Motion meets Attention: Video Motion Prompts

Qixiang Chen u7227010@anu.edu.au
Australian National University

Lei Wang* lei.w@anu.edu.au
Australian National University & Data61/CSIRO

Piotr Koniusz piotr.koniusz@data61.csiro.au
Data61/CSIRO & Australian National University

Tom Gedeon tom.gedeon@curtin.edu.au
Curtin University

Editors: Vu Nguyen and Hsuan-Tien Lin

Abstract

Videos contain rich spatio-temporal information. Traditional methods for extracting motion, used in tasks such as action recognition, often rely on visual contents rather than precise motion features. This phenomenon is referred to as 'blind motion extraction' behavior, which proves inefficient in capturing motions of interest due to a lack of motion-guided cues. Recently, attention mechanisms have enhanced many computer vision tasks by effectively highlighting salient visual areas. Inspired by this, we propose a modified Sigmoid function with learnable slope and shift parameters as an attention mechanism to modulate motion signals from frame differencing maps. This approach generates a sequence of attention maps that enhance the processing of motion-related video content. To ensure temporal continuity and smoothness of the attention maps, we apply pair-wise *temporal attention variation regularization* to remove unwanted motions (*e.g.*, noise) while preserving important ones. We then perform Hadamard product between each pair of attention maps and the original video frames to highlight the evolving motions of interest over time. These highlighted motions, termed *video motion prompts*, are subsequently used as inputs to the model instead of the original video frames. We formalize this process as a *motion prompt layer* and incorporate the regularization term into the loss function to learn better motion prompts. This layer serves as an adapter between the model and the video data, bridging the gap between traditional 'blind motion extraction' and the extraction of relevant motions of interest. We show that our lightweight, plug-and-play motion prompt layer seamlessly integrates into models like SlowFast, X3D, and TimeSformer, enhancing performance on benchmarks such as FineGym and MPII Cooking 2. **[Project website]** **[Code]**

Keywords: Motion; attention; prompt.

1. Introduction

Video-based research has gained popularity over the past several years due to its extensive applications in human-computer interaction, smart video surveillance, sports, and healthcare (Wang et al., 2020; Tang et al., 2023). Videos contain rich information: spatially, they include visual contents such as objects, human subjects, and scene layouts; temporally, they show the dynamics of how these objects and humans interact and evolve over time. Early

* Corresponding author.

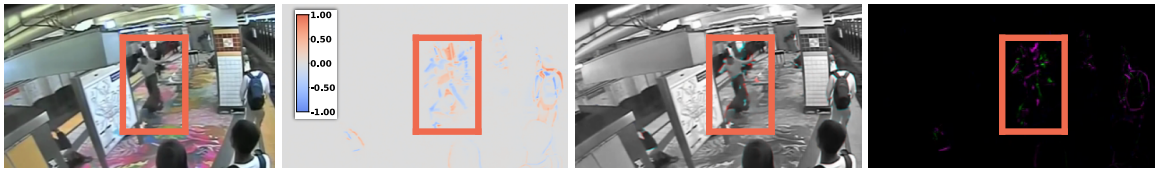


Figure 1: Phenomena of 'blind motion extraction'. 1st: Anomaly *fighting* from UCF-Crime (Sultani et al., 2018), highlighted by the orange bounding box. 2nd: Normalized frame differencing map, 3rd: Time-color reordering frame (Kim et al., 2022), 4th: Taylor video frame (Wang et al., 2024). These methods capture all motions without focusing on the anomaly. Non-motion-capturing methods focus on visual information, which is inefficient for motion-focused video processing.

works focused on using expert-designed, handcrafted descriptors to extract spatio-temporal information from videos (Dalal et al., 2006; Scovanner et al., 2007; Kläser et al., 2008; Wang and Schmid, 2013). However, while they were carefully designed, they could only handle simple contexts and were unable to generalize to other datasets even within the same domain. The major issue is that most descriptors do not focus on motions and heavily rely on visual contents. Compared to human vision systems in extracting information, they are now considered outdated, even though some are still in use (Wang and Schmid, 2013).

Deep learning has significantly advanced video-based research due to its end-to-end learnable nature (Simonyan and Zisserman, 2014; Tran et al., 2015; Feichtenhofer et al., 2016; Carreira and Zisserman, 2018). New architectures, such as CNNs, RNNs, and transformers (O'shea and Nash, 2015; Sherstinsky, 2020; Vaswani et al., 2017), along with components like normalization layers, skip connections, and dropout (Ba et al., 2016; He et al., 2016; Srivastava et al., 2014), and strategies like large-scale pretraining and transfer learning (Carreira and Zisserman, 2018; Zhuang et al., 2020), have greatly supported the evolution of modern video processing techniques. The attention mechanism has recently joined convolutional layers, MLPs, and RNNs as a fundamental building block (Vaswani et al., 2017). Initially used within transformers in natural language processing, the attention mechanism is now effectively applied to image and video processing tasks (Dosovitskiy et al., 2021; Arnab et al., 2021). It helps models focus on the most relevant parts of an image, relevant frames, spatial regions within frames, or significant scenes, thereby understanding both the visual contents and temporal dynamics. However, there are several challenges with attention mechanisms in video processing (Guo et al., 2022; Brauwers and Frasincar, 2021; Niu et al., 2021): (i) they are computationally intensive due to the calculation of spatio-temporal attention weights, (ii) scalability issues when handling videos of varying lengths, and (iii) capturing temporal dependencies is challenging, as the model must focus not only on spatial features within frames but also on the temporal relationships between frames. Moreover, while attentions provide some level of interpretability by highlighting important regions or frames, interpreting why certain regions or frames receive higher attention can be difficult in complex video tasks. Additionally, attention mechanisms must be robust to variations in video quality, lighting, occlusions, and other environmental factors.

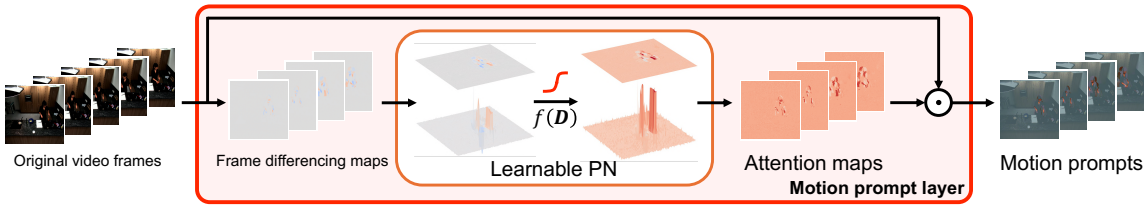


Figure 2: Overview of the motion prompt layer. Learnable Power Normalization (PN) function $f(\cdot)$ modulates motion, influencing how motion is enhanced or dampened in each frame differencing map $\mathbf{D}$ to highlight relevant movements. The resulting attention maps are multiplied element-wise ($\odot$) with original video frame to produce video motion prompts. We introduce a temporal attention variation regularization term (see Sec. 2.2) for smoother attention maps, ensuring better motion prompts. This layer can be inserted between the video input and backbones such as TimeSformer, serving as an adapter. Training involves optimizing both the motion prompt layer and the backbone network using a generic loss function, *e.g.*, cross-entropy, along with the new regularization term.

Despite large-scale datasets allowing attention-driven models to capture a wide range of spatio-temporal patterns, their generalization to unseen video data remains challenging.

Unlike the attention mechanism, prompt engineering involves designing prompts to guide language models in producing desired responses (Brown et al., 2020; Radford et al., 2021; Kim et al., 2021; Zhu et al., 2021). A well-designed prompt provides instructions or contextual clues that direct the model's output (Chen et al., 2023). This allows the attention mechanism to effectively allocate focus to the relevant parts of the input, especially in tasks where there may be many frames and objects, not all of which are relevant. This highlights the potential of using prompts in video processing tasks, where identifying the relevant motions can be challenging. In this paper, we establish a strong connection among video motion, attention and prompt engineering. We introduce the concept of motion prompts to address challenges related to efficiency, interpretability, and generalizability. We use a modified Sigmoid function with learnable slope and shift parameters as a power normalization function[1] to activate the motions: the slope controls the strength of modulation, determining whether to enhance or dampen the motions, and the shift acts as a threshold. The motions to be activated and modulated are represented as a sequence of frame differencing maps computed between consecutive frames. The activated per-frame motions can be viewed as an attention map, due to the enhancement of motion regions both spatially and temporally, as these regions evolve over time. To ensure that the generated attention maps are spatio-temporally smooth and continuous, we introduce a pair-wise temporal attention variation regularization to remove unwanted motions such as noise. We then perform the Hadamard product between each pair of attention maps and the corresponding original video frame, resulting in a sequence of highlighted video sequences, which we refer to as video motion prompts. Thus, our motion prompts are motion-dependent, rather

1. Technically, ours is not a true power normalization (PN), as PN equals 0 for a 0 input. Instead, ours is a shift-enabled, PN-inspired function.

than being dependent solely on the dataset. We formalize this process as a motion prompt layer, a plug-and-play component added between the model and the input data, bridging the gap between traditional, 'blind motion extraction' (see Fig. 1) and the extraction of relevant motions. We show that, with only two additional learnable parameters, our motion prompt layer significantly enhances action recognition. Our contributions are summarized as follows:

i. We introduce video motion prompts, defined as a sequence of spatio-temporally highlighted video frames. We format the extraction of motion prompts as a plug-and-play layer that can be inserted between video data input and the video model architecture, functioning as an adapter. This adapter bridges the gap between 'blind motion extraction' and the extraction of motions of interest.

ii. Our motion prompts are prompt-inspired, motion-dependent, and attention-driven. To generate smooth and continuous attention maps, we introduce a temporal attention variation regularization term. This term removes unwanted motions and enhances the model's generalization ability. We incorporate this regularization term into the loss function to improve the learning of motion prompts.

iii. Experimentally, we demonstrate that our motion prompt layer, despite its simplicity, can be integrated into popular video models to achieve state-of-the-art performance in tasks such as generic action recognition and fine-grained action recognition.

In Appendix A, we review closely related work on motion extraction, attention mechanisms, prompts, and adapter layers for video processing. We also highlight the significant differences of our approach compared to these works. Hereafter, we introduce our approach.

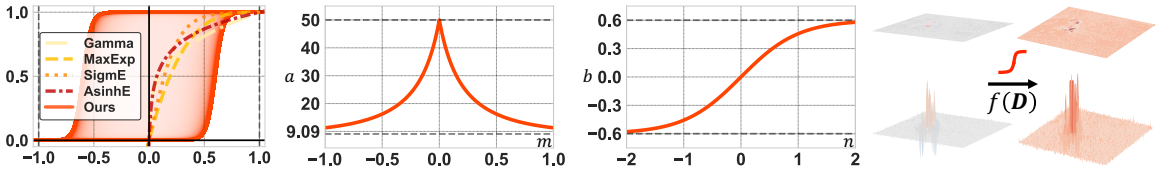
2. Approach

First, we present our notation. An overview of our motion prompt layer is provided in Fig. 2. In Appendix B, we provide preliminary information on activation functions, a mathematical view of attention mechanisms, and the power normalization family.

Notation. Let $\mathcal{I}_T$ stand for the index set $\{1, 2, \ldots, T\}$. Scalars are in regular fonts; vectors are denoted by lowercase boldface letters, *e.g.*, $\boldsymbol{x}$; matrices by uppercase boldface, *e.g.*, $\boldsymbol{X}$; tensors by calligraphic letters, *e.g.*, $\mathbf{X}$. Let $\mathbf{X} \in \mathbb{R}^{d_1 \times d_2 \times d_3}$ denote a third-order tensor, using the Matlab convention, we refer to its t-th slice as $\mathbf{X}_{:,:,t}$, which is a $d_1 \times d_2$ matrix.

2.1. Learnable Power Normalization

Frame differencing maps. For a T-frame video $\mathbf{X} = [\mathbf{F}_1, \mathbf{F}_2, \ldots, \mathbf{F}_T] \in \mathbb{R}^{H \times W \times 3 \times T}$ where $\mathbf{F}_t \in \mathbb{R}^{H \times W \times 3}$ ($t \in \mathcal{I}_T$), H and W denote the frame height and width, respectively, we first convert it into a grayscale video sequence $\mathbf{X}' = [\boldsymbol{F}'_1, \boldsymbol{F}'_2, \ldots, \boldsymbol{F}'_T] \in \mathbb{R}^{H \times W \times T}$. After normalizing the pixel values between 0 and 1, we compute the frame differencing maps between consecutive frames, resulting in $\mathbf{D} = [\boldsymbol{D}_1, \boldsymbol{D}_2, \ldots, \boldsymbol{D}_{T-1}] \in \mathbb{R}^{H \times W \times (T-1)}$, where $\boldsymbol{D}_t = \boldsymbol{F}'_{t+1} - \boldsymbol{F}'_t$ ($t \in \mathcal{I}_{T-1}$). Note that the pixel values in $\boldsymbol{D}_t$ can be either positive or negative. Positive values indicate areas where the pixel intensity has increased from frame t to frame $t+1$, while negative values indicate areas where the pixel intensity has decreased. Note that the pixel values in frame differencing maps are in the range of $[-1, 1]$.



(a) PNs *vs.* ours. (b) $a(m)$ as slope. (c) $b(n)$ as shift. (d) Before & after PN

Figure 3: (a) Comparison of existing well-behaved Power Normalization (PN) functions (Koniusz and Zhang, 2021) and our learnable PN function (Eq. (3)). Our PN function is depicted in orange with shadows, showing its learnable nature and potential shifts. The learnable slope function $a(m)$ in (b) controls the steepness of PN, determining the degree of enhancement or dampening of motions, while the learnable shift function $b(n)$ in (c) determines the threshold for PN, affecting whether the motions are enhanced or dampened. (d) shows a surface plot of pixel value changes before and after applying our PN function.

The frame differencing maps **D**, record a sequence of motions between consecutive video frames, capturing both foreground and background motions, as well as noisy patterns. Depending on the task, we aim to enhance the motions of interest while suppressing the rest. For example, in human action recognition, we want to amplify the motions associated with human actions and reduce irrelevant motions. Below, we introduce our learnable Sigmoid activation function as a Power Normalization (PN) function for motion modulation. Fig. 3 (a) compares existing PN functions with our learnable PN function. Appendix E presents visual comparisons of the motions modulated by these PN functions, including ours. We observe that our PN captures different motions across various video types.

Learnable slope and shift parameters. We opt for a modified Sigmoid function with learnable slope a and shift b as a PN function (see Fig. 3 (a)) on frame differencing maps. For a given frame differencing map $\boldsymbol{D}$, we define:

$$f(\boldsymbol{D}) = \frac{1}{1 + e^{-a(\boldsymbol{D}-b)}} \tag{1}$$

This element-wise function maps the pixel values of frame differencing maps from $[-1, 1]$ to $[0, 1]$. For simplicity, we set $a > 0$ so that $f(\cdot)$ is a monotone increasing function. The parameter a controls the slope of the Sigmoid function[2], and it influences how sharply the function transitions from its minimum to its maximum value (see Fig. 3(b)). The parameter b acts as the threshold or the point of inflection of the Sigmoid function[3], and it determines the position of the Sigmoid curve along the $\boldsymbol{D}$ axis. We allow b to shift either left or right, hence it can be positive or negative (see Fig. 3(c)). However, directly learning the parameters a and b presents challenges, such as initialization sensitivity and uncertainty in the parameter search space. To address these issues, we design the following two mapping

2. The value of a determines the sensitivity of the function $f(\boldsymbol{D})$ to changes in $\boldsymbol{D}$ around the threshold b: large a results in a steep slope with a sharp transition and high sensitivity to changes, whereas small a results in a gentle slope with a smooth transition and lower sensitivity to changes.
3. The value of b determines the horizontal position of the Sigmoid function: larger b shifts the curve to the right, requiring higher $\boldsymbol{D}$ values to reach the midpoint; smaller b shifts the curve to the left, requiring lower $\boldsymbol{D}$ values to reach the midpoint.

functions to learn m and n instead:

$$\begin{cases} a(m) = \dfrac{\alpha}{\beta\,|\tanh(m)| + \epsilon} \\ b(n) = \gamma \tanh(n) \end{cases} \tag{2}$$

Here, $\tanh(\cdot)$ is the hyperbolic tangent function, which maps any given m and n to values between -1 and 1. This alignment with the pixel value range in frame differencing maps facilitates better adjustment of the slope and shift for motion modulation. The symbol $|\cdot|$ denotes the absolute value operation, and ϵ is a small constant for numerical stabilization (we set $\epsilon = 0.1$). Parameters $\alpha > 0$, $\beta > 0$, and $\gamma > 0$ control the characteristics of our PN function: α and β adjusts the rate at which Eq. (1) transitions between 0 and 1, while γ in $b(n)$ controls the shifts of input before it is processed by the exponential function. This setup allows for unrestricted learning of m and n, avoiding potential issues caused by the bounded pixel values in $\boldsymbol{D}$. We prove the boundedness and differentiability of both $a(m)$ and $b(n)$ in Appendix C. Combining Eq. (1) and (2) results in our learnable PN function:

$$f(\boldsymbol{D}) = \frac{1}{1 + e^{-\left(\frac{\alpha}{\beta|\tanh(m)|+\epsilon}\right)(\boldsymbol{D} - \gamma \tanh(n))}}. \tag{3}$$

We provide a detailed analysis of parameter constraints and sensitivity concerning α, β, and γ, their impact on the learning of m and n, and the rationale behind selecting these three parameters in Appendix D. Specifically, we set $\alpha = 5$, $\beta = 0.45$, and $\gamma = 0.6$. Below we show that for pixel values within interval $[-1, 1]$ in $\boldsymbol{D}$, our function $f(\boldsymbol{D})$ is always bounded within $[0, 1]$. Additionally, we show that Eq. (3) is a well-behaved PN function.

Upper and lower bound. Consider the term inside the exponential: $\boldsymbol{A} = a(m)(\boldsymbol{D} - b(n))$, we first examine the derivative of $\boldsymbol{A}$ with respect to $\boldsymbol{D}$: $\frac{\partial \boldsymbol{A}}{\partial \boldsymbol{D}} = a(m) = \frac{\alpha}{\beta|\tanh(m)|+\epsilon}$. Since this derivative is always positive, $\boldsymbol{A}$ is monotonically increasing with respect to $\boldsymbol{D}$. The scaling factor $a(m)$ ranges from $\frac{\alpha}{\beta+\epsilon}$ to $\frac{\alpha}{\epsilon}$ and $\boldsymbol{D} - b(n)$ ranges from $-1-\gamma$ to $1+\gamma$; hence, the lower and upper bounds of $\boldsymbol{A}$ are $-\frac{\alpha(1+\gamma)}{\beta+\epsilon}$ and $\frac{\alpha(1+\gamma)}{\epsilon}$, respectively.
The Sigmoid function $\sigma(\boldsymbol{A}) = \frac{1}{1+e^{-\boldsymbol{A}}}$ approaches 0 as value in $\boldsymbol{A}$ becomes large and negative (*e.g.*, $\sigma(-\frac{5(1+0.6)}{0.45+0.1}) \approx 0.0$), and 1 as value in $\boldsymbol{A}$ becomes large and positive (*e.g.*, $\sigma(\frac{5(1+0.6)}{0.1}) \approx 1.0$). Consequently, for values in $\boldsymbol{D}$ in the interval $[-1, 1]$, the function $f(\boldsymbol{D})$ is always bounded within $[0, 1]$.
Well-behaved power normalization. Eq. (3) is continuous and smooth for all real values of $\boldsymbol{D}$, m, and n. The scaling factor $a(m)$ is always positive, and the exponential term is well-defined, producing a valid, finite value for all input values. The Sigmoid function $\sigma(\boldsymbol{A}) = \frac{1}{1+e^{-\boldsymbol{A}}}$ maps any real number to the interval $[0, 1]$, ensuring that $f(\boldsymbol{D})$ produces values within this range and thereby maintaining proper normalization.
Therefore, Eq. (3) is a well-behaved PN function given: (i) It is continuous and smooth. (ii) The output values are properly normalized within the range $[0, 1]$. (iii) The Sigmoid function ensures that the function maps real numbers to a bounded interval, maintaining normalization.

2.2. Motion Prompt Layer: An Adapter

Video motion prompts. Eq. (3) modulates the motions in frame differencing maps via the learnable m and n, resulting in a sequence of normalized frame differencing maps with pixel values ranging from 0 to 1. This element-wise PN process can also be viewed as activating the motions of interest, guided by the generic loss function, *e.g.*, cross-entropy, hence we call this the attention map. The PN process produces a sequence of attention maps: $\mathbf{A}' = [f(\boldsymbol{D}_1), f(\boldsymbol{D}_2), \ldots, f(\boldsymbol{D}_{T-1})] \in \mathbb{R}^{H\times W\times(T-1)}$, spatially highlighting regions where motions are of interest (*e.g.*, 1) and dampening motions that are not of interest (*e.g.*, 0); temporally showing the evolution of attention maps over time. We then duplicate each attention map three times, resulting in $\mathbf{A}'^{(3)} = [f^{(3)}(\boldsymbol{D}_1), f^{(3)}(\boldsymbol{D}_2), \ldots, f^{(3)}(\boldsymbol{D}_{T-1})] \in \mathbb{R}^{H\times W\times 3\times(T-1)}$, where $f^{(3)}(\boldsymbol{D}_t) \in \mathbb{R}^{H\times W\times 3}$ and $t \in \mathcal{I}_{T-1}$. We perform a channel-wise Hadamard product between each duplicated attention map $f^{(3)}(\boldsymbol{D}_t)$ and the original video frame $\boldsymbol{F}_{(t+1)}$ so that attention attends to each channel of the original video frame, resulting in a sequence of highlighted video frames:

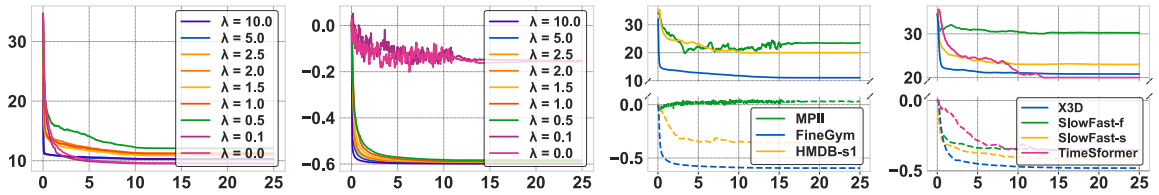
$$\begin{aligned}\mathbf{Z} &= \mathbf{A}'^{(3)} \odot \mathbf{X}_{:,:,2:} \\ &= [f^{(3)}(\boldsymbol{D}_1), f^{(3)}(\boldsymbol{D}_2), \ldots, f^{(3)}(\boldsymbol{D}_{T-1})] \odot [\mathbf{F}_2, \ldots, \mathbf{F}_T] \\ &= [f^{(3)}(\boldsymbol{D}_1) \odot \mathbf{F}_2, f^{(3)}(\boldsymbol{D}_2) \odot \mathbf{F}_3, \ldots, f^{(3)}(\boldsymbol{D}_{T-1}) \odot \mathbf{F}_T],\end{aligned} \tag{4}$$

where $\odot$ denotes the Hadamard (element-wise) product. $\mathbf{Z} \in \mathbb{R}^{H\times W\times 3\times(T-1)}$ denotes the newly generated video, referred to as Video Motion Prompts (VMPs), which are motion-dependent, attention-driven, and provide rich motion cues. $f^{(3)}(\boldsymbol{D}_t) \odot \mathbf{F}_{(t+1)} \in \mathbb{R}^{H\times W\times 3}$ shows the motion prompt for the t-th frame. Below, we show that our video motion prompt generation process is, in fact, connected to the attention mechanism.

Connecting to attention mechanism. We rewrite $f^{(3)}(\boldsymbol{D}_t)$ using the Sigmoid function $\sigma^{(3)}(\cdot)$, replace the frame differencing map $\boldsymbol{D}_t$ with the grayscale conversion function $h(\cdot)$ as $h(\mathbf{F}_{t+1}) - h(\mathbf{F}_t)$, and rewrite Eq (4) in the form of per-frame motion prompt (we use Eq. (2) for the scaling factor and shift parameter and omit m and n for simplicity):

$$\begin{aligned}\mathbf{Z}_t &= f^{(3)}(\boldsymbol{D}_t) \odot \mathbf{F}_{(t+1)} \\ &= \sigma^{(3)}(a[h(\mathbf{F}_{t+1}) - h(\mathbf{F}_t) - b]) \odot \mathbf{F}_{(t+1)}\end{aligned} \tag{5}$$

where $\boldsymbol{A} = a[h(\mathbf{F}_{t+1}) - h(\mathbf{F}_t) - b]$ can be viewed as an attention matrix with the shifting parameter b modulating the pixel intensity changes between each pair of grayscaled consecutive frames. The Sigmoid (denoted as $\sigma^{(3)}$) outputs are similar to the Softmax function outputs in the standard attention mechanism (as in Eq. (9) of Appendix B), which transforms raw attention scores to highlight the most important parts of the h-transformed frame differencing maps. Since we operate on each pixel in frame differencing maps to highlight all relevant motion pixels, there is no need to satisfy the criterion that attention weights sum up to 1, as required by the Softmax function. $\mathbf{F}_{(t+1)}$ is analogous to the value matrix $\boldsymbol{V}$ in Eq. (10) of Appendix B. This shows that our motion prompt generation process is analogous to the standard attention mechanism.



(*a*) *a vs.* epoch (*b*) *b vs.* epoch (*c*) Across datasets (*d*) Across backbones

Figure 4: Effects of the penalty parameter λ on (a) slope a and (b) shift b are evaluated using FineGym with TimeSformer pretrained on Kinetics-600. Larger λ values cause both a and b to approach their lower bounds. In contrast, when smaller λs are used, b becomes noisy and fluctuates, while a assumes lower values. (c) and (d) show the learned a (solid lines) and b (dashed lines) of top performers across different datasets (with TimeSformer) and backbones (on HMDB-51 split 1). In (d), SlowFast-f and SlowFast-s denote motion prompt layer added to the fast-only (green color) and slow-only (yellow color) stream, respectively.

Unlike the traditional attention mechanism, our attention mechanism offers (i) lightweight computation with only two learnable parameters, (ii) interpretability, as the learnable scaling factor and shift parameter have well-explainable functionalities in the motion modulation process, and (iii) generalizability, as $\boldsymbol{A}$ is motion-dependent, relying on frame differencing maps rather than being dataset-dependent.

If attention scores in $\sigma^{(3)}(\boldsymbol{A})$ are all 1, *e.g.*, $\boldsymbol{A}$ becomes large and positive due to motion modulation via slope a and shift b, thus reverting to the use of original video frames.

Temporal attention variation regularization. To ensure temporally the smoothness and continuity of generated attention maps, we introduce temporal attention variation regularization on pair-wise attention maps:

$$\mathcal{V} = \frac{1}{T-2}\sum_{t=1}^{T-2} ||f(\boldsymbol{D}_{t+1}) - f(\boldsymbol{D}_t)||_F^2, \tag{6}$$

where $||\cdot||_F$ denotes the Frobenius norm. Eq. (6) reduces pixel variations between consecutive attention maps, ensuring temporal smoothness while preserving key motion regions.

We design the video motion prompt generation process as a single layer with two learnable parameters that amplify relevant motions while attenuating irrelevant movements. Eq. (6) is incorporated into the original loss function $\mathcal{L}_{\text{ori}}$, such as cross-entropy loss for action recognition, used in models like SlowFast and TimeSformer backbones:

$$\mathcal{L} = \mathcal{L}_{\text{ori}} + \lambda\mathcal{V}, \tag{7}$$

where λ is a penalty parameter that controls the strength of this regularization, balancing the trade-off between temporal smoothness and the maintenance of spatially significant motion regions. We simply insert our motion prompt layer between the video input and the model architecture, using Eq. (7) as the loss function to learn the VMPs as new inputs. The entire model can be learned in an end-to-end manner or fine-tuned on specific layers, including the learning of our motion prompt layer.

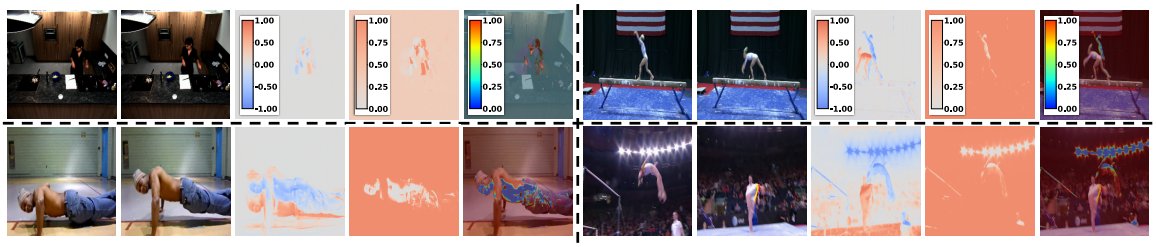


Figure 5: Visualizations of consecutive frames (columns 1-2 and 6-7), frame differencing maps (columns 3 and 8), attention maps (columns 4 and 9), and motion prompts (columns 5 and 10). Top left shows *shake* (MPII), bottom left shows *pushup* (HMDB-51), and top and bottom right show *balance beam* and *uneven bar* (FineGym). Frame differencing maps are noisy; our attention maps, guided by learned slope and shift, are cleaner. Motion prompts (columns 5 and 10) contain richer motion information than the original frames (columns 2 and 7). We notice that for static camera, the attention map shows light orange, indicating that background information is not important for the action (*e.g.*, *shake* in MPII). For moving cameras, the background information is important, hence it appears darker red, receiving higher attention scores. Additional visualizations are in Appendix E.

3. Experiment

3.1. Setup

Dataset. For generic action recognition, we choose the popular and challenging HMDB-51 (Kuehne et al., 2011), which features significant camera and background motion. For fine-grained action recognition, we select the large-scale MPII Cooking 2 (2,881,616 frames, resolution 1624×1224) (Rohrbach et al., 2015) and FineGym (Shao et al., 2020) (Gym99 v1.1: 26,320/8,521 for train/val set, respectively) datasets. FineGym focuses on human actions performed in a gym environment, capturing a wide range of activities with fine granularity (99 classes). In contrast, the MPII Cooking 2 dataset specializes in cooking-related actions (67 classes). We adhere to their standard evaluation protocols in our experiments.

Implementation. The motion prompt layer is initialized with a normal distribution, having a mean of 1×10^{-5} and a standard deviation of 1. The penalty parameter for temporal attention variation regularization is selected from the range [1e-4, 10]. We use SlowFast (Feichtenhofer et al., 2019), C2D, I3D (Carreira and Zisserman, 2018), X3D (Feichtenhofer, 2020), and TimeSformer (Bertasius et al., 2021) as backbones. All experiments use SGD as the optimizer (*e.g.*, with momentum 0.9). The learning rate (*e.g.*, 0.005), weight decay (*e.g.*, 0.0001), decay strategy (*e.g.*, step decay, or cosine decay with a warm-up), and the number of sampled video frames per video follow those specified in the original papers. Note that our layer requires an additional frame to ensure that the resulting motion prompts have exactly the same length as the original input video frames (see Eq. (4)). We fine-tune models pretrained on Kinetics-400 (Kay et al., 2017) (or Kinetics-600 (Long et al., 2020)) as our baseline. All experiments are conducted using the Tesla V100 GPU. Below, we present our evaluations and analysis.

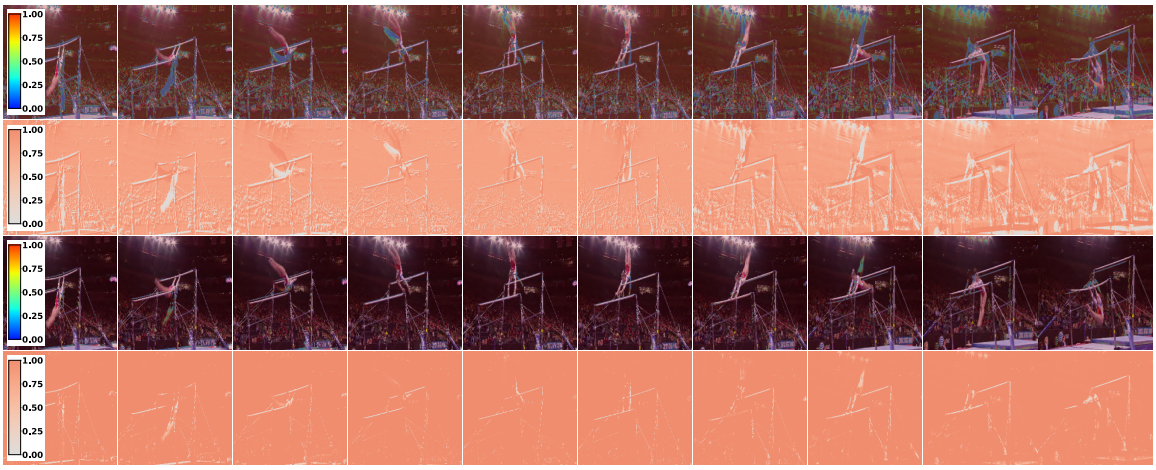


Figure 6: Effects of the regularization term. We use the *uneven bar* action from FineGym for visualization. The first two rows show motion prompts and attention maps without regularization ($\lambda = 0$). The last two rows show results with regularization ($\lambda = 2.5$). Regularization removes unnecessary motion details, resulting in smoother and cleaner attention maps. More visualizations are in Appendix E.

3.2. Evaluation

Analysis of learnable slope and shift. We present the learning process of a and b versus the number of fine-tuning epochs in Fig. 4 with varying regularization penalty parameter λ. We use FineGym with TimeSformer pretrained on Kinetics-600 for fine-tuning with our motion prompts. As shown in the figure, choosing bigger λ results in both slope and shift parameters quickly approaching their lower bounds. Using smaller λ or set λ to 0 results in the noisy and fluctuated learning process for b (red and purple lines in Fig. 4 (b)), and a tends to be slightly bigger (green line in Fig. 4 (a)). Overall, on FineGym, a and b tend to be small, that is to consider some negative motions in the frame differencing maps while ensuring a smooth transition rather than an increase in steepness. This is reasonable as FineGym is captured by moving cameras, hence all positive motions should be considered with varying degrees of attention. The optimal value of λ is 2.5, and the learned values are $a = 11.04$ and $b = -0.59$, resulting in a performance gain of 0.8% compared to the baseline.

Learned a and b across various datasets and backbones. We visualize the pairs of a and b from top performers per dataset using the TimeSformer backbone in Fig. 4 (c). We notice that MPII Cooking 2 tends to have higher a and b values. This is attributed to the dataset being captured by a static camera, making it easier to extract motions related to cooking activities, *e.g.*, with a steep slope. On FineGym, both slope and shift tend to be smaller compared to HMDB-51 split 1 (HMDB-s1). This is because FineGym focuses specifically on gymnastic activities, where significant camera motions occur due to player localization and tracking. In Fig. 4 (d), we observe that the learned a and b vary significantly across different backbones. Moreover, using motion prompts on the SlowFast fast-only (SlowFast-f) stream results in a steeper slope compared to the SlowFast slow-only

Table 1: Evaluations are conducted on (*left*) HMDB-51, and (*right*) FineGym, MPII Cooking 2, using SlowFast, C2D, I3D, X3D and TimeSformer as backbones. For SlowFast, we explore three variants by adding motion prompts into the slow-only stream, fast-only stream, and both slow and fast streams. *K600* denotes that the Kinetics-600 pretrained model is used. We highlight improvements in red.

Model	HMDB-51			Mean
	Split 1	Split 2	Split 3	
SlowFast	75.4	76.2	76.9	76.2
+**VMPs** (slow-only)	**76.8**$^{\uparrow 1.4}$	**77.0**$^{\uparrow 0.8}$	**77.3**$^{\uparrow 0.4}$	**77.0**$^{\uparrow 0.8}$
+**VMPs** (fast-only)	**76.5**$^{\uparrow 1.1}$	**77.4**$^{\uparrow 1.2}$	**77.1**$^{\uparrow 0.2}$	**77.0**$^{\uparrow 0.8}$
+**VMPs** (slow&fast)	**76.2**$^{\uparrow 0.8}$	**76.7**$^{\uparrow 0.5}$	**77.1**$^{\uparrow 0.2}$	**76.6**$^{\uparrow 0.4}$
C2D	67.7	66.9	66.1	66.9
+**VMPs**	**69.4**$^{\uparrow 1.7}$	**68.3**$^{\uparrow 1.4}$	**66.9**$^{\uparrow 0.8}$	**68.2**$^{\uparrow 1.3}$
I3D	70.1	69.7	69.2	69.7
+**VMPs**	**70.5**$^{\uparrow 0.4}$	**70.5**$^{\uparrow 0.8}$	**70.2**$^{\uparrow 1.0}$	**70.4**$^{\uparrow 0.7}$
X3D	75.0	72.6	73.4	73.7
+**VMPs**	**75.8**$^{\uparrow 0.8}$	**73.2**$^{\uparrow 0.6}$	**73.6**$^{\uparrow 0.2}$	**74.2**$^{\uparrow 0.5}$
TimeSformer	70.0	72.1	70.8	71.0
+**VMPs**	**72.7**$^{\uparrow 2.7}$	**73.8**$^{\uparrow 1.7}$	**70.9**$^{\uparrow 0.1}$	**72.5**$^{\uparrow 1.5}$
TimeSformer (*K600*)	72.7	73.1	72.2	72.7
+**VMPs**	**74.2**$^{\uparrow 1.5}$	**74.3**$^{\uparrow 1.2}$	**72.9**$^{\uparrow 0.7}$	**73.8**$^{\uparrow 1.1}$

Model	FineGym		MPII Cooking 2	
	Top-1	Top-5	Top-1	Top-5
SlowFast	**89.8**	99.2	52.9	**86.1**
+**VMPs** (slow-only)	89.7$^{\downarrow 0.1}$	99.2	**55.5**$^{\uparrow 2.6}$	84.5$^{\downarrow 1.6}$
+**VMPs** (fast-only)	**90.3**$^{\uparrow 0.5}$	**99.3**$^{\uparrow 0.1}$	**55.2**$^{\uparrow 2.3}$	84.0$^{\downarrow 2.1}$
+**VMPs** (slow&fast)	**90.1**$^{\uparrow 0.3}$	**99.3**$^{\uparrow 0.1}$	**56.8**$^{\uparrow 3.9}$	**86.6**$^{\uparrow 0.5}$
C2D	79.7	97.5	48.8	**83.4**
+**VMPs**	**81.3**$^{\uparrow 0.6}$	**97.8**$^{\uparrow 0.3}$	**50.9**$^{\uparrow 2.1}$	82.1$^{\downarrow 1.3}$
I3D	82.4	98.3	53.1	82.9
+**VMPs**	**84.7**$^{\uparrow 2.3}$	**98.4**$^{\uparrow 0.1}$	**56.1**$^{\uparrow 3.0}$	**85.7**$^{\uparrow 2.8}$
X3D	83.0	98.4	48.4	**80.8**
+**VMPs**	**83.8**$^{\uparrow 0.8}$	**98.6**$^{\uparrow 0.2}$	**49.1**$^{\uparrow 0.7}$	80.6$^{\downarrow 0.2}$
TimeSformer	83.5	98.5	50.0	79.4
+**VMPs**	**83.8**$^{\uparrow 0.3}$	**98.6**$^{\uparrow 0.1}$	**55.2**$^{\uparrow 5.2}$	**82.5**$^{\uparrow 3.1}$
TimeSformer (*K600*)	83.6	**98.7**	50.6	81.8
+**VMPs**	**84.4**$^{\uparrow 0.8}$	98.5$^{\downarrow 0.2}$	**56.6**$^{\uparrow 6.0}$	**84.4**$^{\uparrow 2.6}$

(SlowFast-s) stream. This is because the fast-only stream samples more frames, offering richer and smoother temporal information that facilitates easier access to motions of interest.

Visualizations of attention maps and motion prompts. We visualize frame difference maps, learned attention maps, and motion prompts in Fig. 5. We also include the original video frames for comparison. As shown in the figure, we observe discrepancies between consecutive frames, attributable to the frame sampling strategy commonly used in video processing tasks. The frame differencing maps show noticeable noise; blue indicates negative motions while orange shows positive motions, especially in videos captured by moving cameras like HMDB-51 and FineGym. Conversely, in static camera scenes such as MPII Cooking 2, the background appears clean. Consequently, in the generated attention maps, the background is depicted in lighter orange, suggesting lower attention scores and less importance relative to the action. In contrast, in scenarios with moving cameras, background motions appear more significant, reflected by darker red shades in the attention maps, indicating higher attention scores. Interestingly, the generated motion prompts reveal rich action information, compared to original frames.

With and without temporal attention variation regularization. Fig. 6 shows a comparison of with and without the use of temporal attention variation regularization. We observe that without the regularization term, the generated attention maps are quite noisy, especially in the background. However, with regularization, the attention maps contain much less noise. This demonstrates that our regularization term contributes to generating smooth and clean attention maps, thereby improving the quality of motion prompts. We also observe that the attention maps exhibit several interesting patterns: (i) they highlight the motion regions in the current frame, (ii) they capture potential movements from the previous

frame, and (iii) they attend to background scenes affected by camera motions. These observations indicate that our attention maps, guided by only two learnable parameters, effectively highlight visual contents of interest while capturing dynamics over short periods of time. More visualizations and discussions can be found in Appendix E.

Generic action recognition. Our evaluations on HMDB-51 are summarized in Table 1 (*left*). Using TimeSformer as the backbone and integrating our motion prompt layer, we achieve accuracy improvements of 1.5%, 1.2%, and 0.7% for split 1, 2 and 3, respectively. On average, this results in a performance gain of 1.1%. Our VMPs show consistent improvements across various recent action recognition backbones.

Fine-grained action recognition. In Table 1 (*right*), we report performance gains on MPII Cooking 2, with Top-1 mean average precision improvements of 3.9% for SlowFast, 2.1% for C2D, 3.0% for I3D, 0.7% for X3D, and 6.0% for TimeSformer. For FineGym, TimeSformer shows a 0.8% increase in Top-1 accuracy. The TimeSformer backbone consistently outperforms the X3D backbone on both datasets, indicating that model performance and the benefits of VMPs depend on the backbone's ability to handle motion.

4. Conclusion

We introduce *video motion prompts* to enhance action recognition. We use a modified Sigmoid activation function with learnable slope and shift as a power normalization function on frame differencing maps to activate motions as attention maps. Additionally, we introduce the *temporal attention variation regularization* term to generate more accurate and smooth motion prompts. We formalize the entire process as a single *motion prompt layer* acting as an adapter, resulting in improved performance across various benchmarks and backbones.

Acknowledgments

Qixiang Chen conducted this research under the supervision of Lei Wang for his final year honors research project at ANU. He is a recipient of The Active Intelligence Research Challenge Award. This work was also supported by the NCI Adapter Scheme Q4 2023, the NCI National AI Flagship Merit Allocation Scheme, and the National Computational Merit Allocation Scheme 2024 (NCMAS 2024), with computational resources provided by NCI Australia, an NCRIS-enabled capability supported by the Australian Government.

References

Anurag Arnab, Mostafa Dehghani, Georg Heigold, Chen Sun, Mario Lučić, and Cordelia Schmid. Vivit: A video vision transformer. In *ICCV*, pages 6836–6846, 2021.

Jimmy Lei Ba, Jamie Ryan Kiros, and Geoffrey E Hinton. Layer normalization. *arXiv preprint arXiv:1607.06450*, 2016.

Gedas Bertasius, Heng Wang, and Lorenzo Torresani. Is space-time attention all you need for video understanding? In *ICML*, July 2021.

Hakan Bilen, Basura Fernando, Efstratios Gavves, Andrea Vedaldi, and Stephen Gould. Dynamic image networks for action recognition. In *CVPR*, June 2016.

Gianni Brauwers and Flavius Frasincar. A general survey on attention mechanisms in deep learning. *IEEE Transactions on Knowledge and Data Engineering*, 35(4):3279–3298, 2021.

Tom Brown, Benjamin Mann, Nick Ryder, Melanie Subbiah, Jared D Kaplan, Prafulla Dhariwal, Arvind Neelakantan, Pranav Shyam, Girish Sastry, Amanda Askell, et al. Language models are few-shot learners. *NeurIPS*, 33:1877–1901, 2020.

João Carreira and Andrew Zisserman. Quo Vadis, Action Recognition? A New Model and the Kinetics Dataset. *CVPR*, pages 1–10, 2018.

Banghao Chen, Zhaofeng Zhang, Nicolas Langrené, and Shengxin Zhu. Unleashing the potential of prompt engineering in large language models: a comprehensive review. *arXiv preprint arXiv:2310.14735*, 2023.

Chun-Fu Chen, Quanfu Fan, and Rameswar Panda. Crossvit: Cross-attention multi-scale vision transformer for image classification. *CoRR*, abs/2103.14899, 2021.

Navneet Dalal, Bill Triggs, and Cordelia Schmid. Human Detection Using Oriented Histogram of Flow and Appearance. *ECCV*, pages 428–441, 2006.

Ning Ding, Yujia Qin, Guang Yang, Fuchao Wei, Zonghan Yang, Yusheng Su, Shengding Hu, Yulin Chen, Chi-Min Chan, Weize Chen, et al. Delta tuning: A comprehensive study of parameter efficient methods for pre-trained language models. *arXiv preprint arXiv:2203.06904*, 2022.

Alexey Dosovitskiy, Lucas Beyer, Alexander Kolesnikov, Dirk Weissenborn, Xiaohua Zhai, Thomas Unterthiner, Mostafa Dehghani, Matthias Minderer, Georg Heigold, Sylvain Gelly, Jakob Uszkoreit, and Neil Houlsby. An image is worth 16x16 words: Transformers for image recognition at scale. In *ICLR*, 2021.

Shiv Ram Dubey, Satish Kumar Singh, and Bidyut Baran Chaudhuri. Activation functions in deep learning: A comprehensive survey and benchmark. *Neurocomputing*, 503:92–108, 2022.

Christoph Feichtenhofer. X3d: Expanding architectures for efficient video recognition. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, pages 203–213, 2020.

Christoph Feichtenhofer, Axel Pinz, and Richard P. Wildes. Spatiotemporal residual networks for video action recognition. In *NeurIPS*, pages 3468–3476, 2016.

Christoph Feichtenhofer, Haoqi Fan, Jitendra Malik, and Kaiming He. Slowfast networks for video recognition. In *ICCV*, pages 6202–6211, 2019.

Meng-Hao Guo, Tian-Xing Xu, Jiang-Jiang Liu, Zheng-Ning Liu, Peng-Tao Jiang, Tai-Jiang Mu, Song-Hai Zhang, Ralph R Martin, Ming-Ming Cheng, and Shi-Min Hu. Attention mechanisms in computer vision: A survey. *CVM*, 8(3):331–368, 2022.

Ryota Hashiguchi and Toru Tamaki. Vision transformer with cross-attention by temporal shift for efficient action recognition, 2022.

Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. Deep residual learning for image recognition. In *CVPR*, pages 770–778, 2016.

Oleksii Hrinchuk, Valentin Khrulkov, Leyla Mirvakhabova, Elena Orlova, and Ivan Oseledets. Tensorized embedding layers for efficient model compression. *arXiv preprint arXiv:1901.10787*, 2019.

Yaosi Hu, Chong Luo, and Zhenzhong Chen. Make it move: Controllable image-to-video generation with text descriptions. In *CVPR*, 2022.

Sergey Ioffe and Christian Szegedy. Batch normalization: Accelerating deep network training by reducing internal covariate shift. In *ICML*, pages 448–456. pmlr, 2015.

Hervé Jégou, Matthijs Douze, and Cordelia Schmid. On the burstiness of visual elements. In *CVPR*, pages 1169–1176. IEEE, 2009.

Menglin Jia, Luming Tang, Bor-Chun Chen, Claire Cardie, Serge Belongie, Bharath Hariharan, and Ser-Nam Lim. Visual prompt tuning. In *ECCV*, pages 709–727. Springer, 2022.

Chen Ju, Tengda Han, Kunhao Zheng, Ya Zhang, and Weidi Xie. Prompting visual-language models for efficient video understanding. In *ECCV*, pages 105–124. Springer, 2022.

Will Kay, Joao Carreira, Karen Simonyan, Brian Zhang, Chloe Hillier, Sudheendra Vijayanarasimhan, Fabio Viola, Tim Green, Trevor Back, Paul Natsev, et al. The kinetics human action video dataset. *arXiv preprint arXiv:1705.06950*, 2017.

Kiyoon Kim, Shreyank N Gowda, Oisin Mac Aodha, and Laura Sevilla-Lara. Capturing temporal information in a single frame: Channel sampling strategies for action recognition. In *BMVC*. BMVA Press, 2022.

Wonjae Kim, Bokyung Son, and Ildoo Kim. Vilt: Vision-and-language transformer without convolution or region supervision. In *ICML*, pages 5583–5594. PMLR, 2021.

Alexander Kläser, Marcin Marszalek, and Cordelia Schmid. A Spatio-Temporal Descriptor Based on 3D-Gradients. *BMCV*, pages 1–10, 2008.

Piotr Koniusz and Hongguang Zhang. Power normalizations in fine-grained image, few-shot image and graph classification. *IEEE TPAMI*, 44(2):591–609, 2021.

Hildegard Kuehne, Hueihan Jhuang, Estíbaliz Garrote, Tomaso Poggio, and Thomas Serre. Hmdb: a large video database for human motion recognition. In *ICCV*, pages 2556–2563. IEEE, 2011.

Cheolhyoung Lee, Kyunghyun Cho, and Wanmo Kang. Mixout: Effective regularization to finetune large-scale pretrained language models. In *ICLR*, 2020.

Hezheng Lin, Xing Cheng, Xiangyu Wu, Fan Yang, Dong Shen, Zhongyuan Wang, Qing Song, and Wei Yuan. CAT: cross attention in vision transformer. *CoRR*, abs/2106.05786, 2021.

Fuchen Long, Ting Yao, Zhaofan Qiu, Xinmei Tian, Jiebo Luo, and Tao Mei. Learning to localize actions from moments. In *Computer Vision–ECCV 2020: 16th European Conference, Glasgow, UK, August 23–28, 2020, Proceedings, Part III 16*, pages 137–154. Springer, 2020.

Zhaoyang Niu, Guoqiang Zhong, and Hui Yu. A review on the attention mechanism of deep learning. *Neurocomputing*, 452:48–62, 2021. ISSN 0925-2312. doi: https://doi.org/10.1016/j.neucom.2021.03.091.

Chigozie Nwankpa, Winifred Ijomah, Anthony Gachagan, and Stephen Marshall. Activation functions: Comparison of trends in practice and research for deep learning. *arXiv preprint arXiv:1811.03378*, 2018.

Keiron O'shea and Ryan Nash. An introduction to convolutional neural networks. *arXiv preprint arXiv:1511.08458*, 2015.

Alec Radford, Jong Wook Kim, Chris Hallacy, Aditya Ramesh, Gabriel Goh, Sandhini Agarwal, Girish Sastry, Amanda Askell, Pamela Mishkin, Jack Clark, et al. Learning transferable visual models from natural language supervision. In *ICML*, pages 8748–8763. PMLR, 2021.

Marcus Rohrbach, Anna Rohrbach, Michaela Regneri, Sikandar Amin, Mykhaylo Andriluka, Manfred Pinkal, and Bernt Schiele. Recognizing fine-grained and composite activities using hand-centric features and script data. *IJCV*, pages 1–28, 2015. ISSN 0920-5691.

Paul Scovanner, Saad Ali, and Mubarak Shah. A 3-Dimentional SIFT Descriptor and its Application to Action Recognition. *CRCV*, pages 1–4, 2007.

Dian Shao, Yue Zhao, Bo Dai, and Dahua Lin. Finegym: A hierarchical video dataset for fine-grained action understanding. In *CVPR*, 2020.

Alex Sherstinsky. Fundamentals of recurrent neural network (rnn) and long short-term memory (lstm) network. *Physica D: Nonlinear Phenomena*, 404:132306, 2020.

Karen Simonyan and Andrew Zisserman. Two-stream convolutional networks for action recognition in videos. In *NIPS*, pages 568–576, 2014.

Nitish Srivastava, Geoffrey Hinton, Alex Krizhevsky, Ilya Sutskever, and Ruslan Salakhutdinov. Dropout: A simple way to prevent neural networks from overfitting. *JMLR*, 15 (56):1929–1958, 2014.

Waqas Sultani, Chen Chen, and Mubarak Shah. Real-world anomaly detection in surveillance videos. In *CVPR*, pages 6479–6488, 2018.

Yunlong Tang, Jing Bi, Siting Xu, Luchuan Song, Susan Liang, Teng Wang, Daoan Zhang, Jie An, Jingyang Lin, Rongyi Zhu, Ali Vosoughi, Chao Huang, Zeliang Zhang, Feng Zheng, Jianguo Zhang, Ping Luo, Jiebo Luo, and Chenliang Xu. Video understanding with large language models: A survey. *arXiv preprint arXiv:2312.17432*, 2023.

Du Tran, Lubomir Bourdev, Rob Fergus, Lorenzo Torresani, and Manohar Paluri. Learning Spatiotemporal Features with 3D Convolutional Networks. *ICCV*, pages 4489–4497, 2015.

Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N Gomez, Łukasz Kaiser, and Illia Polosukhin. Attention is all you need. *NeurIPS*, 30, 2017.

Heng Wang and Cordelia Schmid. Action Recognition with Improved Trajectories. *ICCV*, pages 3551–3558, 2013.

Lei Wang and Piotr Koniusz. 3mformer: Multi-order multi-mode transformer for skeletal action recognition. In *CVPR*, pages 5620–5631, 2023.

Lei Wang and Piotr Koniusz. Flow dynamics correction for action recognition. In *ICASSP*, pages 3795–3799. IEEE, 2024.

Lei Wang, Du Q. Huynh, and Piotr Koniusz. A comparative review of recent kinect-based action recognition algorithms. *IEEE TIP*, 29:15–28, 2020. ISSN 1941-0042.

Lei Wang, Xiuyuan Yuan, Tom Gedeon, and Liang Zheng. Taylor videos for action recognition. In *ICML*, 2024.

Limin Wang, Yuanjun Xiong, Zhe Wang, Yu Qiao, Dahua Lin, Xiaoou Tang, and Luc Van Gool. Temporal segment networks for action recognition in videos. *IEEE TPAMI*, 41(11):2740–2755, 2018.

Limin Wang, Zhan Tong, Bin Ji, and Gangshan Wu. Tdn: Temporal difference networks for efficient action recognition. In *CVPR*, pages 1895–1904, 2021.

Xi Wei, Tianzhu Zhang, Yan Li, Yongdong Zhang, and Feng Wu. Multi-modality cross attention network for image and sentence matching. In *CVPR*, June 2020.

Liwei Wu, Shuqing Li, Cho-Jui Hsieh, and James L Sharpnack. Stochastic shared embeddings: Data-driven regularization of embedding layers. *NeurIPS*, 32, 2019.

Antoine Yang, Antoine Miech, Josef Sivic, Ivan Laptev, and Cordelia Schmid. Zero-shot video question answering via frozen bidirectional language models. *NeurIPS*, 35:124–141, 2022.

Lingfeng Yang, Yueze Wang, Xiang Li, Xinlong Wang, and Jian Yang. Fine-grained visual prompting. *NeurIPS*, 36, 2024.

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

Fuzhen Zhuang, Zhiyuan Qi, Keyu Duan, Dongbo Xi, Yongchun Zhu, Hengshu Zhu, Hui Xiong, and Qing He. A comprehensive survey on transfer learning. *Proceedings of the IEEE*, 109(1):43–76, 2020.