SMV-EAR: Bring Spatiotemporal Multi-View Representation Learning into Efficient Event-Based Action Recognition

Supplementary Material

This supplementary we provides: (i) rigorous mathematical proofs of translation-invariance in TISM (Sec. S1), (ii) complete formulations of fusion strategies in DDCF (Sec. S2), (iii) detailed temporal warping functions in DTW (Sec. S3), (iv) comprehensive implementation and reproduction details (Sec. S4), and (v) extended ablation studies including dynamic weight extraction analysis (Sec. S5).

S1. Rigorous Proofs of TISM

S1.1. Notation and Problem Setting

Let $\mathcal{E} = \{(x_k, y_k, t_k, p_k)\}_{k=0}^{N_e-1}$ be the input event stream, where $(x_k, y_k) \in \mathbb{Z}^2$ denotes spatial coordinates, $t_k \in \mathbb{R}^+$ is the timestamp, and $p_k \in \{-1, +1\}$ represents polarity. For a spatiotemporal view $v \in \{th, tw, hw\}$, we define the orthogonal axis as:

$$z^v = \begin{cases} x, & \text{if } v = th \text{ (Time-Height view)} \\ y, & \text{if } v = tw \text{ (Time-Width view)} \\ t, & \text{if } v = hw \text{ (Height-Width view).} \end{cases} \tag{1}$$

Following the encoding decomposition in Eq.1 of the main paper, we have:

$$F_v(\mathcal{E}) = [F_0, \cdots, F_{N_c-1}],\ F_c = a_c(m_c(w_c(\mathcal{E}))), \tag{2}$$

where $w_c : \mathcal{E} \to \{\mathcal{E}_i\}_{i=0}^{N_b-1}$ is a window function partitioning events into N_b bins, $m_c : \mathcal{E}_i \to \{e_j\}$ is a measurement function extracting event properties, and $a_c : \{e_j\} \to \mathbb{R}$ is an aggregation function reducing to scalar values.

Definition 1 (Translation-Invariance). An encoding function F_v is translation-invariant along axis z^v if and only if for any spatial shift Δz^v when no out-of-bounds conditions:

$$F_v(\mathcal{E}|_{z^v}) = F_v(\mathcal{E}|_{z^v+\Delta z^v}), \tag{3}$$

where $\mathcal{E}|_{z^v+\Delta z^v}$ represents the shifted events with coordinates $(x_k + \Delta x, y_k, t_k, p_k)$ when $z^v = x$ or $(x_k, y_k + \Delta y, t_k, p_k)$ when $z^v = y$.

S1.2. Translation-Invariance of Window Functions

Lemma 1 (Window Invariance). A window function $w : \mathcal{E} \to \{\mathcal{E}_i\}_{i=0}^{N_b-1}$ is translation-invariant along z^v if and only if it does not partition events based on absolute value of z^v.

Proof. Consider two types of window functions:

- *Global window* w_0: assigns all events into a single, global bin, i.e., $w_0(\mathcal{E}) = \{\mathcal{E}\}$.
- *Binning* w_k with $k > 1$: partitions z^v dimension into k intervals $\{[z_i^v, z_{i+1}^v)\}_{i=0}^{k-1}$.

For w_0, shifting all events by Δz^v does not change the bin assignment:

$$w_0(\mathcal{E}|_{z^v}) = \{\mathcal{E}\} = w_0(\mathcal{E}|_{z^v+\Delta z^v}), \tag{4}$$

thus w_0 is translation-invariant. While for w_k with $k > 1$, consider an event $e_j = (x_j, y_j, t_j, p_j)$ near a bin boundary z_i^v. Under shift Δz^v, the event's coordinate becomes $z_j^v + \Delta z^v$, which may cross into an adjacent bin:

$$w_k(e_j|_{z_j^v}) = \text{bin}_i \text{ if } z_j^v \in [z_i^v, z_{i+1}^v), \tag{5}$$

$$w_k(e_j|_{z_j^v+\Delta z^v}) = \text{bin}_{i+1} \text{ if } z_j^v + \Delta z^v \in [z_{i+1}^v, z_{i+2}^v), \tag{6}$$

which violates Eq. (3). Therefore, only w_0 (global, bin-less window) satisfies translation-invariance.

S1.3. Translation-Invariance of Measurements and Aggregations

Lemma 2 (Measurement-Aggregation Pairs). For a global window w_0, the following measurement-aggregation pairs (m_c, a_c) are translation-invariant along z^v:

1. $(c, \textit{sum})$: Count all events and sum, $m_c = 1, a_c = \sum$.
2. $(p, \textit{sum})$: Extract polarities and sum, $m_c = p_k, a_c = \sum$.
3. $(c_+, \textit{sum})$: Count positive events, $m_c = \mathbb{I}[p_k = +1], a_c = \sum$.
4. $(c_-, \textit{sum})$: Count negative events, $m_c = \mathbb{I}[p_k = -1], a_c = \sum$.
5. $(z^v, \textit{variance})$: Measure z^v coordinate and compute variance, $m_c = z_k^v, a_c = \text{Var}$.
6. $(z_+^v, \textit{variance})$: Measure z^v for positive events, $m_c = z_k^v \mathbb{I}[p_k = +1], a_c = \text{Var}$.
7. $(z_-^v, \textit{variance})$: Measure z^v for negative events, $m_c = z_k^v \mathbb{I}[p_k = -1], a_c = \text{Var}$.

Proof. *Case 1-4: Count and polarity measurements with sum aggregation.* For $(c, \textit{sum})$, the encoding is $F_c = \sum_{k=0}^{N_e-1} 1 = N_e$, which is independent of spatial coordinates, thus translation-invariant. Similarly, $(p, \textit{sum})$, $(c_+, \textit{sum})$, $(c_-, \textit{sum})$ depend only on polarities, which are unaffected by spatial shifts.

Case 5-7: Coordinate measurements with variance aggregation. For $(z^v, \textit{variance})$, the variance of a set of values $\{z_k^v\}$ under translation is:

$$\begin{aligned} \text{Var}(\{z_k^v + \Delta z^v\}) &= \frac{1}{N_e} \sum_{k=0}^{N_e-1} (z_k^v + \Delta z^v - \bar{z}^v - \Delta z^v)^2 \\ &= \frac{1}{N_e} \sum_{k=0}^{N_e-1} (z_k^v - \bar{z}^v)^2 = \text{Var}(\{z_k^v\}), \end{aligned} \tag{7}$$

where $\bar{z}^v = \frac{1}{N_e}\sum_k z_k^v$ is the mean. Thus, variance is translation-invariant. Meanwhile, first-order statistics like *mean*, *max*, *min*, or *sum* applied to coordinate measurements are **not** translation-invariant, for example:

$$\text{mean}(\{z_k^v + \Delta z^v\}) = \text{mean}(\{z_k^v\}) + \Delta z^v \neq \text{mean}(\{z_k^v\}). \tag{8}$$

S1.4. Complete Feasible Set

Combining Lemma 1 and Lemma 2, we establish the complete set of translation-invariant encoding functions:

$$\begin{aligned} P_{IV} = \{&(w_0, \textit{sum}, c), (w_0, \textit{sum}, p), (w_0, \textit{sum}, c_+), \\ &(w_0, \textit{sum}, c_-), (w_0, \textit{variance}, z^v), \\ &(w_0, \textit{variance}, z_+^v), (w_0, \textit{variance}, z_-^v)\}. \end{aligned} \tag{9}$$

Theorem 1 (TISM Encoding). The encoding function:

$$F_v = [\text{sum}(c(w_0(\mathcal{E}))), \text{sum}(p(w_0(\mathcal{E})))] \in \mathbb{R}^{2\times U\times V} \tag{10}$$

is translation-invariant along axis z^v and encodes global event distribution and polarity information.

Proof. By Lemma 1, w_0 is translation-invariant. By Lemma 2, $(c, \textit{sum})$ and $(p, \textit{sum})$ are translation-invariant. Therefore, their composition F_v satisfies Eq. (3).

S1.5. Efficiency and Expressiveness Trade-off

As discussed in Sec.3.2, While $(w_0, \textit{variance}, z^v)$ provides additional translation-invariant channels, computing second-order moments incurs higher computational cost. Mathematically speaking, $(p, \textit{sum})$ only requires $O(N_e)$ single-pass accumulation, while $(z^v, \textit{variance})$ requires $O(2N_e)$ two passes (mean computation + variance accumulation). In Tab.4 of main paper, we compare encoding time and accuracy trade-offs to validate this analysis. One can see that given the marginal accuracy gain (+0.4%) but substantial encoding overhead (+172%), hence we adopt the compact $P_{IV}^* = \{(w_0, \textit{sum}, c), (w_0, \textit{sum}, p)\}$ as the default configuration of the proposed TISM.

S1.6. Empirical Validation under Controlled Shifts

We conduct more detailed spatial shift robustness tests on THU-EACT-50-CHL in Tab.5. For test set, we inject test-time spatial translations $\Delta x, \Delta y \in \{0, \pm10, \pm20, \pm30, \pm40\}$ pixels into baseline [2] before encoding, covering 9 shift combinations per sample. The results show the effectiveness of TISM to handle spatial shift.

S1.7. Comparison with Other Representations

Tab.1 compares our TISM against commonly adopted and newly proposed event representation on baseline MVF-Net [2]. Our TISM showcases superior accuracy.

Table 1. Comparison with other representations on F_{th}.

Representation	Window	Measurement	TI	Top-1(%)
Event Frame [7]	1-bin	c	✓	54.5
Voxel Grid [9]	9-bin	c	×	58.2
MVF-Net [2]	3-bin	p	×	56.5
EST [4]	learned	-	×	55.8
EventPillars [3]	4-bin	z,max/min	×	56.3
TISM (ours)	global	c,p	✓	**66.7**

Table 2. Comparison of dynamic weights w_{th}, w_{tw} extraction architecture on THU-EACT-50-CHL.

Architecture	**Input**	**Top-1(%)**	**FLOPs**	**Params**
MLPs	1×1024	65.5(±0.34)	3.6G	23.2M
1D Convs	2×512	64.5(±0.22)	3.5G	23.1M
Attention	1024×1	65.8(±0.28)	3.5G	23.6M
Attention	512×2	65.9(±0.35)	3.6G	23.6M
Attention	2×512	**66.7**(±0.29)	3.6G	23.5M

S2. Formulations of Fusion Strategies in DDCF

S2.1. Problem Formulation

Given dual-branch extracted logits $L_{th} \in \mathbb{R}^C$ and $L_{tw} \in \mathbb{R}^C$ learned from F_{th} and F_{tw} features respectively, where C is the number of action classes, the goal is to design a function $\mathcal{F} : [L_{th}, L_{tw}] \to L \in \mathbb{R}^C$ that maximally exploits **cross-view, sample-specific dynamic complementarity**.

S2.2. Static Fusion Strategies

Early Concatenation (EC). MVF-Net [2] concatenate two temporal feature maps before fed them into network:

$$F = [F_{th}, F_{tw}] \in \mathbb{R}^{2C_f\times H\times W},\ L = \mathcal{R}(F), \tag{11}$$

where C_f is the feature channel dimension. This is our adopted baseline method implemented in the main paper.

Logits Averaging (LA). Simple element-wise averaging:

$$L = \frac{L_{th} + L_{tw}}{2}. \tag{12}$$

View-wise Weighting (VW). Per-view scalar weights:

$$L = w_{th}L_{th} + w_{tw}L_{tw}, \tag{13}$$

$$[w_{th}, w_{tw}] = \text{Softmax}([v_{th}, v_{tw}]), \tag{14}$$

where $v_{th}, v_{tw} \in \mathbb{R}$ are learnable scalars.

Class-wise Weighting (CW). Per-class learnable weights:

$$L = W_{th} \odot L_{th} + W_{tw} \odot L_{tw},\ W_v \in \mathbb{R}^C, \tag{15}$$

where $\odot$ denotes element-wise multiplication.

S2.3. Dynamic Fusion Strategies

Dynamic fusion is referred to learn logits fusion weights w_{tw}, w_{th} that related to $\mathcal{E}$, then apply fusion $L = w_{th} \odot$

$L_{th} + w_{tw} \odot L_{tw}$ for final logits L. As discussed in Sec.3.3, learning from semantics $S_{th}, S_{th} \in \mathbb{R}^{512}$ is a better choice in accuracy-efficiency trade-offs. Meanwhile, the choice of which architecture to learn w_{tw}, w_{th} also needs to be empirically determined. We first formulate each commonly adopted neural architecture compared in Tab.6 as below:

MLPs. 3-layer MLPs on concatenated semantics:

$$S = [S_{th}, S_{tw}] \in \mathbb{R}^{1024}, \tag{16}$$

$$[w_{th}, w_{tw}] = \text{Softmax}(\text{MLPs}(S)) \in \mathbb{R}^{2C}. \tag{17}$$

1-D Convs. 3-layer, 5-kernel 1-D convolutional gating:

$$S' = \text{Reshape}(S, [2, 512]), \tag{18}$$

$$[w_{th}, w_{tw}] = \text{Softmax}(\text{ConvLayers}_k(S')) \in \mathbb{R}^{2C}. \tag{19}$$

1024-Token 1-D Attention. Concat S_{th}, S_{tw} as a $S = [S_{th}, S_{tw}] \in \mathbb{R}^{1024\times 1}$ Self-Attention input:

$$S' = \text{SelfAttention}(S), \tag{20}$$

$$[w_{th}, w_{tw}] = \text{Linear}(S') \in \mathbb{R}^{2C}. \tag{21}$$

512-Token 2-D Attention. Concat S_{th}, S_{tw} as a $S = [S_{th}, S_{tw}] \in \mathbb{R}^{512\times 2}$ Self-Attention input.

2-Token 512-D Attention. (selected) Concat S_{th}, S_{tw} as a $S = [S_{th}, S_{tw}] \in \mathbb{R}^{2\times 512}$ Self-Attention input.

We provide a comprehensive comparison across all above fusion strategies in Tab.2, our selected 2-Token 512-D Self-Attention achieves the best accuracy with comparable computational and parameters overhead. Another argument of the extraction timing of w_{th}, w_{tw} has been ablated in Tab.7 of main paper, showing that extraction from S_{th}, S_{tw} is indeed a better choice in accuracy-efficiency trade-offs.

S2.4. Sample-wise Weight Distribution Analysis

Fig.1 visualizes the learned fusion weight distributions w_{th} and w_{tw} across 29 test samples under an identical action class from DailyDVS-200. The sample-level dynamic fusion weights are successfully learned.

S2.5. Per-Class Weight Visualization

Fig.2 shows per-class average weights, revealing that certain action classes with more vertical motions rely more on T-H view, while horizontal motions depend on T-W view.

S3. Temporal Warping Functions in DTW

Let $t \in [t_0, t_e]$ be the original timestamp and $t' = \mathcal{W}(t; \theta)$ be the warped timestamp, where θ are function-specific parameters. To maintain temporal causality, we require $\mathcal{W}$ to be monotonically increasing: $\frac{d\mathcal{W}}{dt} > 0$. The instantaneous speed scaling factor is $s(t) = \frac{d\mathcal{W}(t)}{dt}$. When $s(t) > 1$, events locally decelerate (become sparser); when $s(t) < 1$, events accelerate (become denser).

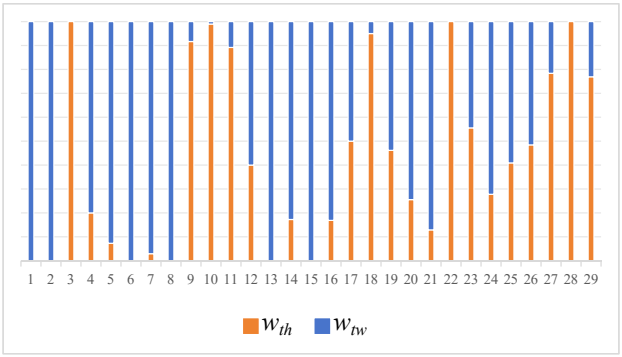

Figure 1. Distribution of learned sample-wise weights w_{th} and w_{tw} on DailyDVS-200 test set. The broad, non-degenerate distributions validate that optimal fusion varies significantly across samples, necessitating dynamic sample-wise fusion.

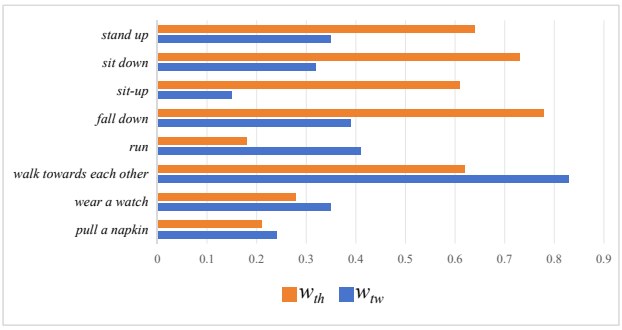

Figure 2. Average fusion weights on DailyDVS-200. Classes with primarily vertical motion (e.g., "Stand up", "Sit down") exhibit higher w_{th}, while horizontal motions (e.g., "Run", "Walk towards each other") show higher w_{tw}.

S3.1. Temporal Warping Function Families

Identity Warping (Id.). No warping is applied, serves as to retain the real sample space of the original data:

$$\mathcal{W}_{\text{id}}(t) = t, s(t) = 1. \tag{22}$$

Linear Warping (Lin.). Uniform speed scaling, where $\alpha > 1$ accelerates actions globally and $\alpha < 1$ decelerates:

$$\mathcal{W}_{\text{lin}}(t; \alpha) = t_0 + \alpha(t - t_0),\ \alpha \in [0.6, 1.4], \tag{23}$$

$$s(t) = \alpha. \tag{24}$$

Power Warping (Pow.). Non-uniform warping, where $\beta > 1$ decelerates early and accelerates late, while $\beta < 1$ accelerates early and decelerates late:

$$\mathcal{W}_{\text{pow}}(t; \beta) = t_0 + (t_e - t_0)\left(\frac{t - t_0}{t_e - t_0}\right)^{\beta},\ \beta \in [0.6, 1.4], \tag{25}$$

$$s(t) = \beta \frac{(t_e - t_0)^{1-\beta}}{(t - t_0)^{1-\beta}}. \tag{26}$$

Table 3. Ablation on number of DTW intervals l.

#Intervals l	Top-1(%)
1 (global)	64.5($\pm$0.33)
2	65.5($\pm$0.27)
3	65.9($\pm$0.26)
4	**66.7**($\pm$0.29)
5	66.3($\pm$0.31)
6	66.1($\pm$0.28)

Exponential Warping (Exp.). Exponential speed modulation, where $\gamma > 0$ creates accelerating actions and $\gamma < 0$ creates decelerating actions:

$$\mathcal{W}_{\text{exp}}(t;\gamma) = t_0 + \frac{e^{\gamma(t-t_0)}-1}{e^{\gamma(t_e-t_0)}-1}(t_e - t_0),\ \gamma \in [-0.5, 0.5], \tag{27}$$

$$s(t) = \frac{\gamma e^{\gamma(t-t_0)}(t_e - t_0)}{e^{\gamma(t_e-t_0)}-1}. \tag{28}$$

Cosine Warping (Cos.). Smooth oscillatory speed variation mimicking natural acceleration-deceleration cycles:

$$\mathcal{W}_{\cos}(t;\eta) = t_0 + \frac{1-\cos(\pi\tau)}{2}(t_e - t_0) + \eta\sin(2\pi\tau)(t_e - t_0), \tau = \frac{t-t_0}{t_e - t_0},\ \eta \in [0, 0.1], \tag{29}$$

$$s(t) = \frac{\pi}{2(t_e - t_0)}\sin(\pi\tau) + \frac{2\pi\eta}{t_e - t_0}\cos(2\pi\tau). \tag{30}$$

S3.2. Ablation on Number of Intervals

We ablate the effect of interval count l in Tab.3. The results lead us to select $l = 4$ as the optimal trade-off between diversity and recognition accuracy.

S4. Reproduction Details

S4.1. Environment and Dependencies

Software Environment:
- Operating System: Ubuntu 20.04 LTS
- CUDA Version: 11.8
- cuDNN Version: 8.7.0
- Python Version: 3.8.10
- PyTorch Version: 2.1.0
- Torchvision Version: 0.16.0
- Additional Libraries: timm 0.9.2, ptflops 0.7, h5py 3.7.0, opencv-python 4.7.0

Hardware Configuration:
- CPU: AMD EPYC 7542 32-Core Processor @ 2.9GHz
- GPU: NVIDIA RTX 3090 (24GB VRAM)
- RAM: 128GB DDR4
- Storage: NVMe SSD 2TB

S4.2. Deterministic Settings

To ensure reproducibility, we enforce strict determinism and report results over three seeds: {0, 11, 42}:

```
import torch
import numpy as np
import random

def set_seed(seed=3407):
    random.seed(seed)
    np.random.seed(seed)
    torch.manual_seed(seed)
    torch.cuda.manual_seed(seed)
    torch.cuda.manual_seed_all(seed)
    torch.backends.cudnn.deterministic = True
    torch.backends.cudnn.benchmark = False
```

S4.3. TISM Encoding Implementation

Pseudocode for efficient TISM encoding:

```
def encode_TISM(events, view='th', T=224, H=260, W=346):
    x, y, t, p = events['x'], events['y'],
                 events['t'], events['p']

    # Normalize timestamps to [0, T-1]
    t_norm = ((t - t.min()) / (t.max() - t.min()) * (T-1))
    t_idx = t_norm.astype(int)

    # Project to T-H or T-W view
    if view == 'th':
        F_c = np.zeros((T, H))
        F_p = np.zeros((T, H))
        for i in range(len(events)):
            F_c[t_idx[i], y[i]] += 1
            F_p[t_idx[i], y[i]] += p[i]
    elif view == 'tw':
        F_c = np.zeros((T, W))
        F_p = np.zeros((T, W))
        for i in range(len(events)):
            F_c[t_idx[i], x[i]] += 1
            F_p[t_idx[i], x[i]] += p[i]

    # Resize to 224×224
    F_c = cv2.resize(F_c, (224, 224))
    F_p = cv2.resize(F_p, (224, 224))

    return np.stack([F_c, F_p], axis=0)
```

Time complexity: $O(N_e)$ with vectorized operations.

S4.4. DDCF Network Architecture

Branch Architecture:
- Backbone: 2 $\times$ ResNet-18 (w/o pretrained)
- Input: $3 \times 2 \times 224 \times 224$ (TISM channels)
- Output after GAP: $S_v \in \mathbb{R}^{512}$
- Classification head: Linear(512, C)

Fusion Module:
- Input: $S = [S_{th}, S_{tw}] \in \mathbb{R}^{2\times512}$
- Transformer: 8 heads, hidden dim 512
- Output projection: Linear(512, $2C$) $\rightarrow [w_{th}, w_{tw}]$

Parameters: 23.5M (11.0M per branch + 1.5M fusion).

S4.5. Training Hyperparameters

As shown in Tab. 4.

S4.6. Baseline Reproduction Notes

MVF-Net [2]:
- Use original 3-bin spatial binning for T-H and T-W views
- Early-concatenation fusion with shared ResNet-34 backbone
- Train with same hyperparameters for fair comparison

CoST [5] and MVFNet [8]:

Table 4. Complete training hyperparameters.

Hyperparameter	THU-EACT	HARDVS	DailyDVS
Epochs	80	80	100
Batch size	32	64	64
Optimizer	AdamW	AdamW	AdamW
Learning rate	1e-4	1e-4	5e-5
Weight decay	1e-5	1e-5	1e-5
LR schedule	Cosine	Cosine	Cosine
Warmup epochs	5	5	10
Label smoothing	0.1	0.1	0.1
Gradient clipping	1.0	1.0	1.0
DTW prob.	0.5	0.5	0.5
DTW intervals	4	4	4

- Adapt from video to event data
- Replace optical flow with event frames
- Use same temporal resolution $T = 224$

Swin-T [6]:

- Use official video Swin-Transformer weights
- Convert event data to pseudo-frames (8 frames)
- Fine-tune with learning rate 5e-5

EventMamba [1]:

- Use official code and pretrained weights
- Adapt to same $T = 224$ temporal resolution
- Report inference time on same hardware

S4.7. Evaluation Protocol

Accuracy Metrics:

- Top-1 and Top-5 accuracy on test set
- Mean $\pm$ standard deviation over 5 seeds
- Per-attribute breakdown (DailyDVS-200 only)

Efficiency Metrics:

- MACs: measured with ptflops on single sample
- FLOPs: measured with ptflops on single sample
- T(cpu): TISM encoding time on AMD EPYC 7542 CPU
- T(gpu): forward pass time on RTX 3090 GPU (batch size 1, averaged over 100 runs)
- T(all) = T(cpu) + T(gpu)
- Params: total trainable parameters

S4.8. Code Release Checklist

Upon acceptance, we will release:

- Complete source code for TISM, DDCF, DTW
- Pretrained model checkpoints for all datasets
- Data preprocessing and loading scripts
- Training and evaluation scripts
- Visualization and analysis notebooks
- Environment configurations (requirements, Dockerfile)
- Detailed README with step-by-step instructions

Table 5. SMV-EAR performance across different backbones on THU-EACT-50-CHL.

Backbone	Top-1(%)	Top-5(%)	Params	MACs
ResNet-10	63.2($\pm$0.4)	86.5($\pm$0.5)	10.8M	2.8G
ResNet-18	**66.7($\pm$0.3)**	**80.5($\pm$0.3)**	23.5M	1.8G
ResNet-34	68.1($\pm$0.3)	90.2($\pm$0.3)	43.7M	10.2G
ResNet-50	68.3($\pm$0.3)	90.4($\pm$0.3)	51.3M	12.8G

Table 6. Scalability analysis with varying temporal resolution T on THU-EACT-50-CHL.

T	Top-1(%)	Top-5(%)	MACs	T(cpu)	T(all)
56	64.2($\pm$0.4)	75.3($\pm$0.5)	1.1G	0.8ms	6.5ms
112	65.8($\pm$0.3)	78.1($\pm$0.4)	1.5G	1.5ms	8.2ms
224	**66.7($\pm$0.3)**	**80.5($\pm$0.3)**	1.8G	2.5ms	10.6ms
336	66.9($\pm$0.3)	80.9($\pm$0.3)	2.8G	3.1ms	13.5ms
448	67.1($\pm$0.4)	81.1($\pm$0.4)	3.5G	4.5ms	17.2ms

Table 7. Training time comparison (on THU-EACT-50-CHL).

Method	Training Time	Time/Epoch	Final Top-1(%)
MVF-Net	2.5 hours	1.9 min	56.5
SlowFast	3.8 hours	2.9 min	52.8
Swin-T	4.2 hours	3.2 min	59.1
SMV-EAR (ours)	2.8 hours	2.1 min	**66.7**

S5. Extended Ablation Studies

S5.1. Hyperparameter Sensitivity

S5.2. Backbone Generalization

We evaluate SMV-EAR with different ResNet backbones in Tab.5. The results show that our SMV-EAR benefits from deeper backbones but shows diminishing returns beyond ResNet-34. We consequently select ResNet-18 for optimal accuracy-efficiency trade-off as the final setup.

S5.3. Temporal Resolution Scalability

In Tab.6 we evaluate scalability with respect to temporal resolution T, showing that the growth of T within a certain range is conducive to EAR accuracy improvement due to the improved representational temporal granularity. However, in order to be consistent with MVF-Net, we adopted 224 as the final scheme setting.

S5.4. Training Efficiency

We further shows training convergence curves comparison in Fig.3 and total training time comparison in Tab.7. Our SMV-EAR also exhibits the best training efficiency.

S5.5. Cross-Dataset Generalization

Tab.8 evaluates the cross-dataset generalization by training on one dataset and testing on another on their common action types. One can see that cross-dataset performance

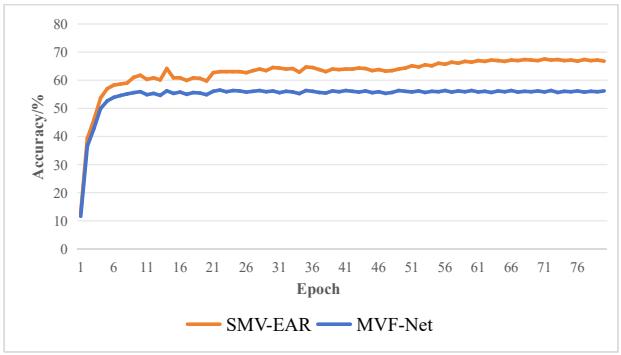

Figure 3. Training and test accuracy curves on THU-EACT-50-CHL. SMV-EAR (orange) converges faster and achieves higher final accuracy than MVF-Net (blue).

Table 8. Cross-dataset generalization (train → test). Diagonal shows in-domain performance.

Train \ Test	THU-EACT	HARDVS	DailyDVS
THU-EACT	**66.7**	42.3	38.1
HARDVS	35.2	**59.6**	45.7
DailyDVS	31.8	48.5	**54.7**

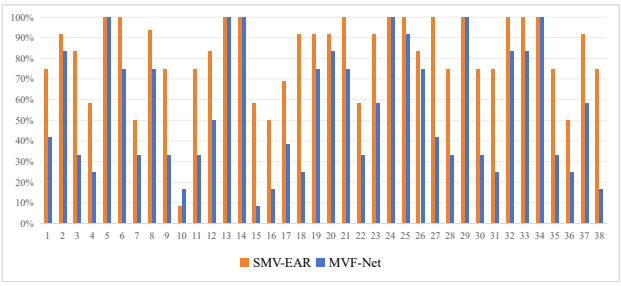

Figure 4. Per-class accuracy comparison on the THU-EACT-50-CHL dataset shows that SMV-EAR (orange) outperforms MVF-Net (blue) on the majority of classes.

drops significantly due to domain or sensor shift, indicating opportunities for domain adaptation techniques.

S6. Additional Visualizations

S6.1. Per-Class Accuracy Breakdown

As shown in Fig.4, we comparing per-class accuracy breakdown of SMV-EAR with MVF-Net. Our SMV-EAR outperforms MVF-Net on all action classes, especially on actions with complex temporal dynamics (e.g., "Sit down", "Jump", "Wave hand").

S6.2. Temporal Progression Visualization

We visualize how our proposed SMV-EAR processes temporal action progression across T-H and T-W views in Fig.5. It can be seen that each branch progressively extracts complementary spatiotemporal motion features from two temporal maps, providing an insightful witness for the inner procedure of SMV-EAR.

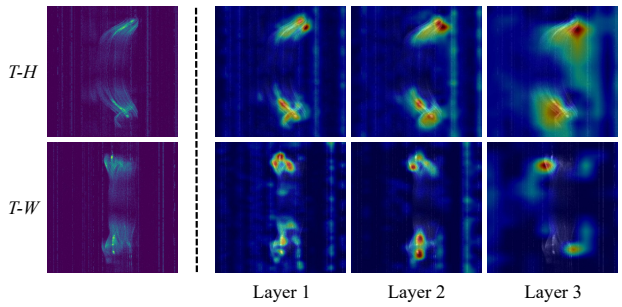

Figure 5. Temporal action progression visualization. Each column shows T-H (top) and T-W (bottom) maps at different feature stages, with attention overlays.

References

[1] Jiaqi Chen, Yan Yang, Shizhuo Deng, Da Teng, and Liyuan Pan. Spikmamba: When snn meets mamba in event-based human action recognition. In *Proceedings of the 6th ACM International Conference on Multimedia in Asia*, pages 1–8, 2024. 5

[2] Yongjian Deng, Hao Chen, and Youfu Li. Mvf-net: A multi-view fusion network for event-based object classification. *IEEE Transactions on Circuits and Systems for Video Technology*, 32(12):8275–8284, 2021. 2, 4

[3] Rui Fan, Weidong Hao, Juntao Guan, Lai Rui, Lin Gu, Tong Wu, Fanhong Zeng, and Zhangming Zhu. Eventpillars: Pillar-based efficient representations for event data. In *Proceedings of the AAAI Conference on Artificial Intelligence*, pages 2861–2869, 2025. 2

[4] Daniel Gehrig, Antonio Loquercio, Konstantinos G Derpanis, and Davide Scaramuzza. End-to-end learning of representations for asynchronous event-based data. In *Proceedings of the IEEE/CVF international conference on computer vision*, pages 5633–5643, 2019. 2

[5] Chao Li, Qiaoyong Zhong, Di Xie, and Shiliang Pu. Collaborative spatiotemporal feature learning for video action recognition. In *Proceedings of the ieee/cvf conference on computer vision and pattern recognition*, pages 7872–7881, 2019. 4

[6] Ze Liu, Jia Ning, Yue Cao, Yixuan Wei, Zheng Zhang, Stephen Lin, and Han Hu. Video swin transformer. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, pages 3202–3211, 2022. 5

[7] Henri Rebecq, Timo Horstschaefer, and Davide Scaramuzza. Real-time visual-inertial odometry for event cameras using keyframe-based nonlinear optimization. 2017. 2

[8] Wenhao Wu, Dongliang He, Tianwei Lin, Fu Li, Chuang Gan, and Errui Ding. Mvfnet: Multi-view fusion network for efficient video recognition. In *Proceedings of the AAAI conference on artificial intelligence*, pages 2943–2951, 2021. 4

[9] Nikola Zubić, Daniel Gehrig, Mathias Gehrig, and Davide

Scaramuzza. From chaos comes order: Ordering event representations for object recognition and detection. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pages 12846–12856, 2023. 2