SV-TAD: Native Sparse Convs for Efficient Temporal Action Detection

Ricardo Pizarro[1], Roberto Valle[2], José M. Buenaposada[3], Luis M. Bergasa[1], and Luis Baumela[2]

[1] Universidad de Alcalá, Alcalá de Henares, Spain
[2] Universidad Politécnica de Madrid, Madrid, Spain
[3] Universidad Rey Juan Carlos, Móstoles, Spain
ricardo.pizarroc@edu.uah.es

Abstract. To adapt billion-parameter Vision Transformers for long-video understanding, recent methods freeze the backbone and train lightweight convolutional modules. While effective for parameter-efficient training, existing adapters do not reduce inference-time computation, leaving scalability with respect to video length largely unaddressed. Token selection can reduce attention cost by pruning redundant tokens, but it breaks the spatial grid structure required by convolutional adapters. This forces an expensive dense reconstruction, nullifying much of the potential speedup. We address this by introducing native sparse 2D convolutions, a primitive that allows these adapters, for the first time, to operate directly and efficiently on dynamically pruned token sets. We integrate this primitive into SV-TAD, an adapter framework for temporal action detection, reducing VideoMAEv2-L computation by up to 64% and achieving 2.2× faster inference, while maintaining state-of-the-art accuracy on THUMOS-14 and ActivityNet-1.3. When scaled to InternVideoNext-L, our approach surpasses the previous state of the art at roughly half its computational cost. Moreover, the sparse formulation naturally supports auxiliary task tokens, which improves fine-grained assembly detection on ATTACH. Code and trained models are available at `https://github.com/pcr-upm/eccv26_tad`.

Keywords: Sparse Convolutions · Token Selection · Vision Transformers · Temporal Action Detection · Efficient Inference · Efficient Training

1 Introduction

Recent advances in large-scale video foundation models have significantly improved video understanding [26]. To adapt these billion-parameter Vision Transformers to temporal action detection (TAD) tasks without prohibitive fine-tuning cost, adapter-based approaches insert lightweight convolutional modules into a frozen backbone, allowing parameter-efficient specialization [16, 19]. However, despite their parameter efficiency, these adapters still operate on dense token grids and their computational cost scales proportionally with the full sequence length of the video. As video resolution and temporal duration increase, this

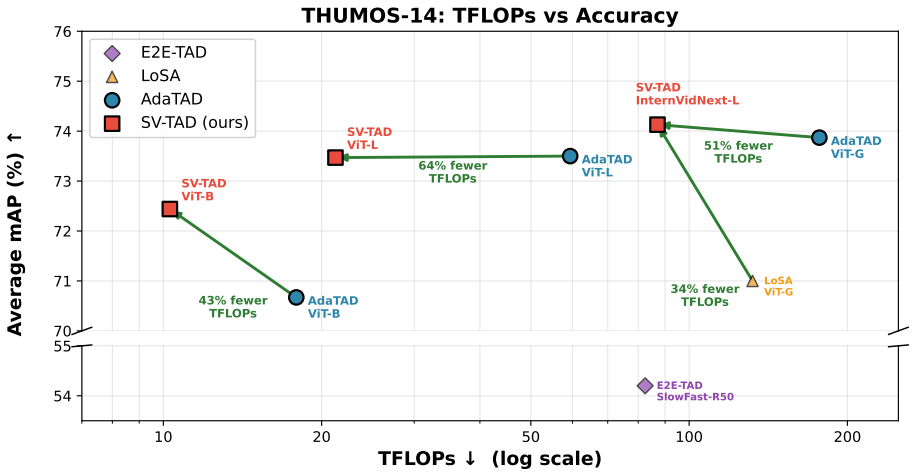


Fig. 1: TFLOPs vs. accuracy on THUMOS-14. SV-TAD (red squares) matches or exceeds AdaTAD [16] (blue circles) at substantially lower compute across three backbone scales: **43%** fewer TFLOPs with VMAEv2-B, **64%** with VMAEv2-L, and **51%** when comparing our InternVideoNext-L against AdaTAD's VMAEv2-G.

dense processing becomes a major computational bottleneck. Moreover, existing adapter-based approaches primarily improve training efficiency and do not reduce the inference-time computational cost of the backbone, leaving scalability with respect to video length largely unaddressed.

In parallel, token selection [3, 14] has emerged as an effective mechanism to reduce the quadratic cost of self-attention by dynamically pruning uninformative tokens. Unlike architectural modifications, such as efficient attention variants [23] or distillation [2], token selection preserves the original model's representations. This makes it uniquely attractive for adapter-based fine-tuning, where any reduction in token count could directly translate into wall-clock savings.

However, while sparsification substantially reduces attention complexity, it disrupts the regular spatial structure required by convolutional adapters, producing a fundamental *representational gap*. This affects convolutions, feature pyramid networks, and any operation that assumes fixed spatial neighborhoods. Existing convolutional adapters reshape tokens into spatial feature maps and apply grid-based convolutions, requiring dense reconstruction when tokens are removed. This reconstruction step substantially diminishes the computational gains of sparsification, so current adapter frameworks cannot fully benefit from token-level sparsity. Additionally, existing 3D sparse convolutional engines [6, 8] cannot be repurposed here, as their format requirements and host-side hash-table operations are architecturally incompatible with the dynamic sparsity of token selection (Sec. 2).

To address this, we introduce SparseConv2D, a *native sparse 2D convolution* that bridges token selection and convolutions without dense reconstruction. Our

key insight is that the sparse token set already contains everything a convolution needs: a convolution does not require the full spatial grid, only the adjacency between each output position and its K^2 neighbors; encoding this adjacency explicitly over the sparse set makes the grid unnecessary. We precompute a *neighbor index table* that maps each retained token to its neighbors' positions in the sparse sequence, then execute the convolution through custom CUDA kernels that gather features directly via this table. The regular 2D patch grid further allows the table to be built in $\mathcal{O}(N_{kept})$, with N_{kept} being the number of tokens selected, translating token reduction into proportional speedups in both memory and computation.

We instantiate this primitive in SparseViT-TAD (SV-TAD), a temporal action detection framework where sparse convolutional adapters are inserted into each Transformer block of a frozen ViT backbone [24, 26], while token selection modules progressively prune sequences in strategic layers. As shown in Fig. 1, SV-TAD achieves better efficiency-accuracy trade-offs than existing adapter methods [9,16]. With ViT-L, it achieves **64%** fewer TFLOPs and **2.2x** faster throughput. Our contributions are:

- We introduce **native sparse 2D convolutions** for Vision Transformers, a computational primitive that processes dynamically sparse token sequences through precomputed neighbor index tables and custom CUDA kernels, without dense grid reconstruction.
- We design **sparse adapter architectures** that integrate this primitive into parameter-efficient fine-tuning, enabling convolutional adapters to operate on token-selected sequences for the first time.
- We demonstrate **64%** reduction of TFLOPs, **2.2x** inference speedup, and **1.7x** training acceleration over the state-of-the-art on THUMOS-14, while maintaining mAP. We validate on ActivityNet-1.3 with consistent gains.
- We show that our sparse framework is **flexible enough to support multi-task learning**: in ATTACH, auxiliary landmark supervision improves over the baseline by +2.41%, demonstrating that secondary tasks can be incorporated with minimal overhead.
- We release our CUDA kernels as a **standalone library** for sparse token processing in PyTorch.

2 Related Work

Parameter-Efficient Fine-Tuning (PEFT). Large-scale ViTs, with hundreds of millions to billions of parameters, cannot be fully fine-tuned for TAD, where each input spans hundreds of frames. Adapter-based methods [11] address this by keeping the backbone weights fixed and learning only small modules inserted between layers. AIM [30] proposes spatial and temporal adapters for video recognition, while ST-Adapter [19] employs 3D convolutions to capture spatio-temporal patterns. LoSA [9] introduces long-short-range adapters and scales to VMAEv2-G, yet underperforms relative to the capacity of its billion-parameter backbone. Other methods such as ViT-TAD [29] adapt a short-term ViT-B for

clip-level processing. While this keeps per-snippet FLOPs low, ViT-TAD relies on full fine-tuning of all backbone parameters, fundamentally limiting scaling to larger backbones. AdaTAD [16] applies temporal-informative 1D convolutional adapters, surpassing both LoSA and ViT-TAD while training only $\sim$2% of parameters. A common limitation across these methods is that their convolutional modules always operate on the complete spatial grid, regardless of token informativeness. Recent image-domain work [13, 31] combines PEFT with token selection but requires explicit attention-matrix computation, incompatible with FlashAttention [7] and infeasible for TAD sequences exceeding 70k tokens.

Token Selection in Vision Transformers. The quadratic complexity of self-attention in the number of tokens N has motivated numerous strategies to alleviate this cost by reducing the number of tokens that traverse the network. EViT [14] ranks patches according to their class-token attention and discards the lowest-scoring ones, while ToMe [3] progressively merges nearby tokens using bipartite matching, eliminating the need for additional training. DynamicViT [21] instead trains a lightweight decision network that predicts per-token keep/drop masks. In video, TokenLearner [22] and STTS [25] extend pruning and merging to the spatio-temporal setting. A shared limitation is that all of these approaches target only the self-attention bottleneck. None specifies how the irregular token sets produced should interact with local operators, such as convolutions. Merging methods carry an additional constraint, as computing the $N \times N$ pairwise similarity matrix is at odds with sub-quadratic attention kernels like FlashAttention [7], limiting applicability to long sequences.

Sparse Convolutions. Processing spatially sparse data with convolutions has a long history in 3D vision. Submanifold sparse convolutions [8] and MinkowskiEngine [6] represent occupied voxels in hash tables, dispatching computation only where data exists, an approach widely adopted for LiDAR point clouds [27]. However, these engines are poorly suited to the dynamic token selection in ViTs: their COO-style sparse-tensor format requires a costly round-trip to the packed dense tensors used by a ViT at every block, and pruning positions invalidates cached kernel maps, forcing the coordinate manager to rebuild all K^d maps via hash-table probes per active site with no reuse across layers whose sparsity patterns differ. This hash-based construction is heavier than the problem requires, since ViT coordinates lie on a known regular 2D patch grid, rendering existing 3D sparse engines impractical for adapter-based video models. Separately, sparse-transformer research [5] sparsifies attention patterns rather than convolutional adapters. To our knowledge, native sparse 2D convolutions for the dynamic, layer-varying sparsity of token selection have not been explored before.

Our Proposal. Each research line reviewed above addresses one dimension of the efficiency challenge, but leaves the others unresolved. PEFT adapters [11, 16, 19, 30] always operate on the full spatial grid, so cost scales with total tokens regardless of informativeness. Token selection [3, 14, 21] reduces the count but does not specify how irregular token sets should interact with local operators,

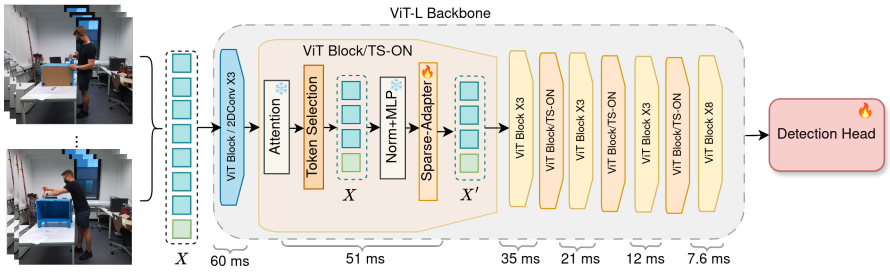


Fig. 2: SV-TAD architecture. Input frames and auxiliary tokens ($\mathbf{X}$) pass through a frozen pretrained ViT backbone with interleaved token selection (TS-ON) blocks. The first three blocks use standard 2D convolutions, with the rest using SparseConv2D. The expanded view shows our modified ViT block: frozen attention computes importance scores, token selection prunes uninformative patches, and the trainable sparse adapter operates directly on the sparse sequence. Timing annotations show per-block latency decreasing from 60ms to 7.6ms as the sequence progressively shrinks.

limiting savings to attention-only models. Existing sparse engines [6, 8, 27] are incompatible with ViT tensor formats and unsuitable for dynamic, layer-varying sparsity. SV-TAD bridges all three: sparse convolutional adapters provide PEFT-style efficiency, token selection reduces the sequence, and our native sparse 2D convolution enables both to work together, translating token reduction into proportional savings without dense reconstruction or external sparse-tensor engines.

3 Method

We present SV-TAD, a framework that bridges token selection and convolutional adapters for TAD. Central to our approach is SparseConv2D, a *native sparse 2D convolution* (Sec. 3.2), this computational primitive processes dynamically sparse token sequences without dense grid reconstruction. We integrate this primitive into a sparse adapter architecture (Sec. 3.4) and show that the resulting framework is flexible enough to support auxiliary task supervision (Sec. 3.3). Fig. 2 provides an overview.

3.1 Token Selection

We employ attention-based token selection [14, 20] to progressively prune uninformative tokens during the forward pass. Crucially, unlike prior work that discards spatial information after pruning, we *preserve the original grid indices* of kept tokens, this positional information is essential for our sparse convolution.

At designated Transformer blocks, we compute importance scores from aggregated attention weights between *auxiliary tokens* and visual tokens. In the standard setting, only the class token serves as the auxiliary token. When more

than one auxiliary task token is present (Sec. 3.3), importance scores incorporate attention from all K tokens with learnable weights $\mathbf{w} \in \mathbb{R}^K$:

$$\mathbf{s} = \frac{1}{n_h} \sum_{h=1}^{n_h} \sum_{k=1}^{K} w_k \cdot \mathbf{A}_k^{(h)}, \quad \mathcal{I}_{kept} = \mathrm{TS}\left(\mathbf{s},\ \lfloor k_r \cdot N \rfloor\right) \tag{1}$$

where TS is the token selection algorithm, N is the number of visual tokens, $\mathbf{A}_k^{(h)} \in \mathbb{R}^N$ denotes the attention weights from the k-th auxiliary token to all visual tokens at head h, n_h is the number of attention heads, and k_r is the keep rate. This formulation allows task-specific tokens (*e.g.*, auxiliary landmarks) to influence which N_{kept} patches are retained, biasing selection toward regions relevant for the auxiliary task. When K=1 (class token only), this reduces to the standard EViT formulation [14]. The tokens kept $\mathbf{X}_{vis} = \mathbf{X}_{vis}[\mathcal{I}_{kept}]$ are passed to subsequent layers, while $\mathcal{I}_{kept}$ encodes their original positions (y, x) in the spatial grid $H \times W$.

Compatibility with FlashAttention. A critical advantage of our token selection is that it requires only the K attention rows from auxiliary tokens to visual tokens, not the full $N \times N$ attention matrix. We extract the required K rows (where $K \ll N$) as a byproduct of the standard attention computation. This allows our entire pipeline, backbone attention, token selection, and sparse adapters to operate with FlashAttention enabled, which is essential for scaling to video sequences.

Following prior work on token selection [14,20,21], the selection is applied at evenly-spaced layers (blocks 4, 8, 12, 16 for ViT-L), progressively reducing the sequence length. Early layers process all tokens to build robust representations before pruning begins.

3.2 Native Sparse 2D Convolution

The straightforward way to combine token selection with convolutional adapters is *dense reconstruction*. It consists of scattering the N_{kept} sparse tokens back to their original grid positions $\mathbf{X}_{dense} \in \mathbb{R}^{T \times H \times W \times D}$, filling pruned locations with zeros, applying standard convolution, and then gathering outputs at kept positions. While functionally correct, this approach has fundamental efficiency limitations: (1) memory scales with $\mathcal{O}(THW)$ regardless of sparsity, (2) convolution computes over all THW positions including empty padding, and (3) empirically, speedup plateaus as reconstruction overhead dominates, pruning more tokens yields no additional benefits. The general incompatibility and reconstruction algorithm are explained in detail in Sec. G of the supplementary material.

We propose a native sparse 2D convolution that operates directly on sparse token sequences, $\mathbf{X_{vis}} \in \mathbb{R}^{N_{kept} \times D/4}$. The key insight is that convolution is inherently *local*, *e.g.* a 3×3 kernel only needs 9 neighbors per output, not the entire grid. We exploit this locality through an indirection table that maps sparse tokens to their sparse neighbors, enabling computation without dense reconstruction. For simplicity in the following sections, we use a 3×3 kernel. Fig. 3 shows

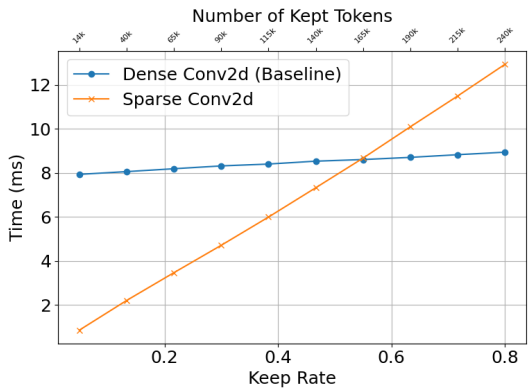


Fig. 3: Kernel Timing comparison on THUMOS-14. DenseConv2D (blue) keeps latency nearly constant across different keep rates, while our SparseConv2D (orange) scales linearly with N_{kept}.

that our sparse kernel achieves parity with dense convolution at ∼55% keep rate; below this threshold, sparse is strictly faster, reaching 8× speedup at 10% keep rate.

The data structure enabling our sparse convolution is a *neighbor index table* $\mathbf{N} \in \mathbb{Z}^{N_{kept} \times 9}$ that maps each kept token to its spatial neighbors' positions in the *sparse* token list. For kept token k at grid position (y_k, x_k) and kernel offsets $\delta_x \in \{-1, 0, 1\}$ and $\delta_y \in \{-1, 0, 1\}$:

$$\mathbf{N}[k, j] = \begin{cases} \text{sparse\_idx}(y_k + \delta_{y,j}, x_k + \delta_{x,j}) & \text{if neighbor is kept} \\ -1 & \text{if pruned or out-of-bounds} \end{cases} \tag{2}$$

where $j \in \{1, \ldots, 9\}$ indexes kernel positions and -1 indicates zero-padding.

We construct this table efficiently in two steps, exploiting the regular 2D grid structure:

1. **Build a lookup map.** Create a dense array $\mathbf{M} \in \mathbb{Z}^{H \times W}$ initialized to -1. For each kept token k at grid position (y_k, x_k), set $\mathbf{M}[y_k, x_k] = k$. This maps grid coordinates to sparse indices.
2. **Fill the neighbor table.** For each kept token k, look up its 8 neighbors via $\mathbf{N}[k, j] = \mathbf{M}[y_k + \delta_{y,j}, x_k + \delta_{x,j}]$. If the neighbor was pruned, $\mathbf{M}$ returns -1, correctly indicating zero-padding.

Both steps are $\mathcal{O}(N_{kept})$, computed on CPU with negligible overhead (less than ∼3ms) relative to the GPU-bound adapter forward pass, and $\mathbf{M}$ is reused across all adapter layers until the next token selection.

Table 1: Effect of keep rate k_r on THUMOS-14 (InternVideoNext-L backbone).

Keep Rate	TFLOPs	vid/s	mAP@0.5	Avg. mAP
0.7	87.05	0.42	78.22	74.13
0.6	74.09	0.50	77.22	73.64
0.5	60.13	0.60	76.40	73.00

Custom CUDA Kernels. We implement forward and backward passes as custom CUDA kernels. The forward pass computes:

$$\mathbf{y}_k = \sum_{j=1}^{9} \mathbf{W}_j \cdot \mathbf{x}[\mathbf{N}[k, j]] + \mathbf{b} \tag{3}$$

where $\mathbf{W}_j \in \mathbb{R}^{C_{out} \times C_{in}}$ is the weight slice for kernel position j, and $\mathbf{x}[\mathbf{N}[k, j]]$ gathers neighbor features (zeros if $\mathbf{N}[k, j] = -1$). We provide two strategies, automatically selected based on C_{out}: an implicit GEMM path via cuBLAS for large channels ($C_{out} \geq 256$) that fully utilizes Tensor Cores, and custom tiled kernels for smaller channels that fuse gather and GEMM into a single launch. Full kernel design details are in Sec. A of the supplementary.

Complexity Analysis. Our sparse convolution requires $\mathcal{O}(N_{kept} \cdot K^2 \cdot C_{in} \cdot C_{out})$ operations versus $\mathcal{O}(THW \cdot K^2 \cdot C_{in} \cdot C_{out})$ for dense convolution. With $N_{kept} = k_r \cdot THW$, compute scales linearly with keep rate k_r. Memory is decoupled from total sequence length through chunked processing, $\mathcal{O}(\text{chunk\_size} \cdot (C_{in} + C_{out}))$. The full model applies token selection at S stages (*e.g.*, $S = 4$ at blocks 4, 8, 12, 16 for ViT-L), with cumulative savings compounding across layers. At k_r=0.6 per stage, the effective rate after two stages drops to 0.36 and after four to 0.13, placing the majority of layers well below the per-kernel crossover point where sparse convolution outperforms dense (Fig. 3).

The default keep rate used for every dataset and backbone is listed in Sec. C of the supplementary.

Keep-Rate Schedule. Table 1 sweeps the per-stage keep rate k_r on THUMOS-14 with the InternVideoNext-L backbone. For this model k_r=0.7 gives the best trade-off; lowering it trades accuracy for throughput gracefully, *e.g.* k_r=0.5 reaches 0.60 videos/s (up from 0.42) at only −1.13% avg. mAP. SV-TAD thus degrades smoothly rather than collapsing as more tokens are pruned, so k_r can be tuned per deployment budget.

3.3 Auxiliary Task Supervision

We introduce auxiliary tokens, learnable embeddings that aggregate task-specific information from visual features via cross-attention. Within each adapter, after the convolution processes the bottleneck features $\mathbf{X}'_{\mathbf{vis}} \in \mathbb{R}^{B \times N_{kept} \times D/4}$, a

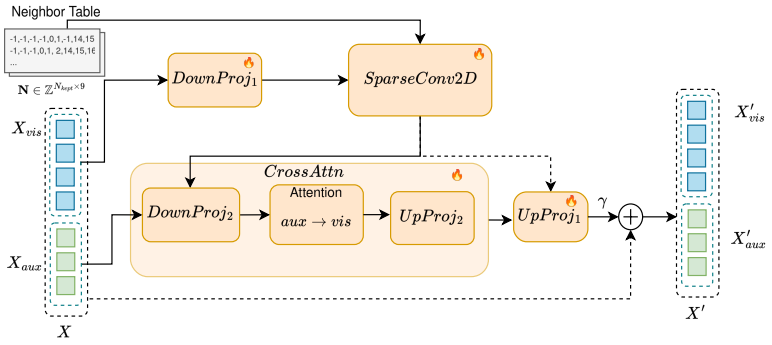


Fig. 4: Sparse adapter architecture. Our bottleneck adapter (orange, trainable; the surrounding ViT block is frozen) is inserted after each Transformer block's attention layer. Visual ($\mathbf{X}_{vis}$) and auxiliary ($\mathbf{X}_{aux}$) tokens are down-projected (DownProj_1) before SparseConv2D gathers each kept token's spatial neighbors via the precomputed neighbor index table $\mathbf{N}$ (-1 marks pruned/out-of-bounds neighbors, top-left inset). CrossAttn then lets auxiliary tokens query the sparse visual features, updating only $\mathbf{X}_{aux}$ at no extra cost to the visual path, before UpProj_1 restores the channel dimension and a learnable scalar γ modulates the adapter's residual contribution.

lightweight cross-attention module lets auxiliary tokens query visual tokens:

$$\mathbf{X}'_{aux} = \text{CrossAttn}(\mathbf{X}_{aux},\ \mathbf{X}'_{\mathbf{vis}}) \tag{4}$$

where only the auxiliary tokens $\mathbf{X_{aux}}$ are updated. In this module, the tokens are further down projected ($D/8$). The queries $\mathbf{Q}_{aux}$ are projected from the auxiliary tokens $\mathbf{X_{aux}}$, while keys $\mathbf{K}_{vis}$ and values $\mathbf{V}_{vis}$ are projected from the visual features $\mathbf{X'_{vis}}$. This asymmetric design enables information flow from visual patches to auxiliary tokens without overhead on the visual token set. The cross-attention module also serves as the adapter's mechanism for modeling temporal dynamics.

This makes our architecture flexible enough to support auxiliary task supervision without requiring per-task dense reconstructions. We demonstrate this capability with landmark-guided token selection and pose heatmap prediction, which we deploy on fine-grained assembly tasks (Sec. 4.5).

3.4 Sparse Adapter Architecture

We integrate our sparse convolution and the auxiliary cross-attention mechanism into a bottleneck adapter, named SV-TAD, inserted after each Transformer block's attention layer. As illustrated in Fig. 4, input features pass through a down-projection, GELU activation, our sparse 2D convolution, and the cross-attention module, followed by an up-projection that restores the original dimen-

sion:

$$\mathbf{X}'_{\mathbf{vis}} = \text{SparseConv2D}(\sigma(\text{DownProj}_1(\mathbf{X}_{\mathbf{vis}})),\ \mathbf{N}) \tag{5}$$

$$\mathbf{X}' = \text{Concat}(\text{CrossAttn}(\mathbf{X}_{aux},\ \mathbf{X}'_{\mathbf{vis}}), \mathbf{X}'_{\mathbf{vis}}) \tag{6}$$

$$\mathbf{X}_{out} = \gamma \cdot \text{UpProj}_1(\mathbf{X}') + \mathbf{X}, \tag{7}$$

where $DownProj_1$: $\mathbb{R}^D \rightarrow \mathbb{R}^{D/4}$, σ is GELU, SparseConv2D uses neighbor table $\mathbf{N}$, CrossAttn lets auxiliary tokens query the sparse bottleneck features $\mathbf{X}_{\mathbf{vis}}$, $UpProj_1$: $\mathbb{R}^{D/4} \rightarrow \mathbb{R}^D$, and γ is a learnable scalar (initialized to 1) that controls adapter influence. For early layers (before token selection), we use a standard 2D convolution, as our sparse convolution offers no gains in this dense regime.

Temporal Context Flow. Temporal information reaches visual tokens through two complementary pathways. First, after the adapter enriches the auxiliary tokens (*e.g.* CLS) at layer ℓ, the frozen ViT self-attention at layer ℓ+1 naturally propagates this enriched representation to every visual token across all frames. Second, the backbone output concatenates the final CLS token along with frame features that the detection head processes directly. The CLS thus serves both as an intermediate temporal relay between adapter layers and as a global temporal summary available to the detection head.

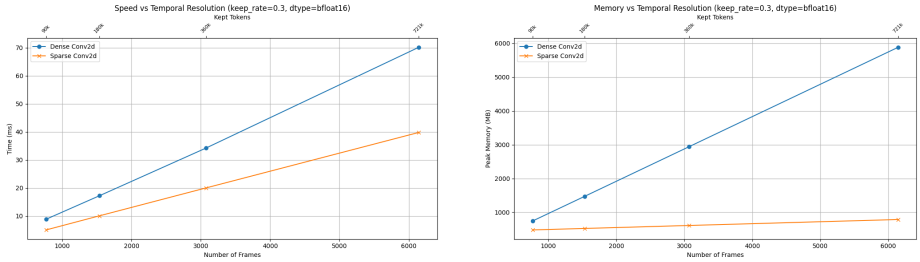
4 Experiments

We demonstrate that native sparse convolutions deliver substantial efficiency gains while maintaining accuracy. We report average mAP over IoU thresholds {0.3, 0.4, 0.5, 0.6, 0.7} for THUMOS-14, {0.5, 0.75, 0.95} for ActivityNet-1.3, and {0.1, 0.2, 0.3, 0.4, 0.5} for ATTACH, following the standard protocol for each benchmark.

4.1 Datasets and Setup

We evaluate SV-TAD on 3 benchmarks for TAD: **THUMOS-14** [12] (200/211 train/test videos, 20 sports classes), **ActivityNet-1.3** [10] (10k/4,728 train/test videos, 200 activity classes), and **ATTACH** [1] (1,200+ assembly videos, 51 fine-grained classes with 68% temporal overlap, person-split: 28/4/10 participants for train/val/test). For THUMOS-14 and ATTACH, we use $T = 768$ frames at 224×224 resolution as input. For ActivityNet-1.3, $T = 192$ at 160×160. Full dataset statistics and preprocessing details are in Sec. B of the supplementary material.

We use frozen video transformer backbones (*i.e.* VideoMAEv2 [26], InternVideoNext [24]), with the OpenTAD framework [17]. Sparse adapters are inserted after each transformer block, with token selection in blocks 4, 8, 12, 16 (ViT-L) using k_r=0.6. We employ ActionFormer [32] as the detection head. Full hyperparameters are provided in Sec. C of the supplementary material.



(a) Speed scaling **(b)** Memory scaling

Fig. 5: Sparse convolution efficiency scaling at k_r=0.3. (a) Our sparse convolution (orange) achieves **1.75**× speedup over dense (blue) at 6,144 frames (40ms vs. 70ms). (b) Dense convolution memory grows to ∼6GB at 6,144 frames regardless of sparsity. Our sparse convolution maintains **constant memory** (∼800MB) via chunked processing, achieving **7.5**× reduction.

4.2 Efficiency Analysis

We evaluate inference of SV-TAD (ViT-L) against AdaTAD [16] (ViT-L) with identical input sizes on THUMOS.

End-to-End Throughput. Figure 1 shows an overview of TFLOPs at different model sizes. SV-TAD (ViT-L) processes **1.57 videos/second** vs. the baseline's 0.73, a **2.2x** improvement: our native sparse convolution translates the **64%** TFLOPs reduction at k_r=0.6 directly into real-world throughput gains, validating the approach for both faster training and practical deployment.

Training Speed. Our efficiency gains also extend to training: measuring average GPU active time, SV-TAD requires **3.01s** per step (forward= 0.70s, backward= 2.31s) vs. the baseline's 5.35s (forward= 1.49s, backward= 3.86s), a 1.7x speedup. With ViT-L, where token selection prunes four times over a 24-layer backbone, SV-TAD requires **9.23GB** peak allocated memory per training step (single RTX 3090, batch size 1) vs. AdaTAD's **13.30GB**, a ∼ 31% reduction. Despite SV-TAD having more parameters than AdaTAD at this scale (361M vs. 346M), progressively shrinking the token count that reaches self-attention at deeper layers reduces activation memory enough to outweigh the parameter count, so the memory advantage of token pruning, like the TFLOPs advantage in Fig. 1, grows with backbone depth.

Kernel-Level Benchmarks. To verify these end-to-end gains stem from the kernel itself, we isolate the convolution module and benchmark it against dense convolution (with scatter-gather reconstruction) across temporal resolutions at k_r=0.3.

Kernel-Level Speed. Fig. 5(a) confirms both kernels scale linearly with sequence length, but ours operates on only $N_{kept} = k_r \cdot N$ tokens, yielding consistent speedups: **2**× at 768 frames (4ms vs. 8ms) and **1.75**× at 6,144 frames (40ms

Table 2: Dense vs. sparse convolution on THUMOS-14.

Conv Type	Avg.
DenseConv1D	**69.60**
DenseConv2D	68.83
SparseConv2D	69.35

Table 3: Backbone-matched comparison on THUMOS-14 with VideoMAEv2-B. All methods use identical settings. * indicates our reproduction.

Method	TFLOPs	Avg.
AdaTAD*	17.9	72.15
SV-TAD (No CrossAttn)	**10.18**	71.80
SV-TAD	10.29	**72.44**

vs. 70ms). Since sparse adapters sit at every Transformer block, these per-layer gains compound across the network.

Kernel Memory Scaling. Dense convolution materializes the full $H \times W \times C$ grid regardless of sparsity, therefore memory grows linearly with sequence length (Fig. 5(b)). Ours instead allocates memory proportional to N_{kept} within a fixed chunk size, holding peak memory near-constant and yielding a **87%** reduction at 6,144 frames (800MB vs. 6GB), enabling sequence lengths that would otherwise run out of memory.

4.3 Convolution Comparison: Dense vs. Sparse

To isolate the effect of our sparse convolution primitive from other architectural choices (*e.g.*, 1D vs. 2D convolution), we compare against dense convolution baselines using the same adapter architecture and token selection, with scatter-gather reconstruction. Only SparseConv2D bypasses reconstruction entirely. For comparison, DenseConv1D is the temporal-convolution adapter identical to AdaTAD [16]. Table 2 presents this comparison on THUMOS-14 with the VideoMAE-B backbone.

DenseConv2D vs. SparseConv2D. Both apply a 3×3 spatial convolution over the same zero-padded grid, yet SparseConv2D outperforms the dense baseline by $+$**0.52%** average mAP. We attribute the gap to numerical divergence, where implementation differences change the gradient accumulation trajectory over training. We verify that in FP32 without token selection (k_r=1), the gap shrinks to 0.10%. This shows that SparseConv2D matches dense accuracy while being substantially more efficient. A detailed analysis is provided in Sec. E of the supplementary.

DenseConv1D vs. SparseConv2D. DenseConv1D achieves a slightly higher average mAP (**69.60** vs. **69.35**, +0.25%), as its temporal kernel directly models inter-frame dynamics, consistent with AdaTAD [16]; however, this comparison is conducted under scatter-gather reconstruction, which drastically reduces the gains obtained by token selection. Implementing an efficient *native sparse* 1D temporal kernel, in contrast, is difficult in practice: temporal neighbors reside in different frame blocks, causing ∼8x larger memory access distances than spatial

Table 4: Ablation of auxiliary landmark supervision on ATTACH (Person Split) using VideoMAEv2-B.

Method	TFLOPs↓	0.1	0.2	0.3	0.4	0.5	Avg.
SV-TAD	**10.29**	20.17	18.69	16.90	14.39	11.25	16.28
SV-TAD + Kp	11.17	**22.58**	**21.10**	**19.21**	**16.80**	**13.76**	**18.69**

neighbors, which breaks cache locality and coalesced warp access (see Sec. G of the supplementary for a detailed analysis).

4.4 Backbone-Matched Comparison

To isolate the contribution of our sparse convolution framework from backbone differences, we conducted controlled experiments using the same VideoMAEv2-B backbone. Table 3 presents this fair comparison on THUMOS-14.

Analysis. The key result is the TFLOPs column: SV-TAD requires only **10.29 TFLOPs** (**43%** fewer than AdaTAD's 17.9) while *maintaining* accuracy (72.44% vs. 72.15%), confirming that native sparse convolutions eliminate reconstruction overhead without sacrificing quality and translate directly into **2.2x** faster inference and **1.7x** faster training (Sec. 4.2).

Effect of Cross-Attention for Temporal Recombination. The variant SV-TAD (No CrossAttn) already achieves the bulk of the computation reduction (10.18 TFLOPs) but drops slightly below AdaTAD in accuracy (71.80% vs. 72.15%). As shown in Table 2, replacing a temporal 1D convolution with a spatial 2D convolution sacrifices inter-frame modeling: DenseConv1D outperforms DenseConv2D by +0.77%. Our cross-attention module compensates for this: a single CLS query aggregates information across retained patches from different frames, recovering global temporal context, and closes the gap at negligible compute cost (+0.11 TFLOPs). SparseConv2D is also robust to the choice of token-selection criterion (*e.g.*, EViT-style attention vs. norm-based), at matched compute (Sec. E of the supplementary).

4.5 Auxiliary Task Tokens Experiments

Sparse representations create a structural advantage beyond efficiency: since our framework explicitly tracks which tokens survive selection and their original grid positions, it naturally enables auxiliary task supervision. Existing convolutional adapters operate exclusively on the visual feature grid, with no mechanism to incorporate task-specific tokens. We demonstrate this on ATTACH [1], a fine-grained assembly actions benchmark. Prior work on ATTACH evaluates only frame-level classification, we are the first to formulate it as a full TAD problem.

Table 4 shows that adding auxiliary landmark supervision (SV-TAD + Kp, Sec. 3.3) improves avg. mAP by +**2.41%** (18.69 vs. 16.28), demonstrating that

Table 5: Results comparison on ActivityNet-1.3 and THUMOS14 (* indicates mAP obtained from official logs). Size-matched comparisons are highlighted.

		ActivityNet-1.3					THUMOS14						
Method	**Backbone**	**TFLOPs↓**	**0.5**	**0.75**	**0.95**	**Avg.**	**TFLOPs↓**	**0.3**	**0.4**	**0.5**	**0.6**	**0.7**	**Avg.**
AFSD [15]	I3D	—	52.40	35.30	6.50	34.40	—	67.3	62.4	55.5	43.7	31.1	52.0
E2E-TAD [18]	SlowFast-R50	82.48	50.47	35.99	10.33	34.10	82.48	69.4	64.3	56.0	46.4	34.9	54.2
BasicTAD [28]	SlowOnly-R50	—	51.20	33.41	7.57	33.12	—	75.5	70.8	63.5	50.9	37.4	59.2
TALLFormer [4]	VideoSwin-B	—	54.10	36.20	7.90	35.60	—	76.0	70.0	63.2	—	34.5	59.4
Re^{2}TAD [33]	Re^{2}VSwin-T	—	54.75	37.81	9.03	36.80	—	77.0	71.5	62.4	49.7	36.3	59.4
LoSA [9]	VMAEv2-G	132	58.5	39.8	7.80	38.6	132	85.0	81.1	74.5	65.1	49.3	71.0
ViT-TAD [29]	ViT-B	—	55.87	38.47	8.80	37.40	—	85.1	80.9	74.2	61.8	45.4	69.5
AdaTAD [16]*	VMAE-B	8.05	56.66	39.49	9.38	38.31	17.90	85.58	81.40	74.31	63.04	49.00	70.67
AdaTAD [16]*	VMAE-L	27.42	57.68	40.52	**9.96**	39.18	59.40	87.11	83.65	76.84	**66.82**	**53.07**	**73.50**
AdaTAD [16]*	VMAEv2-G	83.59	57.40	40.44	9.78	39.20	176.87	87.50	84.02	77.61	**67.05**	**53.19**	73.87
SV-TAD	VMAEv2-B	**4.75**	**57.09**	**39.97**	**9.77**	**38.80**	**10.29**	**87.54**	**83.24**	**75.28**	**65.54**	**50.61**	**72.44**
SV-TAD	VMAEv2-L	**10.10**	**58.40**	**40.81**	9.76	**39.58**	**21.25**	**87.26**	**83.80**	**77.22**	66.21	52.85	73.47
SV-TAD	InternVideoNext-L	**34.00**	**58.79**	**41.30**	**10.41**	**40.06**	**87.05**	**88.38**	**84.40**	**78.22**	66.82	52.85	**74.13**

the sparse adapter's cross-attention injection point is an effective channel for multitask supervision. We provide a detailed ablation of these results in the supplementary material (Sec. F). Sec. E of the supplementary also validates that SparseConv2D generalizes beyond TAD, to image-to-video adaptation on a frozen CLIP ViT-B/16.

4.6 Comparison with State-of-the-Art

Table 5 compares SV-TAD with recent TAD methods on ActivityNet-1.3 and THUMOS-14. The TFLOPs for SV-TAD and AdaTAD [16] are for input with the same number of frames for each data set. For fair comparison, we report AdaTAD last-epoch test mAP from the authors' publicly released training logs.

In THUMOS-14, SV-TAD with VideoMAEv2-B requires **43%** fewer TFLOPs than AdaTAD [16] (10.29 vs. 17.9) while achieving **72.44%** average mAP, surpassing AdaTAD's 70.67% by **+1.77%**. We note that ViT-TAD's 69.5% with ViT-B is obtained under a substantially reduced input regime (256 frames at 160×160 vs. our 768 frames at 224×224), and its full fine-tuning design prevents scaling beyond ViT-B. Scaling to ViT-L widens the efficiency gap: SV-TAD uses **64% fewer TFLOPs** (21.25 vs. 59.4) while matching AdaTAD's accuracy (73.47% vs. 73.50%), demonstrating that our efficiency gains grow with model size.

When scaling to InternVideoNext-L, the efficiency advantage becomes most pronounced. On THUMOS-14, SV-TAD requires only **87.05** TFLOPs vs.**176.87** for AdaTAD's billion-parameter VMAEv2-G, while surpassing its 73.87% Avg. mAP by +0.26%: a Large-sized backbone with native sparse convolutions surpasses Giant-level performance at **half the computational cost**. On ActivityNet-1.3, SV-TAD with VideoMAEv2-B uses **41% fewer TFLOPs** (4.75 vs. 8.05) while achieving **38.80%** average mAP, surpassing AdaTAD (38.39%) by **+0.41%**. The most compelling result is with InternVideoNext-L: SV-TAD requires **59% fewer TFLOPs** (34 vs. 83.59) while achieving **40.06%** average mAP, surpassing AdaTAD with VMAEv2-G (39.20%) by +0.86% at every IoU threshold (Table 5). To isolate this gain from the backbone switch itself, we also

run AdaTAD with the *same* InternVideoNext-L backbone: it reaches 39.72% average mAP at 95.3 TFLOPs, confirming that, at matched backbone, SV-TAD still cuts compute substantially (34 vs. 95.3 TFLOPs) while improving accuracy (+0.34%), so the gains stem from SV-TAD itself rather than from the choice of backbone.

5 Conclusion

We introduced a native sparse 2D convolution primitive that enables convolutional adapters to operate on dynamically pruned token sets without reconstructing the dense spatial grid. This makes sparsity effective across the entire model rather than only within attention layers. Integrated into our SV-TAD framework, this design yields substantial reductions in compute and inference time while maintaining or improving detection accuracy across multiple benchmarks.

Crucially, the gains grow with model capacity: with ViT-L, we match top performing accuracy at 64% fewer TFLOPs, while with InternVideoNext-L we set a new state of the art on both THUMOS-14 and ActivityNet-1.3, achieving the best accuracy while requiring roughly half the compute (51% and 59% fewer TFLOPs, respectively). In ATTACH, auxiliary landmark supervision enabled by the sparse adapter's cross-attention injection point, sets a new state of the art and improves our baseline model by +2.41% mAP, demonstrating that the framework extends naturally to multitask learning on fine-grained assembly.

Beyond temporal action detection, our primitive is applicable wherever convolutional modules follow token selection in Vision Transformers, including spatial adapters, feature pyramid networks, and task-specific heads.

Limitations and Future Work. Our implementation targets NVIDIA GPUs; porting to other accelerators requires replacing the backend, although the core algorithm is architecture-agnostic. Additionally, the sparse kernel becomes faster than its dense counterpart only below the ∼55% keep rate (Fig. 3), so the primitive is most beneficial when progressive pruning through the layers is acceptable. Exploring learned, content-adaptive keep-rate schedules and extending the primitive to 3D spatio-temporal neighborhoods are promising directions for future work.

Acknowledgements

This work was supported by projects PID2022-137581OB-I00, PLEC2023-010343 (INARTRANS 4.0) and PID2024-161576OB-I00, funded by the Spanish MICIU/AEI/10.13039/501100011033 and FEDER, UE, co-funded by the European Regional Development Fund (ERDF, "A way of making Europe"), and from iRoboCity2030-CM project (grant TEC-2024/TEC-62), awarded by the Community of Madrid. RP, JMB, LMB and LB are members of the Madrid ELLIS Unit, funded by the Autonomous Community of Madrid, Spain.

References

1. Aganian, D., Stephan, B., Eisenbach, M., Stretz, C., Gross, H.M.: Attach dataset: Annotated two-handed assembly actions for human action understanding. In: 2023 IEEE International Conference on Robotics and Automation (ICRA). pp. 11367–11373 (2023). https://doi.org/10.1109/ICRA48891.2023.10160633
2. Beyer, L., Zhai, X., Royer, A., Markeeva, L., Anil, R., Kolesnikov, A.: Knowledge distillation: A good teacher is patient and consistent. In: IEEE/CVF Conference on Computer Vision and Pattern Recognition, CVPR 2022, New Orleans, LA, USA, June 18-24, 2022. pp. 10915–10924. IEEE (2022). https://doi.org/10.1109/CVPR52688.2022.01065, https://doi.org/10.1109/CVPR52688.2022.01065
3. Bolya, D., Fu, C., Dai, X., Zhang, P., Feichtenhofer, C., Hoffman, J.: Token merging: Your vit but faster. In: The Eleventh International Conference on Learning Representations, ICLR 2023, Kigali, Rwanda, May 1-5, 2023. OpenReview.net (2023), https://openreview.net/forum?id=JroZRaRw7Eu
4. Cheng, F., Bertasius, G.: Tallformer: Temporal action localization with a long-memory transformer. In: Avidan, S., Brostow, G.J., Cissé, M., Farinella, G.M., Hassner, T. (eds.) Computer Vision - ECCV 2022 - 17th European Conference, Tel Aviv, Israel, October 23-27, 2022, Proceedings, Part XXXIV. Lecture Notes in Computer Science, vol. 13694, pp. 503–521. Springer (2022). https://doi.org/10.1007/978-3-031-19830-4_29, https://doi.org/10.1007/978-3-031-19830-4_29
5. Child, R., Gray, S., Radford, A., Sutskever, I.: Generating long sequences with sparse transformers. arXiv preprint arXiv:1904.10509 (2019)
6. Choy, C.B., Gwak, J., Savarese, S.: 4d spatio-temporal convnets: Minkowski convolutional neural networks. In: IEEE Conference on Computer Vision and Pattern Recognition, CVPR 2019, Long Beach, CA, USA, June 16-20, 2019. pp. 3075–3084. Computer Vision Foundation / IEEE (2019). https://doi.org/10.1109/CVPR.2019.00319, http://openaccess.thecvf.com/content_CVPR_2019/html/Choy_4D_Spatio-Temporal_ConvNets_Minkowski_Convolutional_Neural_Networks_CVPR_2019_paper.html
7. Dao, T., Fu, D., Ermon, S., Rudra, A., Ré, C.: Flashattention: Fast and memory-efficient exact attention with io-awareness. Advances in neural information processing systems **35**, 16344–16359 (2022)
8. Graham, B., Van der Maaten, L.: Submanifold sparse convolutional networks. arXiv preprint arXiv:1706.01307 (2017)
9. Gupta, A., Mittal, G., Magooda, A., Yu, Y., Taylor, G.W., Chen, M.: Losa: Long-short-range adapter for scaling end-to-end temporal action localization. In: IEEE/CVF Winter Conference on Applications of Computer Vision, WACV 2025, Tucson, AZ, USA, February 26 - March 6, 2025. pp. 2092–2102. IEEE (2025). https://doi.org/10.1109/WACV61041.2025.00210, https://doi.org/10.1109/WACV61041.2025.00210
10. Heilbron, F.C., Escorcia, V., Ghanem, B., Niebles, J.C.: Activitynet: A large-scale video benchmark for human activity understanding. In: IEEE Conference on Computer Vision and Pattern Recognition, CVPR 2015, Boston, MA, USA, June 7-12, 2015. pp. 961–970. IEEE Computer Society (2015). https://doi.org/10.1109/CVPR.2015.7298698, https://doi.org/10.1109/CVPR.2015.7298698
11. Houlsby, N., Giurgiu, A., Jastrzebski, S., Morrone, B., de Laroussilhe, Q., Gesmundo, A., Attariyan, M., Gelly, S.: Parameter-efficient transfer learning for NLP. In: Chaudhuri, K., Salakhutdinov, R. (eds.) Proceedings of the 36th International

Conference on Machine Learning, ICML 2019, 9-15 June 2019, Long Beach, California, USA. Proceedings of Machine Learning Research, vol. 97, pp. 2790–2799. PMLR (2019), http://proceedings.mlr.press/v97/houlsby19a.html

12. Idrees, H., Zamir, A.R., Jiang, Y., Gorban, A., Laptev, I., Sukthankar, R., Shah, M.: The THUMOS challenge on action recognition for videos "in the wild". Comput. Vis. Image Underst. **155**, 1–23 (2017). https://doi.org/10.1016/J.CVIU.2016.10.018, https://doi.org/10.1016/j.cviu.2016.10.018
13. Kim, K., Park, J., Kim, J., Kwon, H., Sohn, K.: Faster parameter-efficient tuning with token redundancy reduction. In: IEEE/CVF Conference on Computer Vision and Pattern Recognition, CVPR 2025, Nashville, TN, USA, June 11-15, 2025. pp. 30189–30198. Computer Vision Foundation / IEEE (2025). https://doi.org/10.1109/CVPR52734.2025.02810, https://openaccess.thecvf.com/content/CVPR2025/html/Kim_Faster_Parameter-Efficient_Tuning_with_Token_Redundancy_Reduction_CVPR_2025_paper.html
14. Liang, Y., Ge, C., Tong, Z., Song, Y., Wang, J., Xie, P.: Evit: Expediting vision transformers via token reorganizations. In: The Tenth International Conference on Learning Representations, ICLR 2022, Virtual Event, April 25-29, 2022. OpenReview.net (2022), https://openreview.net/forum?id=BjyvwnXXVn_
15. Lin, C., Xu, C., Luo, D., Wang, Y., Tai, Y., Wang, C., Li, J., Huang, F., Fu, Y.: Learning salient boundary feature for anchor-free temporal action localization. In: IEEE Conference on Computer Vision and Pattern Recognition, CVPR 2021, virtual, June 19-25, 2021. pp. 3320–3329. Computer Vision Foundation / IEEE (2021). https://doi.org/10.1109/CVPR46437.2021.00333
16. Liu, S., Zhang, C., Zhao, C., Ghanem, B.: End-to-end temporal action detection with 1b parameters across 1000 frames. In: IEEE/CVF Conference on Computer Vision and Pattern Recognition, CVPR 2024, Seattle, WA, USA, June 16-22, 2024. pp. 18591–18601. IEEE (2024). https://doi.org/10.1109/CVPR52733.2024.01759, https://doi.org/10.1109/CVPR52733.2024.01759
17. Liu, S., Zhao, C., Zohra, F., Soldan, M., Pardo, A., Xu, M., Alssum, L., Ramazanova, M., Alcázar, J.L., Cioppa, A., Giancola, S., Hinojosa, C., Ghanem, B.: Opentad: A unified framework and comprehensive study of temporal action detection. In: IEEE/CVF Conference on Computer Vision and Pattern Recognition Workshops, CVPR Workshops 2025, Nashville, TN, USA, June 11-15, 2025. pp. 2625–2635. Computer Vision Foundation / IEEE (2025), https://openaccess.thecvf.com/content/CVPR2025W/PVUW/html/Liu_OpenTAD_A_Unified_Framework_and_Comprehensive_Study_of_Temporal_Action_CVPRW_2025_paper.html
18. Liu, X., Bai, S., Bai, X.: An empirical study of end-to-end temporal action detection. In: IEEE/CVF Conference on Computer Vision and Pattern Recognition, CVPR 2022, New Orleans, LA, USA, June 18-24, 2022. pp. 19978–19987. IEEE (2022). https://doi.org/10.1109/CVPR52688.2022.01938, https://doi.org/10.1109/CVPR52688.2022.01938
19. Pan, J., Lin, Z., Zhu, X., Shao, J., Li, H.: St-adapter: Parameter-efficient image-to-video transfer learning. In: Koyejo, S., Mohamed, S., Agarwal, A., Belgrave, D., Cho, K., Oh, A. (eds.) Advances in Neural Information Processing Systems 35: Annual Conference on Neural Information Processing Systems 2022, NeurIPS 2022, New Orleans, LA, USA, November 28 - December 9, 2022 (2022), http://papers.nips.cc/paper_files/paper/2022/hash/a92e9165b22d4456fc6d87236e04c266-Abstract-Conference.html

20. Pizarro, R., Valle, R., Buenaposada, J.M., Bergasa, L.M., Baumela, L.: Pose-guided token selection for the recognition of activities of daily living. Image and Vision Computing **162**, 105686 (2025). https://doi.org/https://doi.org/10.1016/j.imavis.2025.105686, https://www.sciencedirect.com/science/article/pii/S0262885625002744
21. Rao, Y., Zhao, W., Liu, B., Lu, J., Zhou, J., Hsieh, C.: Dynamicvit: Efficient vision transformers with dynamic token sparsification. In: Ranzato, M., Beygelzimer, A., Dauphin, Y.N., Liang, P., Vaughan, J.W. (eds.) Advances in Neural Information Processing Systems 34: Annual Conference on Neural Information Processing Systems 2021, NeurIPS 2021, December 6-14, 2021, virtual. pp. 13937–13949 (2021), https://proceedings.neurips.cc/paper/2021/hash/747d3443e319a22747fbb873e8b2f9f2-Abstract.html
22. Ryoo, M.S., Piergiovanni, A.J., Arnab, A., Dehghani, M., Angelova, A.: Tokenlearner: Adaptive space-time tokenization for videos. In: Ranzato, M., Beygelzimer, A., Dauphin, Y.N., Liang, P., Vaughan, J.W. (eds.) Advances in Neural Information Processing Systems 34: Annual Conference on Neural Information Processing Systems 2021, NeurIPS 2021, December 6-14, 2021, virtual. pp. 12786–12797 (2021), https://proceedings.neurips.cc/paper/2021/hash/6a30e32e56fce5cf381895dfe6ca7b6f-Abstract.html
23. Tay, Y., Dehghani, M., Bahri, D., Metzler, D.: Efficient transformers: A survey. ACM Comput. Surv. **55**(6), 109:1–109:28 (2023). https://doi.org/10.1145/3530811, https://doi.org/10.1145/3530811
24. Wang, C., Zhu, Y., Xu, Y., Yang, J., Lin, L., Yan, Z., Wang, Y., Wang, Y., Wang, L.: Internvideo-next: Towards general video foundation models without video-text supervision. arXiv preprint arXiv:2512.01342 (2025)
25. Wang, J., Yang, X., Li, H., Liu, L., Wu, Z., Jiang, Y.: Efficient video transformers with spatial-temporal token selection. In: Avidan, S., Brostow, G.J., Cissé, M., Farinella, G.M., Hassner, T. (eds.) Computer Vision - ECCV 2022 - 17th European Conference, Tel Aviv, Israel, October 23-27, 2022, Proceedings, Part XXXV. Lecture Notes in Computer Science, vol. 13695, pp. 69–86. Springer (2022). https://doi.org/10.1007/978-3-031-19833-5_5, https://doi.org/10.1007/978-3-031-19833-5_5
26. Wang, L., Huang, B., Zhao, Z., Tong, Z., He, Y., Wang, Y., Wang, Y., Qiao, Y.: Videomae V2: scaling video masked autoencoders with dual masking. In: IEEE/CVF Conference on Computer Vision and Pattern Recognition, CVPR 2023, Vancouver, BC, Canada, June 17-24, 2023. pp. 14549–14560. IEEE (2023). https://doi.org/10.1109/CVPR52729.2023.01398, https://doi.org/10.1109/CVPR52729.2023.01398
27. Yan, Y., Mao, Y., Li, B.: SECOND: sparsely embedded convolutional detection. Sensors **18**(10), 3337 (2018). https://doi.org/10.3390/S18103337, https://doi.org/10.3390/s18103337
28. Yang, M., Chen, G., Zheng, Y., Lu, T., Wang, L.: Basictad: An astounding rgb-only baseline for temporal action detection. Comput. Vis. Image Underst. **232**, 103692 (2023). https://doi.org/10.1016/J.CVIU.2023.103692, https://doi.org/10.1016/j.cviu.2023.103692
29. Yang, M., Gao, H., Guo, P., Wang, L.: Adapting short-term transformers for action detection in untrimmed videos. In: IEEE/CVF Conference on Computer Vision and Pattern Recognition, CVPR 2024, Seattle, WA, USA, June 16-22, 2024. pp. 18570–18579. IEEE (2024). https://doi.org/10.1109/CVPR52733.2024.01757, https://doi.org/10.1109/CVPR52733.2024.01757

30. Yang, T., Zhu, Y., Xie, Y., Zhang, A., Chen, C., Li, M.: AIM: adapting image models for efficient video action recognition. In: The Eleventh International Conference on Learning Representations, ICLR 2023, Kigali, Rwanda, May 1-5, 2023. OpenReview.net (2023), https://openreview.net/forum?id=CIoSZ_HKHS7
31. Yuan, X., Fei, H., Baek, J.: Efficient transformer adaptation with soft token merging. In: IEEE/CVF Conference on Computer Vision and Pattern Recognition, CVPR 2024 - Workshops, Seattle, WA, USA, June 17-18, 2024. pp. 3658–3668. IEEE (2024). https://doi.org/10.1109/CVPRW63382.2024.00369, https://doi.org/10.1109/CVPRW63382.2024.00369
32. Zhang, C., Wu, J., Li, Y.: Actionformer: Localizing moments of actions with transformers. In: Avidan, S., Brostow, G.J., Cissé, M., Farinella, G.M., Hassner, T. (eds.) Computer Vision - ECCV 2022 - 17th European Conference, Tel Aviv, Israel, October 23-27, 2022, Proceedings, Part IV. Lecture Notes in Computer Science, vol. 13664, pp. 492–510. Springer (2022). https://doi.org/10.1007/978-3-031-19772-7_29, https://doi.org/10.1007/978-3-031-19772-7_29
33. Zhao, C., Liu, S., Mangalam, K., Ghanem, B.: Re^{2}tal: Rewiring pretrained video backbones for reversible temporal action localization. In: IEEE/CVF Conference on Computer Vision and Pattern Recognition, CVPR 2023, Vancouver, BC, Canada, June 17-24, 2023. pp. 10637–10647. IEEE (2023). https://doi.org/10.1109/CVPR52729.2023.01025, https://doi.org/10.1109/CVPR52729.2023.01025