Backtrace Mamba: Reviving Critical Temporal Contexts via Hierarchical Memory Compression for Online Action Detection

Su Yan, Jiahua Li, Kun Wei, Cheng Deng*

School of Electronic Engineering, Xidian University, Xi'an, Shaanxi, China
ys@stu.xidian.edu.cn, {ljhxdu,weikunsk,chdeng.xd}@gmail.com

Abstract

Online Action Detection (OAD) requires real-time prediction of ongoing actions without access to future frames, posing challenges in balancing computational efficiency and long-term dependencies modeling. Existing methods either suffer from slow training and limited temporal receptive fields, or face high computational costs and delayed inference, lacking the capability to tackle extra-long video inputs. Thus, we present a novel Mamba-based OAD framework (MOAD) that efficiently and effectively performs OAD. The hierarchical memory mechanism is introduced to intelligently store high-value scene and action frames based on motion-aware similarity metrics, preserving essential historical knowledge in an online manner. To further reduce storage space, we design a memory quantization method to compress the stored historical features. Additionally, the temporal soft pruning strategy built upon the memory bank is proposed to dynamically remove redundant features, reducing temporal redundancy while maintaining temporal coherence. Sufficient experiments on four challenging benchmarks prove our method significantly outperforms existing methods.

Introduction

Online Action Detection (OAD) (De Geest et al. 2016) aims to identify actions in real time from streaming video without access to future frames. Since its practicality, OAD has attracted considerable attention and shows significant potential in various applications in real-world, including autonomous driving and human-robot interaction systems (Bahl et al. 2023). Unlike offline approaches (Li et al. 2025; Mu et al. 2025) utilizing the entire video sequence, OAD methods detect ongoing actions based solely on partial observation of the action. Therefore, the crucial challenge lies in *effectively* modeling long-range temporal dependencies.

Early OAD methods (Eun et al. 2020; Dey and Salem 2017; Xu et al. 2019) primarily apply RNN (Cho 2014). However, their sequential processing nature limits parallelization and causes memory decay over time. Recent approaches (Wang et al. 2021; Xu et al. 2021; Zhao and Krähenbühl 2022; Wang et al. 2023) apply Transformers (Vaswani 2017) to better capture long-range temporal structures. However, Transformers suffer from quadratic computational complexity, leading to excessive parameters and slow inference speeds, which fail to meet the real-time requirements. Therefore, a module capable of efficiently and effectively tackling long videos is essential for OAD task.

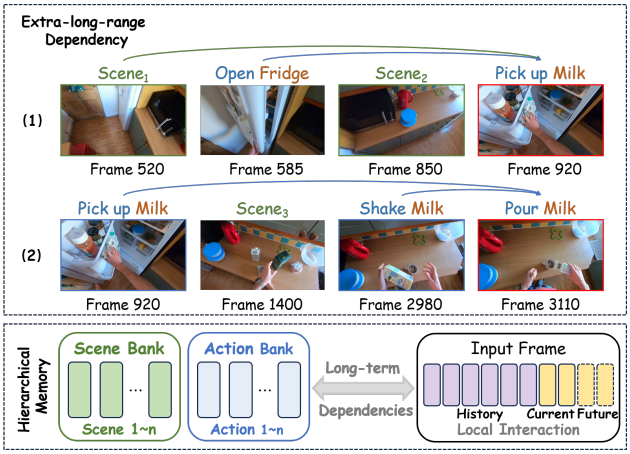

Figure 1: Examples of how extra-long-range dependencies and local features influence current action detection: (1) The ongoing action depends on both long-range scene and action features. (2) For example, the action of pouring milk relies on extra-long-range memory (over 2,000 frames), such as picking up the milk. Thus, we introduce a hierarchical memory mechanism to store high-value features, including both scenes and actions to support long-term historical dependency as a complement to local interaction.

We further analyze the influence of extra-long-range features for scene and action features. As shown in figure 1, the ongoing action depends on both scene and action features and relies on extra-long-range memory (over 2,000 frames). Therefore, more refined extraction of historical memory is critically important, which is lacking in existing methods.

Besides, we observe temporal redundancy in long sequences. In figure 2: the distribution of action frame ratios varies both within and across datasets. Some videos contain a large proportion of non-action or irrelevant frames, particularly in the THUMOS'14 dataset, where most videos contain only 30% action frames. Thus, how to use key frames and remove redundant frames is another problem for OAD.

*Corresponding author.

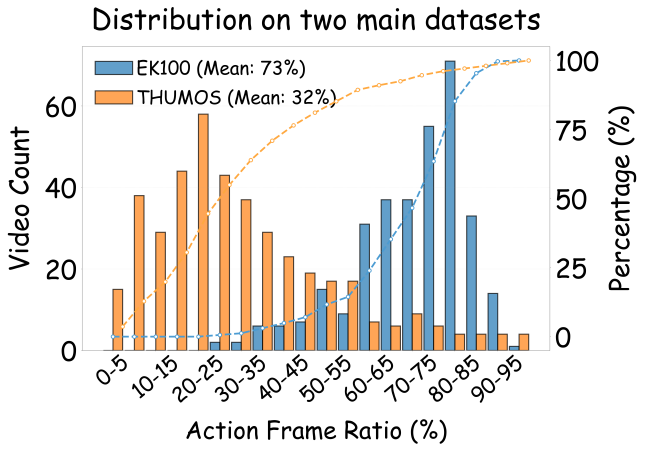

Figure 2: Action frame ratio on THUMOS'14 and EK100 datasets. The distribution of action frame ratios varies both within and across datasets.

To address the mentioned challenges above, we introduce a novel framework named **MOAD**, which tackles OAD with the trade-off between accuracy and efficiency. Based on Mamba (Gu and Dao 2023), an innovative State Space Model (SSM) to model long-range sequences with linear complexity, we propose a novel framework with a hierarchical memory mechanism to store critical historical frames.

Specifically, the hierarchical memory mechanism consists of both action and scene memory banks, which preserve the features of key frames and the surrounding ones, constraining the evolution of behaviors or events. Memory banks are updated using similarity metrics with a motion-aware trigger to more accurately capture changes in actions. We further employ feature dimensionality reduction and quantization to improve efficiency. A pruning strategy integrated with the Mamba block is proposed to reduce temporal redundancy and improve inference speed. Since direct physical pruning may disrupt the temporal coherence of input sequences, we introduce a temporal soft pruning algorithm that uses quantized features as compensatory information to replace the pruned frames.

In summary, the main contributions of this work are:

- We pioneer Mamba-based modeling for Online Action Detection, introducing MOAD—a unified module that integrates two key innovation for effective long-term dependencies modeling.
- A hierarchical memory mechanism and a temporal soft-pruning algorithm are introduced to precisely model critical scene and action memories and remove temporal redundancy.
- The proposed method is tested on 4 challenging datasets and proves effectiveness and efficiency with 5% accuracy improvement and 4% computation burden.

Related Work

Online Video Understanding. Online Action Detection (OAD) (De Geest et al. 2016) aims to predict ongoing actions in real-time by utilizing information from past frames up to the current moment, without accessing future frames. Early approaches primarily relied on Recurrent Neural Networks (RNNs) (Cho 2014) such as TRN (Xu et al. 2019) and IDN (Eun et al. 2020). While MiniROAD (An et al. 2023) introduced a RNN-based architecture that effectively addressed the training-inference discrepancy to tackle the task. Recently, transformer-based architectures (Wang et al. 2021; Xu et al. 2021; Zhao and Krähenbühl 2022; Kim, Kang, and Kim 2022; Rangrej et al. 2023; Wang et al. 2023; Cao et al. 2023; Reza et al. 2025; Guermal et al. 2024; Xu et al. 2024b,a) become the mainstream. For example, OadTR (Wang et al. 2021) employed a transformer encoder-decoder architecture to capture long-term temporal structure through self-attention, while LSTR (Xu et al. 2021) improved performance by introducing a two-stage feature compression framework. TeSTra (Zhao and Krähenbühl 2022) improved computational efficiency by integrating temporal smoothing kernels. MAT (Wang et al. 2023) introduced a novel paradigm to model the entire temporal structure. While HAT (Reza et al. 2025) enhances the synergy between long-term and short-term information by integrating historical context.

Mamba for Vision. The evolution of sequence modeling has a significant impact on computer vision, with models like ViT (Dosovitskiy et al. 2020) successfully adapted for visual tasks. Gu et al. (Gu, Goel, and Ré 2021) proposed the Structured State-Space Sequence (S4) model, which have demonstrated strong potential in modeling long-range dependencies with only a linear increase in computational cost. Recently, State Space Models (SSMs)—particularly Mamba (Gu and Dao 2023) emerged as competitive alternatives for long-sequence modeling due to their linear computational complexity. Mamba-based architectures (Huang et al. 2024; Li, Singh, and Grover 2024; Yang et al. 2024) demonstrate state-of-the-art performance across major vision benchmarks. Vim (Zhu et al. 2024) introduces a new vision backbone incorporating bidirectional Mamba blocks, enhancing spatially-aware processing. Similarly, VMamba (Liu et al. 2024) integrates Visual State-Space (VSS) and 2D Selective Scan modules to collect contextual information from multiple perspectives by bridging 1D and 2D data structures. MambaVision (Hatamizadeh and Kautz 2025) enhances long-range spatial dependency capture by integrating Mamba with Vision Transformers (ViTs). And EfficientVMamba (Pei, Huang, and Xu 2025) combines an atrous-based selective scan method to capture both global and local representative features.

Preliminaries

Problem Formulation

In Online Action Detection (OAD), the objective is to predict ongoing actions from a video stream in real time without access to future frames. Given a video stream $\mathcal{V} = \{\mathbf{v}_t\}_{t=0}^{T}$, where $\mathbf{v}_t$ denotes the video frame at time step t, the goal is to predict the action instances at each time step t. Let $\mathcal{Y} = \{y_1, y_2, \ldots, y_k\}$ denote the set of predefined action categories. The task can be formally described as learning a

function f that maps a sequence of frames to an action label $y_t \in \mathcal{Y}$ at time step t:

$$f(\{\mathbf{v}_1, \mathbf{v}_2, \ldots, \mathbf{v}_t\}) \rightarrow y_t. \tag{1}$$

State Space Models (S4)

SSMs are based on continuous systems that map one-dimensional input sequences $x(t) \in \mathbb{R}^L$ to outputs $\mathrm{y}(t) \in \mathbb{R}^L$ through a hidden state $\mathrm{h}(t) \in \mathbb{R}^N$:

$$\begin{aligned} h'(t) &= Ah(t) + Bx(t), \\ y(t) &= Ch'(t), \end{aligned} \tag{2}$$

with matrix $A(t) \in \mathbb{R}^{N\times N}$, $B(t) \in \mathbb{R}^{N\times 1}$ and $C(t) \in \mathbb{R}^{1\times N}$. Models like S4 and Mamba use an ordinary differential equation to model input data, Mamba discretizes continuous systems by using a timescale parameter Δ, converting parameters through zero-order hold (ZOH) as follows:

$$\begin{aligned} h(t) &= \exp(\Delta\mathbf{A})h(t-1) + \\ &\quad (\Delta\mathbf{A})^{-1}(\exp(\Delta\mathbf{A}) - \mathbf{E}) \cdot \Delta\mathbf{B}x(t), \\ y(t) &= \mathbf{C}h(t), \end{aligned} \tag{3}$$

where matrix $\mathbf{A}$ defines the evolution of the hidden state, $\mathbf{B}$ and $\mathbf{C}$ control how input x_t influences the state h_t and how the state affects the output y_t. The step size Δ determines the input discretization resolution of SSMs, which forms the foundation for heuristic gating mechanisms.

Inspired by S5 (Smith, Warrington, and Linderman 2022), Mamba introduced *parallel scan algorithm* rather than convolution to perform model iteration, reducing the time complexity to $O(n/t)$:

$$\begin{aligned} \bar{K} &= (\mathbf{C}\bar{\mathbf{B}}, \mathbf{C}\bar{\mathbf{A}}\bar{\mathbf{B}}, \ldots, \mathbf{C}\bar{\mathbf{A}}^{\mathbf{L-1}}\bar{\mathbf{B}}), \\ y &= x * \bar{K}. \end{aligned} \tag{4}$$

Method

Figure 3 presents our proposed architecture, which consists of three core components. The first part combines the extracted features with positional encoding, completing the data preprocessing. Next, the hierarchical memory bank is managed and updated through feature quantization and similarity metrics. Finally, the temporal soft pruning algorithm is introduced to reduce temporal redundancy, retain key information. We now introduce the details of each component.

Data Preprocessing

We use each input video features extracted by a pre-trained feature encoder (e.g., I3D (Carreira and Zisserman 2017)) as the input of our model. At each time step t, the feature can be represented as $\mathbf{F} = \{f_t\}_{t=-L+1}^{0} \in \mathbb{R}^{l\times d}$, where l and D are the length and dimension of the feature sequence, respectively (Xu, Yan, and Deng 2025). To be more specific, the feature sequence is concatenated by RGB feature and optic flow feature, $\mathbf{F} = [\mathbf{F}_{\text{RGB}}, \mathbf{F}_{\text{flow}}]$, where $[\cdot,\cdot]$ denotes concatenation along the feature dimension.

To better capture different temporal features, following (Xu et al. 2021), the feature sequence is divided into two continuous and non-overlapping memory segments: historical features $\mathbf{F}_h = \{f_t\}_{t=-l_c-l_h+1}^{-l_c}$ and current features $\mathbf{F}_c = \{f_t\}_{t=-l_c+1}^{0}$, with $l_c \ll l_h$. Since the feature extractor lacks order information, we incorporate sinusoidal position encoding $E_{\text{pos}} \in \mathbb{R}^{l\times d}$:

$$\tilde{\mathbf{F}} = \text{Linear}(\mathbf{F}) + E_{\text{pos}} \tag{5}$$

where $\tilde{\mathbf{F}}$ represents the position-augmented feature. For simplicity, we omit channel indices in this formulation.

Hierarchical Memory Mechanism (HMM)

To enable fine-scale memory management, we introduce a hierarchical memory mechanism which decomposes stored frames into discrete action (n_a slots) and scene (n_s slots) feature banks, enabling controlled memory consumption. It first performs feature dimension reduction, compression and quantization to minimize storage requirements. Subsequently, we employ similarity metrics to identify and retain high-value frames. While a motion-aware change detection module is designed to precisely update the action bank.

Dimensionality reduction and quantization First, we employ a Mamba module to perform global information modeling (Dong et al. 2022) on the input sequence $\tilde{\mathbf{F}} = \tilde{\mathbf{f}}_{-l+1}, \tilde{\mathbf{f}}_{-l+2}, \ldots, \tilde{\mathbf{f}}_0\} = \{\tilde{\mathbf{f}}_1, \tilde{\mathbf{f}}_2, \ldots, \tilde{\mathbf{f}}_l\}$, obtaining coarse-grained features:

$$\mathbf{X}_i = \text{Mamba}(\tilde{\mathbf{F}}_i), \quad \forall i \in \{1, 2, \ldots, l\} \tag{6}$$

where $\mathbf{X} = \{\mathbf{x}_1, \mathbf{x}_2, \ldots, \mathbf{x}_L\} \in \mathbb{R}^{l\times d}$ is the output feature sequence. To reduce computational overhead and enhance inference efficiency, we compress features before storing them in the memory bank through dimensionality reduction. PCA is applied to each frame feature:

$$\mathbf{z}_i = \mathbf{W}^T(\mathbf{X}_i - \boldsymbol{\mu}) \tag{7}$$

where $\mathbf{W} \in \mathbb{R}^{d\times k}$ is the PCA projection matrix and $\boldsymbol{\mu}$ is the mean vector. This yields compressed feature sequence $\mathbf{Z} = \{\mathbf{z}_1, \mathbf{z}_2, \ldots, \mathbf{z}_l\} \in \mathbb{R}^{l\times k}$ with reduced dimension k.

To further minimize storage requirements and accelerate similarity computations, we quantize the compressed features $\mathbf{Z}$ using a discrete codebook. This involves:

1. **Codebook Construction**: Learning m centroids $\mathcal{C} = \{\mathbf{c}_1, \mathbf{c}_2, \ldots, \mathbf{c}_m\} \subset \mathbb{R}^{m\times k}$;
2. **K-Means Optimization**: Optimizing centroid positions to minimize feature quantization error:

$$\min_{\mathcal{C}} \sum_{i=1}^{n} \min_{j\in\{1,\ldots,m\}} \|\mathbf{z}_i - \mathbf{c}_j\|_2^2;$$

3. **Feature Mapping**: Encoding features via nearest-neighbor assignment:

$$q_i = \underset{j\in\{1,\ldots,m\}}{\arg\min} \|\mathbf{z_i} - \mathbf{c}_j\|_2,$$

where $\mathcal{C}$ represents the codebook of centroids, and q_i is the index of the nearest centroid to the feature $\mathbf{z}_i$. In the vector quantization stage, instead of storing the entire feature vector, we only store the integer index q_i, which reduces the storage space to $\log_2 m$ bits, where m is the number of centroids in the codebook.

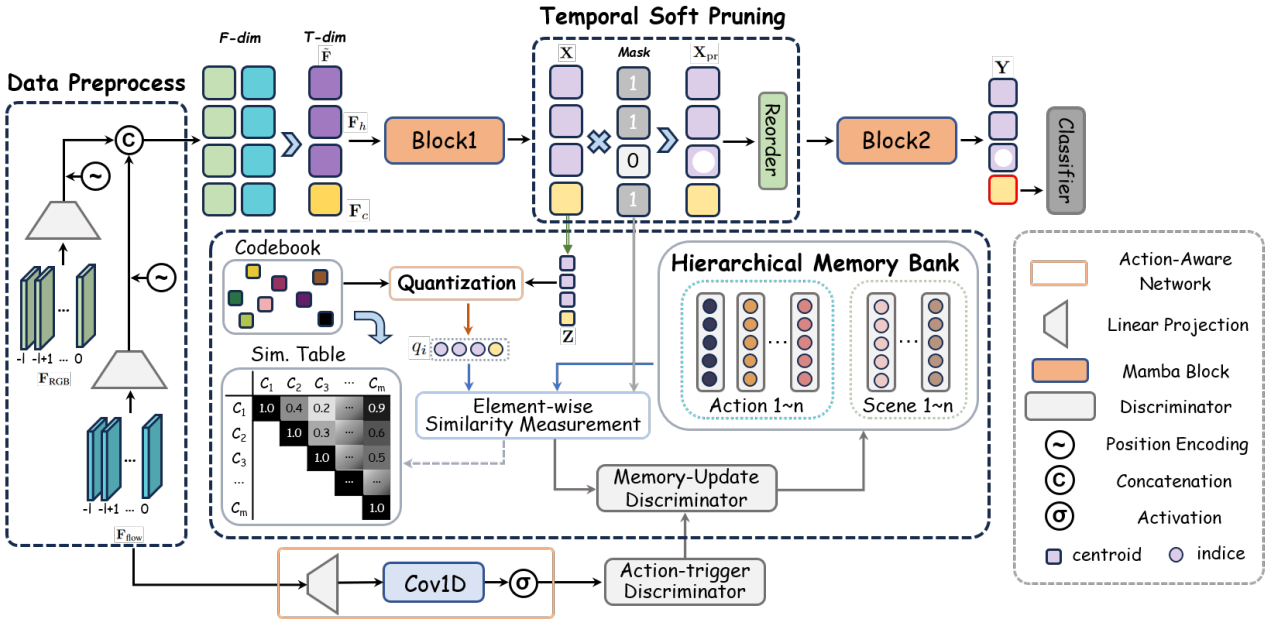

Figure 3: The proposed MOAD architecture. We first process the extracted RGB and optical flow features through a linear layer with positional encoding, then concatenate them as the input to the first block. The output undergoes feature quantization. By computing the similarity between the quantized features and a learnable codebook with an auxiliary motion-aware network, we determine the update triggers for the hierarchical memory bank. Based on feature similarity, a mask matrix is generated to prune redundant frames. Finally, the output is passed to a classifier for action detection.

Similarity Measurement We employ a similarity-based selection criterion for memory bank updates (Dong et al. 2024). Following feature quantization to discrete indices, we compute the cosine similarity between each frame's assigned centroid vector in the codebook:

$$\text{sim}(q_x, q_m) = \cos(\mathbf{c}_{q_x}, \mathbf{c}_{q_m}) = \frac{\mathbf{c}_{q_x} \cdot \mathbf{c}_{q_m}}{\|\mathbf{c}_{q_x}\| \|\mathbf{c}_{q_m}\|} \tag{8}$$

q_x and q_m denote the quantized indices of the input frame feature and a memory bank-stored feature, respectively, $\mathbf{c}_{\mathbf{q_x}}$ and $\mathbf{c}_{\mathbf{q_m}}$ represent their corresponding codebook centroids.

To further accelerate the inference stage, we can precompute the similarity matrix of codebook centroids $S \in \mathbb{R}^{m \times m}$, where $S_{ij} = \cos(\mathbf{c_i}, \mathbf{c_j})$. In this way, for feature similarity computation S_t at each time step, we can directly refer to the precomputed table, eliminating the need for real-time computation. Thus, at each time step the similarity-based update flag $S(t) = \min\left(\text{sim}(q_{x_i}, q_{m_k})\right)$.

To better and more accurately capture the action categories, we have designed an action-aware network as auxiliary information:

$$\mathbf{X}_{flow} = \sigma(\text{Cov1d}(\text{Linear}(\mathbf{F}_{flow}))) \in \mathbb{R}^{l \times c}, \tag{9}$$

where Cov1d represents the Causal Depthwise Conv1d, which can effectively preserve the temporal information of 1-D time sequences. It also prevents future information from affecting current anticipation. And σ is the SiLU activation function (Ramachandran, Zoph, and Le 2017). We employ a dynamic threshold mechanism to precisely determine which frames should be retained or updated, thereby enhancing both inference efficiency and accuracy. First, an energy signal $E(t)$ is calculated for each timestep:

$$E(t) = \frac{1}{C} \sum_{c=1}^{C} |M(t, c)| \quad \forall t \in [1, L], \tag{10}$$

where C is the number of channels. An adaptive threshold $\tau(t)$ is computed using local statistics of the energy signal:

$$\tau(t) = \mu_{E(t)} + \alpha \cdot \sigma_{E(t)}, \tag{11}$$

with μ_E being the temporal mean of the energy signal, σ_E its temporal standard deviation, and α a scaling factor controlling the threshold sensitivity. The action-based update flag $a(t)$ is determined by comparing the energy signal against this dynamic threshold:

$$a(t) = \begin{cases} 1 & \text{if } E(t) > \tau(t), \\ 0 & \text{otherwise.} \end{cases} \tag{12}$$

The final update procedure incorporates dual criteria: First, similarity-based update flag $S(t)$ must fall below a predefined threshold θ, indicating novel event occurrence. When this condition is met, the update flag $a(t)$ determines the target memory bank:

$$M(t) = \begin{cases} \text{ActionBank} & \text{if } S(t) < \theta \wedge u(t) = 1, \\ \text{SceneBank} & \text{if } S(t) < \theta \wedge u(t) = 0, \\ \text{NoUpdate} & \text{if } S(t) \geq \theta, \end{cases} \tag{13}$$

where $S(t)$ denotes the cosine similarity between current features and memory prototypes, θ is the novelty threshold,

and $\wedge$ represents the logical AND operator. For each current timestep t, we store the corresponding indexs as one memory prototype: $m_i = \{q_{-n}, \ldots, q_0, \ldots, q_n\}$.

Temporal Soft Pruning (TSP)

To reduce temporal redundancy and computational cost while maintain temporal coherence, we propose a temporal soft pruning algorithm to adaptively mask some frames based on the memory bank. For each input sequence, we employ memory similarity as the core criterion for pruning decisions, replacing the generic predictor P from DynamicViT (Rao et al. 2021). The process maintains a binary mask $M \in \{0,1\}^L$ for the sequence. Given the feature sequence $X \in \mathbb{R}^{L\times D}$, the maximum similarity between each frame and the memory prototypes is computed as:

$$\Gamma_i = \max_{k\in\mathcal{M}} \left(\text{sim}(q_{x_i}, q_{m_k})\right) \quad \forall i \in [1, l], \tag{14}$$

where the similarity vector $\Gamma \in \mathbb{R}^L$ measures frame relevance to stored memory. The retention probability ρ_i is derived from Γ_i via a sharpening transformation:

$$\begin{aligned} \rho_i &= \frac{e^{\gamma\Gamma_i}}{e^{\gamma\Gamma_i} + e^{\gamma(1-\Gamma_i)}}, \\ \Pi &= \text{Softmax}\left([\boldsymbol{\rho} \quad \mathbf{1}-\boldsymbol{\rho}]\right) \in \mathbb{R}^{L\times 2}, \end{aligned} \tag{15}$$

where γ controls the sharpness of the decision boundary, with $\pi_{i,0}$ representing retention probability and $\pi_{i,1}$ corresponding to pruning probability. Then, the mask matrix $\hat{M} \in \{0,1\}^L$ is formulated as :

$$\hat{M} = \text{Gumbel-Softmax}(\Pi), \tag{16}$$

we adopt the gumbel-softmax (Jang, Gu, and Poole 2016) trick to make the sampling process differentiable, therefore enabling the end to-end training. To enforce temporal coherence, the final mask is refined as:

$$M_i = \begin{cases} 1 & \text{if } \hat{M}_i = 1, \\ 1 & \text{if } \hat{M}_i = 0 \wedge \left(\hat{M}_{i-1} = 1, \vee \hat{M}_{i+1} = 1\right) \\ 0 & \text{otherwise.} \end{cases} \tag{17}$$

Given the per-frame classification requirement , physical frame mask would compromise temporal indexing. We compensate for pruned positions by incorporating memory similarity vectors as lightweight semantic representations, implementing a soft-pruning approach. The final pruned feature matrix is constructed as:

$$\mathbf{X}_{\text{pr}} = \mathbf{X} \odot M_i + \Gamma \otimes (1 - M_i), \tag{18}$$

where $\odot$ denotes element-wise multiplication and $\otimes$ represents the tensor product. This formulation preserves H_{pr} for direct compatibility with classification heads.

Next, the pruned features $\mathbf{X}_{\text{re}}$ are reordered as $\mathbf{X}' = [\mathbf{x_1}, \mathbf{x_2}, \ldots, \mathbf{x}_{l'}, \gamma_1 \otimes \mathbf{1}_D, \gamma_2 \otimes \mathbf{1}_D, \ldots]^\top$ and fed into the second Mamba block to generate the final output $\mathbf{Y}$. While We maintain temporal consistency by sharing matrix $\mathbf{A}$ between the two SSMs:

$$\begin{aligned} \mathbf{Y}_1 &= \mathbf{C}_1 \left(\bar{\mathbf{A}}^i \bar{\mathbf{B}}_1 \mathbf{x}_1 + \bar{\mathbf{A}}^{i-1} \bar{\mathbf{B}}_1 \mathbf{x}_2 + \cdots\right), \\ \mathbf{Y}_2 &= \mathbf{C}_2 \left(\bar{\mathbf{A}}^i \bar{\mathbf{B}}_2 \mathbf{x}_1 + \bar{\mathbf{A}}^{i-1} \bar{\mathbf{B}}_2 \mathbf{x}_2 + \cdots\right), \\ \mathbf{Y} &= g(\mathbf{Y}_1, \mathbf{Y}_2), \end{aligned} \tag{19}$$

where $g(\cdot)$ merges the two channel outputs.

Training

Then we take the current l_c of the final output $\mathbf{Y}_c \in \mathbb{R}^{l_c\times k}$ into to a classifier to generate the predicted action probabilities $\widehat{\mathbf{Y}}_c = \{\hat{y}_{c,k}\}_{c=1}^{l_c}$. Finally, we conduct the empirical cross entropy loss between the predicted action probability distribution $\hat{y}_{c,k}$ at time c across all K classes and the ground truth action label $\{g_{c,k}\}_{c=1}^{l_c}$ as:

$$\mathcal{L}_c = -\sum_{c=1}^{l_c}\sum_{k=0}^{K} g_{c,k} \log(\hat{y}_{c,k}) \tag{20}$$

where $\mathcal{L}_c$ is the final loss function, and K and l_c are the number of all action classes and the length of the current clip, respectively.

Inference

To accelerate inference phase and keep training-inference consistency, we reorder the pruned feature matrix with a permutation matrix $\mathbf{P}$:

$$\mathbf{X}' = \mathbf{P}\mathbf{X}_{\text{pr}} = \begin{bmatrix} \mathbf{X}_{\text{re}} \\ \Gamma_{\text{pr}} \otimes \mathbf{1}_D \end{bmatrix}, \tag{21}$$

where X_{re} is the matrix of features for the retained frames, $\Gamma_{\text{pruned}} \otimes \mathbf{1}_D$ represents the similarity compensations for pruned frames. Then, the output Y can be formulated as:

$$\begin{aligned} \mathbf{Y} = \mathbf{X}'\mathbf{W} &= \begin{bmatrix} \mathbf{X}_{\text{re}} \\ \Gamma_{\text{pr}} \otimes \mathbf{1}_D \end{bmatrix} \mathbf{W} \\ &= \sum_{i\in\text{re}} \mathbf{x}_i \mathbf{W} + \sum_{j\in\text{pr}} \gamma_j \times \mathbf{1}_D^\top \mathbf{W}, \end{aligned} \tag{22}$$

where $\mathbf{1}_D^\top W$ is the column sum of the matrix. Thus, the computation for retained frames is a standard matrix multiplication, while for pruned frames, it is a scalar multiplication (complexity $O(1)$).

Experiments

Datasets and Metrics

Datasets. Following prior works, we evaluate TIM on four accessible datasets. *THUMOS'14* (Idrees et al. 2017) includes over 20 hours of sports videos, each annotated with 20 actions. Following previous studies (Wang et al. 2023; An et al. 2023), we train our model on the validation set, which consists of 200 untrimmed videos, and evaluate it on the test set, comprising 213 videos. *TVSeries* (De Geest et al. 2016) features 27 episodes from 6 popular TV series, each approximately 150 minutes long, amounting to nearly 16 hours of content. The videos in this dataset describe 30 realistic actions, each occurring at least 50 times, set against a wide variety of backgrounds. *EPIC-Kitchens-100* (Damen et al. 2022) dataset comprises 700 long videos, annotating nearly 90k segments of fine-grained actions, with a total duration of 100 hours. Narrations are categorized into 97 verb classes and 300 noun classes. *FineAction* (Liu et al. 2022) is a challenging dataset featuring densely annotated instances and diverse fine-grained human actions. It contains 103,324 temporal instances across 106 action categories, derived from 16,732 untrimmed videos, totaling 705 hours of footage.

Method	Modality	Overall			Unseen			Tail		
		Verb	Noun	Action	Verb	Noun	Action	Verb	Noun	Action
LSTR	RGB + OF	39.6	44.1	22.6	34.3	35.3	18.7	39.0	41.6	20.7
TesTra		39.7	45.6	25.1	36.3	37.2	19.2	39.0	42.4	22.2
MAT		44.5	48.3	26.3	39.9	38.1	20.3	43.4	46.6	23.7
Ours		47.1	50.5	**28.4**	42.6	40.0	**21.7**	45.7	48.8	**25.4**

Table 1: Online action detection performances on EPIC-Kitchens-100 (Damen et al. 2022). Accuracy is measured by class-mean recall@5(%). Data augmentation is operated on all models.

Evaluation Protocols Following previous works (Wang et al. 2023; An et al. 2023; Zhao and Krähenbühl 2022; Xu et al. 2021), we measure the performance of our method with per-frame *mean Average Precision (mAP)* on THUMOS'14, EPIC-Kitchens-100 and FineAction dataset, and per-frame *mean calibrated Average Precision (mcAP)* on TVSeries. The *mAP* is widely used, which needs to calculate the *Average Precision (AP)* for each class with precision and recall values. While the *mcAP*, introduced by (De Geest et al. 2016) with the TVSeries dataset, was proposed to address the imbalance between positive and negative samples. Similar to *AP*, it is calculated as:

$$\text{cPrec} = \frac{\alpha \cdot \text{TP}}{\alpha \cdot \text{TP} + \text{FP}}, \quad \text{cAP} = \frac{\sum_i \text{cPrec}(i) \times I(i)}{\sum \text{TP}}, \tag{23}$$

where $I(i)$ is 1 if frame i is a true positive, and α equal to the ratio between negative and positive frames.

Implementation Details

Following prior works, we conduct our experiments using pre-extracted features. For *TVSeries* and *THUMOS'14*, videos are resampled at 24 FPS, and frames are extracted at 4 FPS for training and validation. We adopt a two-stream network (Xu et al. 2019) pretrained on ActivityNet v1.3 (Caba Heilbron et al. 2015) or Kinetics-400 (Kay et al. 2017), where RGB features adopts ResNet (He et al. 2016) and optical flow fields uses BN-Inception (Ioffe 2015). On *EPIC-Kitchens-100*, we preprocess the videos at 30 FPS and fine-tune the two-stream TSN on the classification task with ImageNet-pretrained parameters. For *FineAction* Dataset, we officially use two separate I3D (Carreira and Zisserman 2017) models pretrained on Kinetics as the feature extractor, following (An et al. 2023).

Specifically, we configure the size of hierarchical memory bank. For THUMOS'14, the action bank maintains 16 memory slots each storing a 10-frame temporal window centered on a key frame. While the scene bank employs 4 memory slots with extended 30-frames, accommodating scene features' lower temporal volatility.

Comparisons with State-of-the-Art Methods

Online Action Detection We first report the results for online action detection on the EPIC-Kitchens-100 dataset in Tab. 1. Following (Zhao and Krähenbühl 2022) and (Wang et al. 2023), all methods are conducted with data augmentation. Our approach achieves new state-of-the-art performance across all metrics, particularly excelling in action-level recognition. Most notably, we observe significant gains in action recall for Overall (+2.1%), Unseen (+1.4%), and Tail (+1.7%) categories, demonstrating our model's superior capability in modeling comprehensive action semantics under diverse conditions.

As shown in Tab. 2, our model achieves significant accuracy improvements over both transformer-based and RNN-based models. On THUMOS'14, it attains gains of 2.2% and 2.7% with ActivityNet and Kinetics (Kinet.) features respec-

Method	Arch	THUMOS'14		TVSeries	
		ANet	Kinet.	ANet	Kinet.
RED (Gao, Yang, and Nevatia 2017)	RNN	45.3	-	79.2	-
TRN (Xu et al. 2019)	RNN	47.2	62.1	83.7	86.2
IDN (Eun et al. 2020)	RNN	50.0	60.3	84.7	86.1
MiniROAD (An et al. 2023)	RNN	69.3	71.8	88.5	89.6
OadTR (Wang et al. 2021)	Trans	58.3	65.2	85.4	87.2
LSTR (Xu et al. 2021)	Trans	65.3	69.5	88.1	89.1
GateHUB (Chen et al. 2022)	Trans	69.1	70.7	88.4	89.6
TeSTra (Zhao and Krähenbühl 2022)	Trans	68.2	71.2	-	-
MAT (Wang et al. 2023)	Trans	70.4	71.6	88.6	89.7
JOADAA (Guermal et al. 2024)	Trans	-	72.6	-	-
BEDL (Guo, Wang, and Ji 2024)	Trans	70.6	72.7	88.9	90.1
HAT† (Reza et al. 2025)	Trans	70.2	71.4	-	-
Ours	**Mamba**	**72.4**	**74.3**	**89.8**	**91.2**

Table 2: Comparison of online action detection performance evaluated in (mAP %) for THUMOS'14 and (mcAP %) for TVSeries, with feature extracted from both Kinetics and ANet pretrained models.

Method	mAP (%)
OadTR(Wang et al. 2021)	31.8
LSTR(Xu et al. 2021)	31.6
TeSTra(Zhao and Krähenbühl 2022)	31.2
MiniROAD(An et al. 2023)	37.1
Ours	**41.2**

Table 3: Comparison with other OAD methods on FineAction dataset (Liu et al. 2022).

Dim.	FLOPs	Inf.	mAP
32	1.165	15	71.2
48	1.358	21	74.3
64	1.484	30	74.4

(a) The reduced dimension (k)

θ	mAP
0.5	70.4
0.3	72.7
0.2	**74.3**

(b) The threshold (θ)

# Centroid	FLOPs	Inf.	mAP
128	1.265	19	72.1
192	1.358	21	74.3
256	1.452	24	74.5

(c) The number of centroids (m)

$g(\cdot)$	mAP
one channel*	70.6
Sum.	**74.3**
Average	72.8

(d) Aggregation Function

Table 4: Ablation studies on (a): the reduced dimension; (b): the novelty threshold of updating the memory bank (c): the number of learnable codebook; (d): the aggregation function of the final result.

tively. Consistent improvements are further demonstrated on TVSeries, where our approach substantially outperforms baselines using both features under identical settings.

As illustrated in Tab. 3, our model outperforms other methods on FineAction dataset, which offers a more realistic environment with diverse action instances across similar categories. This indicates that our model is capable of capturing more discriminative features, which enhances its suitability for real-life applications.

Ablation Study

To facilitate model analysis, we conduct detailed ablation studies on the THUMOS'14 test set using Kinetics-pretrained features to evaluate individual components of our proposed framework.

		HMM	TSP	FLOPs	FPS	mAP
MOAD	(a)	✗	✗	17.9	25.0	32.4
	(b)	✓	✗	16.8	20.4	36.8
	(c)	✓	✓	10.2	34.0	**41.2**

Table 5: Computational efficiency on the proposed method and TeSTra. All results are tested on FineAction datasets.

Effect of Each Component. Based on Tab. 5, our comprehensive component analysis on FineAction reveals that the proposed Hierarchical Memory Mechanism (HMM) and Temporal Soft Pruning (TSP) cooperatively enhance performance-efficiency trade-offs for long-form video understanding. The Mamba-based baseline (config. (a)) provides a competitive starting point , enabling HMM (config. (b)) reduces computational cost while substantially boosting accuracy by 4.4 mAP (36.8 vs. 32.4), confirming its effectiveness in preserving essential temporal knowledge. Further activating TSP (config. (c)) dramatically reduces FLOPs by 39.3% (10.2G vs. 16.8G) while accelerating inference by 66.7% (34.0 vs. 20.4) and achieving a peak accuracy of 41.2 mAP—demonstrating its dual capability to eliminate redundancy while enhancing temporal modeling. Crucially, the full framework achieves an 8.8 mAP gain over baseline with 43% lower computation, establishing a superior efficiency-accuracy balance for long-duration action detection.

Effect of Reduced Dimension. As shown in Tab. 4a, we examine the impact of feature dimensionality reduction on model efficiency and accuracy with FLOPS (G) and Inference time. Increasing dimensions from 32 to 64 improves mAP from 71.2% to 74.4%, but at a significant computational cost with FLOPs increasing from 1.165G to 1.484G. The optimal balance is achieved at dimension 48, delivering 74.3 mAP with moderate FLOPs and inference time.

Effect of Threshold Selection. Tab. 4b demonstrates the critical role of the novelty threshold θ in memory bank updating. Lower thresholds substantially enhance performance, with θ=0.2 achieving peak mAP, which is a 3.9% improvement over θ=0.5. This confirms our threshold's effectiveness in preserving discriminative action features.

Effect of Memory Capacity. Analysis in Tab. 4c reveals the relationship between the codebook size and performance. Increasing the number of centroids from 128 to 256 boosts mAP from 72.1% to 74.5%, computational costs rise up to 0.19G FLOPs. The 192-centroid configuration provides optimal efficiency while maintaining accuracy.

Effect of Aggregation Function. As evidenced in Tab. 4d, the feature aggregation strategy significantly impacts performance. Simple summation achieves superior results, outperforming averaging by 1.5% and single-channel methods by 3.7%, validating its effectiveness in integrating critical spatiotemporal information.

Online Action Anticipation

We compare our model with prior methods on EPIC-Kitchens-100 and THUMOS'14 for action anticipation, as shown in the Appendix. Our model achieves an average 2% performance gain in both verb and action detection, demonstrating its enhanced capability to capture temporal action evolution.

Conclusion

In this work, we focus on problem of Online Action Detection (OAD), requiring real-time action prediction without future frames and balancing efficiency with long-term dependency modeling. Existing methods suffer from slow training, limited temporal fields, high costs, delayed inference, and inability to handle extra-long videos. Thus, we present MOAD, a novel Mamba-based OAD framework for efficient and effective detection. MOAD introduces a hierarchical memory mechanism to intelligently store high-value historical frames based on motion-aware similarity. To further optimize, we design memory quantization for compression and temporal soft pruning to dynamically remove redundancy while preserving coherence. Sufficient experiments on four benchmarks demonstrate that MOAD significantly outperforms existing methods.

Acknowledgments

Our work is supported in part by the National Key R&D Program of China (No. 2023YFC3305600), the Joint Fund of Ministry of Education of China (8091B022149, 8091B02072404), the National Natural Science Foundation of China (62132016, 62406238) and the Fundamental Research Funds for the Central Universities ZYTS25149.

References

An, J.; Kang, H.; Han, S. H.; Yang, M.-H.; and Kim, S. J. 2023. Miniroad: Minimal RNN Framework for Online Action Detection. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, 10341–10350.

Bahl, S.; Mendonca, R.; Chen, L.; Jain, U.; and Pathak, D. 2023. Affordances from Human Videos as a Versatile Representation for Robotics. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 13778–13790.

Caba Heilbron, F.; Escorcia, V.; Ghanem, B.; and Carlos Niebles, J. 2015. Activitynet: A Large-Scale Video Benchmark for Human Activity Understanding. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, 961–970.

Cao, S.; Luo, W.; Wang, B.; Zhang, W.; and Ma, L. 2023. E2e-Load: End-to-End Long-Form Online Action Detection. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, 10422–10432.

Carreira, J.; and Zisserman, A. 2017. Quo Vadis, Action Recognition? A New Model and the Kinetics Dataset. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, 6299–6308.

Chen, J.; Mittal, G.; Yu, Y.; Kong, Y.; and Chen, M. 2022. Gatehub: Gated History Unit with Background Suppression for Online Action Detection. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 19925–19934.

Cho, K. 2014. Learning Phrase Representations using RNN Encoder-Decoder for Statistical Machine Translation. *arXiv preprint arXiv:1406.1078*.

Damen, D.; Doughty, H.; Farinella, G. M.; Furnari, A.; Kazakos, E.; Ma, J.; Moltisanti, D.; Munro, J.; Perrett, T.; Price, W.; et al. 2022. Rescaling Egocentric Vision: Collection, Pipeline and Challenges for Epic-Kitchens-100. *International Journal of Computer Vision*, 1–23.

De Geest, R.; Gavves, E.; Ghodrati, A.; Li, Z.; Snoek, C.; and Tuytelaars, T. 2016. Online Action Detection. In *Computer Vision–ECCV 2016: 14th European Conference, Amsterdam, The Netherlands, October 11-14, 2016, Proceedings, Part V 14*, 269–284. Springer.

Dey, R.; and Salem, F. M. 2017. Gate-Variants of Gated Recurrent Unit (GRU) Neural Networks. In *2017 IEEE 60th International Midwest Symposium on Circuits and Systems (MWSCAS)*, 1597–1600. IEEE.

Dong, J.; Li, H.; Cong, Y.; Sun, G.; Zhang, Y.; and Van Gool, L. 2024. No One Left Behind: Real-World Federated Class-Incremental Learning. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 46(4): 2054–2070.

Dong, J.; Wang, L.; Fang, Z.; Sun, G.; Xu, S.; Wang, X.; and Zhu, Q. 2022. Federated Class-Incremental Learning. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 10164–10173.

Dosovitskiy, A.; Beyer, L.; Kolesnikov, A.; Weissenborn, D.; Zhai, X.; Unterthiner, T.; Dehghani, M.; Minderer, M.; Heigold, G.; Gelly, S.; et al. 2020. An Image is Worth 16x16 Words: Transformers for Image Recognition at Scale. *arXiv preprint arXiv:2010.11929*.

Eun, H.; Moon, J.; Park, J.; Jung, C.; and Kim, C. 2020. Learning to Discriminate Information for Online Action Detection. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 809–818.

Gao, J.; Yang, Z.; and Nevatia, R. 2017. RED: Reinforced Encoder-Decoder Networks for Action Anticipation. *arXiv preprint arXiv:1707.04818*.

Gu, A.; and Dao, T. 2023. Mamba: Linear-Time Sequence Modeling with Selective State Spaces. *arXiv preprint arXiv:2312.00752*.

Gu, A.; Goel, K.; and Ré, C. 2021. Efficiently Modeling Long Sequences with Structured State Spaces. *arXiv preprint arXiv:2111.00396*.

Guermal, M.; Ali, A.; Dai, R.; and Bremond, F. 2024. JOADAA: Joint Online Action Detection and Action Anticipation. In *Proceedings of the IEEE/CVF Winter Conference on Applications of Computer Vision*, 6889–6898.

Guo, H.; Wang, H.; and Ji, Q. 2024. Bayesian Evidential Deep Learning for Online Action Detection. In *European Conference on Computer Vision*, 283–301. Springer.

Hatamizadeh, A.; and Kautz, J. 2025. MambaVision: A Hybrid Mamba-Transformer Vision Backbone. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, 25261–25270.

He, K.; Zhang, X.; Ren, S.; and Sun, J. 2016. Deep Residual Learning for Image Recognition. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, 770–778.

Huang, T.; Pei, X.; You, S.; Wang, F.; Qian, C.; and Xu, C. 2024. LocalMamba: Visual State Space Model with Windowed Selective Scan. *arXiv preprint arXiv:2403.09338*.

Idrees, H.; Zamir, A. R.; Jiang, Y.-G.; Gorban, A.; Laptev, I.; Sukthankar, R.; and Shah, M. 2017. The THUMOS Challenge on Action Recognition for Videos "in the Wild". *Computer Vision and Image Understanding*, 155: 1–23.

Ioffe, S. 2015. Batch Normalization: Accelerating Deep Network Training by Reducing Internal Covariate Shift. *arXiv preprint arXiv:1502.03167*.

Jang, E.; Gu, S.; and Poole, B. 2016. Categorical Reparameterization with Gumbel-Softmax. *arXiv preprint arXiv:1611.01144*.

Kay, W.; Carreira, J.; Simonyan, K.; Zhang, B.; Hillier, C.; Vijayanarasimhan, S.; Viola, F.; Green, T.; Back, T.; Natsev, P.; et al. 2017. The Kinetics Human Action Video Dataset. *arXiv preprint arXiv:1705.06950*.

Kim, Y. H.; Kang, H.; and Kim, S. J. 2022. A Sliding Window Scheme for Online Temporal Action Localization. In *European Conference on Computer Vision*, 653–669. Springer.

Li, J.; Wei, K.; Xu, Z.; Wang, L.; and Deng, C. 2025. Robust Temporal Action Localization with Meta Boundary Refinement. *IEEE Transactions on Multimedia*.

Li, S.; Singh, H.; and Grover, A. 2024. Mamba-nd: Selective State Space Modeling for Multi-Dimensional Data. In *European Conference on Computer Vision*, 75–92. Springer.

Liu, Y.; Tian, Y.; Zhao, Y.; Yu, H.; Xie, L.; Wang, Y.; Ye, Q.; Jiao, J.; and Liu, Y. 2024. VMamba: Visual State Space Model. *Advances in Neural Information Processing Systems*, 37: 103031–103063.

Liu, Y.; Wang, L.; Wang, Y.; Ma, X.; and Qiao, Y. 2022. Fineaction: A Fine-Grained Video Dataset for Temporal Action Localization. *IEEE Transactions on Image Processing*, 31: 6937–6950.

Mu, C.; Li, J.; Wei, K.; and Deng, C. 2025. Energy vs. Noise: Towards Robust Temporal Action Localization in Open-World. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 39, 6164–6172.

Pei, X.; Huang, T.; and Xu, C. 2025. Efficientvmamba: Atrous Selective Scan for Light Weight Visual Mamba. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 39, 6443–6451.

Ramachandran, P.; Zoph, B.; and Le, Q. V. 2017. Searching for Activation Functions. *arXiv preprint arXiv:1710.05941*.

Rangrej, S. B.; Liang, K. J.; Hassner, T.; and Clark, J. J. 2023. GliTr: Glimpse Transformers with Spatiotemporal Consistency for Online Action Prediction. In *Proceedings of the IEEE/CVF Winter Conference on Applications of Computer Vision*, 3413–3423.

Rao, Y.; Zhao, W.; Liu, B.; Lu, J.; Zhou, J.; and Hsieh, C.-J. 2021. Dynamicvit: Efficient Vision Transformers with Dynamic Token Sparsification. *Advances in Neural Information Processing Systems*, 34: 13937–13949.

Reza, S.; Zhang, Y.; Moghaddam, M.; and Camps, O. 2025. HAT: History-Augmented Anchor Transformer for Online Temporal Action Localization. In *European Conference on Computer Vision*, 205–222. Springer.

Smith, J. T.; Warrington, A.; and Linderman, S. W. 2022. Simplified State Space Layers for Sequence Modeling. *arXiv preprint arXiv:2208.04933*.

Vaswani, A. 2017. Attention Is All You Need. *Advances in Neural Information Processing Systems*.

Wang, J.; Chen, G.; Huang, Y.; Wang, L.; and Lu, T. 2023. Memory-and-Anticipation Transformer for Online Action Understanding. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, 13824–13835.

Wang, X.; Zhang, S.; Qing, Z.; Shao, Y.; Zuo, Z.; Gao, C.; and Sang, N. 2021. Oadtr: Online Action Detection with Transformers. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, 7565–7575.

Xu, C.; Lyu, G.; Yan, J.; Yang, M.; and Deng, C. 2024a. LLM Knows Body Language, Too: Translating Speech Voices into Human Gestures. In *ACL*, 5004–5013.

Xu, C.; Yan, J.; and Deng, C. 2025. Keep and Extent: Unified Knowledge Embedding for Few-Shot Image Generation. *IEEE Transactions on Image Processing*.

Xu, C.; Yan, J.; Yang, M.; and Deng, C. 2024b. Rethinking Noise Sampling in Class-Imbalanced Diffusion Models. *IEEE Transactions on Image Processing*.

Xu, M.; Gao, M.; Chen, Y.-T.; Davis, L. S.; and Crandall, D. J. 2019. Temporal Recurrent Networks for Online Action Detection. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, 5532–5541.

Xu, M.; Xiong, Y.; Chen, H.; Li, X.; Xia, W.; Tu, Z.; and Soatto, S. 2021. Long Short-Term Transformer for Online Action Detection. *Advances in Neural Information Processing Systems*, 34: 1086–1099.

Yang, C.; Chen, Z.; Espinosa, M.; Ericsson, L.; Wang, Z.; Liu, J.; and Crowley, E. J. 2024. Plainmamba: Improving Non-Hierarchical Mamba in Visual Recognition. *arXiv preprint arXiv:2403.17695*.

Zhao, Y.; and Krähenbühl, P. 2022. Real-Time Online Video Detection with Temporal Smoothing Transformers. In *European Conference on Computer Vision*, 485–502. Springer.

Zhu, L.; Liao, B.; Zhang, Q.; Wang, X.; Liu, W.; and Wang, X. 2024. Vision Mamba: Efficient Visual Representation Learning with Bidirectional State Space Model. *arXiv preprint arXiv:2401.09417*.