M}_0 \leftarrow m_0$
8: $\quad \mathcal{B}_0 \leftarrow \mathcal{B} \setminus \mathcal{A}_0$
9: **end if**
10: **for** $t = 1$ to T **do**
11: $\quad E_t \leftarrow X_{t-1} + \texttt{CondGenerator}(\mathcal{A}_{t-1}, \mathcal{M}_{t-1})$
12: $\quad F_t \leftarrow \texttt{Compressor}(E_t)$
13: $\quad G_t \leftarrow \texttt{Decoder}(Q = X_t, KV = F_t)$
14: $\quad \hat{y}_t^{\text{trk}}, \hat{y}_t^{\text{det}} \leftarrow \texttt{Decoder}\big(Q = \text{concat}(\mathcal{A}_{t-1}, \mathcal{V}),\ KV = G_t\big)$
15: $\quad \mathcal{D} \leftarrow \texttt{filter\_fg}(\hat{y}_t^{\text{det}})$
16: $\quad \mathcal{A}' \leftarrow \texttt{Sampler}(\mathcal{B}_{t-1}, |\mathcal{D}|)$
17: $\quad \mathcal{A}_t \leftarrow \texttt{concat}(\mathcal{A}_{t-1}, \mathcal{A}')$
18: $\quad \mathcal{B}_t \leftarrow \mathcal{B}_{t-1} \setminus \mathcal{A}'$
19: $\quad \mathcal{M}_t \leftarrow \texttt{concat}(\hat{m}_t^{\text{trk}}, \mathcal{D})$
20: **end for**

Unification of Tasks. By altering the initialization of $\mathcal{A}$ and $\mathcal{M}$, AUSM handles both prompted and unprompted video segmentation. For unprompted video segmentation, both $\mathcal{A}_0$ and $\mathcal{M}_0$ are initialized as empty sets (Line 3 in Alg. 1). The initialization for prompted video segmentation involves $\texttt{Sampler}(\mathcal{B}, n)$, which uniformly samples n vectors from $\mathcal{B}$ without replacement and returns them as a matrix whose rows are the sampled vectors. Formally,

$$\texttt{Sampler}(\mathcal{B}, n) = [b_1, \ldots, b_n]^\top \in \mathbb{R}^{n \times D}, \text{where}$$
$$(b_1, \ldots, b_n) \sim \text{Unif}\{(b'_1, \ldots, b'_n) \in \mathcal{B}^n : b'_i \neq b'_j, \forall i \neq j\}.$$

In the prompted setting, $\mathcal{A}_0 = \texttt{Sampler}(\mathcal{B}, |y_0|)$ and $\mathcal{M}_0 = m_0$, while $\mathcal{B}_0 = \mathcal{B} \setminus \mathcal{A}_0$ (Lines 6–7 in Alg. 1).

History Marker. The vectorization of objects widely used in VIS [14, 15, 24] loses spatial detail due to excessive compression of object-wise information. In contrast, History Marker preserves fine details similarly to memory-based methods in VOS [31]. Specifically, it leverages Token Mark [16] to dissolve instance masks into a spatial feature map, minimizing information loss [17, 54]. At time t, given allocated vectors $\mathcal{A}_{t-1}$ and segmentation masks $\mathcal{M}_{t-1}$, this module operates as:

$$\texttt{HistoryMarker}(\mathcal{A}_{t-1}, \mathcal{M}_{t-1}) = S_t \in \mathbb{R}^{H \times W \times D},$$
$$S_t[h, w, :] = \frac{\sum_{i=1}^{|\mathcal{A}_{t-1}|} \mathcal{M}_{t-1}^i[h, w] \cdot \mathcal{A}_{t-1}^i}{\epsilon + \sum_{i=1}^{|\mathcal{A}_{t-1}|} \mathcal{M}_{t-1}^i[h, w]},$$

where ϵ is a small number to prevent division by zero. The output S_t is added to X_{t-1} to form E_t, which is then passed to the History Compressor.

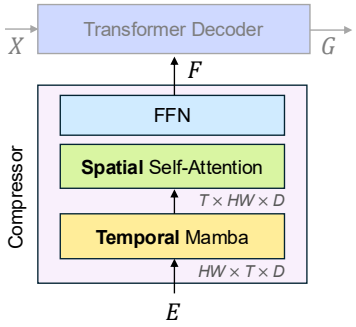

Figure 2. `Compressor` module. Mamba encodes temporal dependencies, while self-attention captures spatial structure, enabling recurrent compression of E with constant memory.

History Compressor. This module encodes visual features from previous frames and instance-specific masks into a single spatial state, enabling inference over arbitrarily long videos with constant memory. As shown in Fig. 2, each layer in the History Compressor consists of three components: Mamba [11], Self-Attention [41], and a feed-forward network. By decomposing video features into spatial and temporal dimensions, these components operate on different axes: Mamba processes the temporal dimension while self-attention handles the spatial dimension. We choose Mamba for the temporal dimension for two reasons: (1) videos are inherently sequential in time, aligning with SSM architectures; and (2) modeling videos is memory-intensive on GPUs because each frame contributes many tokens. The recurrent design of Mamba enables a single spatial state that is updated each frame, eliminating the need to store spatio-temporal features from all previous frames.

History Decoder. The History Decoder is a stack of Transformer decoder layers that outputs G_t by taking the current-frame features X_t as queries and the compressed state F_t as keys and values. This spatial feature G_t has two important properties: (1) it incorporates current-frame information through the image encoder, and (2) it retains fine-grained information about objects from previous frames through the compressed state.

Pixel Decoder and Update Process. The Pixel Decoder follows [5], comprising Transformer decoder layers with masked attention, and takes object queries as in Hungarian matching–based detection methods [3, 4, 19]. Specifically, the final predictions for frame t are obtained using both previously allocated ID vectors $\mathcal{A}_{t-1}$ and ob-

ject queries $\mathcal{V}$ as inputs, with G_t providing keys and values. Each ID vector in $\mathcal{A}_{t-1}$ tracks its corresponding object, and $\mathcal{V}$ detects instances not yet assigned to any vector in $\mathcal{A}_{t-1}$. Therefore, this process yields two types of predictions: tracking predictions $\hat{y}_t^{\text{trk}}$ from $\mathcal{A}_{t-1}$ and detection predictions $\hat{y}_t^{\text{det}}$ from $\mathcal{V}$.

After prediction, the sets $\mathcal{A}$, $\mathcal{B}$, and $\mathcal{M}$ are updated for the next frame. We define the following operations used in Alg. 1:

- `filter_fg`$(\hat{y}_t^{\text{det}})$: Filters predictions from $\hat{y}_t^{\text{det}}$ to retain only foreground objects, returning a set $\mathcal{D}$ of newly detected objects.
- `concat`(A, B): Concatenates two vector sets A and B.

The update process proceeds as follows: (1) foreground detections $\mathcal{D}$ are filtered from $\hat{y}_t^{\text{det}}$; (2) $|\mathcal{D}|$ new vectors $\mathcal{A}'$ are sampled from the remaining buffer $\mathcal{B}_{t-1}$; (3) $\mathcal{A}_t$ is formed by concatenating $\mathcal{A}_{t-1}$ and $\mathcal{A}'$, while $\mathcal{B}_t$ is obtained by removing $\mathcal{A}'$ from $\mathcal{B}_{t-1}$; (4) $\mathcal{M}_t$ is updated by concatenating mask predictions from $\hat{y}_t^{\text{trk}}$ and newly detected objects $\mathcal{D}$.

3.3. Parallel Training as Language Models

Modern decoder-only language models [10, 29] significantly benefit from parallel training by adopting building blocks (*e.g.,* Transformers and SSMs) that support the teacher-forcing technique [40]. However, this parallel training cannot be readily applied to existing video segmentation methods that rely on frame-by-frame propagation of outputs, such as those employing query propagation [15, 17, 30]. Therefore, existing frameworks train video segmentation models by recurrently processing frames, leading to severely inefficient training.

Algorithm 2 Parallel Form for Training. $A_{B:B+C}$ denotes $(A_B, \ldots, A_{B+C})$ for brevity.

Initialize $\mathcal{B} \in \mathbb{R}^{N^{\text{id}} \times D}$, $\mathcal{V} \in \mathbb{R}^{N^{\text{det}} \times D}$
$\mathcal{A} \leftarrow$ `Sampler`$(\mathcal{B}, N^{\text{gt}})$
$(y_{1:T}^{\text{trk}}, y_{1:T}^{\text{det}}), \mathcal{A}_{0:T-1}, \mathcal{M}_{0:T-1} \leftarrow$ `Preprocess`$(y_{1:T}, \mathcal{A})$
$E_{1:T} \leftarrow (X_{t-1} +$ `CondGenerator`$(\mathcal{A}_{t-1}, \mathcal{M}_{t-1}))_{1:T}$
$F_{1:T} \leftarrow$ `Compressor`$(E_{1:T})$
$G_{1:T} \leftarrow$ `Decoder`$(Q = X_t, KV = F_t)_{1:T}$
$(\hat{y}_t^{\text{trk}}, \hat{y}_t^{\text{det}})_{t=1}^{T} \leftarrow$ `Decoder`$(Q =$ `Concat`$(\mathcal{A}_{t-1}, \mathcal{V}), KV = G_t)_{1:T}$
$\mathcal{L}_{\text{total}} = \sum_{t=1}^{T} \left[L^{\text{trk}}(y_t^{\text{trk}}, \hat{y}_t^{\text{trk}}) + L^{\text{det}}(y_t^{\text{det}}, \hat{y}_t^{\text{det}}) \right]$

In contrast, as shown in Alg. 2, all modules in AUSM are compatible with teacher forcing. Therefore, AUSM supports parallel training across the temporal dimension, yielding substantial improvements in training efficiency. The parallel training begins by sampling N^{gt} vectors from the buffer $\mathcal{B}$ to form the set $\mathcal{A}$, with a one-to-one mapping to ground-truth instances. A critical component is the `Preprocess` function, which prepares $y_{1:T}^{\text{trk}}$, $y_{1:T}^{\text{det}}$, $\mathcal{M}_{0:T-1}$, and $\mathcal{A}_{0:T-1}$.

As illustrated in Fig. 3, `Preprocess` randomly samples a timestep t_{sample}^i for each instance i (highlighted with colored contours). This timestep determines when an instance shifts from being treated as a detection target to a tracking target. Using the ground truth $y_{1:T}^i$ and sampled index t_{sample}^i, it constructs, for each instance i and frame t,

$$y_t^{\text{det},i} = \begin{cases} y_t^i & \text{if } t \le t_{\text{sample}}^i \\ \varnothing & \text{otherwise} \end{cases}$$

$$y_t^{\text{trk},i} = \begin{cases} y_t^i & \text{if } t > t_{\text{sample}}^i \\ \varnothing & \text{otherwise} \end{cases}$$

$$\mathcal{M}_t^i = \begin{cases} y_t^i & \text{if } t \ge t_{\text{sample}}^i \\ \varnothing & \text{otherwise} \end{cases}$$

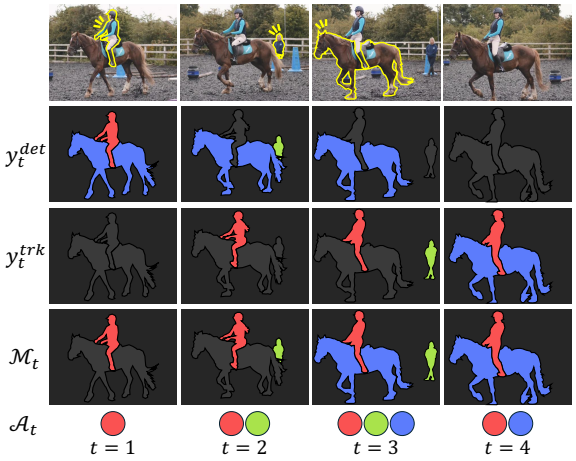

Figure 3. Schematic of `Preprocess`. The video contains three ground-truth instances: `person1` (the person riding the horse), `person2` (the person standing), and `horse`. Each instance is matched with a vector from $\mathcal{A}$, represented by colored circles: red for `person1`, green for `person2`, and blue for `horse`. The highlighted contours indicate the randomly sampled timesteps (t_{sample}^i) for each instance.

The set $\mathcal{A}_{t-1}$ is then constructed to contain vectors for foreground instances in $\mathcal{M}_{t-1}$, maintaining the one-to-one mapping between allocated vectors and instances.

Once preprocessing is complete, all subsequent operations, from applying History Marker to calculating losses, can be executed in parallel across frames. The training loss decomposes into tracking loss L^{trk} and detection loss L^{det}. For tracking, the one-to-one mapping between y_t^{trk} and $\mathcal{A}_{t-1}$ enables direct loss computation. For detection, where no predetermined mapping exists between y_t^{det} and the detection queries $\mathcal{V}$, we employ the Hungarian algorithm [3] to obtain optimal assignments before computing the loss.

4. Experiments

4.1. Datasets

Training Datasets. We train AUSM on a diverse collection of segmentation datasets that are publicly available to ensure robustness across both prompted and unprompted

Table 1. Quantitative results on Prompted and Unprompted Video Segmentation benchmarks. We report $\mathcal{J}\&\mathcal{F}$ for DAVIS and MOSE, $\mathcal{G}$ for YouTube-VOS, and AP for YouTube-VIS and OVIS. † denotes methods additionally trained on private data. "✗" marks tasks architecturally incompatible with the models, and "–" marks feasible tasks without reported results.

Method	Backbone	*Prompted*				*Unprompted*		
		DAVIS	MOSE	YTVOS18	YTVOS19	YTVIS19	YTVIS21	OVIS
Task-specialized Models								
Xmem [6]	ResNet50	86.2	57.6	85.7	85.5	✗	✗	✗
DeAOT [53]	ResNet50	85.2	59.4	86.0	85.9	✗	✗	✗
DeAOT [53]	Swin-B	86.2	–	86.2	86.1	✗	✗	✗
SAM2 [37]†	Hiera-B+	90.2	76.6	–	88.6	✗	✗	✗
SAM2 [37]†	Hiera-L	90.7	77.9	–	89.3	✗	✗	✗
UniRef++ [48]	Swin-L	83.9	59.0	83.2	83.0	✗	✗	✗
GenVIS [15]	Swin-L	✗	✗	✗	✗	64.0	59.6	45.2
DVIS [56]	Swin-L	✗	✗	✗	✗	63.9	58.7	47.1
VISAGE [22]	Swin-L	✗	✗	✗	✗	64.2	59.6	46.5
Video K-Net [25]	Swin-B	✗	✗	✗	✗	54.1	–	–
Universal Models - offline								
TarViS [2]	Swin-T	82.8	–	–	–	–	50.9	34.0
TarViS [2]	Swin-L	85.3	–	–	–	–	60.2	43.2
Universal Models - online								
UNINEXT [51]	ResNet50	74.5	–	77.0	–	53.0	–	34.0
UNINEXT [51]	ConvNeXt-L	77.2	–	78.1	–	**64.3**	–	41.1
UniVS [24]	Swin-T	71.7	–	70.3	–	52.4	51.6	33.0
UniVS [24]	Swin-B	75.0	–	70.9	–	57.8	56.5	39.0
UniVS [24]	Swin-L	76.2	–	71.5	–	60.0	57.9	41.7
AUSM	Swin-T	76.4	58.8	79.5	78.3	54.9	52.1	39.4
AUSM	Swin-B	**81.6**	**62.1**	**80.2**	**79.1**	62.6	**58.6**	**45.5**

segmentation paradigms. Specifically, we use COCO [26], DAVIS 2017 [32], MOSE [9], and SA-V [37], YouTube-VIS 2019 & 2021 [52], and OVIS [34]. For datasets that include semantic labels [26, 34, 52], we additionally supervise a classification objective, allowing our model to perform category-aware segmentation.

Evaluation Benchmarks and Metrics. For prompted tasks, we follow standard semi-supervised VOS protocols on four benchmarks: DAVIS 2017, YouTube-VOS 2018 & 2019 [50], and MOSE, a recent benchmark focusing on multi-object segmentation with complex instance interactions. For DAVIS and MOSE, we report the standard $\mathcal{J}\&\mathcal{F}$ metric, which averages region similarity and contour accuracy. For YouTube-VOS, we report $\mathcal{G}$, the average of $\mathcal{J}\&\mathcal{F}$ computed across both seen and unseen categories. For unprompted tasks, we evaluate on VIS benchmarks: YouTube-VIS 2019 & 2021 and OVIS, with the latter featuring challenging heavy occlusions with long videos. Following standard practice, we report the mean Average Precision (AP).

4.2. Training Details

A core design of AUSM is that it enables *parallel* training over frame sequences. Our architecture concurrently processes all frames within a clip while preserving an autoregressive supervision objective over the output sequence. The training process is divided into three stages, progressively increasing temporal complexity and data diversity.

Stage 1 (Pseudo-video pretraining): We begin by pretraining AUSM on COCO [26] using a pseudo-video augmentation strategy (COCO-pseudo). Each image sample is randomly augmented into a 3-frame sequence using spatial transformations following [14]. **Stage 2 (Multi-source short-clip training):** The second stage introduces real video data to the model. We train on 5-frame clips sampled from a mixture of COCO-pseudo, MOSE [9], SA-V [37], YouTube-VIS 2019 & 2021 [52], and OVIS [34]. **Stage 3 (Long-clip adaptation):** In the final stage, AUSM is fine-tuned on 16-frame video clips to enhance long-range temporal modeling. To reduce memory cost, we freeze the image backbone and update only the temporal modules and prediction heads. This stage uses the same datasets as Stage 2, with the addition of the DAVIS 2017 training set [32].

4.3. Main Results

Tab. 1 reports AUSM's performance across a range of prompted and unprompted video segmentation benchmarks. We divide the table into two broad categories: (1) the top section includes specialized methods tailored for a single task – prompted or unpropted video segmentation; (2) the bottom section highlights universal frameworks that support both paradigms within a single architecture. We note that our results are obtained using a *single model* trained with

our joint learning framework, without any task-specific fine-tuning. We evaluate AUSM using Swin-Tiny and Base [27].

Prompted Video Segmentation. The strongest specialized method for prompted video segmentation is SAM2 [37], a memory-based mask propagation approach built upon the core design of STM [31], where each object is processed independently using a dedicated memory buffer. By incorporating additional private datasets, SAM2 achieves high performance across standard benchmarks. In contrast, AUSM processes multiple objects jointly in a single forward pass and operates without explicit memory buffers. Among online universal frameworks, AUSM demonstrates strong performance, showcasing the potential of our autoregressive formulation. Notably, we surpass the latest UniVS [24] (Swin-L variant) by a large margin on YouTube-VOS 2018 (+8.7), despite using a smaller Swin-B backbone. This highlights AUSM as a unified, memory-efficient alternative that balances generality and scalability.

Unprompted Video Segmentation. Specialized models for VIS achieve strong performance across benchmarks. However, their decoupled architectures are often tailored for object query propagation [15] or short-term tracking during training [22], which limits their extensibility to prompted tasks. Despite being a unified model, AUSM achieves competitive results with these specialized approaches, demonstrating its ability to capture complex object dynamics without relying on task-specific architectural constraints. This underscores the strength of our autoregressive formulation in handling unprompted segmentation while preserving compatibility with prompted settings. Notably, AUSM achieves the highest score on OVIS among universal models, a challenging benchmark characterized by heavy occlusion and long-range object interactions.

4.4. Ablation Studies

Parallel vs. Iterative Training Efficiency. To assess the empirical benefit of parallel training, we measure training time per iteration (sec/iter) over sequence lengths of 1, 2, 4, 8, and 16 using the Swin-B backbone. As shown in Fig. 4, our parallel approach scales substantially better with increasing sequence length than the iterative baseline. While the iterative method shows a steep rise in training time (increasing from 1.47s to 8.75s per iteration), our parallel approach demonstrates much slower growth (from 1.47s to only 3.45s per iteration at length 16). This results in a 2.5× faster training at sequence length 16, with even greater advantages expected at longer horizons. These results highlight the efficiency and scalability of our parallel training pipeline, making it especially well-suited for learning from long video sequences where temporal modeling is critical.

Effect of Training with Longer Sequences. To evaluate the impact of our long-clip adaptation strategy, we compare performance between Stage 2 (trained with 5-frame clips) and Stage 3 (trained with 16-frame clips) of our training pipeline. As shown in Fig. 5, the transition to longer sequences consistently improves performance across all datasets. The most substantial gains are observed on MOSE (+4.52) and OVIS (+5.2), demonstrating that longer temporal context enables better understanding of complex object dynamics and appearance variations. These long-range modeling capabilities are directly facilitated by our parallel training mechanism, which enables efficient learning over extended sequences. Importantly, these improvements are achieved without any explicit memory buffer (*e.g.* FIFO-style spatio-temporal caches). Instead, the model leverages the extrapolative capacity of the Compressor module, enabling efficient long-term reasoning with constant memory.

Table 2. Effect of varying the foreground threshold during inference on Unprompted Video Segmentation performance.

Thres.	YTVIS19	YTVIS21	OVIS
0.3	62.4	57.8	44.5
0.4	**62.6**	<u>58.1</u>	45.1
0.5	**62.6**	**58.6**	45.5
0.6	<u>61.8</u>	<u>58.1</u>	<u>46.4</u>
0.7	61.6	57.9	**46.5**

Effect of Foreground Threshold at Inference. When AUSM performs inference for Unprompted Video Segmentation, new objects are introduced into the buffer via $\texttt{filter\_fg}(\hat{y}_t^{\text{det}})$, which selects confident detection predictions based on a foreground probability threshold. To assess the impact of this component, we conduct an ablation study by varying the threshold during inference. As shown in Tab. 2, the overall results suggest that our model is relatively robust across a wide range of threshold values (0.3–0.7). While a higher threshold benefits performance on OVIS (up to 46.5 AP at 0.7), Youtube-VIS datasets achieve the best results around 0.4–0.5. We use a fixed threshold of 0.5 for all benchmarks to ensure consistency and avoid dataset-specific tuning.

Scaling Inference Compute in AUSM. Recent advances in language modeling have demonstrated that scaling inference-time computation [45] can significantly improve model performance. Building on this paradigm, studies have found that even simple input repetition techniques [1, 39] enhance model performance across various architectures [11, 41].

Inspired by these findings, we demonstrate how our model can improve its predictions through scaling inference computation. The approach involves creating repeated sequences that allow the model to refine its predictions iteratively. For a single image $\mathcal{I}$, we construct a pseudo video sequence $\mathcal{I}^{\text{aug}} = (\mathcal{I}_1, \dots, \mathcal{I}_T)$ where $\forall t \in \{1, \dots, T\}, \mathcal{I}_t = \mathcal{I}$. By processing this sequence, our model can refine predictions through allocating more attention to previously unexplored regions in subsequent iterations and iteratively

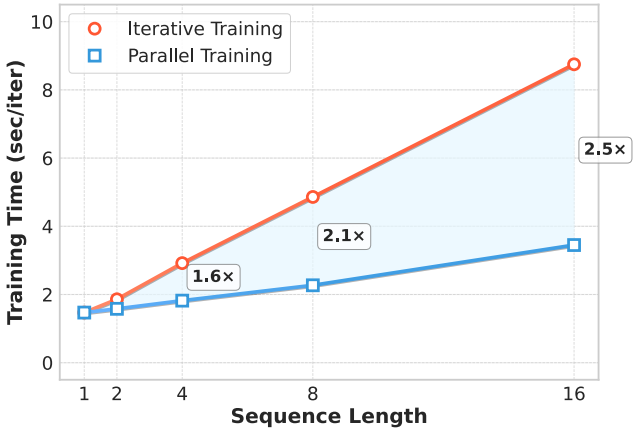

Figure 4. Comparison of training time (sec/iter) between iterative and parallel training approaches across different sequence lengths.

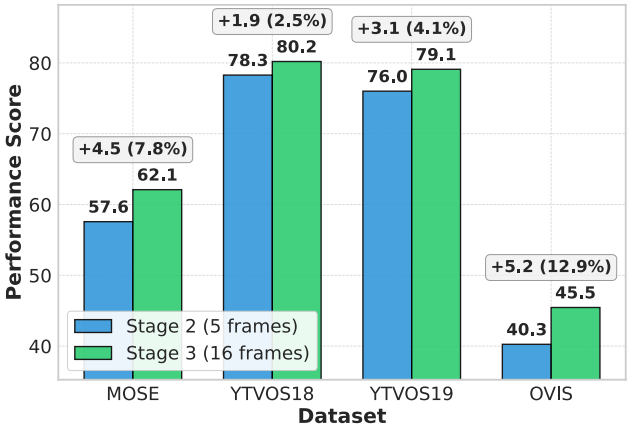

Figure 5. Performance comparison between Stage 2 (5-frame) and Stage 3 (16-frame) training across four benchmark datasets.

Table 3. Results measured using Swin-B backbone. We relegate more experimental details in the supplementary.

# Repetition	COCO	YTVIS19
×1	34.2	62.6
×2	34.9	63.3
×3	35.0	63.5

leveraging object interactions.

We extend this concept to video sequences as well. Given a video $(\mathcal{I}_1, \ldots, \mathcal{I}_T)$, we construct an augmented sequence by repeating the video frames as:

$$\mathcal{I}^{\text{aug}} = (\mathcal{I}_1, \ldots, \mathcal{I}_T, \mathcal{I}_{T-1}, \ldots, \mathcal{I}_1, \mathcal{I}_2, \ldots, \mathcal{I}_T)$$

When processing this augmented sequence, we extract predictions from only the final T frames as our refined output.

As shown in Tab. 3, we observe improved performance from scaling inference compute. While earlier sections focused on unifying various video segmentation tasks within a single framework, this extension shows that our model can serve as a recursive universal segmentation model.

5. Discussion

Future Directions. Our reformulation of video segmentation as an autoregressive modeling problem establishes a conceptual connection to language modeling, opening several promising research directions. One particularly important direction is length extrapolation [33, 49] – the ability to extend a model's effective context beyond its training sequence length. Although our model demonstrates strong performance and theoretical support for processing infinite frames, we expect performance degradation when processing extremely long sequences, a challenge shared with language models. Techniques for extending context in language models can be valuable for maintaining performance across arbitrarily long videos.

Another promising direction is the extension of our unified framework to encompass additional video understanding tasks. While this work addresses both Prompted and Unprompted Video Segmentation, other tasks such as Referring Video Object Segmentation [38] could be integrated into our framework with minimal modifications. For instance, natural language prompting could be incorporated by initializing the Mamba state in our Compressor with text embeddings, leveraging the flexible conditioning capabilities of our autoregressive formulation.

Limitations. While our model achieves state-of-the-art performance on Unprompted Video Segmentation benchmarks, we observe a modest performance gap on Prompted Video Segmentation tasks compared to specialized task-specific models. We attribute this discrepancy primarily to our architectural balance, which currently emphasizes a deep per-frame image encoder optimized for object-level understanding. Future work could explore architectural reconfigurations; for example, reducing backbone depth while strengthening temporal modeling components, which may enhance performance on prompted tasks without compromising capabilities on unprompted tasks.

6. Conclusion

We introduce AUSM as a step toward a more unified, scalable, and general-purpose formulation of video segmentation. AUSM combines parallel training, temporally-aware modeling without explicit memory buffers, and state space models to enable efficient long-range reasoning with constant-time inference. Experimental results across diverse benchmarks show that autoregressive modeling provides a practical and scalable solution for video segmentation. We hope this autoregressive perspective offers both a strong baseline and a fresh direction for future research in video perception.

Acknowledgements

We thank De-An Huang for feedback and helpful discussions. This work was supported by the National Research Foundation of Korea (NRF) grant funded by the Korea government (MSIT) (RS-2025-00554790).

References

[1] Simran Arora, Aman Timalsina, Aaryan Singhal, Benjamin Spector, Sabri Eyuboglu, Xinyi Zhao, Ashish Rao, Atri Rudra, and Christopher Ré. Just read twice: closing the recall gap for recurrent language models. *arXiv preprint arXiv:2407.05483*, 2024. 7

[2] Ali Athar, Alexander Hermans, Jonathon Luiten, Deva Ramanan, and Bastian Leibe. Tarvis: A unified approach for target-based video segmentation. In *CVPR*, 2023. 1, 3, 6

[3] Nicolas Carion, Francisco Massa, Gabriel Synnaeve, Nicolas Usunier, Alexander Kirillov, and Sergey Zagoruyko. End-to-end object detection with transformers. In *ECCV*, 2020. 2, 4, 5

[4] Bowen Cheng, Alex Schwing, and Alexander Kirillov. Per-pixel classification is not all you need for semantic segmentation. In *NeurIPS*, 2021. 4

[5] Bowen Cheng, Ishan Misra, Alexander G Schwing, Alexander Kirillov, and Rohit Girdhar. Masked-attention mask transformer for universal image segmentation. In *CVPR*, 2022. 2, 4, 11

[6] Ho Kei Cheng and Alexander G Schwing. Xmem: Long-term video object segmentation with an atkinson-shiffrin memory model. In *ECCV*, 2022. 2, 6

[7] Marius Cordts, Mohamed Omran, Sebastian Ramos, Timo Rehfeld, Markus Enzweiler, Rodrigo Benenson, Uwe Franke, Stefan Roth, and Bernt Schiele. The cityscapes dataset for semantic urban scene understanding. In *CVPR*, 2016. 3

[8] Tri Dao and Albert Gu. Transformers are SSMs: Generalized models and efficient algorithms through structured state space duality. In *ICML*, 2024. 2

[9] Henghui Ding, Chang Liu, Shuting He, Xudong Jiang, Philip HS Torr, and Song Bai. Mose: A new dataset for video object segmentation in complex scenes. In *ICCV*, 2023. 2, 6

[10] Aaron Grattafiori, Abhimanyu Dubey, Abhinav Jauhri, Abhinav Pandey, Abhishek Kadian, Ahmad Al-Dahle, Aiesha Letman, Akhil Mathur, Alan Schelten, Alex Vaughan, et al. The llama 3 herd of models. *arXiv preprint arXiv:2407.21783*, 2024. 3, 5

[11] Albert Gu and Tri Dao. Mamba: Linear-time sequence modeling with selective state spaces. *arXiv preprint arXiv:2312.00752*, 2023. 2, 3, 4, 7

[12] Albert Gu, Karan Goel, and Christopher Ré. Efficiently modeling long sequences with structured state spaces. *arXiv preprint arXiv:2111.00396*, 2021.

[13] Albert Gu, Karan Goel, Ankit Gupta, and Christopher Ré. On the parameterization and initialization of diagonal state space models. In *NeurIPS*, 2022. 3

[14] Miran Heo, Sukjun Hwang, Seoung Wug Oh, Joon-Young Lee, and Seon Joo Kim. Vita: Video instance segmentation via object token association. In *NeurIPS*, 2022. 2, 4, 6

[15] Miran Heo, Sukjun Hwang, Jeongseok Hyun, Hanjung Kim, Seoung Wug Oh, Joon-Young Lee, and Seon Joo Kim. A generalized framework for video instance segmentation. In *CVPR*, 2023. 2, 4, 5, 6, 7

[16] Miran Heo, Min-Hung Chen, De-An Huang, Sifei Liu, Subhashree Radhakrishnan, Seon Joo Kim, Yu-Chiang Frank Wang, and Ryo Hachiuma. Omni-rgpt: Unifying image and video region-level understanding via token marks. In *CVPR*, 2025. 2, 4

[17] Miran Heo, Seoung Wug Oh, Seon Joo Kim, and Joon-Young Lee. Robust and consistent online video instance segmentation via instance mask propagation. In *AAAI*, 2025. 2, 4, 5

[18] De-An Huang, Zhiding Yu, and Anima Anandkumar. Minvis: A minimal video instance segmentation framework without video-based training. In *NeurIPS*, 2022. 1, 2

[19] Sukjun Hwang, Miran Heo, Seoung Wug Oh, and Seon Joo Kim. Video instance segmentation using inter-frame communication transformers. In *NeurIPS*, 2021. 2, 4

[20] Dahun Kim, Sanghyun Woo, Joon-Young Lee, and In So Kweon. Video panoptic segmentation. In *CVPR*, 2020. 3

[21] Dahun Kim, Jun Xie, Huiyu Wang, Siyuan Qiao, Qihang Yu, Hong-Seok Kim, Hartwig Adam, In So Kweon, and Liang-Chieh Chen. Tubeformer-deeplab: Video mask transformer. In *CVPR*, 2022. 2

[22] Hanjung Kim, Jaehyun Kang, Miran Heo, Sukjun Hwang, Seoung Wug Oh, and Seon Joo Kim. Visage: Video instance segmentation with appearance-guided enhancement. In *ECCV*, 2024. 1, 2, 6, 7

[23] Alexander Kirillov, Eric Mintun, Nikhila Ravi, Hanzi Mao, Chloe Rolland, Laura Gustafson, Tete Xiao, Spencer Whitehead, Alexander C. Berg, Wan-Yen Lo, Piotr Dollar, and Ross Girshick. Segment anything. In *ICCV*, 2023. 2

[24] Minghan Li, Shuai Li, Xindong Zhang, and Lei Zhang. Univs: Unified and universal video segmentation with prompts as queries. In *CVPR*, 2024. 1, 2, 3, 4, 6, 7

[25] Xiangtai Li, Wenwei Zhang, Jiangmiao Pang, Kai Chen, Guangliang Cheng, Yunhai Tong, and Chen Change Loy. Video k-net: A simple, strong, and unified baseline for video segmentation. In *CVPR*, 2022. 2, 6

[26] 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, 11

[27] 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. In *ICCV*, 2021. 7, 11

[28] Ilya Loshchilov and Frank Hutter. Decoupled weight decay regularization. In *ICLR*, 2019. 11

[29] Ben Mann, N Ryder, M Subbiah, J Kaplan, P Dhariwal, A Neelakantan, P Shyam, G Sastry, A Askell, S Agarwal, et al. Language models are few-shot learners. *arXiv preprint arXiv:2005.14165*, 1:3, 2020. 3, 5

[30] Tim Meinhardt, Alexander Kirillov, Laura Leal-Taixe, and Christoph Feichtenhofer. Trackformer: Multi-object tracking with transformers. In *CVPR*, 2022. 5

[31] Seoung Wug Oh, Joon-Young Lee, Ning Xu, and Seon Joo Kim. Video object segmentation using space-time memory networks. In *ICCV*, 2019. 2, 4, 7

[32] Jordi Pont-Tuset, Federico Perazzi, Sergi Caelles, Pablo Arbeláez, Alex Sorkine-Hornung, and Luc Van Gool. The 2017 davis challenge on video object segmentation. *arXiv preprint arXiv:1704.00675*, 2017. 2, 3, 6

[33] Ofir Press, Noah A Smith, and Mike Lewis. Train short, test long: Attention with linear biases enables input length extrapolation. *arXiv preprint arXiv:2108.12409*, 2021. 8

[34] 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. *IJCV*, 2022. 2, 6, 11

[35] Alec Radford, Jeffrey Wu, Rewon Child, David Luan, Dario Amodei, Ilya Sutskever, et al. Language models are unsupervised multitask learners. *OpenAI blog*, 1(8):9, 2019. 3

[36] 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 *ICML*, 2021. 11

[37] Nikhila Ravi, Valentin Gabeur, Yuan-Ting Hu, Ronghang Hu, Chaitanya Ryali, Tengyu Ma, Haitham Khedr, Roman Rädle, Chloe Rolland, Laura Gustafson, Eric Mintun, Junting Pan, Kalyan Vasudev Alwala, Nicolas Carion, Chao-Yuan Wu, Ross Girshick, Piotr Dollár, and Christoph Feichtenhofer. Sam 2: Segment anything in images and videos. In *ICLR*, 2025. 2, 6, 7

[38] Seonguk Seo, Joon-Young Lee, and Bohyung Han. Urvos: Unified referring video object segmentation network with a large-scale benchmark. In *ECCV*, 2020. 8

[39] Jacob Mitchell Springer, Suhas Kotha, Daniel Fried, Graham Neubig, and Aditi Raghunathan. Repetition improves language model embeddings. *arXiv preprint arXiv:2402.15449*, 2024. 7

[40] Ilya Sutskever, Oriol Vinyals, and Quoc V Le. Sequence to sequence learning with neural networks. In *NeurIPS*, 2014. 5

[41] Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N Gomez, Lukasz Kaiser, and Illia Polosukhin. Attention is all you need. In *NeurIPS*, 2017. 2, 3, 4, 7

[42] Paul Voigtlaender, Michael Krause, Aljosa Osep, Jonathon Luiten, Berin Balachandar Gnana Sekar, Andreas Geiger, and Bastian Leibe. Mots: Multi-object tracking and segmentation. In *CVPR*, 2019. 3

[43] Huiyu Wang, Yukun Zhu, Hartwig Adam, Alan Yuille, and Liang-Chieh Chen. Max-deeplab: End-to-end panoptic segmentation with mask transformers. In *CVPR*, 2021. 2

[44] Yuqing Wang, Zhaoliang Xu, Xinlong Wang, Chunhua Shen, Baoshan Cheng, Hao Shen, and Huaxia Xia. End-to-end video instance segmentation with transformers. In *CVPR*, 2020. 2

[45] Jason Wei, Xuezhi Wang, Dale Schuurmans, Maarten Bosma, Fei Xia, Ed Chi, Quoc V Le, Denny Zhou, et al. Chain-of-thought prompting elicits reasoning in large language models. In *NeurIPS*, 2022. 7

[46] Junfeng Wu, Qihao Liu, Yi Jiang, Song Bai, Alan Yuille, and Xiang Bai. In defense of online models for video instance segmentation. In *ECCV*, 2022. 1, 2

[47] Jiannan Wu, Yi Jiang, Bin Yan, Huchuan Lu, Zehuan Yuan, and Ping Luo. Segment every reference object in spatial and temporal spaces. In *ICCV*, 2023. 2

[48] Jiannan Wu, Yi Jiang, Bin Yan, Huchuan Lu, Zehuan Yuan, and Ping Luo. Uniref++: Segment every reference object in spatial and temporal spaces. *arXiv preprint arXiv:2312.15715*, 2023. 2, 6

[49] Guangxuan Xiao, Yuandong Tian, Beidi Chen, Song Han, and Mike Lewis. Efficient streaming language models with attention sinks. *arXiv preprint arXiv:2309.17453*, 2023. 8

[50] Ning Xu, Linjie Yang, Yuchen Fan, Dingcheng Yue, Yuchen Liang, Jianchao Yang, and Thomas Huang. Youtube-vos: A large-scale video object segmentation benchmark. *arXiv preprint arXiv:1809.03327*, 2018. 2, 6

[51] 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. 1, 2, 3, 6

[52] Linjie Yang, Yuchen Fan, and Ning Xu. Video instance segmentation. In *ICCV*, 2019. 1, 2, 3, 6, 11

[53] Zongxin Yang and Yi Yang. Decoupling features in hierarchical propagation for video object segmentation. In *NeurIPS*, 2022. 2, 6

[54] Zongxin Yang, Yunchao Wei, and Yi Yang. Associating objects with transformers for video object segmentation. In *NeurIPS*, 2021. 2, 4

[55] Kaining Ying, Qing Zhong, Weian Mao, Zhenhua Wang, Hao Chen, Lin Yuanbo Wu, Yifan Liu, Chengxiang Fan, Yunzhi Zhuge, and Chunhua Shen. Ctvis: Consistent training for online video instance segmentation. In *ICCV*, 2023. 1, 2

[56] Tao Zhang, Xingye Tian, Yu Wu, Shunping Ji, Xuebo Wang, Yuan Zhang, and Pengfei Wan. Dvis: Decoupled video instance segmentation framework. In *ICCV*, 2023. 6