International Journal of Computer Vision
https://doi.org/

Feature Hallucination for Self-supervised Action Recognition

Lei Wang · **Piotr Koniusz**

Received: 16.06.2024 / Revised: 24.11.2024 / Revised: 05.03.2025 / Accepted: 25.06.2025

Abstract Understanding human actions in videos requires more than raw pixel analysis; it relies on high-level semantic reasoning and effective integration of multimodal features. We propose a deep translational action recognition framework that enhances recognition accuracy by jointly predicting action concepts and auxiliary features from RGB video frames. At test time, hallucination streams infer missing cues, enriching feature representations without increasing computational overhead. To focus on action-relevant regions beyond raw pixels, we introduce two novel domain-specific descriptors. *Object Detection Features* (ODF) aggregate outputs from multiple object detectors to capture contextual cues, while *Saliency Detection Features* (SDF) highlight spatial and intensity patterns crucial for action recognition. Our framework seamlessly integrates these descriptors with auxiliary modalities such as optical flow, Improved Dense Trajectories, skeleton data, and audio cues. It remains compatible with state-of-the-art architectures, including I3D, AssembleNet, Video Transformer Network, FASTER, and recent models like VideoMAE V2 and InternVideo2. To handle uncertainty in auxiliary features, we incorporate aleatoric uncertainty modeling in the hallucination step and introduce a robust loss function to mitigate feature noise. Our multimodal self-supervised action recognition framework achieves state-of-the-art performance on multiple benchmarks, including Kinetics-400, Kinetics-600, and Something-Something V2, demonstrating its effectiveness in capturing fine-grained action dynamics.

· L. Wang is a Research Fellow (Grade 2) in the School of Engineering and Built Environment — Electrical and Electronic Engineering at Griffith University, and a Visiting Scientist at Data61/CSIRO. E-mail: l.wang4@griffith.edu.au.
· P. Koniusz is a Principal Research Scientist at Data61/CSIRO, and an Honorary Associate Professor (Level D) at ANU, and an Adjunct Associate Professor (Level D) at the University of New South Wales (UNSW). E-mail: piotr.koniusz@data61.csiro.au.

Fig. 1: We use object detectors and saliency maps to identify action-relevant regions within video frames. Fig. 1a shows bounding boxes from four detectors. The faster R-CNN detector with ResNet101 is focused on human-centric actions, such as *stand*, *watch*, *talk*, *etc*.. The other three detectors identify objects, such as *oven*, *sink*, *clock*, *etc*.. Fig. 1b shows how the MNL saliency detector [185] emphasizes spatial regions. Fig. 1c shows how the ACLNet saliency detector [184] highlights motion regions.

1 Introduction

Action recognition (AR) has evolved significantly, transitioning from handcrafted feature-based methods [30, 127, 80, 148, 149, 150, 134, 29, 112, 113, 152, 155, 156] to deep learning-driven approaches [133, 144, 44, 15, 153, 18, 189, 167]. Early techniques such as Histogram of Gradients (HOG) [48, 80], Histogram of Optical Flow (HOF) [30], and Improved Dense Trajectories (IDT) [149, 150] effectively captured spatial and temporal patterns but suffered from high computational costs and poor scalability. The emergence of deep learning models, particularly two-stream networks [133] and Inflated 3D CNNs (I3D) [15], introduced significant improvements by learning end-to-end representations from raw RGB and optical flow. However, despite these advancements, action recognition remains an open challenge due to the absence of well-established models that can effectively address three fundamental issues: (i) incomplete and imbalanced multimodal data, (ii) inefficient

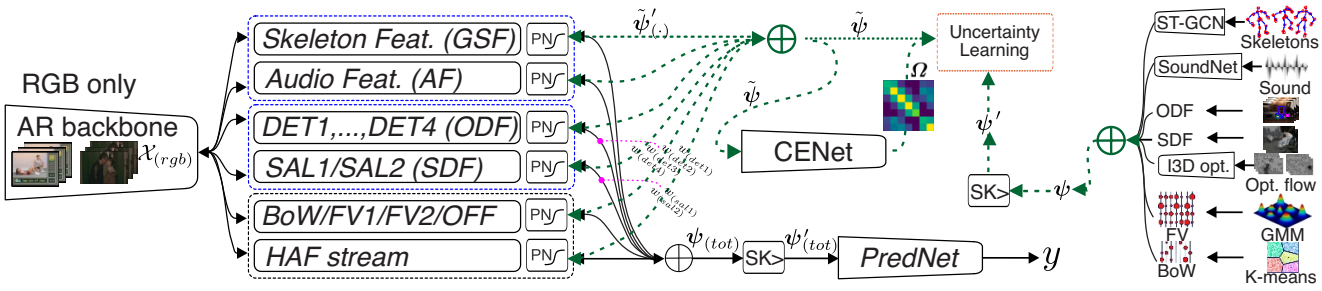


Fig. 2: We use six mainstream action recognition (AR) backbones: I3D, AssembleNet/AssembleNet++, Video Transformer Network (VTN), the lightweight FASTER framework, and recent models like VideoMAE V2 and InternVideo2. The prediction layer (*e.g.*, the final 1D covolutional layer of I3D) is removed from each backbone, and the intermediate representation (*e.g.*, the pooled spatiotemporal token embeddings of VideoMAE V2 / InternVideo2 encoder), $\mathscr{X}_{(rgb)}$, is passed into the following streams: *Bag-of-Words* (*BoW*), *Fisher Vector* (*FV*), *Optical Flow Features* (*OFF*), and *High Abstraction Features* (*HAF*), followed by the *Power Normalization* (*PN*) block (dashed black). The OFF stream is supervised by features extracted from the pre-trained I3D optical flow network (on Kinetics-400). Additionally, we introduce new feature streams: *Object Detection Features* (*ODF*) (from detector-based descriptors *DET1*,...,*DET4*), *Saliency Detection Features* (*SDF*) (from saliency-based descriptors *SAL1* and *SAL2*), *spatio-temporal GCN-encoded Skeleton Features* (*GSF*), and *Audio Features* (*AF*) (dashed blue). The *GSF* stream is supervised by skeleton features from the pre-trained ST-GCN (on Kinetics-Skeletons), and the *AF* stream is supervised by audio features from SoundNet (pre-trained on 2-million unlabeled videos). The resulting feature vectors, $\tilde{\boldsymbol{\psi}}_{(\cdot)}$, where $(\cdot)$ denotes the stream name, are aggregated ($\oplus$), followed by *Sketching* (*SK*) block, and passed into the *Prediction Network* (*PredNet*). The *ODF* and *SDF* features are reweighted by corresponding weights $w_{(\cdot)}$ (magenta lines)). Green dashed arrows show the feature hallucination process. During training, we use either MSE loss or our uncertainty learning loss (dashed red) for hallucination streams. The *Covariance Estimation Network* (*CENet*) takes the concatenated hallucinated features and produces a precision matrix $\boldsymbol{\Omega}$ (the inverse of the covariance matrix). This matrix, along with the hallucinated features $\tilde{\boldsymbol{\psi}}$ and ground truth features $\boldsymbol{\psi}'$, is fed into the uncertainty learning module. During testing, the hallucinated features $\tilde{\boldsymbol{\psi}}$ are input into PredNet to obtain the predicted labels y.

feature fusion across modalities, and (iii) the lack of structured motion descriptors in deep learning models.

A key limitation of existing robust AR models is their dependence on multimodal data, such as RGB, optical flow, and skeletons. While multimodal learning can enhance action recognition performance by using complementary cues [35], most benchmark datasets only provide RGB videos, leading to missing or imbalanced modalities. Many datasets do not provide all possible modalities. For example, some datasets only contain RGB videos, while others may include skeleton or depth data but lack optical flow. This inconsistency forces models to either rely exclusively on RGB-based representations, leading to suboptimal motion reasoning, or incorporate handcrafted features like IDT, which, despite their effectiveness, are computationally prohibitive and incompatible with modern deep learning frameworks [47, 23, 24, 151, 27]. These constraints prevent existing models from fully exploiting multimodal cues in a scalable and efficient manner [47, 23, 24, 151, 27, 165, 153].

Beyond the challenge of missing modalities, current feature fusion strategies suffer from fundamental inefficiencies [35]. Late fusion techniques process each modality separately before combining predictions, failing to model fine-grained cross-modal interactions. Early fusion, on the other hand, directly combines raw features, often introducing modality misalignment and redundant information. Existing hallucination-based approaches [162, 141, 157] attempt to synthesize missing modalities (*e.g.*, estimating optical flow), but they remain constrained to a fixed set of features and fail to generalize across datasets with diverse action categories (*e.g.*, DEEP-HAL [162]). Even when multiple modalities are present, they may not be equally available or useful. Some modalities might be noisy, sparse, or unreliable. Consequently, there is no well-established approach that efficiently integrates multimodal information while handling missing data in a robust and scalable manner.

Moreover, modern deep learning models [170] primarily focus on RGB and text-based semantics while neglecting motion-specific domain knowledge [162, 157, 18, 37, 36]. Unlike handcrafted methods that explicitly model motion dynamics, deep networks, especially 3D CNNs [15, 125, 124], Vision Transformers [91, 116, 181, 135] and Masked Autoencoder (MAE) [143, 154], learn representations implicitly, often discarding structured motion cues that are critical for distinguishing similar actions, such as trajectories or motion boundaries [150]. While handcrafted descriptors like IDT effectively capture motion, they rely on fixed heuristics and are computationally expensive. Deep learning models, in contrast, lack explicit mechanisms to track movement over time, making them susceptible to failures in fast or subtle motion scenarios [18]. Additionally, they require massive amounts of data to learn motion cues [37, 36], which is impractical for many real-world applications. This fundamental gap in motion reasoning limits model performance, particularly in challenging scenarios where appearance-based features alone are insufficient. The trade-off between efficiency and accuracy remains unresolved, as highly expressive models demand extensive feature engineering and computationally expensive training, hindering large-scale deployment[107, 5, 11, 101, 41, 178, 125, 124, 111].

To address these limitations, we propose a self-supervised multimodal framework that enhances feature integration, reduces reliance on handcrafted descriptors, and enables robust action recognition even in incomplete multimodal settings. Our approach introduces two novel domain-specific descriptors. First, *Object Detection Features* (ODF) capture action-relevant entities and contextual information using object detection outputs, such as those from Faster R-CNN [120], improving spatial awareness. Figure 1a shows examples of bounding boxes detected by four object detectors. Second, *Saliency Detection Features* (SDF) extract salient motion regions, helping the model focus on task-relevant patterns and improving action recognition accuracy. Figures 1b and 1c show saliency maps from region-wise and temporal saliency detectors. These descriptors serve as semantic priors, guiding our model to attend to informative regions in video frames. Unlike existing methods that rely solely on RGB and optical flow, our approach dynamically integrates motion and structural cues into deep learning pipelines. More importantly, ODF and SDF introduce structured, learnable motion-aware features that enhance deep learning models with explicit spatial and motion cues while maintaining computational efficiency.

Beyond feature enhancement, our framework incorporates a self-supervised hallucination mechanism, allowing the model to synthesize missing modalities (*e.g.*, skeleton data [179] and audio cues [6], *etc.*) at test time. This enables robust performance even when certain modalities are absent, making the model scalable and practical. Additionally, we introduce aleatoric uncertainty modeling, which mitigates feature noise and ensures stable predictions when auxiliary data is unreliable. By addressing the three fundamental challenges of multimodal action recognition, our framework offers a generalizable, efficient, and scalable solution. Figure 2 provides a conceptual overview of our approach. Our method achieves state-of-the-art results on benchmark datasets such as Kinetics-400, Kinetics-600, and Something-Something V2, demonstrating its ability to bridge the gap between handcrafted and deep learning-based approaches. Our contributions can be summarized as follows:

i. We introduce **a novel multimodal action recognition framework** that integrates diverse auxiliary features while reducing the reliance on computationally expensive handcrafted descriptors during inference.
ii. We propose Object Detection Features (ODF) and Saliency Detection Features (SDF) as **domain-specific descriptors** that guide the model toward action-relevant regions, improving motion reasoning and action recognition accuracy.
iii. We develop **a self-supervised hallucination mechanism** to synthesize missing cues at test time, addressing the challenge of incomplete multimodal data.
iv. We incorporate **aleatoric uncertainty modeling** and a robust loss function to mitigate feature noise, enhancing performance on fine-grained action recognition tasks.

2 Related Work

We review early video descriptors, deep learning pipelines, object and human detectors, saliency and audio features, as well as uncertainty in vision and Power Normalization (PN) for mitigating feature burstiness in Action Recognition (AR).

2.1 Early Video Descriptors and Encoding Schemes

Early video descriptors. Early approaches relied on spatio-temporal interest point detectors [94, 38, 16, 173, 95, 148] and spatio-temporal descriptors [30, 127, 146, 148, 149, 150] which capture various appearance and motion statistics. However, spatio-temporal interest point detectors struggle to capture long-term motion patterns. To address this, the Dense Trajectory (DT) [148] approach is developed, which densely samples feature points in each frame and tracks them across the video frames using optical flow. Multiple descriptors are then extracted along these trajectories to capture shape, appearance and motion cues. Despite its utility, DT cannot account for camera motion. The Improved Dense Trajectory (IDT) approach [150, 149] overcomes this limitation by estimating and removing global background motion caused by the camera. Additionally, IDT filters out inconsistent matches using a human detector. For spatio-temporal descriptors, IDT uses HOG [48], HOF [30] and MBH [149]. HOG [48] contains statistics of the amplitude of image gradients w.r.t. the gradient orientation, thus it captures the static appearance cues. In contrast, HOF [30] captures histograms of optical flow while MBH [149] captures derivatives of the optical flow, thus it is highly resilient to the global camera motion whose cues cancel out due to derivatives. Thus, HOF and MBH contain the zero- and first-order optical flow statistics. Other notable spatio-temporal descriptors include HOG-3D [80], SIFT3D [127], SURF3D [173] and LTP [182].

BoW/FV encoding. The Bag-of-Words (BoW) method [134, 29] creates a visual vocabulary using k-means clustering, where local descriptors are assigned to specific clusters. Variants include Soft Assignment (SA) [50, 83] and Localized Soft Assignment (LcSA) [99, 87]. Following DEEP-HAL [162], we use BoW encoding [29] with Power Normalization [87]. Additionally, we use Fisher Vectors (FV) [112, 113], which capture first- and second-order statistics of local descriptors assigned to Gaussian Mixture Model (GMM) clusters.

2.2 Deep Learning in Action Recognition

CNN-based. Early AR models using CNNs relied on frame-wise features with average pooling [73], which discarded the temporal order. To address this, frame-wise CNN scores are input to LSTMs [39], while two-stream networks [133] compute separate representations for RGB frames and 10 stacked optical flow frames. Spatio-temporal patterns are later modeled using 3D CNN filters [72, 144, 44, 147].

While two-stream networks [133] overlook temporal order, approaches like rank pooling [46, 47, 24, 151] and higher-order pooling [23, 82, 86, 42, 84, 165] gained popularity. The I3D model [15] introduces spatio-temporal 'inflation', where 2D CNN filters pre-trained on ImageNet-1K [34] are adapted to 3D, incorporating temporal pooling. PAN [183] proposes the Persistence of Appearance motion cue, which distills motion information directly from adjacent RGB frames. A bootstrapping approach [100] uses long-range temporal context attention, while another [93] introduces a graph attention model to explore semantic relationships. Slow-I-Fast-P (SIFP) [96] uses dual pathways for compressed AR, with sparse sampling on I-frames and dense sampling using pseudo optical flow clips.

Optical flow. Optical flow remains a cornerstone in AR [133, 15, 46, 151, 161]. Early methods tackle small displacements [62, 110], while new methods address larger displacements, such as Large Displacement Optical Flow (LDOF) [10]. Recent methods involve non-rigid descriptor or segment matching [172, 9] and edge-preserving interpolation [121]. We use LDOF [110] for optical flow estimation.

GCNs for skeletons. Graph Convolutional Networks (GCNs) have shown great success in skeletal AR [131, 13, 129, 20, 19, 128, 163, 158, 159, 164, 153]. These methods construct skeleton graphs, where joints are vertices, and bones are edges, enabling GCNs to model dependencies [78]. The spatio-temporal GCN (ST-GCN) [179] simultaneously learns spatial and temporal features. Subsequent advancements include Actional-Structural GCN (AS-GCN) [97], Context-Aware GCN (CA-GCN) [186], Shift-GCN [21], dynamic directed GCN [90], part-level GCN [65], and other specialized GCNs [109, 130, 2, 118]. In this work, we use pre-trained ST-GCN on Kinetics-skeleton to extract skeleton features for feature hallucination.

NAS. AssembleNet [125] uses Neural Architecture Search (NAS) to identify an optimal architecture for spatio-temporal feature interactions in AR. AssembleNet++ [124] extends this by dynamically learning attention weights to explore interactions between appearance, motion, and spatial object information. We use AssembleNet++ in this work.

Transformer-based. The AR field is increasingly adopting transformer-based models [41]. Pure transformer architectures have achieved state-of-the-art accuracy on major video recognition benchmarks [5, 107, 11, 101, 81, 160] by globally connecting spatial and temporal patches. Lightweight transformers, such as the Video Transformer Network (VTN) [91], enable real-time AR on low-power devices, including edge computing scenarios. Recent advancements include TubeViT [116], Side4Video [181], and OmniVec2 [135], which enhance video understanding via multimodal and multitask learning.

Lightweight models. While recent methods [107, 5, 11, 101, 41, 178], such as AssembleNet [125, 124] and Motionformer [111], have achieved state-of-the-art performance in AR, their high computational makes them unsuitable for real-time or resource-constrained applications. To address this, lightweight models like SqueezeNet [68], Xception [26], ShuffleNet [103], EfficientNet [140], MobileNet [64], as well as frameworks like FASTER [188] and Video Transformer Networks (VTN) [91], have been proposed to mitigate these challenges. Recent efforts have focused on optimizing architectures at the clip level [64] to further reduce computational overhead. Given the strong temporal structure and high redundancy in video data, the FASTER framework [188] tackles these challenges by emphasizing the temporal aggregation stage. By using a lightweight model, it effectively captures scene changes over time while minimizing redundant computations.

Masked vision modeling. Self-supervised video pre-training methods like VideoMAE [143] and its successor VideoMAE V2 [154] have significantly improved tasks like action classification and spatial-temporal detection. VideoMAE uses high-ratio video tube masking to enhance representation learning. VideoMAE V2 introduces dual masking to reduce decoder input length, improving both computational efficiency and learning performance. InternVideo2 [170] further advances AR through multi-stage training on web and YouTube datasets, achieving state-of-the-art performance across 60+ video and audio tasks.

Building on [162, 157], we investigate the use of various backbones, including AssembleNet++, lightweight VTN, FASTER, and recent advancements such as VideoMAE V2 and InternVideo2. These backbones are used to hallucinate computationally expensive handcrafted features, such as optical flow and skeleton features, effectively reducing the need for explicit feature extraction during testing stage.

2.3 Object and Saliency Detectors

Object detectors. Modern deep learning-based object detection methods include Region-based Convolutional Neural Networks (R-CNN) [54], its faster variants [53, 120], mask-based extensions [58], and the YOLO family [119], including YOLO v2 and YOLO v3, which prioritize efficiency by using a single network architecture.

In this work, we use the faster R-CNN detector [120] with several backbones: (i) Inception V2 [139], (ii) Inception ResNet V2 [138], (iii) ResNet101 [59] and (iv) NASNet [191]. Among these, Inception V2, Inception ResNet V2, and NASNet are pre-trained on the COCO dataset [98], enabling detection across 91 object classes. These models are particularly adept at summarizing environments, such as indoor settings, and associating scene context with actions. ResNet101, on the other hand, is pre-trained on the AVA v2.1 dataset [56], which includes 80 human actions, making it highly suited for human-centric AR tasks. In addition to detection scores, each bounding box is further described using ImageNet-1K [123] scores from a pre-trained Inception ResNet V2 model [138].

Saliency detectors. Saliency detectors identify image regions that correlate with human visual attention, typically represented as saliency maps. Deep learning-based saliency models [166, 63] outperform traditional methods [190] but often require pixel-wise annotations. Recent advancements include MNL [185] (a weakly-supervised model), RFCN [166] (a fully-supervised model), and a Robust Background Detector (RBD) [190] (refer to [8] for a detailed survey).

For spatial saliency, we use MNL [185], which is trained on noisy labels derived from weak or unsupervised handcrafted saliency models. For temporal saliency, we rely on ACLNet [184], a CNN-LSTM-based architecture designed to capture dynamic saliency patterns over time.

2.4 Audio Modality in Action Recognition

The visual and audio modalities are highly correlated yet contain distinct and complementary information. Numerous studies [1, 175, 136, 169, 49] have explored the integration of audio and visual cues for AR, as audio can provide strong complementary evidence for certain actions. This strong correlation enables accurate semantic predictions of one modality from the other. At the same time, their intrinsic differences make cross-modal prediction a valuable pretext task for self-supervised learning, offering advantages over within-modality learning. Building on this idea, Cross-Modal Deep Clustering (XDC) [4] uses both the semantic correlation and distinct characteristics of visual and audio modalities to enhance AR. Similarly, a fused multisensory representation [108] has been introduced to jointly model visual and audio components, leading to a richer and more robust understanding of video content.

SoundNet [6] transfers discriminative knowledge from visual recognition models to the sound modality. Using two million unlabeled videos, it bridges the gap between vision and audio, making it ideal for extracting audio features as ground truth in our hallucination process. More recently, OmniVec2 [135], a multimodal multitask transformer-based model, has been introduced. It employs modality-specialized tokenizers, a shared transformer architecture, and cross-attention mechanisms to project different modalities, including audio, into a unified embedding space. Additionally, InternVideo2 [170], a new family of video foundation models, incorporates video-audio correspondence in its second-stage training, encouraging deeper semantic learning across modalities.

2.5 Uncertainty in Computer Vision

Uncertainty in computer vision is generally categorized into *aleatoric* and *epistemic* uncertainty [105, 79, 69, 66, 75]. Aleatoric uncertainty is typically modeled by a Gaussian distribution over the predictions, while epistemic uncertainty is represented by a distribution over the model weights, as seen in Bayesian Neural Networks. In simple terms, aleatoric (or statistical) uncertainty refers to randomness or inherent variability in the data, whereas epistemic (or systematic) uncertainty refers to uncertainty stemming from a lack of knowledge (*e.g.*, uncertainty about the best model, essentially representing ignorance).

In this work, we primarily focus on heteroscedastic aleatoric uncertainty, which has become popular in many applications. Examples include uncertainty-weighted multitask loss in depth regression and segmentation [76], bounding box regression with uncertainty in Faster R-CNN [61] and YOLOv3 [25], deep learning-assisted methods for measuring uncertainty in action recognition [3], uncertainty-aware audio-visual action recognition [137], uncertainty quantification for deep context-aware mobile AR [67], structured uncertainty prediction networks for face images [40], and recent few-shot keypoint detection with uncertainty learning [102]. However, many of these approaches treat multiple variables independently, while we model uncertainty with covariance to capture the underlying relationships between variables. Specifically, we model the aleatoric uncertainty of features during the hallucination step to further enhance the performance of action recognition.

2.6 Power Normalization Family

BoW, FV and even CNN-based descriptors must address the phenomenon of burstiness, which is defined as '*the property that a given visual element appears more times in an image than a statistically independent model would predict*' [71]. This phenomenon is also present in video descriptors. Power Normalization [87, 85] is known to mitigate burstiness and has been extensively studied in the context of BoW [87, 85, 86, 89]. Additionally, a connection to max-pooling was identified in [87], which demonstrates that the so-called

MaxExp pooling is, in fact, a detector of '*at least one particular visual word being present in an image*'. According to the studies [87, 89], many Power Normalization functions are closely related. The Power Normalizations used in our work are outlined in Section 3.

3 Background

Below, we provide the necessary background information for our framework, beginning with an introduction to our notations.

Notations. We use boldface uppercase letters to represent matrices, *e.g.*, $\boldsymbol{M},\boldsymbol{P}$; regular uppercase letters with a subscript to represent matrix elements, *e.g.*, P_{ij}, which denotes the $(i,j)^{\text{th}}$ element of $\boldsymbol{P}$; boldface lowercase letters for vectors, *e.g.*, $\mathbf{x},\boldsymbol{\phi},\boldsymbol{\psi}$; and regular lowercase letters for scalars. Vectors may be numbered, *e.g.*, $\mathbf{x}_n$, while regular lowercase letters with a subscript represent an element of a vector, *e.g.*, $\mathbf{x}_i$ is the i^{th} element of $\mathbf{x}$. The operators ';' and ',' are used to concatenate vectors along the first and second modes, respectively. For example, $\circledcirc_{i\in\mathscr{I}_K}\mathbf{v}_i=[\mathbf{v}_1;...;\mathbf{v}_K]$ and $\circledcirc^2_{i\in\mathscr{I}_K}\mathbf{v}_i=[\mathbf{v}_1,...,\mathbf{v}_K]$ concatenate a group of vectors along the first and second modes, respectively. $\uparrow\otimes_r$ denotes the r-th Kronecker power. The operator $\oplus$ denotes aggregation (sum), while $\mathscr{I}_d$ represents an index set of integers $\{1,...,d\}$.

3.1 Descriptor Encoding Schemes

Bag-of-Words [134, 29] assigns each local descriptor $\boldsymbol{x}$ to the closest visual word from $\boldsymbol{M}=[\boldsymbol{m}_1,...,\boldsymbol{m}_K]$, which is built using k-means. To obtain the mid-level feature $\boldsymbol{\phi}$, we solve the following optimization problem:

$$\begin{aligned}&\boldsymbol{\phi}=\arg\min_{\boldsymbol{\phi}'}\|\boldsymbol{x}-\boldsymbol{M}\boldsymbol{\phi}'\|_2^2,\\&s.t.\ \ \boldsymbol{\phi}'\in\{0,1\},\mathbf{1}^T\boldsymbol{\phi}'=1.\end{aligned}\tag{1}$$

Fisher Vector Encoding [112, 113] uses a mixture of K Gaussians from a GMM as a dictionary. It encodes descriptors with respect to the Gaussian components $G(w_k,\boldsymbol{m}_k,\boldsymbol{\sigma}_k)$, which are parametrized by the mixing probability, mean, and diagonal standard deviation. The first- and second-order features $\boldsymbol{\phi}_k,\boldsymbol{\phi}'_k\in\mathbb{R}^D$ are given by:

$$\boldsymbol{\phi}_k=(\boldsymbol{x}-\boldsymbol{m}_k)/\boldsymbol{\sigma}_k,\ \ \boldsymbol{\phi}'_k=\boldsymbol{\phi}_k^2-1.\tag{2}$$

The concatenation of per-cluster features $\boldsymbol{\phi}^*_k\in\mathbb{R}^{2D}$ forms the mid-level feature $\boldsymbol{\phi}\in\mathbb{R}^{2KD}$:

$$\boldsymbol{\phi}=[\boldsymbol{\phi}^*_1;...;\boldsymbol{\phi}^*_K],\ \ \boldsymbol{\phi}^*_k=\frac{p(\boldsymbol{m}_k|\boldsymbol{x},\theta)}{\sqrt{w_k}}\left[\boldsymbol{\phi}_k;\boldsymbol{\phi}'_k/\sqrt{2}\right],\tag{3}$$

where p and θ represent the component membership probabilities and the parameters of the GMM, respectively. For each descriptor $\boldsymbol{x}$ with dimensionality D (after PCA), its encoding $\boldsymbol{\phi}$ has a dimensionality of $2KD$, as it captures both first- and second-order statistics.

3.2 Pooling a.k.a. Aggregation

Traditionally, pooling is performed by averaging mid-level feature vectors $\boldsymbol{\phi}(\mathbf{x})$ corresponding to local descriptors $\mathbf{x}\in\mathscr{X}$ from a video sequence $\mathscr{X}$, expressed as $\boldsymbol{\psi}=\text{avg}_{\mathbf{x}\in\mathscr{X}}\boldsymbol{\phi}(\mathbf{x})$, with optional ℓ_2-normalization. In this paper, we apply this approach to both full sequences $\mathscr{X}$ and subsequences.

Proposition 1 *For subsequence pooling, let $\mathscr{X}_{s,t}=\mathscr{X}_{0,t}\setminus\mathscr{X}_{0,s-1}$, where $\mathscr{X}_{s,t}$ denotes the set of descriptors in the sequence $\mathscr{X}$ from frame s to frame t, with $0\le s\le t\le\tau$, $\mathscr{X}_{0,-1}\equiv\emptyset$, and τ is the length of $\mathscr{X}$. Let us compute an integral mid-level feature $\boldsymbol{\phi}'_t=\boldsymbol{\phi}'_{t-1}+\sum_{\mathbf{x}\in\mathscr{X}_{t,t}}\boldsymbol{\phi}(\mathbf{x})$, which aggregates mid-level feature vectors from frame 0 to frame t, with $\boldsymbol{\phi}'_{-1}$ initialized as an all-zeros vector. The pooled subsequence is then given by:*

$$\boldsymbol{\psi}_{s,t}=\frac{\boldsymbol{\phi}'_t-\boldsymbol{\phi}'_{s-1}}{\|\boldsymbol{\phi}'_t-\boldsymbol{\phi}'_{s-1}\|_2+\varepsilon}=\frac{\sum_{\mathbf{x}\in\mathscr{X}_{s,t}}\boldsymbol{\phi}(\mathbf{x})}{\|\sum_{\mathbf{x}\in\mathscr{X}_{s,t}}\boldsymbol{\phi}(\mathbf{x})\|_2+\varepsilon},\tag{4}$$

where $0\le s\le t\le\tau$ are the starting and ending frames of subsequence $\mathscr{X}'_{s,t}\subseteq\mathscr{X}$, and ε is a small constant. We normalize the pooled sequences/subsequences as described next.

3.3 Power Normalization

As discussed in Section 2, we apply power normalization functions to each stream, such as ODF and SDF. We investigate three operators $g(\boldsymbol{\psi},\cdot)$, detailed in Remarks 1–3.

Remark 1 The AsinhE function [89] is an extension of the well-known power normalization (Gamma) [89], defined as $g(\boldsymbol{\psi},\gamma)=\text{Sgn}(\boldsymbol{\psi})|\boldsymbol{\psi}|^\gamma$ for $0<\gamma\le1$, with a smooth derivative and a parameter γ'. The AsinhE function is defined as the normalized Arcsin hyperbolic function:

$$g(\boldsymbol{\psi},\gamma')=\text{arcsinh}(\gamma'\boldsymbol{\psi})/\text{arcsinh}(\gamma').\tag{5}$$

Remark 2 Sigmoid (SigmE), a max-pooling approximation [89], is an extension of the MaxExp operator defined as $g(\boldsymbol{\psi},\eta)=1-(1-\boldsymbol{\psi})^\eta$ for $\eta>1$. This operator is extended to have a smooth derivative and a response defined for real-valued $\boldsymbol{\psi}$ (rather than $\boldsymbol{\psi}\ge0$), with a parameter η' and a small constant ε':

$$g(\boldsymbol{\psi},\eta')=\frac{2}{1+\text{e}^{-\eta'\boldsymbol{\psi}/(\|\boldsymbol{\psi}\|_2+\varepsilon')}}-1.\tag{6}$$

Remark 3 AxMin, a piecewise linear form of SigmE [89], is given as $g(\boldsymbol{\psi},\eta'')=\text{Sgn}(\boldsymbol{\psi})\min(\eta''\boldsymbol{\psi}/(\|\boldsymbol{\psi}\|_2+\varepsilon'),1)$ for $\eta''>1$ and a small constant ε'.

Although these three pooling operators serve similar roles, we investigate each one because their interplay with end-to-end learning differs. Specifically, $\lim_{\psi\to\pm\infty} g(\psi,\cdot)$ for AsinhE and SigmE are $\pm\infty$ and ± 1, respectively, thus their asymptotic behaviors differ. Moreover, AxMin is non-smooth and relies on the same gradient re-projection properties as ReLU.

3.4 Count Sketches

Sketching vectors via the count sketch [28, 171] is a technique for dimensionality reduction, which we apply in this paper.

Proposition 2 *Let d and d' denote the dimensionality of the input and sketched output vectors, respectively. Let the vector $\boldsymbol{h}\in\mathscr{I}_{d'}^{d}$ contain d integers uniformly drawn from $\{1,...,d'\}$, and let the vector $\boldsymbol{s}\in\{-1,1\}^d$ contain d values uniformly drawn from $\{-1,1\}$. The sketch projection matrix $\boldsymbol{P}\in\{-1,0,1\}^{d'\times d}$ is given by:*

$$P_{ij}=\begin{cases} s_i & \text{if } h_i=j,\\ 0 & \text{otherwise,}\end{cases} \tag{7}$$

where s_i is the corresponding value from $\boldsymbol{s}$. The sketch projection $p:\mathbb{R}^d\to\mathbb{R}^{d'}$ is a linear operation, defined as $p(\psi)=\boldsymbol{P}\psi$ (or $p(\psi;\boldsymbol{P})=\boldsymbol{P}\psi$ to explicitly highlight $\boldsymbol{P}$).
Proof This follows directly from the definition of the count sketch, as explained in Definition 1 of [171].

Remark 4 Count sketches are unbiased estimators: $\mathbb{E}_{\boldsymbol{h},\boldsymbol{s}}(p(\psi,\boldsymbol{P}(\boldsymbol{h},\boldsymbol{s})),p(\psi',\boldsymbol{P}(\boldsymbol{h},\boldsymbol{s})))=\langle\psi,\psi'\rangle$. The variance is given by: $\mathbb{V}_{\boldsymbol{h},\boldsymbol{s}}(p(\psi),p(\psi'))\leq\frac{1}{d'}\left(\langle\psi,\psi'\rangle^2+\|\psi\|_2^2\|\psi'\|_2^2\right)$, so larger sketches reduce noisy. Thus, for each modality, we use a separate sketch matrix $\boldsymbol{P}$.
Proof For the first and second properties, see Appendix A of [171] and Lemma 3 of [114].

3.5 Sketching the Power Normalized Vectors

Proposition 3 *Sketching PN vectors increases the sketching variance (normalized by ℓ_2 vector norms) by a factor of $1\leq\kappa\leq 2$.*

Proof The variance $\mathbb{V}$ from Remark 4 is normalized by the norms $\|\psi\|_2^2\|\psi'\|_2^2$. Let $\mathbb{V}^{(\gamma)}$ denote the variance for d-dimensional vectors $\{(\psi^\gamma,\psi'^\gamma):\psi\geq 0,\psi'\geq 0\}$, where the vectors are power-normalized by Gamma as described in Remark 1. This variance is similarly normalized by $\|\psi^\gamma\|_2^2\|\psi'^\gamma\|_2^2$. For extreme PN ($\gamma\to 0$), the variance simplifies as follows:

$$\lim_{\gamma\to 0}\mathbb{V}^{(\gamma)}=\frac{1}{d'}\lim_{\gamma\to 0}\left(\frac{\langle\psi^\gamma,\psi'^\gamma\rangle^2}{\|\psi^\gamma\|_2^2\|\psi'^\gamma\|_2^2}+1\right)=\frac{2}{d'}. \tag{8}$$

Now, assume the d-dimensional vectors ψ and ψ' are ℓ_2-norm normalized. The ratio of variances can then be expressed as:

$$\kappa=\mathbb{V}/\mathbb{V}^{(\gamma)}=2/(\langle\psi,\psi'\rangle^2+1), \tag{9}$$

The factor κ depends on the pair (ψ,ψ') and varies smoothly within the range $[1,2]$ as γ changes between 0 and 1. The Gamma is a monotonically increasing function, and for typical values such as $\gamma=0.5$, empirical data suggests $\kappa\approx 1.3$.

3.6 Positional Embedding

Let $G_\sigma(\mathbf{x}-\mathbf{x}')=\exp(-\|\mathbf{x}-\mathbf{x}'\|_2^2/2\sigma^2)$ denote the standard Gaussian RBF kernel centered at $\mathbf{x}'$ with bandwidth σ. Kernel linearization refers to rewriting G_σ as the inner-product of two infinite-dimensional feature maps. To obtain these maps, we use a fast approximation method based on probability product kernels [70]. Specifically, we use the inner product of d''-dimensional isotropic Gaussians for $\mathbf{x},\mathbf{x}'\in\mathbb{R}^{d''}$. Thus, we have:

$$G_\sigma(\mathbf{x}-\mathbf{x}')=\left(\frac{2}{\pi\sigma^2}\right)^{\frac{d''}{2}}\int_{\boldsymbol{\zeta}\in\mathbb{R}^{d''}} G_{\sigma/\sqrt{2}}(\mathbf{x}-\boldsymbol{\zeta})\,G_{\sigma/\sqrt{2}}(\mathbf{x}'-\boldsymbol{\zeta})\,\mathrm{d}\boldsymbol{\zeta}. \tag{10}$$

Eq. (10) is approximated by replacing the integral with a sum over Z pivots $\boldsymbol{\zeta}_1,...,\boldsymbol{\zeta}_Z$, yielding the feature map $\boldsymbol{\phi}$ as:

$$\boldsymbol{\phi}(\mathbf{x};\{\boldsymbol{\zeta}_i\}_{i\in\mathscr{I}_Z})=\left[G_{\sigma/\sqrt{2}}(\mathbf{x}-\boldsymbol{\zeta}_1),...,G_{\sigma/\sqrt{2}}(\mathbf{x}-\boldsymbol{\zeta}_Z)\right]^T, \tag{11}$$

$$\text{and } G_\sigma(\mathbf{x}-\mathbf{x}')\approx\left\langle\sqrt{c}\boldsymbol{\phi}(\mathbf{x}),\sqrt{c}\boldsymbol{\phi}(\mathbf{x}')\right\rangle, \tag{12}$$

where c is a constant. Eq. (12) is the linearization of the RBF kernel, and Eq. (11) defines the feature map. The pivots $\{\boldsymbol{\zeta}_i\}_{i\in\mathscr{I}_Z}$ are evenly spaced in the interval $[0;1]$, with Z total pivots. For simplicity, we drop the explicit notation of the pivots $\{\boldsymbol{\zeta}_i\}_{i\in\mathscr{I}_Z}$ and write $\boldsymbol{\phi}(\mathbf{x})$, *etc.*.

3.7 Action Recognition Backbones

Below, we present six mainstream models used as backbones for feature hallucination.

I3D [15] (Inflated 3D ConvNet) is a two-stream architecture that extends 2D ConvNet (pre-trained on ImageNet-1K [34]) by inflating their kernels to 3D, enabling spatiotemporal feature extraction. The model has become one of the most widely adopted frameworks for video processing tasks. Variants such as S3D [176] build on the I3D architecture

by introducing modifications to its modules to enhance efficiency and performance. In our work, we adapt I3D by removing its final convolutional layer and classifier to construct the backbone network. The pre-trained weights from Kinetics-400 are used for feature hallucination, allowing us to leverage its robust video representation capabilities.

AssembleNet [125] is a family of learnable models designed to optimize the 'connectivity' among features across input modalities, tailored for specific target tasks such as AR. Its extension, **AssembleNet++** [124], introduces peer-attention mechanisms that enable the model to learn the interactions and relative importance of features, particularly between semantic object information and raw appearance and motion features. By removing the classification layer, we extract a 2048-dimensional output from the 3D average pooling layer as the intermediate representation for feature hallucination. For our backbone, we use AssembleNet++ pre-trained on Kinetics-400, using its robust capability to capture spatiotemporal and multimodal information.

VTN [91] uses the latest advancements in Vision Transformer (ViT) for computer vision tasks, applying it to AR. VTN consists of two main components: (i) an encoder that processes each frame of the input sequence independently using a 2D CNN to generate frame embeddings (pre-trained models are used to maximize the benefits of transfer learning from image classification tasks), and (ii) a decoder that integrates intra-frame temporal information in a fully-attentional, feed-forward manner, ultimately producing classification scores for the video clip. For our backbone, we remove the final classification layer of VTN. We adopt the default model hyperparameters as specified in [91], such as 4 stacked decoder blocks, 8 attention heads, and frame embeddings of size 512. For the encoder, we use MobileNet V2 [126] and ResNet-34 [60], forming two different backbones: VTN-MobileNet and VTN-ResNet.

FASTER [188] is a general framwork designed to aggregate both expensive and lightweight representations from different clips. It combines an expensive model, which captures detailed action information (*e.g.*, subtle motion), and a lightweight model, which tracks scene changes over time to minimize redundant computation between neighboring clips. This approach ensures global coverage of the entire video at a low cost by using FAST-GRU to aggregrate representations from different clip models. For the clip-level backbone in FASTER, we follow the method outlined in [188], selecting R(2+1)D-50 [145] as the expensive model and R2D-26 [145] as the lightweight model. We form our backbone by removing the final classification layer. In our experiments, we choose a clip length of $L=8$ and use 8 clips.

VideoMAE [143] is a cutting-edge self-supervised video pre-training framework that introduces video tube masking with a high masking ratio. This design makes video reconstruction a more challenging and meaningful self-supervision task, encouraging the extraction of more effective video representations during pre-training. Its advanced version, **VideoMAE V2** [154], scales VideoMAE further by introducing a dual masking strategy for improved efficiency. This approach applies a masking map to both the encoder and decoder, significantly reducing the decoder's input length while maintaining effectiveness. For feature hallucination, we use the VideoMAE V2 pre-trained encoder on UnlabeledHybrid [154], with ViT-g as the backbone.

InternVideo2 [170] is a state-of-the-art video foundation model excelling in video recognition, video-text tasks, and video-centric dialogue. It employs a progressive training approach that integrates masked video modeling, cross-modal contrastive learning, and next-token prediction, scaling the video encoder to an impressive 6 billion parameters. The training unfolds in three stages: first, video data is fed into the model to reconstruct unmasked video tokens, maximizing the capture of spatiotemporal visual concepts. Second, the model aligns video with audio, speech, and text through cross-modal contrastive learning, enriching it with semantic information. Finally, the model predicts the next token using video-centric inputs, embedding even deeper semantics. For our feature hallucination backbone, we use InternVideo2-1B, trained in the first stage, denoted as InternVideo2$_{s1}$.

4 Approach

Our pipeline, illustrated in Fig. 2, comprises the following components: (i) BoW/FV/OFF streams (dashed black), (ii) Object Detection Features (ODF) and Saliency Detection Features (SDF) streams, (iii) the GCN-encoded Skeleton Features (GSF) stream, (iv) the SoundNet-encoded Audio Features (AF) stream, (v) the High Abstraction Features (HAF) stream, (vi) the Prediction Network (abbreviated as PredNet), and (vii) the Covariance Estimation Network (CENet), which facilitates uncertainty-aware feature descriptor learning.

Each stream begins by processing the intermediate representations generated by the backbone from the RGB frames. These representations are refined through a hallucination process to approximate features using Mean Squared Error (MSE) loss between the ground-truth features and the outputs of the hallucinated streams. The HAF stream enhances the backbone representations before integrating them with the hallucinated streams. PredNet then combines the outputs from all streams, BoW, FV, OFF, HAF, ODF, SDF, GSF and AF, to learn action concepts for classification.

The following sections detail our method: we start by describing the extraction of ODF and SDF descriptors. Next, we explain the hallucination streams and the computation of ground-truth features, followed by a discussion

on uncertainty-aware learning and the design of CENet. Finally, we introduce the objective function used for feature hallucination.

4.1 Statistical Motivation

Before introducing our ODF and SDF descriptors, we highlight the importance of higher-order statistics in capturing the nuanced characteristics of video data [82, 88, 17, 84]. Comparing videos requires more than simple feature matching; it necessitates robust representations of the underlying distribution of local features (*e.g.*, detection scores) or descriptors. The characteristic function, $\varphi_{\Upsilon}(\boldsymbol{\omega})$, provides a comprehensive description of this distribution by representing the probability density $f_{\Upsilon}(\boldsymbol{\upsilon})$ of the features $\boldsymbol{\upsilon} \sim \Upsilon$: $\varphi_{\Upsilon}(\boldsymbol{\omega}) = \mathbb{E}_{\boldsymbol{\upsilon}\sim\Upsilon}\left(\exp(i\boldsymbol{\omega}^T\boldsymbol{\upsilon})\right)$. Using a Taylor series expansion, the characteristic function can be expressed as:

$$\mathbb{E}_{\boldsymbol{\upsilon}\sim\Upsilon}\left(\sum_{r=0}^{\infty}\frac{i^j}{r!}\langle\boldsymbol{\upsilon},\boldsymbol{\omega}\rangle^r\right) \approx \frac{1}{N}\sum_{n=0}^{N}\sum_{r=0}^{\infty}\frac{i^r}{r!}\langle\uparrow\otimes_r\boldsymbol{\upsilon}_n,\uparrow\otimes_r\boldsymbol{\omega}\rangle = \tag{13}$$
$$\sum_{r=0}^{\infty}\frac{i^r}{r!}\left\langle\frac{1}{N}\sum_{n=0}^{N}\uparrow\otimes_r\boldsymbol{\upsilon}_n,\uparrow\otimes_r\boldsymbol{\omega}\right\rangle = \sum_{r=0}^{\infty}\left\langle\mathscr{X}^{(r)},\frac{i^r}{r!}\uparrow\otimes_r\boldsymbol{\omega}\right\rangle,$$

where i is the imaginary unit, $\uparrow\otimes_r$ is the r-th Kronecker power, and $\mathscr{X}^{(r)}$, defined as $\mathscr{X}^{(r)} = \frac{1}{N}\sum_{n=0}^{N}\uparrow\otimes_r\boldsymbol{\upsilon}_n$, is a tensor capturing the r-th order moment of the feature distribution.

In principle, with infinite data and infinite moments, one can fully capture $f_{\Upsilon}(\boldsymbol{\upsilon})$. In practice, first-, second- and third-order moments are typically sufficient, however, second- and third-order tensors grow quadratically and cubically with respect to the size of $\boldsymbol{\upsilon}$. Thus, in what follows, we represent second-order moments not by a covariance matrix but by the subspace corresponding to the top n' leading eigenvectors. We also make use of the corresponding eigenvalues of the signal. Finally, it suffices to notice that $\boldsymbol{\kappa}^{(r)} = \mathrm{diag}\left(\mathscr{X}^{(r)}\right)$ corresponds to the notion of order r cumulants used in calculations of skewness ($r=3$) and kurtosis ($r=4$) but it grows linearly with respect to the size of $\boldsymbol{\upsilon}$. Thus, in what follows, we use the ℓ_2 norm normalized mean, leading eigenvectors (and trace-normalized eigenvalues), skewness and kurtosis (rather than coskewness and cokurtosis) to obtain compact representation of ODF and SDF.

4.2 Object Detection Features

Each object bounding box is described by the feature vector:

$$\boldsymbol{\upsilon} = \left[\boldsymbol{\delta}(y_{(det)}); \mathbf{y}_{(inet)}; \boldsymbol{\phi}(\varsigma); \circledcirc_{i\in\mathscr{I}_4}\boldsymbol{\phi}(v_i); \boldsymbol{\phi}\left(\frac{t-1}{\tau-1}\right)\right] \in \mathbb{R}^d, \tag{14}$$

where $\boldsymbol{\delta} = [0,...,1,...,0]^T$ is a vector with all zeros except for a single 1 at the location y. Since there are 91 object classes for detectors trained on the COCO dataset and 80 classes for those trained on the AVA v2.1 dataset, we assume $y_{(det)} \in \mathscr{I}_{91+80}$, where the labels $0,...,90$ correspond to COCO classes, and classes $91,...,79+91$ correspond to AVA v2.1 classes. Additionally, $\mathbf{y}_{(inet)} \in \mathbb{R}^{1001}$ represents an ℓ_1-norm normalized ImageNet-1K classification score, $0 \leq \varsigma \leq 1$ is the detector confidence score, $v_0,...,v_4$ are the normalized Cartesian coordinates (top-left and bottom-right) of a bounding box in the range $[0;1]$, and $(t-1)/(\tau-1)$ is the normalized frame index with respect to the total sequence length τ. For feature maps $\boldsymbol{\phi}(\cdot)$ defined in Eq. (11), we use $Z=7$ pivots and set the RBF σ to 0.5.

For all detections per video from a given detector, we first compute the mean $\boldsymbol{\mu}([\boldsymbol{\upsilon}_1,...,\boldsymbol{\upsilon}_N]) \in \mathbb{R}^d$ (denoted as $\boldsymbol{\mu}$), where N is the total number of detections. We then form a matrix $\boldsymbol{\Upsilon} \in \mathbb{R}^{d\times N}$:

$$\boldsymbol{\Upsilon} = \frac{1}{J}\left[\frac{1}{K_1}\left[\circledcirc^2_{i\in\mathscr{I}_{K_1}}(\boldsymbol{\upsilon}_{i1}-\boldsymbol{\mu})\right],...,\frac{1}{K_J}\left[\circledcirc^2_{i\in\mathscr{I}_{K_J}}(\boldsymbol{\upsilon}_{iJ}-\boldsymbol{\mu})\right]\right], \tag{15}$$

where K_j denotes the number of detections per frame $j \in \mathscr{I}_J$. From this, we extract higher-order statistical moments as described below. Since N is large and its size varies across videos, hallucinating $\boldsymbol{\Upsilon}$ directly is infeasible (and lacks invariance properties).

First, we compute $\boldsymbol{U}\boldsymbol{\lambda}\boldsymbol{V} = \mathrm{svd}(\boldsymbol{\Upsilon})$, rather than $\boldsymbol{U}\boldsymbol{\lambda}^2\boldsymbol{U}^T = \mathrm{eig}\left(\boldsymbol{\Upsilon}\boldsymbol{\Upsilon}^T\right)$ since $N \ll d$, where $\boldsymbol{U} = [\mathbf{u}_1, \mathbf{u}_2, ...]$. We take $\mathscr{X}^{(r)}\left(\{\mathbf{v}-\boldsymbol{\mu}\}_{n=0}^N\right)$ (abbreviated as $\mathscr{X}^{(r)}$) and define $\boldsymbol{\kappa}^{(r)} = \mathrm{diag}\left(\mathscr{X}^{(r)}\right)$ as described in Section 4.1. We then form our multi-moment descriptor $\boldsymbol{\psi}_{(det)} \in \mathbb{R}^{d(4+n')}$, $n' \geq 1$:

$$\boldsymbol{\psi}_{(det)} = \left[\frac{\boldsymbol{\mu}}{||\boldsymbol{\mu}||_2}; \circledcirc^2_{i\in\mathscr{I}_{n'}}\mathbf{u}_i\left(\mathscr{X}^{(2)}\right); \frac{\boldsymbol{\kappa}^{(3)}}{\left(\boldsymbol{\kappa}^{(2)}\right)^{3/2}}; \frac{\boldsymbol{\kappa}^{(4)}}{\left(\boldsymbol{\kappa}^{(2)}\right)^2}; \frac{\mathrm{diag}(\boldsymbol{\lambda}^2)}{\sum_i\lambda_{ii}^2}\right]. \tag{16}$$

The composition of Eq. (16) is explained in Section 4.1. It is easy to verify that $\frac{\boldsymbol{\kappa}^{(3)}}{\left(\boldsymbol{\kappa}^{(2)}\right)^{3/2}}$ and $\frac{\boldsymbol{\kappa}^{(4)}}{\left(\boldsymbol{\kappa}^{(2)}\right)^2}$ are the empirical versions of skewness and kurtosis, given by $\frac{\mathbb{E}_{\boldsymbol{\upsilon}\sim\Upsilon}\left((\boldsymbol{\upsilon}-\boldsymbol{\mu})^3\right)}{\mathbb{E}^{3/2}_{\boldsymbol{\upsilon}\sim\Upsilon}\left((\boldsymbol{\upsilon}-\boldsymbol{\mu})^2\right)}$ and $\frac{\mathbb{E}_{\boldsymbol{\upsilon}\sim\Upsilon}\left((\boldsymbol{\upsilon}-\boldsymbol{\mu})^4\right)}{\mathbb{E}^2_{\boldsymbol{\upsilon}\sim\Upsilon}\left((\boldsymbol{\upsilon}-\boldsymbol{\mu})^2\right)}$, respectively.

4.3 Saliency Detection Features

We extract directional gradients from saliency frames using discretized gradient operators $[-1,0,1]$ and $[-1,0,1]^T$, obtaining gradient amplitude and orientation maps, $\boldsymbol{\Lambda}$ and $\boldsymbol{\theta}$, for each frame, encoded as follows:

$$\boldsymbol{\upsilon}'_{(sal)} = \sum_{i\in\mathscr{I}_W, j\in\mathscr{I}_H}\Lambda_{ij}\boldsymbol{\phi}(\theta_{ij}/(2\pi)) \otimes \boldsymbol{\phi}\left(\frac{i-1}{W-1}\right) \otimes \boldsymbol{\phi}\left(\frac{j-1}{H-1}\right), \tag{17}$$

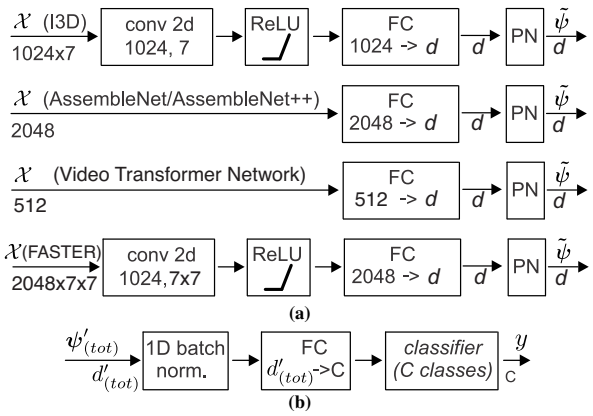


Fig. 3: Stream details. Figure 3a shows the architecture we use for the BoW, FV, OFF, HAF, ODF, SDF, GSF, and AF streams for four different backbones. Figure 3b shows the architecture of our PredNet. The operations and their parameters are specified within each block (*e.g.*, *conv2d* with its filter count/size, and *Power Normalization* (*PN*)). The input and output sizes are indicated under the corresponding arrows. For our experiments, we select $d = 128$, 256 or 512, depending on the dataset.

where $\otimes$ represents the Kronecker product, and $\boldsymbol{\phi}(\theta)$ follows Eq. (11), except that the assignment to Gaussians is performed in the modulo ring to respect the periodic nature of θ. We encode $\boldsymbol{\phi}(\theta)$ with 12 pivots to capture the orientation of gradients. The remaining maps $\boldsymbol{\phi}(\cdot)$ are encoded with 5 pivots each, corresponding to spatial binning. Note that $\boldsymbol{\upsilon}'_{(sal)}$ (denoted as $\boldsymbol{\upsilon}'$) is conceptually similar to a single CKN layer [104], but simpler: for one-dimensional variables, we sample pivots (similar to learning) for the maps $\boldsymbol{\phi}(\cdot)$. Each saliency frame is represented as a feature vector: $\boldsymbol{\upsilon}^\dagger = [\boldsymbol{\upsilon}'/||\boldsymbol{\upsilon}'||_2; \mathbf{I}_:/||\mathbf{I}_:||_1] \in \mathbb{R}^{d^\dagger}$, where $\mathbf{I}_:$ is a vectorized low-resolution saliency map. Thus, $\boldsymbol{\upsilon}^\dagger$ captures both the directional gradient statistics and the intensity-based gist of the saliency maps.

Next, we compute the mean $\boldsymbol{\mu}([\boldsymbol{\upsilon}_1^\dagger, ..., \boldsymbol{\upsilon}_J^\dagger]) \in \mathbb{R}^{d^\dagger}$ (denoted as $\boldsymbol{\mu}$), where J is the total number of frames per video. We then obtain $\boldsymbol{\Upsilon}^\dagger = \left[\boldsymbol{\upsilon}_1^\dagger, ..., \boldsymbol{\upsilon}_J^\dagger\right]/J \in \mathbb{R}^{d^\dagger \times J}$, which is compactly described by the multi-moment expression in Eq. (16), resulting in the multi-moment descriptor $\boldsymbol{\psi}_{(sal)} \in \mathbb{R}^{d(4+n^\dagger)}$.

4.4 Hallucinating Streams

Each stream processes the intermediate representation, $\mathscr{X}_{(rgb)}$, which is obtained either by removing the classifier and/or the final 1D convolutional layer of the backbone network, as detailed in Section 3.7, or by pooling spatiotemporal token embeddings, as used in VideoMAE V2 and InternVideo2. To implement the pipeline for each stream, we follow the approach in [162], using a Fully Connected (FC) unit. Furthermore, each stream is equipped with a PN module. For the PN implementation, we explore three variants: AsinhE, SigmE, and AxMin, described in Remarks 1, 2 and 3, respectively. Below, we provide a detailed explanation of each stream and its corresponding ground truth. It is important to note that ground-truth features are used exclusively during the training phase to train the hallucination streams.

BoW/FV. As FV captures both first- and second-order statistics, we use a separate stream for each type of statistic. For BoW, we follow the process outlined in Section 3.1. Specifically, we apply k-means to build a 1000-dimensional dictionary using the same descriptors employed for precomputing FV. The descriptors are then encoded according to Eq. (1), aggregated using the steps described in Section 3.2, and normalized with PN as discussed in Section 3.3. Where applicable, we use 4000-dimensional dictionary for BoW and apply sketching to reduce the vector size to d dimensions. To train Fisher Vectors, we compute 256-dimensional GMM-based dictionaries on descriptors derived from IDT [150], following the steps detailed in Sections 2 and 3.1. PCA is applied to the trajectory (30 dimensions), HOG (96 dimensions), HOF (108 dimensions), MBHx (96 dimensions) and MBHy (96 dimensions) features, yielding a final 213-dimensional local descriptor. The encoded first- and second-order FV representations, each of size $256 \times 213 = 54528$, are sketched to d dimensions as described in Section 3.4. For this purpose, we prepare matrices $\boldsymbol{P}_{(fv1)}$ and $\boldsymbol{P}_{(fv2)}$ as defined in Proposition 2. The sketched first- and second-order representations, $\boldsymbol{\psi}'_{(fv1)} = \boldsymbol{P}_{(fv1)}\boldsymbol{\psi}_{(fv1)}$ and $\boldsymbol{\psi}'_{(fv2)} = \boldsymbol{P}_{(fv2)}\boldsymbol{\psi}_{(fv2)}$, can then be seamlessly integrated with the loss functions detailed in Section 4.7.

I3D OFF. We use the I3D optical flow pre-trained on Kinetics-400 as the feature extractor to obtain the OFF (Fig. 2), denoted as $\boldsymbol{\psi}_{(off)}$. These features have a dimensionality of 1024, which we sketch to d dimensions before using them as training ground truth for the OFF layer. LDOF [10] is used as it effectively handles large displacements.

ODF and SDF. The ODF ground-truth training representations are of size $1214 \times N$, where N is the total number of bounding boxes per video (ranging from 50 to 10,000). The 1214-dimensional features consist of: $80+91$ one-hot encoding for detection classes, 6×7 values representing $\boldsymbol{\phi}(\cdot)$-embedded confidence scores, bounding box coordinates, and frame numbers, and 1001-dimensional ImageNet scores. An alternative representation without the RBF embedding, $\boldsymbol{\phi}(\mathbf{x}) = \mathbf{x}$, results in features of size $1178 \times N$. The SDF ground-truth training representations have a size of $556 \times J$, where J is the number of frames per video. These features include: 300 dimensions ($12 \times 5 \times 5$) capturing spatio-angular gradient distributions, and 256 dimensions (16×16) capturing the luminance of saliency maps. Both ODF and SDF are encoded per video using the multi-moment descriptor in Eq. (16), producing compact representations of size $1178 \times (4 + n')$ and $556 \times (4 + n^\dagger)$, respectively, where n' and $n^\dagger$ are varied between 1 and 5.

These representations are Power Normalized with SigmE and sketched to d dimensions via $\boldsymbol{\psi}'_{(\cdot)} = \boldsymbol{P}_{(\cdot)}\boldsymbol{\psi}_{(\cdot)}$. The resulting features are fed into loss functions such as MSE. Ground-truth representations are used only during training.

Skeleton features. We extract skeleton features (GSF) using ST-GCN pre-trained on large-scale Kinetics-skeletons. These 400-dimensional features are sketched to d dimensions as $\boldsymbol{\psi}'_{(gsf)} = \boldsymbol{P}_{(gsf)}\boldsymbol{\psi}_{(gsf)}$ and used as ground truth for training. For datasets lacking skeleton data (*e.g.*, MPII Cooking Activities, Charades, and EPIC-Kitchens), GSF training is disabled. However, during testing, this stream can leverage pre-trained models (*e.g.*, from Toyota Smarthome) to generate hallucinated features.

Audio features. Audio features (AF) are extracted using the 8-layer SoundNet model pre-trained on two-million unlabeled videos [6]. For datasets with audio, the audio is extracted from video using FFmpeg [142] with a bit rate of 160 kbps, two audio channels, and a sample rate of 44100 Hz. If a video contains sound, a corresponding .wav file is generated. Follow [6], the audio is downsampled to 22 kHZ and converted to a single channel for efficiency, with a minor trade-off in sound quality. The waveform is scaled to the range [-256, 256] for feature extraction from the `pool5` layer. These features are sketched to d dimensions as ground truth. The AF stream is trained only if the dataset contains audio.

HAF. Each stream applies PN to hallucinated features, aligning the hallucinated output $\tilde{\boldsymbol{\psi}}_{(\cdot)}$ (of d dimensions) with the ground-truth features $\boldsymbol{\psi}'_{(\cdot)}$ described earlier. High Abstraction Features (HAF) follow the same steps, combining with other streams and passing the backbone features into PredNet (see Fig. 2). While hallucinated streams co-supervise the backbone through external ground-truth tasks, HAF directly processes backbone features for PredNet.

PredNet. The final component of the pipeline, PredNet, is illustrated in Figure 3b. Its input is $\boldsymbol{\psi}_{(tot)}$ (unsketching) or $\boldsymbol{\psi}'_{(tot)}$ (sketched). This input is passed through batch normalization, followed by a fully connected (FC) layer that produces a C-dimensional representation. This output is optimized using cross-entropy loss.

4.5 Uncertainty Learning

We concatenate the ground truth features of BoW, FV, I3D OFF, ODF, SDF, GSF, and AF into a column feature vector: $\boldsymbol{\psi}' = [\boldsymbol{\psi}'_{bow}, \boldsymbol{\psi}'_{fv1}, \boldsymbol{\psi}'_{fv2}, \boldsymbol{\psi}'_{off}, \boldsymbol{\psi}'_{odf}, \boldsymbol{\psi}'_{sdf}, \boldsymbol{\psi}'_{gsf}, \boldsymbol{\psi}'_{af}]^T \in \mathbb{R}^{d'}$. Similarly, the hallucinated features are concatenated into a column vector: $\tilde{\boldsymbol{\psi}} = [\tilde{\boldsymbol{\psi}}_{bow}, \tilde{\boldsymbol{\psi}}_{fv1}, \tilde{\boldsymbol{\psi}}_{fv2}, \tilde{\boldsymbol{\psi}}_{off}, \tilde{\boldsymbol{\psi}}_{odf}, \tilde{\boldsymbol{\psi}}_{sdf}, \tilde{\boldsymbol{\psi}}_{gsf}, \tilde{\boldsymbol{\psi}}_{af}]^T \in \mathbb{R}^{d'}$.

Mean Squared Error (MSE) assumes the errors across all features are independent and identically distributed (i.i.d.) or, equivalently, that the covariance matrix is diagonal [12, 77]. While this approach enables local noise level estimation, it makes a limiting and often flawed assumption that residuals (errors) across features are uncorrelated. To address this, we extend the noise model to use a multivariate Gaussian likelihood with a full covariance matrix. This matrix captures feature correlations, allowing for structured residual sampling. A maximum likelihood approach is used to train the covariance prediction. Given the hallucinated features $\tilde{\boldsymbol{\psi}}'$ and the ground truth features $\boldsymbol{\psi}'$, we replace the MSE loss with the following uncertainty learning objective:

$$
\begin{aligned}
&\arg\max \log \mathcal{N}(\tilde{\boldsymbol{\psi}}; \boldsymbol{\psi}', \boldsymbol{\Sigma}) \\
&= \arg\max \log \frac{1}{(2\pi)^{\frac{d'}{2}} |\boldsymbol{\Sigma}|^{\frac{1}{2}}} e^{-\frac{1}{2}(\tilde{\boldsymbol{\psi}}-\boldsymbol{\psi}')^{\mathrm{T}}\boldsymbol{\Sigma}^{-1}(\tilde{\boldsymbol{\psi}}-\boldsymbol{\psi}')} \\
&= \arg\min \frac{d'}{2}\log(2\pi) + \frac{1}{2}\log(|\boldsymbol{\Sigma}|) + \frac{1}{2}(\tilde{\boldsymbol{\psi}}-\boldsymbol{\psi}')^{\mathrm{T}}\boldsymbol{\Sigma}^{-1}(\tilde{\boldsymbol{\psi}}-\boldsymbol{\psi}') \\
&\approx \arg\min \log(|\boldsymbol{\Sigma}|) + (\tilde{\boldsymbol{\psi}}-\boldsymbol{\psi}')^{\mathrm{T}}\boldsymbol{\Sigma}^{-1}(\tilde{\boldsymbol{\psi}}-\boldsymbol{\psi}'), \qquad (18)
\end{aligned}
$$

where $\boldsymbol{\Sigma}$ is the covariance matrix, and d' is the feature dimension. Note that the constant term is removed, as it does not influence the optimisation. To estimate the covariance matrix $\boldsymbol{\Sigma}$, we use a Covariance Estimation Network (CENet), discribed in the following section.

4.6 Covariance Estimation Network

A deep neural network is used to estimate the covariance matrix $\boldsymbol{\Sigma}$, using the latent representation $\mathscr{X}_{(rgb)}$ as input. By definition, $\boldsymbol{\Sigma}$ is symmetric and positive definite. For a feature vector $\boldsymbol{\psi}'$ of dimensionality d', the covariance matrix contains $(d'^2 - d')/2 + d'$ unique parameters. This matrix captures structured information about reconstruction uncertainty, allowing the hallucinated output to more closely approximate the ground truth features.

Since $\boldsymbol{\Sigma}$ appears in its inverted form ($\boldsymbol{\Omega} = \boldsymbol{\Sigma}^{-1}$) in the negative log-likelihood calculation (Eq. (18)), it is more practical to estimate the precision matrix $\boldsymbol{\Omega}$ directly. This also simplifies the log determinant computation, as $\log(|\boldsymbol{\Sigma}|) = -\log(|\boldsymbol{\Omega}|)$. Rewriting Eq. (18), the objective function for feature hallucination becomes:

$$
\arg\min (\tilde{\boldsymbol{\psi}}-\boldsymbol{\psi}')^{\mathrm{T}}\boldsymbol{\Omega}(\tilde{\boldsymbol{\psi}}-\boldsymbol{\psi}') - \kappa\log(|\boldsymbol{\Omega}|), \qquad (19)
$$

where $\kappa \geq 0$ (typically $\kappa \neq d'$) adjusts the penalty for large uncertainty.

Using Cholesky decomposition, the precision matrix can be represented as $\boldsymbol{\Omega} = \boldsymbol{\omega}\boldsymbol{\omega}^{\mathrm{T}}$, where $\boldsymbol{\omega}$ is a lower triangular matrix. The covariance network estimates $\boldsymbol{\omega}$ explicitly. With this decomposition, it is trivial to evaluate two terms in Eq. (18): (i) the reconstruction error: $\mathbf{y}^{\mathrm{T}}\mathbf{y}$, where $\mathbf{y} = \boldsymbol{\omega}^{\mathrm{T}}(\tilde{\boldsymbol{\psi}} - \boldsymbol{\psi}')$, and (ii) the log determinant: $\log(|\boldsymbol{\Sigma}|) =$

$-2\sum_i^{d'} \log \omega_{i,i}$, where $\omega_{i,i}$ represents the i-th diagonal element of $\boldsymbol{\omega}$. To ensure $\boldsymbol{\Omega}$ is positive-definite, the diagonal elements are constrained to be strictly positive, for instance, by estimating $\log \omega_{i,i}$ using a network.

Computational complexity. Sampling from $\boldsymbol{\Sigma}$ involves solving a triangular system of equations via backward substitution, which requires $O(d'^2)$ operations. The number of parameters to estimate grows quadratically with d', making direct estimation feasible only for features with small dimensionality.

Sparse Cholesky decomposition. To scale to higher-dimensional feature vectors, we impose a fixed sparsity pattern on $\boldsymbol{\omega}$, estimating only the non-zero values. Specifically, $\omega_{i,j}$ is non-zero only if i and j are within the same batch of sampled feature indices. This reduces the maximum number of non-zero elements in $\boldsymbol{\omega}$ to $(d^{*2}-d^*)/2+d'$, where $d^* \ll d'$ represnts the batch size. We set d^* to be the number of feature types, ensuring each sampled index corresponds to a feature within a specific type. With the sparsity, the uncertainty model for each feature representation resembles a Gaussian Random Field for residuals. A zero value in the precision matrix $\boldsymbol{\Omega}$ at position (i,j) indicates conditional independence between feature i and j.

Parallel computing. This sparse representation allows efficient evaluation of the uncertainty model without constructing a full dense matrix. Similarly, sampling can be performed by solving a sparse system of equations. The method is GPU-parallelizable, as each batch can be evaluated independently.

CENet architecture. The Covariance Estimation Network (CENet) is composed of several key components designed to estimate the precision matrix $\boldsymbol{\Omega}$ efficiently and ensure it is symmetric and positive-definite. First, the network begins with a simple Multi-Layer Perceptron (MLP) that includes two fully connected (FC) layers, a batch normalization layer, and a ReLU activation function applied between them. This MLP processes the input feature vector to generate a refined representation.

Next, a zero padding layer expands the output dimensions from $(d^{*2}-d^*)/2+d'$ to $d' \cdot d'$, aligning it with the dimensionality required for the precision matrix. The expanded vector is then reshaped into a square matrix of size $d' \times d'$. To enforce the positive-definiteness of the precision matrix, an exponential block is applied to the diagonal elements of the reshaped matrix. This ensures that the diagonal entries are strictly positive by transforming them to remove the logarithmic scale introduced during estimation.

Finally, the precision matrix $\boldsymbol{\Omega}$ is derived using the Cholesky decomposition, $\boldsymbol{\Omega} = \boldsymbol{\omega}\boldsymbol{\omega}^{\mathrm{T}}$, where $\boldsymbol{\omega}$ is a lower triangular matrix estimated by the network. This decomposition guarantees the symmetry and positive-definiteness of $\boldsymbol{\Omega}$.

The input to the CENet can be either (i) the intermediate representation $\mathscr{X}_{(rgb)}$, or (ii) the concatenated hallucinated feature $\tilde{\boldsymbol{\psi}}$, as described earlier. Both variants are evaluated to determine the optimal input data for CENet's operation.

4.7 Objective and its Optimization

Objective function. During training, we optimize a combined loss function that incorporates an uncertainty learning term for training hallucination streams and a classification loss:

$$
\begin{aligned}
\ell^*(\mathscr{X},\mathbf{y};\bar{\boldsymbol{\Theta}}) = \alpha\,\big((\tilde{\boldsymbol{\psi}}-\boldsymbol{\psi}')^{\mathrm{T}}\boldsymbol{\Omega}(\tilde{\boldsymbol{\psi}}-\boldsymbol{\psi}')-\kappa\log(|\boldsymbol{\Omega}|)\big) \\
+\ell\Big(f(\boldsymbol{\psi}'_{(tot)};\boldsymbol{\Theta}_{(pr)}),\mathbf{y};\boldsymbol{\Theta}_{(\ell)}\Big), \\
\text{where: } \forall i\in\mathscr{H},\tilde{\boldsymbol{\psi}}_i = g(\hbar(\mathscr{X},\boldsymbol{\Theta}_i),\eta),\boldsymbol{\psi}'_i=\boldsymbol{P}_i\boldsymbol{\psi}_i, \\
\tilde{\boldsymbol{\psi}}=\oplus_{i\in\mathscr{H}}\tilde{\boldsymbol{\psi}}_i,\boldsymbol{\psi}'=\oplus_{i\in\mathscr{H}}\boldsymbol{\psi}'_i, \\
\boldsymbol{\psi}_{(haf)} = g\big(\hbar(\mathscr{X},\boldsymbol{\Theta}_{(haf)}),\eta\big), \\
\boldsymbol{\psi}'_{(tot)} = \boldsymbol{P}_{(tot)}\Big[\tilde{\boldsymbol{\psi}};\boldsymbol{\psi}_{(haf)}\Big], \\
\boldsymbol{\Omega}=\boldsymbol{\omega}\boldsymbol{\omega}^{\mathrm{T}},\boldsymbol{\omega}=c(\mathscr{X},\boldsymbol{\Theta}_{(cov)}) \\
\text{or } \boldsymbol{\omega}=c(\tilde{\boldsymbol{\psi}},\boldsymbol{\Theta}_{(cov)}).
\end{aligned}
\tag{20}
$$

The equation above represents a trade-off between the uncertainty learning term, $(\tilde{\boldsymbol{\psi}}-\boldsymbol{\psi}')^{\mathrm{T}}\boldsymbol{\Omega}(\tilde{\boldsymbol{\psi}}-\boldsymbol{\psi}')-\kappa\log(|\boldsymbol{\Omega}|)$, and the classification loss, $\ell(\cdot,\mathbf{y};\boldsymbol{\Theta}_{(\ell)})$, with labels $\mathbf{y}\in\mathscr{Y}$ and parameters $\boldsymbol{\Theta}_{(\ell)}\equiv\{\boldsymbol{W},\mathbf{b}\}$. The trade-off is controlled by the constant $\alpha\geq 0$. Uncertainty is computed over hallucination streams $i\in\mathscr{H}$, where $\mathscr{H}\equiv\{(bow),(fv1),(fv2),(off),(odf),(sdf),(gsf),(af)\}$, a set of hallucination streams that can be adjusted based on the task at hand. Additionally, $g(\cdot,\eta)$ is the Power Normalization function described in Section 3.3, and $c(\cdot,\boldsymbol{\Theta}_{(cov)})$ is the CENet module with parameters $\boldsymbol{\Theta}_{(cov)}$. The PredNet module, $f(\cdot;\boldsymbol{\Theta}_{(pr)})$, has learnable parameters $\boldsymbol{\Theta}_{(pr)}$. The hallucination streams $\{\hbar(\cdot,\boldsymbol{\Theta}_i),i\in\mathscr{H}\}$ produce the corresponding hallucinated BoW/FV/OFF/ODF/SDF/GSF/AF representations $\{\tilde{\boldsymbol{\psi}}_i,i\in\mathscr{H}\}$. The HAF stream is denoted by $\boldsymbol{\psi}_{(haf)}$, which is generated by $\hbar(\cdot,\boldsymbol{\Theta}_{(haf)})$.

The parameters $\{\boldsymbol{\Theta}_i,i\in\mathscr{H}\}$ are learned for the hallucination streams, while $\boldsymbol{\Theta}_{(haf)}$ is learned for the HAF stream. The complete set of parameters is denoted as $\bar{\boldsymbol{\Theta}}\equiv(\{\boldsymbol{\Theta}_i,i\in\mathscr{H}\},\boldsymbol{\Theta}_{(haf)},\boldsymbol{\Theta}_{(pr)},\boldsymbol{\Theta}_{(cov)},\boldsymbol{\Theta}_{(\ell)})$. Furthermore, the projection matrices $\{\boldsymbol{P}_i,i\in\mathscr{H}\}$ are used for count sketching of the ground-truth BoW/FV/OFF/ODF/SDF/GSF/AF feature vectors $\{\boldsymbol{\psi}_i,i\in\mathscr{H}\}$, and the corresponding sketched/compressed representations are $\{\boldsymbol{\psi}'_i,i\in\mathscr{H}\}$. The projection matrix $\boldsymbol{P}_{(tot)}$ handles the concatenation of the hallucinated BoW/FV/OFF/ODF/SDF/GSF/AF representations with HAF. This results in $\boldsymbol{\psi}_{(tot)}=\Big[\tilde{\boldsymbol{\psi}};\boldsymbol{\psi}_{(haf)}\Big]$, and its sketched counterpart $\boldsymbol{\psi}'_{(tot)}$ is fed into the PredNet module f.

Section 3.4 explains how to select the matrices $\boldsymbol{P}$. If sketching is not needed, we simply set $\boldsymbol{P}$ to be the identity matrix, $\boldsymbol{P}=\mathbb{I}$. In our experiments, we set $\alpha=1$.

We also explore a variant in which we use a weighted average of several streams fed into the PredNet module f:

$$\boldsymbol{\psi}'_{(tot)}=\frac{1}{|\mathscr{H}^*|+1}\Big(w_{(haf)}\boldsymbol{\psi}_{(haf)}+\sum_{i\in\mathscr{H}^*}w_i\tilde{\boldsymbol{\psi}}_i\Big),$$
$$\tilde{\boldsymbol{\psi}}_{(det)}=\frac{1}{|\mathscr{D}|}\sum_{i\in\mathscr{D}}w_i\tilde{\boldsymbol{\psi}}_i,\ \text{and}\ \tilde{\boldsymbol{\psi}}_{(sal)}=\frac{1}{|\mathscr{S}|}\sum_{i\in\mathscr{S}}w_i\tilde{\boldsymbol{\psi}}_i, \tag{21}$$

where $\mathscr{H}^*\equiv\{(fv1),(fv2),(bow),(off),(odf),(sdf)\}$, $\mathscr{D}\equiv\{(det1),...,(det4)\}$, and $\mathscr{S}\equiv\{(sal1),(sal2)\}$. Let $\mathscr{T}$ be set to $\mathscr{H}^*$, $\mathscr{D}$ or $\mathscr{S}$, and the weights are defined as:

$$w_i=\frac{1}{|\mathscr{T}|}\frac{\max(w_i'^{\beta},\rho)}{\sum_{j\in\mathscr{T}}\max(w_j'^{\beta},\rho)}. \tag{22}$$

Optimization. Before training the CNN, we first train an SVM for each ground-truth stream separately (using a manageable subset of the data). The weights w' are set proportionally to the accuracies obtained on the validation set. For the HAF stream, we set $w'_{(haf)}=\frac{1}{|\mathscr{H}^*|+1}$ and $\rho=0.1$. In the first few epochs (*i.e.*, 10), we set $\beta=0$, ensuring that all streams receive equal weights. Subsequently, we perform a Golden-section search to determine the optimal $\beta\geq 0$ in each epoch. We start with boundary values $\beta\in\{0,50\}$, train an SVM on a manageable subset of training data, evaluate β on the validation set, and update the boundary values for the next epoch.

Eq. (22) has an interesting property: for $\beta=0$, all weights are equal, $w_i=1/|\mathscr{T}|$. As $\beta\to\infty$, the weights become binary: $w_i=1$ if $w_i=\max(\{w_i\}_{i\in\mathscr{T}})$, and $w_i=0$ otherwise. Thus, β interpolates between equal weighting and a winner-takes-all approach.

We minimize $\ell^*(\mathscr{X},\mathbf{y};\bar{\boldsymbol{\Theta}})$ with respect to the parameters of each stream: $\{\boldsymbol{\Theta}_i,i\in\mathscr{H}\}$ for hallucination streams, $\boldsymbol{\Theta}_{(haf)}$ for the HAF stream, $\boldsymbol{\Theta}_{(cov)}$ for CENet, $\boldsymbol{\Theta}_{(pr)}$ for PredNet, and $\boldsymbol{\Theta}_{(\ell)}$ for the classification loss. In practice, we alternate between two minimization steps: one forward and backward pass to update the parameters $\{\boldsymbol{\Theta}_i,i\in\mathscr{H}\}$ and $\boldsymbol{\Theta}_{(cov)}$ for uncertainty learning, followed by another forward and backward pass for the classification loss ℓ. This can be viewed as a multi-task learning process, where we simultaneously learn BoW/FV/OFF/ODF/SDF/GSF/AF and label tasks. We use the Adam minimizer with an initial learning rate of 10^{-4}, halved every 10 epochs, and train for 50–100 epochs, depending on the dataset.

5 Experiment

Below, we demonstrate the effectiveness of our method. For smaller datasets, such as HMDB-51 and YUP++, we use the I3D, VTN, and FASTER backbones. For Charades, EPIC-KITCHENS-55, and Toyota Smarthome, we also investigate the AssembleNet++ backbones. For large-scale Kinetics-400, Kinetics-600, and Something-Something V2, we use the recent, popular VideoMAE V2 and InternVideo2.

5.1 Datasets and Evaluation Protocols

HMDB-51 [92] consists of 6766 internet videos across 51 classes, with each video containing approximately 20 to 1000 frames. We report the mean accuracy across three splits.

YUP++ [45] contains 20 scene classes of video textures, with 60 videos per class. The splits include scenes captured by either static or moving cameras. We use the standard splits (1/9 of the dataset for training) for evaluation.

MPII Cooking Activities [122] includes high-resolution videos of people cooking various dishes. The 64 activities span 3748 clips, including coarse actions such as *opening refrigerator*, and fine-grained actions like *peel*, *slice*, and *cut apart*. We report the mean Average Precision (mAP) over 7-fold cross validation. For the human-centric protocol [22, 24], we use Faster R-CNN [120] to crop the video around human subjects.

Charades [132] consists of of 9848 videos of daily indoor activities, 66,500 clip annotations, and 157 classes.

EPIC-KITCHENS-55 [31] is a multi-class, egocentric dataset with approximately 28K training videos associated with 331 noun and 125 verb classes. The dataset contains of 39,594 segments across 432 videos. We follow the protocol in [7] and evaluate our model on the validation set, as well as the standard seen (S1: 8,047 videos), and unseen (S2: 2,929 videos) test sets.

Toyota Smarthome [32] consists of 16,115 RGB+D video clips spanning 31 activity classes. This dataset poses several challenges, such as high intra-class variation, high class imbalance, simple and composite activities, and activities with similar motion and variable duration. Activities are annotated with both coarse and fine-grained labels. There are two evaluation protocols for activity classification: cross-subject and cross-view. Follow [32], we report the mean per-class accuracy.

Kinetics-400 [74] contains 400 human action classes with over 300K video clips, each containing 180 frames. The dataset covers a wide range of actions, including sports, everyday tasks, and human-object interactions, and is split into training, validation, and test sets.

Kinetics-600 [14] extends the Kinetics-400 dataset, with over 500K video clips. It includes a more diverse set of actions compared to Kinetics-400, covering a broad spectrum of human activities from both indoor and outdoor environments.

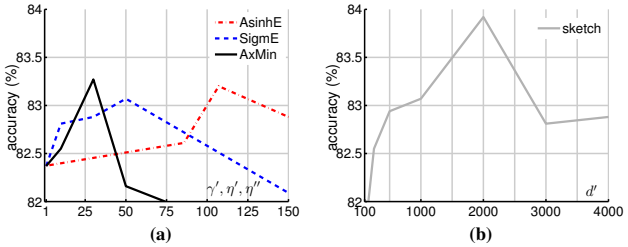


Fig. 4: Evaluations of (fig. 4a) Power Normalization and (fig. 4b) sketching on the HMDB-51 dataset (split 1 only).

Something-Something V2 [55] is a large-scale action recognition dataset consisting of 220K videos across 174 action classes. The dataset focuses on human-object interactions, where actions are typically described by phrases like *putting something in something* or *picking something up*.

5.2 Evaluations of Various Design Components

Sketching and Power Normalization. Since PredNet uses a fully connected (FC) layer (see Figure 3b), we expect that limiting the input size to this layer through count sketching, as described in Section 3.4, should improve performance. Additionally, given that visual and video representations often suffer from burstiness, we investigate the AsinhE, SigmE, and AxMin methods, as outlined in Remarks 1, 2 and 3.

Figure 4a investigates the classification accuracy on the HMDB-51 dataset (split 1) when our HAF and BoW/FV feature vectors $\{\tilde{\boldsymbol{\psi}}_i, i \in \mathscr{H}\}$ and $\boldsymbol{\psi}_{(haf)}$ (described in Sections 4.4 and 4.7) are processed through Power Normalizing functions: AsinhE, SigmE and AxMin, prior to concatenation (see Figure 2). Our experiments show that all PN functions perform similarly, improving results from a baseline 82.29% to approximately 83.20% accuracy. A similar improvement is observed on YUP++ (*static*), with accuracy rising from 93.15% to 94.44%. For simplicity, we use AsinhE for PN in the following experiments.

Figure 4b illustrates the effect of applying count sketching to the concatenated HAF and BoW/FV feature vectors $\boldsymbol{\psi}_{(tot)}$, resulting in $\boldsymbol{\psi}'_{(tot)}$ (see Section 4.7 for reference to symbols), on the HMDB-51 dataset (split 1). This approach improves the accuracy from 82.88% to 83.92% for $d' = 2000$. This improvement is expected, as reduced size of $\boldsymbol{\psi}'_{(tot)}$ results in fewer parameters for the FC layer in PredNet, reducing overfitting. Similarly, on the YUP++ dataset (split *static*), the accuracy increases from 93.15% to 94.81%.

ODF+SVM. Firstly, we evaluate our ODF on SVM using the HMDB-51 dataset. We set $n' = 3$ for Eq. (16) and compare various detector backbones and pooling strategies. Table 1 shows that all detectors perform similarly, with

	sp1	*sp2*	*sp3*	mean acc.
det1	42.00	39.74	40.39	40.72
det1	40.49	40.13	39.67	40.09
det3	43.78	44.05	41.97	**43.26**
det4	41.08	39.22	40.39	40.23
all+avg	42.50	41.05	41.01	41.52
all+max	43.25	42.32	42.09	42.55
all+wei	45.80	44.52	44.09	**44.80**
DEEP-HAL+all+avg	83.25	82.24	82.84	82.77
DEEP-HAL+all+max	83.18	81.86	82.84	82.62
DEEP-HAL+all+wei	84.01	83.25	83.10	**83.45**

Table 1: Evaluations of ODF on HMDB-51. (Top) Performance is evaluated using backbones: (*det1*) Inception V2, (*det2*) Inception ResNet V2, (*det3*) ResNet101, and (*det4*) NASNet. (Middle) Results for average pooling, max pooling, and weighted mean combinations of all detectors are reported as (*all+avg*), (*all+max*), and (*all+wei*), respectively. (Bottom) Pre-trained DEEP-HAL combined with all four detectors is evaluated using average pooling, max pooling, and weighted mean.

	avg	*max*	*wei*
all	55.12	42.34	**60.52**
DEEP-HAL+all	74.22	71.85	**75.74**

Table 2: Pooling on YUP++. Results are shown for average pooling (*avg*), max pooling (*max*), and weighted mean (*wei*) of all detectors (*all*) compared to pre-trained DEEP-HAL combined with all detectors using average pooling, max pooling, and weighted mean.

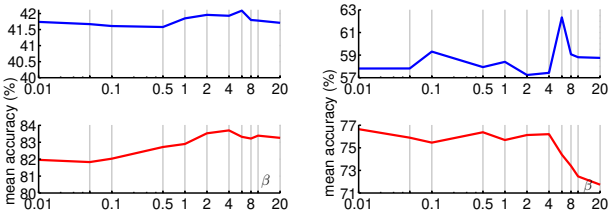


Fig. 5: Impact of β in the weighted mean on classification performance. Figure 5a presents results for HMDB-51 with (*top*) four combined detectors + SVM and (*bottom*) DEEP-HAL with four combined detectors + SVM. Figure 5b shows results for YUP++.

(*det3*) slightly outperforming the others. Additionally, max-pooling on ODFs from all four detectors marginally outperforms average-pooling. However, only the weighted mean (*all+wei*), as defined in Eq. (22), outperforms (*det3*), highlighting the importance of robust aggregation of ODFs. Similarly, when combining pre-trained DEEP-HAL with all detectors, the weighted mean (*DEEP-HAL+all+wei*) achieves the best performance. Table 2 shows a similar trend on YUP++.

We train SVM only on videos where at least one detection occurred, so the resulting accuracy of 75.74% is lower than the main results reported on the full pipeline. Finally, Figure 5 demonstrates that $\beta \neq 1$ has a positive impact on reweighting.

SDF. The SDF achieves accuracies of 24.35% on HMDB-51 and 32.68% on YUP++. This is expected, as SDF does not capture discriminative information directly, but instead iden-

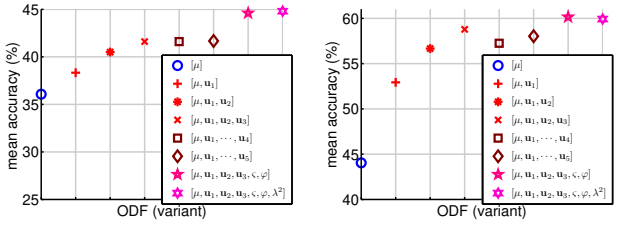


Fig. 6: Evaluation of ODF with SVM using the weighted mean across four detectors. Figure 6a and Figure 6b show results on HMDB-51 and YUP++, respectively. The parameters $\boldsymbol{\mu}$, $\mathbf{u}_1,\ldots,\mathbf{u}_i$, $\boldsymbol{\varsigma}$, $\boldsymbol{\varphi}$, and $\boldsymbol{\lambda}^2$ correspond to the entries in Eq. (16).

tify salient spatial and temporal regions to focus the main network's attention on.

Multi-moment descriptor. Figure 6 shows that concatenating the mean and three eigenvectors, as defined in Eq. (16), yields good results. However, adding additional vectors leads to a deterioration in performance. Adding skewness and kurtosis ($\boldsymbol{\varsigma}$ and $\boldsymbol{\varphi}$) further improves the results, while the inclusion of eigenvalues has a limited impact.

ImageNet (global score) *vs*.object detectors. Various scores from the object and saliency detectors we use cannot be directly integrated into DEEP-HAL due to the varying number of detected objects and frames. Therefore, we propose and use ODF and SDF descriptors. Additionally, We note that using a simplified version of ODF, which stacks ImageNet-1K scores per frame into a matrix (without detectors) and applies our multi-moment descriptor, yields approximately 4% worse results on Charades compared to our DEEP-HAL+ODF (detectors-based approach), which achieves 48.0% mAP. This is expected, as ImageNet-1K is trained in a multi-class setting (one object per image), while detectors allow us to robustly model the distribution of object classes and locations per frame.

Reweighting mechanism. In this experiment, we employ the DEEP-HAL pipeline and hallucinate ODF and SDF ($d=512$). Typically, we use three levels of weighting mean pooling, applied to: (i) four object detectors constituting the ODF, (ii) two saliency detectors constituting the SDF, and (iii) the final combination of HAF/BoW/FV/OFF/ODF/SDF. Below, we investigate the performance of a single weighted mean pooling step, applied simultaneously to four object detectors, two saliency detectors, and the remaining streams.

Table 3 shows that using a flat, single-level weighted mean pooling yields 86.1% accuracy on the HMDB-51, which is approximately 1.4% lower than using three levels of weighted mean pooling. We also observe that the improvement from using *wei+3 levels* is very minimal (around 0.2%). We expect that applying a single weighted mean pooling per modality is a reasonable strategy, as object category detectors, for example, may yield similar responses.

	sp1	*sp2*	*sp3*	mean acc.
wei+flat	86.47	85.56	86.27	86.10
wei+1 level	88.00	86.33	87.20	87.18
wei+2 levels	88.20	86.50	87.33	87.34
wei+3 levels	88.37	86.80	87.52	**87.56**

Table 3: Evaluation of the flat single-level weighted mean (*wei+flat*) versus different hierarchical levels of weighted mean pooling on HMDB-51. *wei+1 level* indicates that weighted mean pooling is applied solely to the four object detectors in ODF. *wei+2 levels* extends this by applying weighted pooling to both ODF and SDF. *wei+3 levels* adds a final weighted combination of HAF, BoW, FV, OFF, ODF, and SDF.

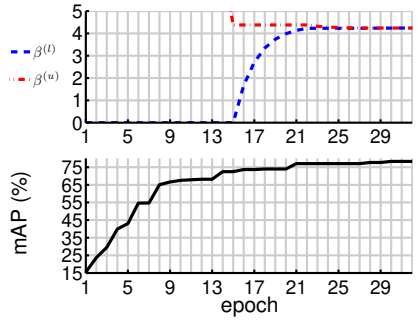


Fig. 7: Visualization of the Golden-section search for the weighting mechanism at the final level. (*top*) Depicts the convergence of the lower and upper estimates, $\beta^{(l)}$ and $\beta^{(u)}$, over the course of training epochs. (*bottom*) For each epoch, β is set as the midpoint $\beta=0.5(\beta^{(l)}+\beta^{(u)})$, and the corresponding validation score (mAP) is evaluated on MPII (*split1*). As training progresses, mAP steadily improves and stabilizes, reflecting the convergence of the Golden-section search algorithm.

Therefore, they should first be reweighted for the best 'combined detector' performance before being combined with more complementary modalities.

Finally, Figure 7 (top) demonstrates how our Golden-search selects the optimal β on the MPII validation set (*split1*). Figure 7 (bottom) shows the corresponding validation mAP (note that this is not the mAP score on the test set). For the first 10 epochs we use $\beta=0$, and we start the Golden-search from epoch 11.

Input to CENet. The input to our CENet can either be the intermediate representation $\mathscr{X}_{(rgb)}$ from the backbone or the concatenated feature representations $\tilde{\boldsymbol{\psi}}$ from hallucination (see Sec. 4.6), we compare the performance of these two variants on Charades. As shown in Table 7, using concatenated feature representations as input to CENet improves the performance of AR by more than 0.5% compared to the use of intermediate representation. Both variants use the same experimental settings, *i.e.*, the same number of multiple sketches (MSK) and the same feature dimensionality d through sketching, across all four backbones. The reasons for this improvement are: (i) the intermediate representations are 'raw' and not sufficiently representative for CENet to identify the relationships between each feature descriptor and its corresponding uncertainty, and (ii) the hallucinated

features are more informative than the intermediate representations. Therefore, we use the hallucinated feature representations as input to CENet (see Fig. 2).

Sketched feature dimension per modality. We evaluate the effects of feature dimension d through sketching on both Charades and EPIC-KITCHENS-55. On Charades, $d=128$ achieves almost identical performance to $d=256$ (within 0.3% difference, see Table 7) across all backbones. Considering both the performance gain and the computational cost, we choose $d=128$ for subsequent experiments on Charades. On the validation set of EPIC-KITCHENS-55, with AssembleNet++ and VTN backbones, $d=128$ yields very similar results to $d=256$. However, with I3D and FASTER backbones, $d=128$ decreases performance by more than 0.5%. Thus, for the seen and unseen test protocols, we choose $d=128$ for AssembleNet++ and VTN backbones and $d=256$ for I3D and FASTER backbones to achieve better performance.

Multiple sketches. On Charades, we observe that using more multiple sketches (MSK) helps improve AR performance. Specifically, 4× MSK outperforms 2× MSK by 0.2%, and 8× MSK outperforms 4× MSK by 0.9%. However, the improvement becomes smaller when 12× MSK is used compared to 8× MSK across all backbones (approximately 0.1% improvement, see Table 7), except the FASTER framework. Considering both the performance gain and computational cost, we set the number of MSK to 8 for the experiments on EPIC-KITCHENS-55 and Toyota Smarthome.

Backbone	Modality					Hal. loss	Mean acc.(%)
	BoW/FV/OFF	ODF	SDF	GSF	AF		
I3D						-	74.8
	✓					MSE	83.3
	✓	✓				MSE	84.3
	✓		✓			MSE	83.9
	✓	✓	✓			MSE	85.2
	✓	✓	✓			MSE	87.0
	✓	✓	✓			MSE	87.6
	✓	✓	✓	✓		MSE	88.0
	✓	✓	✓	✓	✓	MSE	88.2
	✓	✓	✓	✓	✓	Uncert.	**88.6**
VTN-MobileNet						-	59.2
	✓					MSE	67.3
	✓					Uncert.	68.1
	✓	✓	✓			Uncert.	70.9
	✓	✓	✓	✓		Uncert.	72.2
	✓	✓	✓	✓	✓	MSE	71.9
	✓	✓	✓	✓	✓	Uncert.	**72.5**
VTN-ResNet						-	63.1
	✓					MSE	73.3
	✓					Uncert.	75.2
	✓	✓	✓			Uncert.	77.0
	✓	✓	✓	✓		Uncert.	78.3
	✓	✓	✓	✓	✓	MSE	77.9
	✓	✓	✓	✓	✓	Uncert.	**78.6**
FASTER						-	74.9
	✓					MSE	80.3
	✓					Uncert.	81.9
	✓	✓	✓			Uncert.	83.7
	✓	✓	✓	✓		Uncert.	85.8
	✓	✓	✓	✓	✓	MSE	85.6
	✓	✓	✓	✓	✓	Uncert.	**86.2**
ADL+I3D 81.5% [151]				Full-FT I3D 81.3% [15]			
EvaNet (Ensemble) 82.3% [115]				PA3D + I3D 82.1% [177]			
DEEP-HAL (exact) 82.50% [162]				DEEP-HAL 82.48% [162]			

Table 4: Evaluation results for (*top*) our proposed methods and (*bottom*) comparisons with state-of-the-art approaches on HMDB-51.

5.3 Discussion on Fine-grained Action Recognition

In this section, we choose the I3D backbone to discuss fine-grained AR.

MPII. Table 6 shows a ∼3.0% mAP improvement over the (*DEEP-HAL*) baseline due to the detectors capturing human-object interaction, which helps model fine-grained AR. Additionally, incorporating skeleton and audio modalities further improves the performance by approximately 1%, even though these modalities are not available in the MPII dataset.

Charades. Table 7 (top) presents the relative gains of our hallucination pipeline using the I3D backbone. We evaluate both (*ODF*) and (*SDF*) with 512-dimensional sketching (*SK512*) and observe that ODF outperforms SDF, with both methods surpassing the baseline (*DEEP-HAL*) [162].

We also observe that combining ODF and SDF (with SK512) achieves a 49.1% mAP, representing a ∼**6**% gain over the baseline. This demonstrates the high complementarity of ODF and SDF. Using a larger sketch (ODF/SDF, SK1000) results in a 50.1% mAP, which closely matches the performance of (*DEEP-HAL with ODF/SDF (exact)*, 50.16%), where 'exact' denotes late fusion by concatenation of ODF and SDF streams with the DEEP-HAL stream fed into PredNet. The matching results between (*DEEP-HAL with ODF/SDF (SK1000)*) and (*DEEP-HAL+ODF/SDF (exact)*) show that we can hallucinate ODF and SDF at test time while maintaining full performance, thus saving computational time and boosting reults on Charades by ∼**6**% over the baseline. Moreover, with additional skeleton (GSF) and audio information (AF), our model achieves 52.1%, further improving performance by approximately 2%. In contrast, SlowFast networks [43] and AssembleNet [125] achieve 45.2% and 51.6% on Charades, respectively.

EPIC-KITCHENS-55. Table 8 shows the experimental results. Our model learns human-like semantic features due to ODF/SDF, and even skeleton and audio information can be synthesized. There is no evidence suggesting that a backbone alone can discover these features without guidance. Comparing MPII (3748 clips) with the larger EPIC-KITCHENS-55 (39594 clips), both related to cooking tasks, SDF+ODF boosts MPII from 81.8 to 84.8%, and boosts EPIC-KITCHENS-55 from 32.51% (DEEP-HAL) to 35.88% (on the seen classes protocol), and from 22.33% (DEEP-HAL) to 27.32% (on the unseen classes protocol). This demonstrates a ∼**3**% improvement on both MPII and EPIC-KITCHENS-55, with EPIC-KITCHENS-55 contain-

Backbone	Modality				Hal. loss	Mean acc. (%)
	BoW/FV/OFF	ODF	SDF	AF		
I3D					-	89.9
	✓				MSE	92.6
	✓	✓			MSE	93.2
	✓		✓		MSE	92.8
	✓	✓	✓		MSE	94.4
	✓	✓	✓		Uncert.	94.9
	✓	✓	✓	✓	MSE	95.5
	✓	✓	✓	✓	Uncert.	**96.0**
VTN-ResNet					-	85.0
	✓				MSE	89.7
	✓	✓	✓		MSE	90.1
	✓	✓	✓		Uncert.	90.3
	✓	✓	✓	✓	MSE	92.0
	✓	✓	✓	✓	Uncert.	**92.8**
FASTER					-	90.1
	✓				MSE	93.0
	✓	✓	✓		MSE	93.2
	✓	✓	✓		Uncert.	93.8
	✓	✓	✓	✓	MSE	94.9
	✓	✓	✓	✓	Uncert.	**95.3**
T-ResNet 87.6% [45]					ADL I3D 91.7% [151]	
DEEP-HAL 92.6% [162]					MSOE-two-stream 91.9% [57]	

Table 5: Evaluation results for (*top*) our proposed methods and (*bottom*) comparisons with state-of-the-art approaches on YUP++. Note that GSF is not applicable to this dataset.

Backbone	Modality					Hal. loss	mAP(%)
	BoW/FV/OFF	ODF	SDF	GSF	AF		
I3D						-	74.8
	✓					MSE	80.4
	✓	✓	✓			MSE	84.8
	✓	✓	✓			Uncert.	85.6
	✓	✓	✓	✓		Uncert.	86.2
	✓	✓	✓	✓	✓	MSE	86.0
	✓	✓	✓	✓	✓	Uncert.	**86.5**
VTN-ResNet						-	66.3
	✓	✓	✓			Uncert.	75.5
	✓	✓	✓	✓		Uncert.	76.1
	✓	✓	✓	✓	✓	Uncert.	**76.7**
FASTER						-	75.5
	✓	✓	✓			Uncert.	82.9
	✓	✓	✓	✓		Uncert.	83.7
	✓	✓	✓	✓	✓	Uncert.	**84.3**
KRP-FS+IDT 76.1% [24]						GRP+IDT 75.5% [22]	
I3D+BoW/OFF MTL 79.1% [162]						DEEP-HAL 81.8% [162]	

Table 6: Evaluation results for (*top*) our proposed methods and (*bottom*) comparisons with state-of-the-art approaches on the MPII dataset.

ing nearly 10× more clips than MPII. With the addition of GSF and AF, we achieve an extra performance gain of around 8%.

Toyota Smarthome. Table 9 presents results for Toyota Smarthome. With ODF/SDF, our model outperforms I3D baseline by 4% on average. Since this dataset provides reliable skeleton information, adding GSF improves the performance by an additional 4%. This suggests that robust skeletons enhance AR. Moreover, hallucinating AF further boosts performance by 1–3%. Even with the I3D backbone (using only RGB), our model still achieves very competitive results, outperforming recent methods such as VPN [33] and UNIK [180] by 3% and 6% (on average), respectively.

Backbone	Modality						SK dim. d	num. of input to MSK	num. of input to CENet	mAP (%)
	BoW/FV	OFF	ODF	SDF	GSF	AF				
I3D							-	-	-	37.2
	✓						1000	-	-	41.9
	✓	✓					1000	2	-	42.0
	✓	✓					1000	4	-	42.2
	✓	✓					1000	8	-	43.1
	✓	✓	✓				512	8	-	47.2
	✓	✓		✓			512	8	-	45.3
	✓	✓	✓	✓			512	8	-	49.1
	✓	✓	✓	✓			1000	8	-	50.1
	✓	✓	✓	✓	✓		512	8	-	50.9
	✓	✓	✓	✓	✓		256	8	-	50.5
	✓	✓	✓	✓	✓		128	8	-	50.3
	✓	✓	✓	✓	✓		128	8	$\mathscr{X}_{(rgb)}$	50.5
	✓	✓	✓	✓	✓		128	8	Ψ	51.0
	✓	✓	✓	✓	✓		128	12	Ψ	51.0
	✓	✓	✓	✓	✓	✓	512	8	-	51.6
	✓	✓	✓	✓	✓	✓	256	8	-	51.5
	✓	✓	✓	✓	✓	✓	128	8	-	51.2
	✓	✓	✓	✓	✓	✓	128	8	$\mathscr{X}_{(rgb)}$	51.3
	✓	✓	✓	✓	✓	✓	128	8	Ψ	52.0
	✓	✓	✓	✓	✓	✓	128	12	Ψ	**52.1**
AssembleNet++							-	-	-	53.8
							-	-	-	56.7*
	✓	✓	✓	✓			512	8	-	55.8
	✓	✓	✓	✓			512	8	-	60.7*
	✓	✓	✓	✓			1000	8	-	56.9
	✓	✓	✓	✓			1000	8	-	62.0*
	✓	✓	✓	✓	✓		512	8	-	58.0
	✓	✓	✓	✓	✓	✓	512	8	-	58.5
	✓	✓	✓	✓	✓	✓	256	8	-	58.2
	✓	✓	✓	✓	✓	✓	128	8	-	58.0
	✓	✓	✓	✓	✓	✓	128	8	$\mathscr{X}_{(rgb)}$	58.3
	✓	✓	✓	✓	✓	✓	128	8	$\mathscr{X}_{(rgb)}$	64.7*
	✓	✓	✓	✓	✓	✓	128	8	Ψ	58.8
	✓	✓	✓	✓	✓	✓	128	12	Ψ	**59.0**
	✓	✓	✓	✓	✓	✓	128	12	Ψ	**65.3***
VTN							-	-	-	43.5
	✓	✓	✓	✓			512	8	-	48.3
	✓	✓	✓	✓			256	8	-	48.0
	✓	✓	✓	✓			128	8	-	47.8
	✓	✓	✓	✓	✓		128	8	-	50.2
	✓	✓	✓	✓	✓	✓	128	8	-	53.2
	✓	✓	✓	✓	✓	✓	128	8	$\mathscr{X}_{(rgb)}$	53.7
	✓	✓	✓	✓	✓	✓	128	8	Ψ	54.3
	✓	✓	✓	✓	✓	✓	128	12	Ψ	**54.5**
FASTER							-	-	-	40.7
	✓	✓	✓	✓			512	8	-	46.0
	✓	✓	✓	✓			256	8	-	45.2
	✓	✓	✓	✓			128	8	-	45.0
	✓	✓	✓	✓	✓		128	8	-	47.8
	✓	✓	✓	✓	✓	✓	128	8	-	49.7
	✓	✓	✓	✓	✓	✓	128	8	$\mathscr{X}_{(rgb)}$	51.2
	✓	✓	✓	✓	✓	✓	128	8	Ψ	52.7
	✓	✓	✓	✓	✓	✓	128	12	Ψ	**53.3**
LFB 42.5% [174]							ActionCLIP 44.3% [168]			
En-VidTr-L 47.3% [187]							MoViNet-A4 48.5% [81]			
SlowFast 45.2% [43]							AssembleNet 51.6% [125]			
AssembleNet-101 58.6% [125]							AssembleNet++ 50 59.8% [124]			

*These results are obtained without pre-training on Kinetics-400.

Table 7: Evaluation results for (*top*) our methods and (*bottom*) comparisons with state-of-the-art approaches on the Charades dataset.

5.4 Discussion on the Uncertainty Learning

We now evaluate the performance of using MSE and uncertainty learning for feature hallucination. We observe that uncertainty learning helps improve the AR performance across all datasets. On HMDB-51 and YUP++ (Table 4 and 5), uncertainty learning loss improves performance by approximately 1% on average for all backbones. On MPII (Table 6), hallucinating additional ODF/SDF with uncertainty learning outperforms using MSE by 0.8% with the I3D backbone. Furthermore, hallucinating all features with

Backbone +BoW/FV/OFF	Modality ODF/SDF	GSF	AF	SK dim. d	Verbs	Nouns	Actions
					Validation		
I3D	✓			1000	55.4	33.3	21.5
	✓	✓		512	59.2	38.1	26.4
	✓	✓		256	59.0	37.7	26.0
	✓	✓		128	58.6	37.1	25.6
	✓	✓	✓	512	**62.0**	**40.9**	**30.5**
	✓	✓	✓	256	**61.7**	**40.4**	**30.3**
	✓	✓	✓	128	60.9	39.0	29.7
AssembleNet++	✓			512	57.2	34.8	23.2
	✓			256	56.8	34.6	22.9
	✓			128	56.6	34.0	22.5
	✓	✓		128	63.5	41.2	40.9
	✓	✓	✓	128	**68.2**	**46.1**	**36.3**
VTN	✓			256	54.3	32.7	19.8
	✓			128	54.0	32.6	19.7
	✓	✓		256	58.0	36.2	25.2
	✓	✓		128	57.6	36.0	24.7
	✓	✓	✓	256	**60.0**	**38.8**	**29.4**
	✓	✓	✓	128	**59.6**	**38.2**	**29.3**
FASTER	✓			256	55.0	33.1	20.9
	✓			128	54.2	32.7	19.5
	✓	✓		256	58.5	36.7	24.3
	✓	✓		128	58.1	36.0	23.9
	✓	✓	✓	256	**62.0**	**41.8**	**31.5**
	✓	✓	✓	128	61.3	41.6	31.0
LFB Max [174]					52.6	31.8	22.8
WeakLargeScale [51]					58.4	36.9	26.1
					Test s1 (seen)		
I3D	✓	✓	✓	256	70.0	50.7	38.7
AssembleNet++	✓	✓	✓	128	76.5	57.2	47.9
VTN	✓	✓	✓	128	67.7	49.2	38.0
FASTER	✓	✓	✓	256	71.0	51.1	39.2
TSN Fusion [31]					48.2	36.7	20.5
LFB Max [174]					60.0	45.0	32.7
WeakLargeScale [51]					65.2	45.1	34.5
					Test s2 (unseen)		
I3D	✓	✓	✓	256	61.7	40.0	30.3
AssembleNet++	✓	✓	✓	128	68.9	48.2	39.1
VTN	✓	✓	✓	128	58.8	39.6	29.0
FASTER	✓	✓	✓	256	63.6	40.9	30.9
TSN Fusion [31]					39.4	22.7	10.9
LFB Max [174]					50.9	31.5	21.2
WeakLargeScale [51]					57.3	35.7	25.6

Table 8: Experimental results on the EPIC-KITCHENS-55 dataset.

Backbone +BoW/FV/OFF	Modality ODF/SDF	GSF	AF	Hal. loss	CS	CV_1	CV_2
I3D				-	53.4	34.9	45.1
	✓			MSE	57.6	38.2	49.3
	✓			Uncert.	58.2	39.0	50.1
	✓	✓		Uncert.	62.3	43.1	55.2
	✓	✓	✓	Uncert.	**65.1**	**44.3**	**56.3**
AssembleNet++				-	63.6	45.2	55.8
	✓			MSE	65.5	48.0	59.0
	✓			Uncert.	66.0	48.7	59.9
	✓	✓		Uncert.	70.8	52.7	65.5
	✓	✓	✓	Uncert.	**72.3**	**54.1**	**68.8**
VTN				-	53.0	33.2	43.7
	✓			MSE	57.0	38.2	48.7
	✓			Uncert.	57.8	39.3	49.2
	✓	✓		Uncert.	61.3	43.0	54.1
	✓	✓	✓	Uncert.	**62.6**	**43.9**	**55.7**
FASTER				-	53.7	35.0	46.7
	✓			MSE	59.1	41.2	53.2
	✓			Uncert.	60.0	42.1	54.0
	✓	✓		Uncert.	63.9	46.5	57.7
	✓	✓	✓	Uncert.	**65.5**	**48.8**	**59.4**
Separable STA [32]					54.2	35.2	50.3
NPL [117]					-	39.6	54.6
VPN [33]					60.8	43.8	53.5
UNIK [180]					63.1	22.9	61.2

Table 9: Experimental results on the Toyota Smarthome dataset.

uncertainty learning improves performance by an additional 0.5%.

As mentioned earlier, skeleton data often contains noise, which affects the GSF, we evaluate the impact of noisy GSF on AR. On Toyota Smarthome, hallucinating GSF improves AR by 4–5% across all backbones, as this dataset provides skeleton data. Using GSF on Charades improves results by 2–3% on average, despite Charades not having skeleton data; in this case, we use the GSF stream pre-trained on skeleton data from Toyota Smarthome for hallucination. The use of skeleton information improves AR performance by approximately 1% and 4% on MPII and EPIC-KITCHENS-55, respectively. Note that these two cooking datasets are fine-grained AR datasets focusing on specific regions of actions (*e.g.*, hands), making it more difficult to obtain full human skeleton data. Nevertheless, our proposed model, with uncertainty, is still able to hallucinate some skeleton features even without full human skeleton data.

Additionally, videos typically contain background sounds that are irrelevant to the actions observed, such as music, TV, or the noise from coffee or washing machines. In Table 8, we see that the inclusion of audio features boosts AR by more than 3% across all backbones. The largest performance gain (around 5%) is observed when using the AssembleNet++ backbone. Our experimental results demonstrate that these irrelevant sounds do not confuse the model, highlighting the robustness of our approach to noisy and unconstrained audio sources.

5.5 Discussion on Modalities

We observe that using additional modalities helps improve AR. The performance gain from the use of ODF/SDF is approximately 2%, 1.5%, 4%, and 3% on HMDB-51, YUP++, MPII, and Charades, respectively. On fine-grained AR tasks, the performance gain from using ODF/SDF averages around 3%. This suggests that object detection and saliency information contribute significantly to AR, particularly in fine-grained AR, where the surrounding objects of performers provide important cues to actions.

The inclusion of skeleton features further improves performance by approximately 1–2% on average for HMDB-51, MPII and Charades. On EPIC-KITCHENS-55 and Toyota Smarthone, the performance gain from skeleton features is more substantial, exceeding 4% and 3–4%, respectively. Since YUP++ is a natural scene classification dataset and does not require skeleton information, we do not hallucinate skeleton features for this dataset.

	K400	K600	SSv2
VideoMAE V2 [154]	87.2	88.8	77.0
+ BoW/FV	87.5	88.9	77.0
+ BoW/FV + OFF	**87.6**	**89.1**	77.3
+ BoW/FV + OFF + ODF/SDF	87.5	**89.1**	**77.4**
InternVideo2$_{s1}$ [170]	91.3	91.4	77.1
+ BoW/FV	**91.8**	91.6	77.0
+ BoW/FV + OFF	**91.8**	**91.7**	77.2
+ BoW/FV + OFF + ODF/SDF	91.6	91.5	**77.3**

Table 10: Experimental results on the large-scale Kinetics-400 (K400), Kinetics-600 (K600), and Something-Something V2 (SSv2) datasets. For both VideoMAE V2 (ViT-g) and InternVideo2$_{s1}$ (1B), finetuned model weights specific to each dataset are used. Ground-truth descriptors for BoW/FV, OFF, and ODF/SDF used in feature hallucination are generated using 80,000 training samples from Mini-Kinetics-200 [176], balancing computational cost and feature storage constraints.

We also find that the use of sound features further boosts AR performance. Adding sound information increases performance by approximately 3.5% on average for EPIC-KITCHENS-55 and Toyota Smarthome. This is expected, as visual and audio modalities are often highly correlated. For fine-grained AR, audio information is especially important for action-related tasks. On Charades, the use of the VTN backbone yields the largest performance gain (around 3%) for hallucinated sound information, as VTN is particularly well-suited for handling audio features. On HMDB-51 and MPII, the performance gain is smaller (around 0.5–1%) due to the lack of reliable sound information. For natural scene classification on YUP++, the sound features improve performance by 1–2%.

Our analysis highlights that AR benefits from additional modalities such as skeleton and sound, in addition to the commonly used RGB and optical flow videos. Our proposed model can synthesize multiple modalities through a simple hallucination step, boost AR performance without introducing extra computational cost during the test stage.

5.6 Discussion on the Large-scale Datasets

We also evaluate our model on three widely-used large-scale action recognition datasets: Kinetics-400, Kinetics-600, and Something-Something V2. The experimental results are summarized in Table 10.

To address computational costs and feature storage constraints, we use 80,000 training samples from Mini-Kinetics-200 [176] to generate ground-truth descriptors for BoW, FV, OFF, ODF and SDF, which are used for feature hallucination. For this process, we use our uncertainty loss and set the sketching dimension to $d = 128$ for the power-normalized ground-truth descriptors. These descriptors are hallucinated from the pooled spatiotemporal token embeddings of pretrained VideoMAE V2 (ViT-g) and InternVideo2$_{s1}$-1B encoders. After training the hallucination streams, we freeze their weights and proceed to finetune the HAF and PredNet components for each dataset. This approach allows HAF and PredNet to adapt to the hallucinated features of different datasets, enhancing both computational and storage efficiency.

Notably, despite the hallucinated streams being trained on feature descriptors derived from a subset of Kinetics-400 (Mini-Kinetics-200), they still enhance action recognition performance on large-scale datasets. Interestingly, the improvements brought by BoW/FV and BoW/FV+OFF are particularly significant, yielding gains of over 0.4%, 0.3%, and 0.1% on Kinetics-400, Kinetics-600, and Something-Something V2, respectively. Adding ODF and SDF contributes only marginal improvements. This is likely because both VideoMAE V2 and InternVideo2 are self-supervised learning frameworks that effectively capture deep semantic features through spatiotemporal masking and video frame reconstruction. However, handcrafted descriptors like those encoded via BoW and FV still play a valuable role in boosting performance, even when derived from a subset of Kinetics-400 and pretrained hallucinated streams.

On Something-Something V2, we observe that BoW/FV does not improve performance for either VideoMAE V2 or InternVideo2. However, incorporating OFF significantly enhances performance, underscoring the challenging nature of Something-Something V2, which requires robust temporal reasoning. Motion-related information, such as that captured by OFF, proves crucial for these improvements.

In contrast, on Kinetics-400 and Kinetics-600, adding OFF results in negligible performance gains. This indicates that motion information is far less relevant for the Kinetics datasets, as also demonstrated in recent studies [111, 52]. These findings highlight the differing demands of these datasets, with temporal motion cues playing a pivotal role in Something-Something V2 but being less critical for the Kinetics datasets. This highlights the need for video understanding researchers to collect and curate datasets where motion, temporal information, and reasoning are crucial, fostering advancements that better serve the video understanding research community.

5.7 Discussion on Action Recognition Backbones

We observe that the top performance of our model depends on the backbone used. Based on our comparisons, the AssembleNet++ backbone performs the other three backbones (I3D, VTN and FASTER). Models with AssembleNet++ achieve a performance boost of approximately 13%, 5%, and 10% over I3D and FASTER backbones on Charades, EPIC-KITCHENS-55, and Toyota Smarthome, respectively.

The VTN backbone (with ResNet) performs slightly worse than the other three backbones (AssembleNet, I3D and FASTER), as it is a lightweight model designed for AR on hardware with limited computational power (*e.g.*, mobile devices). The FASTER framework is also a lightweight model that avoids redundant computation between neighboring clips. However, it uses an efficient model (R(2+1)D-50) for subtle motions, performing better than VTN by approximately 8%, 3%, 8%, 2%, and 4% on HMDB-51, YUP++, MPII, EPIC-KITCHENS-55, and Toyota Smarthome, respectively. On charades, VTN performs slightly better than FASTER (by about 1%). This can be attributed to: (i) the lack of available skeleton data and unreliable pose estimators on this dataset (due to pose complexity and mobile camera movements), (ii) noise in the sound data, and (iii) VTN's ability to handle time series data (*e.g.*, audio and skeletons) more effectively.

We also observe that, for the hallucination task, the AssembleNet++ backbone generally outperforms the FASTER and I3D backbones, which perform equally well and both outperform VTN. This is expected because: (i) AssembleNet++ is optimized for hallucination tasks, (ii) both FASTER and I3D backbones use (2+1)D or 3D ConvNets, which are better suited for hallucinating spatio-temporal features (*e.g.*, OFF, GSF), and (iii) VTN only uses 2D CNNs for video frame embedding, which is less efficient compared to the use of (2+1)D and 3D CNNs.

Although the VTN backbone performs worse than the others, its performance remains competitive compared to most existing state-of-the-art methods, especially on fine-grained AR tasks such as Charades, EPIC-KITCHENS-55, and Toyota Smarthome (Table 7, 8 and 9).

Recent self-supervised pretraining frameworks, such as VideoMAE V2 and InternVideo2 (Table 10), have emerged as powerful video learners, effectively capturing self-supervisory features that were traditionally the domain of handcrafted methods. These video foundation models, pretrained on large-scale visual and motion datasets, are designed to extensively use spatiotemporal information, excelling in various video processing tasks. Interestingly, despite the remarkable capabilities of these models, we observe that handcrafted descriptors still provide a meaningful performance boost. This can be attributed to their unique design and specialized focus, which complement the broader but more generalized feature representations learned by these video foundation models.

Our approach is 'orthogonal' to these developments, which focus on extensive mining for combinations of neural blocks/dataflows to obtain an 'optimal' pipeline. We achieve similar results with a simpler approach based on self-supervised learning. Our pipeline is more lightweight by comparison, as it does not require computations of optical flow, detections, or segmentation masks at test time.

	DET1: Inception V2	*DET2: Inception ResNet V2*	*DET3: ResNet101 AVA*	*DET4: NASNet*	ODF total (+SVD)
sec. per frame	0.07	0.38	0.10	0.91	1.46 (+0.09)
s.p.c. HMDB-51	6.5	35.3	9.3	84.5	135.6 (+0.5)
s.p.c. YUP++	9.7	52.7	13.9	126.2	202.5 (+0.8)
s.p.c. MPII	12.4	67.1	17.7	160.8	258.0 (+1.3)
s.p.c. Charades	21.0	114.2	30.0	273.5	438.7 (+2.6)
s.p.c. EPIC-Kitchens	20.3	110.4	29.0	264.3	424.0 (+2.6)
s.p.c. Toyota Smarthome	16.9	92.0	24.2	220.2	353.3 (+2.1)

Table 11: Statistics for the object detectors used in our experiments. The table provides timings such as seconds per frame (denoted as *sec. per frame*) and seconds per clip (denoted as *s.p.c.*) for detectors used by ODF. Additionally, the total time incurred by a combined detector (*ODF total*) is shown. We also report the time taken for the full Singular Value Decomposition (SVD) and all other ODF operations, assuming approximately 5 detections per frame.

	SAL1: MNL	*SAL2: ACLNet*	SDF *total* (+Eq. (17))	ODF+SDF total (+Eq. (17)+SVD)
sec. per frame	0.60	0.30	0.90 (+0.003)	2.36 (+0.1)
s.p.c. HMDB-51	55.7	27.9	83.6 (+0.3)	219.2 (+0.8)
s.p.c. YUP++	83.2	41.6	124.8 (+0.4)	327.3 (+1.2)
s.p.c. MPII	106.0	53.0	159.0 (+0.5)	417.0 (+1.8)
s.p.c. Charades	180.3	90.1	270.4 (+0.9)	709.1 (+3.5)
s.p.c. EPIC-Kitchens	174.3	87.1	261.4 (+0.9)	685.4 (+2.9)
s.p.c. Toyota Smarthome	145.2	72.6	217.8 (+0.7)	571.1 (+2.4)

Table 12: Statistics for the saliency detectors used in our experiments. The table presents timings such as seconds per frame (*sec. per frame*) and seconds per clip (*s.p.c.*) for detectors used by SDF. The total time incurred by the combined detector (*SDF total*) is also provided. Additionally, we report the time taken for the descriptor in Eq. (17) and all other SDF operations. Finally, the combined time for both ODF and SDF operations (*SDF+ODF total*) is included.

5.8 Computational Costs and Efficiency

Table 11 shows the timing for object detectors used by ODF descriptors during training. The detections from all four object detectors we use take approximately 1.47 seconds per frame. Therefore, obtaining four ODF descriptors per clip (a uniquely annotated sequence for training or classification) takes between 136 and 441 seconds. Table 12 presents the timing for saliency detectors used in our SDF descriptors during training. The detections for both saliency detectors take around 0.9 seconds per frame, with obtaining both SDF descriptors per clip taking between 84 and 271 seconds.

It is important to note that the majority of the computational cost arises from the detectors rather than from the ODF and SDF descriptors, whose computational cost is minimal. Moreover, the idea of learning these computationally expensive representations during training proves highly valuable. While the total computation time per training clip ranges from 220 to 712 seconds, these representations are obtained virtually for free (in milliseconds) during testing, thanks to the DET1, ..., DET4 and SAL1/SAL2 units, as shown in Figure 2. Assuming that 25% of clips in charades are used for testing, this results in a savings of 137 days of

	Video	IncV2	IncResV2	Res101	NASNet	MNL	ACLNet
HMDB-51	2.2 GB	64.5 GB	64.7 GB	69.3 GB	69.9 GB	6.7 GB	2.4 GB
YUP++	788.6 MB	12.9 GB	13.5 GB	4.2 GB	14.6 GB	1.6 GB	687.7 MB
MPII	8.7 GB	65.7 GB	83.8 GB	48.2 GB	97.3 GB	11.3 GB	2.5 GB
Charades	59.0 GB	453.6 GB	473.1 GB	155.8 GB	490.2 GB	210.8 GB	76.6 GB

Table 13: Storage statistics for original videos and extracted features. We present the storage sizes of raw object detection features extracted using Inception V2 (IncV2), Inception ResNet V2 (IncResV2), ResNet101 (Res101), and NASNet, as well as raw saliency detection features obtained from MNL and ACLNet. All extracted features are stored in HDF5 format, while the original videos are in formats such as AVI.

	no. of frames	*av. frame count*	*no. of videos*	*no. of clips*	*no. of classes*
HMDB-51	628635	92.91	6766	6766	51
YUP++	166463	138.72	1200	1200	20
MPII	662394	176.73	44	3748	60
Charades	19978821	300.51	9848	66500	157
EPIC-Kitchens	$\sim$ 11.5M	290.43	432	39596	149
Toyota Smarthome	3.9M	242.01	16115	16115	31
Kinetics-400	54M	180	300K	300K	400
Kinetics-600	90M	180	500K	500K	600
Something-Something V2	22M	100	220K	220K	174

Table 14: Statistics of the datasets used in our experiments.

computation on a single GPU (or the equivalent of 1 day's savings on 137 GPUs). Given the 6% improvement on Charades over the baseline (without ODF and SDF descriptors), coupled with these substantial computational savings, we believe these statistics highlight the value of our approach.

5.9 Limitations and Challenges

While our framework performs well on several benchmarks, it does face practical limitations, particularly in terms of computational costs, scalability, and sensitivity to parameter choices.

Computational complexity is a major challenge, especially during the training stage when extracting ground-truth feature descriptors, such as ODF and SDF (see Table 11 and Table 12). These feature extraction processes can be computationally expensive and result in high storage consumption, particularly for large-scale datasets. For example, the original Charades dataset is 59.0 GB, whereas the extracted object detection and saliency detection features require a total of 1.54 TB and 287.4 GB of storage, respectively, which are approximately 26.7 and 4.87 times the size of the original videos (see Table 13). Table 14 presents basic statistics for the datasets used in our experiments. Notebly, Kinetics-400, Kinetics-600 and Something-Something V2 are among the largest datasets in our study, with approximately 54M, 90M, and 22M frames, respectively.

The framework's sensitivity to parameter choices also plays a crucial role in its effectiveness. For instance, the sketch size (d') is key in determining both computational efficiency and the quality of feature representation. A small d' can introduce noise, leading to poor approximations of the original feature set, which may negatively impact the model's performance.

Additionally, the framework faces challenges when applied to recent self-supervised pretraining frameworks such as VideoMAE V2 and InternVideo2. These models can use self-supervised learning techniques and are trained end-to-end, eliminating the need for manually extracted handcrafted features during training. However, these handcrafted features, although not required, still provide significant improvements to these models. The heavy computational cost and storage constraints of extracting the full set of handcrafted features make it infeasible to fully exploit their potential (see the total number of frames in Table 14), which in turn complicates training models with the complete set of semantically rich features. This issue limits the ability to optimize hallucination stream weights, making training more challenging.

Finally, robustness to noisy data is another potential limitation. In real-world scenarios, noisy or poorly extracted ground truth features can degrade performance, particularly in training the hallucination streams. This may lead to poor-quality hallucinated features during testing, further affecting the model's overall effectiveness.

6 Conclusion

In this work, we introduced a novel multimodal action recognition framework that enhances recognition accuracy by integrating diverse auxiliary features while reducing reliance on computationally expensive handcrafted descriptors at inference. To guide the model toward action-relevant regions, we proposed two domain-specific descriptors: Object Detection Features (ODF), which capture contextual cues from multiple object detectors, and Saliency Detection Features (SDF), which emphasize spatial and intensity patterns critical for action understanding. To handle incomplete multimodal data, we developed a self-supervised hallucination mechanism that synthesizes missing cues at test time, enriching feature representations without increasing computational overhead. Furthermore, we incorporated aleatoric uncertainty modeling and a robust loss function to mitigate feature noise, improving the robustness of our model in fine-grained action recognition tasks. Our framework remains compatible with state-of-the-art architectures, including I3D, AssembleNet, Video Transformer Network, FASTER, and recent models such as VideoMAE V2 and InternVideo2. Extensive experiments on Kinetics-400, Kinetics-600, and Something-Something V2 confirm that our method achieves state-of-the-art performance, demonstrating its effectiveness in capturing fine-grained action dynamics and advancing multimodal action recognition.

Acknowledgements. The authors thank CSIRO Scientific Computing for help. This work was also supported by 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. We sincerely thank the anonymous reviewers for their invaluable insights and constructive feedback, which have greatly contributed to improving our work.
Data availability statement: All datasets used and studied in this paper are publicly available.

A Hallucination Quality

Below, we analyze the quality of hallucinated BoW and FV streams compared to their corresponding ground-truth feature vectors.

Figure 8 presents histograms of the squared differences between hallucinated features and ground-truth ones. Specifically, we plot histograms of $\{(\tilde{\psi}_{(bow),mn} - \psi_{(bow),mn})^2, m \in \mathscr{I}_{1000}, n \in \mathscr{N}\}$, where m iterates over 1000 features and n spans all videos in the dataset. For clarity, counts for training and testing splits are normalized by 1000 (the number of features) and the number of videos in each split, respectively. The histograms use bins of size 0.01, creating smooth, continuous-like plots.

Figure 8a demonstrates that during training, our BoW hallucination unit based on fully connected (*FC*) layers closely approximates the ground-truth BoW descriptors. Histograms at epochs 1, 5, 15, and 25 are shown with colors transitioning from red to blue. Early epochs exhibit a modest peak near the first bin, but as training progresses, this peak intensifies while subsequent bins diminish. This pattern reflects the reduction in approximation error over time.

Similarly, Figure 8b illustrates that hallucinated BoW descriptors also closely approximate ground-truth descriptors in the testing split. Comparisons between testing and training histograms for BoW, as well as first- and second-order FV descriptors, reveal only minor differences. A ratio analysis of testing to training bins shows variations between 0.8× and 1.25×. For clarity, we omit plots of FV testing split comparisons, as they align closely with training results.

Figures 8c and 8d show that the first- and second-order FV terms (*FV1* and *FV2*) are also well-learned by our hallucinating units. The results are displayed for the training split, as the testing behavior closely mirrors the training performance.

Finally, Figures 8e, 8f, 8g, and 8h highlight similar learning trends for BoW training and testing splits, as well as for the first- and second-order FV terms (training split only), using our hallucination unit based on *FC* layers without sketching or power normalization (*-SK/PN*).

B Visualization using UMAP

Figure 9 presents a UMAP [106] visualization of the YUP++ dataset. In Fig. 9a, the top-left corner shows samples from three classes, represented in red, green, and blue. These classes exhibit partial overlap in this representation. In contrast, Fig. 9b depicts the same region, but the samples from the red, green, and blue classes are now more distinctly separated, indicating improved class-wise clustering.

Figure 10 illustrates a UMAP visualization of the HMDB-51 dataset. In Fig. 10a, the bottom-left corner contains samples from two overlapping classes, shown in red and blue. However, in Fig. 10b, the samples from these two classes are better separated, and their respective clusters appear more clearly delineated, demonstrating improved class-wise organization in this visualization.

	sp1	*sp2*	*sp3*	*sp4*	*sp5*	*sp6*	*sp7*	mAP
HAF*+BoW halluc.	78.8	75.0	84.1	76.0	77.0	78.3	75.2	77.8
HAF*+BoW hal.+MSK/PN	80.1	79.2	84.8	83.9	80.9	78.5	75.5	80.4
HAF•+BoW halluc.	78.8	78.3	84.2	77.4	77.1	78.3	75.2	78.5
HAF•+BoW hal.+MSK/PN	80.8	80.9	85.0	83.9	82.0	79.8	79.6	**81.7**

Table 15: Evaluations on MPII. The (*HAF*+BoW halluc.*) represents our pipeline using the BoW stream, where (*) indicates human-centric pre-processing with a 256-pixel height resolution. The (*HAF*+BoW hal.+MSK/PN*) extends this pipeline by incorporating multiple sketches per BoW followed by Power Normalization (*PN*). Similarly, (•) denotes human-centric pre-processing with an increased 512-pixel height resolution.

C Higher Resolution Frames on MPII

For the human-centric pre-processing applied to MPII, denoted by (*), we observe that the bounding boxes used for extracting human subjects are of low resolution. To address this, we first resize the RGB frames to 512 pixels in height (instead of 256 pixels), compute the corresponding optical flow, and then extract the human subjects. This adjustment effectively increases the resolution by a factor of 2×.

In Table 15, results for (*HAF*+BoW halluc.*), our pipeline incorporating the BoW stream, and (*HAF*+BoW hal.+MSK/PN*), which includes multiple sketches and PN, are shown for the standard 256 pixels height resolution. These results, denoted by (*), are taken from [162].

The (*HAF•+BoW halluc.*), which also includes the BoW stream, and (*HAF•+BoW hal.+MSK/PN*) pipelines are analogous but computed with the increased 512-pixel height resolution, denoted by (•). As shown in the table, increasing the resolution by 2× before human detection, extracting higher-resolution subjects, and then scaling them to a 256-pixel height for yields a 1.3% improvement in accuracy.

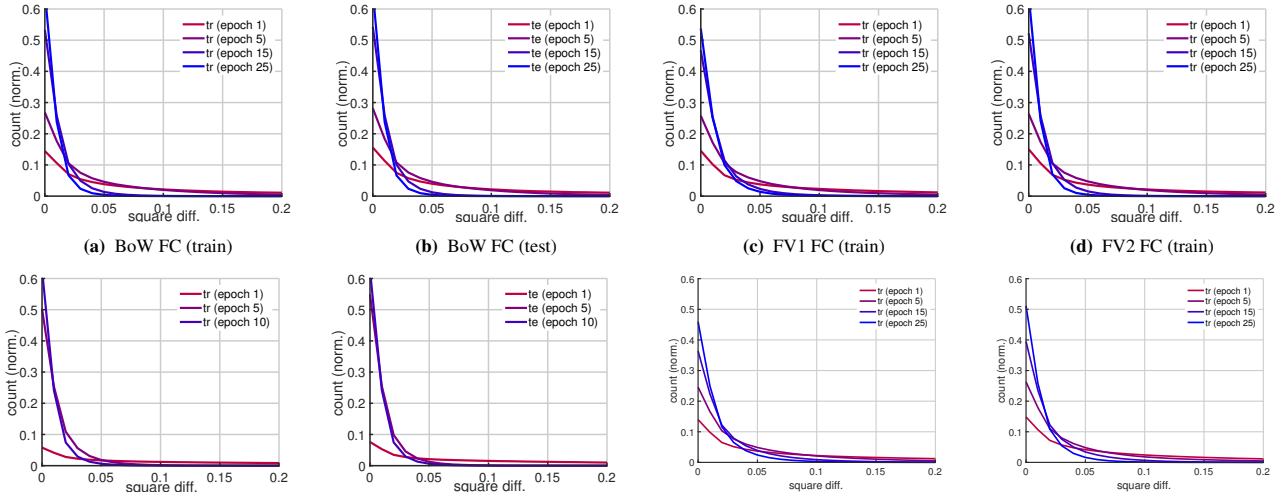
D Data Pre-processing

For all video datasets, we apply a data augmentation strategy that includes random cropping of videos and left-right flips on both RGB and optical flow frames. During testing, we use center cropping and avoid flipping.

For the MPII dataset, which involves human-centric pre-processing, we first apply a human detector. Next, we randomly crop around the bounding box containing the human subject. This crop is included in the final sequence. We also allow scaling, zooming in, and left-right flipping. For longer videos, we sample sequences to create a 64-frame clip. For shorter videos (fewer than 64 frames), we repeat the sequence multiple times to match the expected input length. Finally, we scale the pixel values of both RGB and optical flow frames to the range between -1 and 1.

References

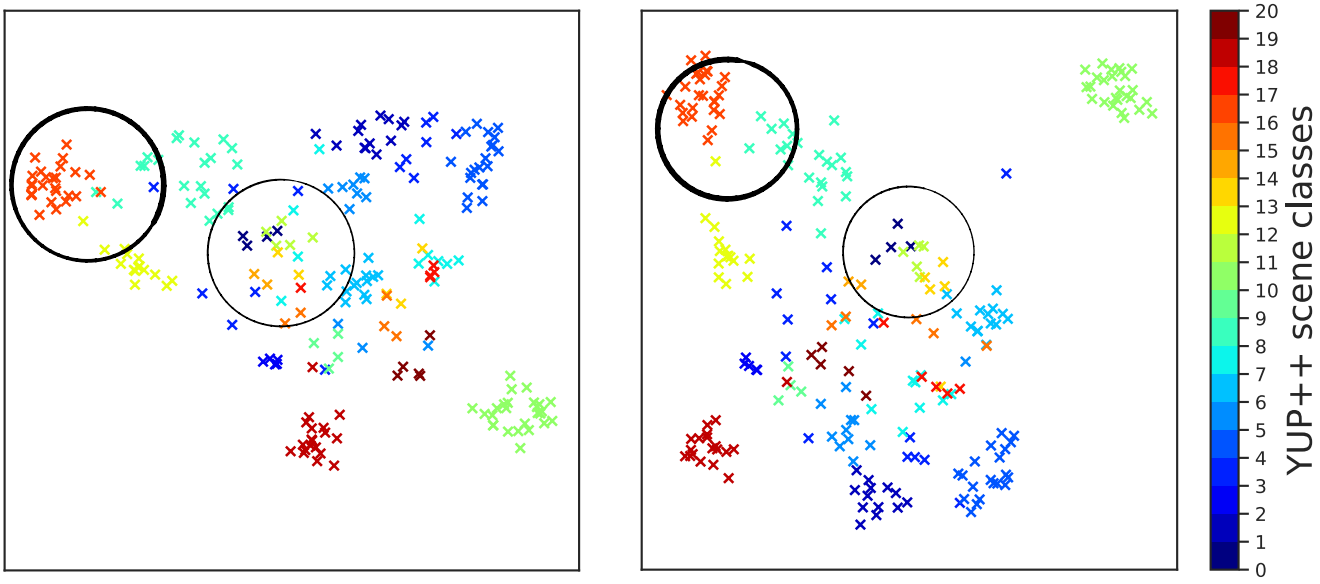
1. Abdullah, L.N., Noah, S.A.M.: Integrating audio visual data for human action detection. In: 2008 Fifth International Conference on Computer Graphics, Imaging and Visualisation, pp. 242–246 (2008). DOI 10.1109/CGIV.2008.65 5
2. Ahmad, T., Jin, L., Zhang, X., Lai, S., Tang, G., Lin, L.: Graph convolutional neural network for human action recognition: A comprehensive survey. IEEE Transactions on Artificial Intelligence **2**(2), 128–145 (2021). DOI 10.1109/TAI.2021.3076974 4
3. Akbari, A., Jafari, R.: A deep learning assisted method for measuring uncertainty in activity recognition with wearable sensors. In: 2019 IEEE EMBS International Conference on Biomedical



(a) BoW FC (train) **(b)** BoW FC (test) **(c)** FV1 FC (train) **(d)** FV2 FC (train)

(e) BoW Conv (train) **(f)** BoW Conv (test) **(g)** FV1 FC (train), -SK/PN **(h)** FV2 FC (train), -SK/PN

Fig. 8: Evaluation of the squared difference between hallucinated and ground-truth representations on HMDB-51 (split 1). Experiments in the top row use (*FC*) streams with sketching and PN. The two leftmost plots in the bottom row use (*Conv*) streams, while the two rightmost plots in the bottom row examine (*FC*) streams without sketching or PN (*-SK/PN*).

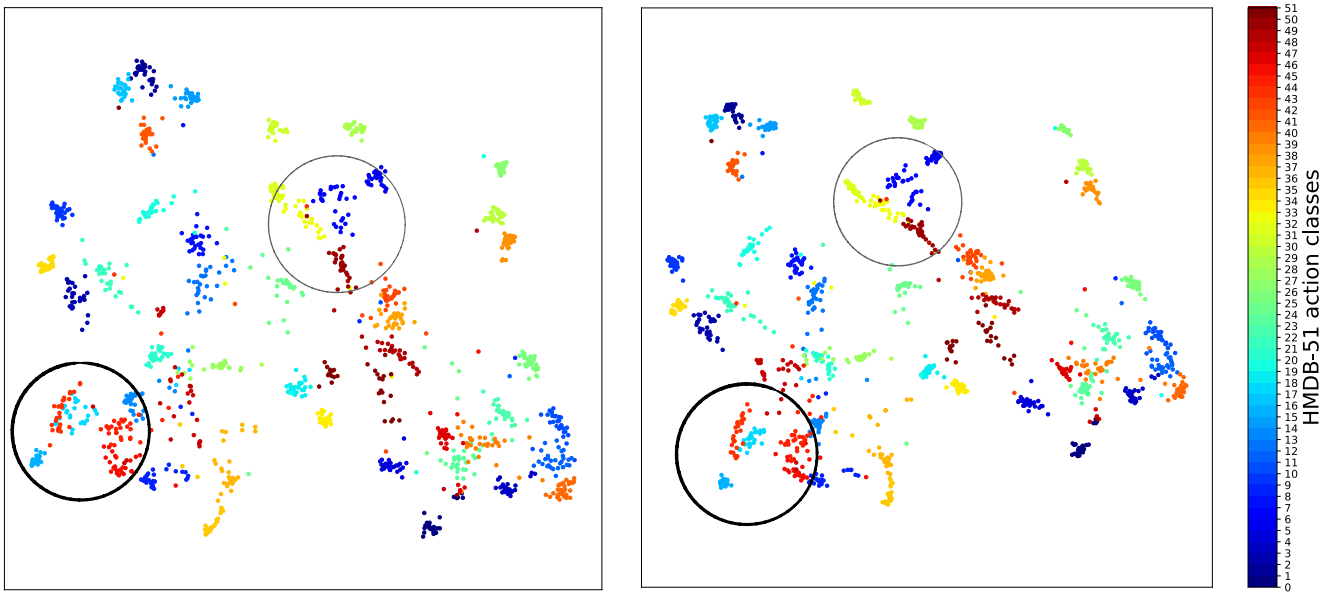


(a) **(b)**

Fig. 9: Visualization of the feature space (extracted from PredNet) for DEEP-HAL in Fig. 9a and DEEP-HAL+ODF in Fig. 9b on the YUP++ dataset. For comparison, regions with notable differences are circled to highlight significant changes.

Health Informatics (BHI), pp. 1–5 (2019). DOI 10.1109/BHI.2019.8834505 5

4. Alwassel, H., Mahajan, D., Korbar, B., Torresani, L., Ghanem, B., Tran, D.: Self-supervised learning by cross-modal audio-video clustering. In: Advances in Neural Information Processing Systems (NeurIPS) (2020) 5
5. Arnab, A., Dehghani, M., Heigold, G., Sun, C., Lučić, M., Schmid, C.: Vivit: A video vision transformer. In: Proceedings of the IEEE/CVF International Conference on Computer Vision (ICCV), pp. 6836–6846 (2021) 2, 4
6. Aytar, Y., Vondrick, C., Torralba, A.: Soundnet: Learning sound representations from unlabeled video. In: Proceedings of the 30th International Conference on Neural Information Processing Systems, NIPS'16, p. 892–900. Curran Associates Inc., Red Hook, NY, USA (2016) 3, 5, 11
7. Baradel, F., Neverova, N., Wolf, C., Mille, J., Mori, G.: Object level visual reasoning in videos. In: ECCV, pp. 1–16. Springer Science+Business Media, Munich, Germany (2018) 13
8. Borji, A., Cheng, M.M., Jiang, H., Li, J.: Salient object detection: A benchmark. TIP **24**(12), 5706–5722 (2015). DOI 10.1109/TIP.2015.2487833 5
9. Braux-Zin, J., Dupont, R., Bartoli, A.: A general dense image matching framework combining direct and feature-based costs. In: ICCV, pp. 185–192. IEEE, Sydney, NSW, Australia (2013) 4
10. Brox, T., Malik, J.: Large displacement optical flow: Descriptor matching in variational motion estimation. TPAMI **33**(3), 500–



(a) (b)

Fig. 10: Visualization of the feature space from PredNet for DEEP-HAL (Fig. 10a) and DEEP-HAL+ODF (Fig. 10b) on the HMDB-51 dataset. Regions with notable differences are highlighted for comparison.

513 (2011). DOI 10.1109/TPAMI.2010.143. URL http://dx.doi.org/10.1109/TPAMI.2010.143 4, 10

11. Bulat, A., Perez-Rua, J.M., Sudhakaran, S., Martinez, B., Tzimiropoulos, G.: Space-time mixing attention for video transformer. In: A. Beygelzimer, Y. Dauphin, P. Liang, J.W. Vaughan (eds.) Advances in Neural Information Processing Systems (2021). URL https://openreview.net/forum?id=QgX15Mdi1E_ 2, 4
12. Burda, Y., Grosse, R.B., Salakhutdinov, R.: Importance weighted autoencoders. In: Y. Bengio, Y. LeCun (eds.) 4th International Conference on Learning Representations, ICLR 2016, San Juan, Puerto Rico, May 2-4, 2016, Conference Track Proceedings (2016). URL http://arxiv.org/abs/1509.00519 11
13. Cao, Z., Simon, T., Wei, S.E., Sheikh, Y.: Realtime multi-person 2d pose estimation using part affinity fields. In: The IEEE Conference on Computer Vision and Pattern Recognition (CVPR) (2017) 4
14. Carreira, J., Noland, E., Banki-Horvath, A., Hillier, C., Zisserman, A.: A short note about kinetics-600. arXiv preprint arXiv:1808.01340 (2018) 13
15. Carreira, J., Zisserman, A.: Quo Vadis, Action Recognition? A New Model and the Kinetics Dataset. In: CVPR, pp. 1–10. IEEE, Honolulu, HI, USA (2018) 1, 2, 4, 7, 16
16. Chakraborty, B., Holte, M.B., Moeslund, T.B., Gonzàlez, J.: Selective spatio-temporal interest points. CVIU **116**(3), 396–410 (2012) 3
17. Chen, C., Fu, Z., Chen, Z., Jin, S., Cheng, Z., Jin, X., Hua, X.S.: Homm: Higher-order moment matching for unsupervised domain adaptation. In: Proceedings of the AAAI conference on artificial intelligence, pp. 3422–3429 (2020) 9
18. Chen, Q., Wang, L., Koniusz, P., Gedeon, T.: Motion meets attention: Video motion prompts. In: The 16th Asian Conference on Machine Learning (Conference Track) (2024) 1, 2
19. Chen, Z., Li, S., Yang, B., Li, Q., Liu, H.: Multi-scale spatial temporal graph convolutional network for skeleton-based action recognition. Proceedings of the AAAI Conference on Artificial Intelligence **35**(2), 1113–1122 (2021). URL https://ojs.aaai.org/index.php/AAAI/article/view/16197 4
20. Cheng, K., Zhang, Y., He, X., Chen, W., Cheng, J., Lu, H.: Skeleton-based action recognition with shift graph convolutional network. In: Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR) (2020) 4
21. Cheng, K., Zhang, Y., He, X., Chen, W., Cheng, J., Lu, H.: Skeleton-based action recognition with shift graph convolutional network. In: 2020 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR), pp. 180–189 (2020). DOI 10.1109/CVPR42600.2020.00026 4
22. Cherian, A., Fernando, B., Harandi, M., Gould, S.: Generalized rank pooling for action recognition. In: CVPR, pp. 3222–3231. IEEE, Honolulu, HI, USA (2017) 13, 17
23. Cherian, A., Koniusz, P., Gould, S.: Higher-order pooling of CNN features via kernel linearization for action recognition. In: WACV, pp. 130–138. IEEE, Santa Rosa, CA, USA (2017). DOI 10.1109/WACV.2017.22 2, 4
24. Cherian, A., Sra, S., Gould, S., Hartley, R.: Non-linear temporal subspace representations for activity recognition. In: CVPR, pp. 2197–2206. IEEE, Salt Lake City, UT, USA (2018). DOI 10.1109/CVPR.2018.00234 2, 4, 13, 17
25. Choi, J., Chun, D., Kim, H., Lee, H.J.: Gaussian yolov3: An accurate and fast object detector using localization uncertainty for autonomous driving. In: The IEEE International Conference on Computer Vision (ICCV) (2019) 5
26. Chollet, F.: Xception: Deep learning with depthwise separable convolutions. In: Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR) (2017) 4
27. Choutas, V., Weinzaepfel, P., Revaud, J., Schmid, C.: PoTion: Pose motion representation for action recognition. In: CVPR, pp. 7024–7033. IEEE, Salt Lake City, UT, USA (2018) 2
28. Cormode, G., Hadjieleftheriou, M.: Finding frequent items in data streams. Proc. VLDB Endow. **1**(2), 1530–1541 (2008). DOI 10.14778/1454159.1454225. URL http://dx.doi.org/10.14778/1454159.1454225 7
29. Csurka, G., Dance, C.R., Fan, L., Willamowski, J., Bray, C.: Visual categorization with bags of keypoints. In: ECCV Workshop, pp. 1–22. Springer Science+Business Media, Prague, Czech Republic (2004) 1, 3, 6

30. Dalal, N., Triggs, B., Schmid, C.: Human Detection Using Oriented Histogram of Flow and Appearance. In: ECCV, pp. 428–441. Springer Science+Business Media, Graz, Austria (2006) 1, 3
31. Damen, D., Doughty, H., Farinella, G.M., Fidler, S., Furnari, A., Kazakos, E., Moltisanti, D., Munro, J., Perrett, T., Price, W., Wray, M.: Scaling egocentric vision: The epic-kitchens dataset. In: ECCV, pp. 1–17. Springer Science+Business Media, Munich, Germany (2018) 13, 18
32. Das, S., Dai, R., Koperski, M., Minciullo, L., Garattoni, L., Bremond, F., Francesca, G.: Toyota smarthome: Real-world activities of daily living. In: Proceedings of the IEEE/CVF International Conference on Computer Vision (ICCV) (2019) 13, 18
33. Das, S., Sharma, S., Dai, R., Bremond, F., Thonnat, M.: Vpn: Learning video-pose embedding for activities of daily living (2020) 17, 18
34. Deng, J., Dong, W., Socher, R., Li, L.J., Li, K., Fei-Fei, L.: Imagenet: A large-scale hierarchical image database. In: 2009 IEEE Conference on Computer Vision and Pattern Recognition, pp. 248–255 (2009). DOI 10.1109/CVPR.2009.5206848 4, 7
35. Ding, D., Wang, L., Zhu, L., Gedeon, T., Koniusz, P.: Learnable expansion of graph operators for multi-modal feature fusion. In: The Thirteenth International Conference on Learning Representations (2025). URL https://openreview.net/forum?id=SMZqIOSdlN 2
36. Ding, X., Wang, L.: Do language models understand time? WWW '25 Companion. Association for Computing Machinery, New York, NY, USA (2025). DOI 10.1145/3701716.3717744. URL https://doi.org/10.1145/3701716.3717744 2
37. Ding, X., Wang, L.: The journey of action recognition. In: Companion Proceedings of the ACM Web Conference 2025, WWW '25 Companion. Association for Computing Machinery, New York, NY, USA (2025). DOI 10.1145/3701716.3717746. URL https://doi.org/10.1145/3701716.3717746 2
38. Dollár, P., Rabaud, V., Cottrell, G., Belongie, S.: Behavior recognition via sparse spatio-temporal features. In: ICCCN, pp. 65–72. IEEE, Honolulu, HI, USA (2005). URL http://dl.acm.org/citation.cfm?id=1259587.1259830 3
39. Donahue, J., Hendricks, L.A., Guadarrama, S., Rohrbach, M., Venugopalan, S., Darrell, T., Saenko, K.: Long-term recurrent convolutional networks for visual recognition and description. In: CVPR, pp. 2625–2634. IEEE, Boston, MA, USA (2015) 4
40. Dorta, G., Vicente, S., Agapito, L., Campbell, N.D.F., Simpson, I.: Structured uncertainty prediction networks. In: 2018 IEEE/CVF Conference on Computer Vision and Pattern Recognition, pp. 5477–5485 (2018). DOI 10.1109/CVPR.2018.00574 5
41. Dosovitskiy, A., Beyer, L., Kolesnikov, A., Weissenborn, D., Zhai, X., Unterthiner, T., Dehghani, M., Minderer, M., Heigold, G., Gelly, S., Uszkoreit, J., Houlsby, N.: An image is worth 16x16 words: Transformers for image recognition at scale. In: International Conference on Learning Representations (2021). URL https://openreview.net/forum?id=YicbFdNTTy 2, 4
42. Fang, P., Zhou, J., Kumar Roy, S., Petersson, L., Harandi, M.: Bilinear attention networks for person retrieval. In: ICCV, pp. 8030–8039. IEEE, Seoul, Korea (2019) 4
43. Feichtenhofer, C., Fan, H., Malik, J., He, K.: Slowfast networks for video recognition. In: ICCV, pp. 6202–6211. IEEE, Seoul, Korea (2019) 16, 17
44. Feichtenhofer, C., Pinz, A., Wildes, R.P.: Spatiotemporal residual networks for video action recognition. In: NIPS, pp. 3468–3476. MIT Press, Barcelona, Spain (2016) 1, 4
45. Feichtenhofer, C., Pinz, A., Wildes, R.P.: Temporal residual networks for dynamic scene recognition. In: CVPR, pp. 4728–4737. IEEE, Honolulu, HI, USA (2017) 13, 17
46. Fernando, B., Gavves, E., M., J.O., Ghodrati, A., Tuytelaars, T.: Modeling video evolution for action recognition. In: CVPR, pp. 5378–5387. IEEE, Boston, MA, USA (2015) 4
47. Fernando, B., Gould, S.: Learning end-to-end video classification with rank-pooling. In: ICML, vol. 48, pp. 1187–1196. ACM, New York City, NY, USA (2016) 2, 4
48. Freeman, W.T., Roth, M.: Orientation histograms for hand gesture recognition. Tech. Rep. TR94-03, MERL - Mitsubishi Electric Research Laboratories, Cambridge, MA 02139 (1994). URL http://www.merl.com/publications/TR94-03/ 1, 3
49. Gao, R., Oh, T.H., Grauman, K., Torresani, L.: Listen to look: Action recognition by previewing audio. In: IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR) (2020) 5
50. van Gemert, J.C., Veenman, C.J., Smeulders, A.W.M., Geusebroek, J.M.: Visual word ambiguity. TPAMI **32**(7), 1271–1283 (2010). DOI 10.1109/TPAMI.2009.132. URL http://dx.doi.org/10.1109/TPAMI.2009.132 3
51. Ghadiyaram, D., Tran, D., Mahajan, D.: Large-scale weakly-supervised pre-training for video action recognition. In: CVPR, pp. 12046–12055. IEEE, Long Beach,California,USA (2019) 18
52. Girdhar, R., El-Nouby, A., Singh, M., Alwala, K.V., Joulin, A., Misra, I.: Omnimae: Single model masked pretraining on images and videos. In: Proceedings of the IEEE/CVF conference on computer vision and pattern recognition, pp. 10406–10417 (2023) 19
53. Girshick, R.: Fast r-cnn. In: ICCV, pp. 1440–1448. IEEE, Santiago, Chile (2015) 4
54. Girshick, R., Donahue, J., Darrell, T., Malik, J.: Region-based convolutional networks for accurate object detection and segmentation. TPAMI **38**(1), 142–158 (2016) 4
55. Goyal, R., Ebrahimi Kahou, S., Michalski, V., Materzynska, J., Westphal, S., Kim, H., Haenel, V., Fruend, I., Yianilos, P., Mueller-Freitag, M., et al.: The" something something" video database for learning and evaluating visual common sense. In: Proceedings of the IEEE international conference on computer vision, pp. 5842–5850 (2017) 14
56. Gu, C., Sun, C., Ross, D.A., Vondrick, C., Pantofaru, C., Li, Y., Vijayanarasimhan, S., Toderici, G., Ricco, S., Sukthankar, R., Schmid, C., Malik, J.: Ava: A video dataset of spatio-temporally localized atomic visual actions. In: CVPR, pp. 6047–6056. IEEE, Salt Lake City, UT, USA (2018) 5
57. Hadji, I., Wildes, R.P.: A new large scale dynamic texture dataset with application to ConvNet understanding. In: ECCV. Springer Science+Business Media, Munich, Germany (2018) 17
58. He, K., Gkioxari, G., Dollár, P., Girshick, R.B.: Mask r-cnn. In: ICCV, pp. 2980–2988. IEEE, Venice, Italy (2017) 4
59. He, K., Zhang, X., Ren, S., Sun, J.: Deep residual learning for image recognition. In: CVPR, pp. 1–12. IEEE, Las Vegas, NV, USA (2016) 5
60. He, K., Zhang, X., Ren, S., Sun, J.: Deep residual learning for image recognition. In: 2016 IEEE Conference on Computer Vision and Pattern Recognition (CVPR), pp. 770–778 (2016). DOI 10.1109/CVPR.2016.90 8
61. He, Y., Zhu, C., Wang, J., Savvides, M., Zhang, X.: Bounding box regression with uncertainty for accurate object detection. In: The IEEE Conference on Computer Vision and Pattern Recognition (CVPR) (2019) 5
62. Horn, B.K.P., Schunck, B.G.: Determining optical flow. Artificial Intelligence **17**, 185–203 (1981) 4
63. Hou, Q., Cheng, M.M., Hu, X., Borji, A., Tu, Z., Torr, P.H.S.: Deeply supervised salient object detection with short connections. In: Proc. IEEE Conf. Comp. Vis. Patt. Recogn., pp. 3203–3212. IEEE, Honolulu, HI, USA (2017) 5

64. Howard, A., Sandler, M., Chen, B., Wang, W., Chen, L., Tan, M., Chu, G., Vasudevan, V., Zhu, Y., Pang, R., Adam, H., Le, Q.: Searching for mobilenetv3. In: 2019 IEEE/CVF International Conference on Computer Vision (ICCV), pp. 1314–1324. IEEE Computer Society, Los Alamitos, CA, USA (2019). DOI 10.1109/ICCV.2019.00140. URL https://doi.ieeecomputersociety.org/10.1109/ICCV.2019.00140 4
65. Huang, L., Huang, Y., Ouyang, W., Wang, L.: Part-level graph convolutional network for skeleton-based action recognition. Proceedings of the AAAI Conference on Artificial Intelligence **34**(07), 11045–11052 (2020). DOI 10.1609/aaai.v34i07.6759. URL https://ojs.aaai.org/index.php/AAAI/article/view/6759 4
66. Hüllermeier, E., Waegeman, W.: Aleatoric and epistemic uncertainty in machine learning: an introduction to concepts and methods. Mach. Learn. **110**(3), 457–506 (2021). DOI 10.1007/s10994-021-05946-3. URL https://doi.org/10.1007/s10994-021-05946-3 5
67. Huo, Z., Pakbin, A., Chen, X., Hurley, N.C., Yuan, Y., Qian, X., Wang, Z., Huang, S., Mortazavi, B.: Uncertainty quantification for deep context-aware mobile activity recognition and unknown context discovery. In: AISTATS, pp. 3894–3904 (2020). URL http://proceedings.mlr.press/v108/huo20a.html 5
68. Iandola, F.N., Moskewicz, M.W., Ashraf, K., Han, S., Dally, W.J., Keutzer, K.: Squeezenet: Alexnet-level accuracy with 50x fewer parameters and <1mb model size. CoRR **abs/1602.07360** (2016). URL http://arxiv.org/abs/1602.07360 4
69. Indrayan, A.: Medical biostatistics, 2nd ed. edn. Chapman & Hall/CRC,, Boca Raton : (c2008.). URL http://www.loc.gov/catdir/toc/ecip0723/2007030353.html 5
70. Jebara, T., Kondor, R., Howard, A.: Probability product kernels. JMLR **5**, 819–844 (2004) 7
71. Jégou, H., Douze, M., Schmid, C.: On the Burstiness of Visual Elements. In: CVPR, pp. 1169–1176. IEEE, Long Beach,alifornia, USA (2009) 5
72. Ji, S., Xu, W., Yang, M., Yu, K.: 3D convolutional neural networks for human action recognition. TPAMI **35**, 221–231 (2010) 4
73. Karpathy, A., Toderici, G., Shetty, S., Leung, T., Sukthankar, R., Fei-Fei, L.: Large-scale video classification with convolutional neural networks. In: CVPR, pp. 1725–1732. IEEE, Columbus, OH, USA (2014). DOI 10.1109/CVPR.2014.223. URL https://doi.org/10.1109/CVPR.2014.223 4
74. Kay, W., Carreira, J., Simonyan, K., Zhang, B., Hillier, C., Vijayanarasimhan, S., Viola, F., Green, T., Back, T., Natsev, P., et al.: The kinetics human action video dataset. arXiv preprint arXiv:1705.06950 (2017) 13
75. Kendall, A., Gal, Y.: What uncertainties do we need in bayesian deep learning for computer vision? In: I. Guyon, U.V. Luxburg, S. Bengio, H. Wallach, R. Fergus, S. Vishwanathan, R. Garnett (eds.) Advances in Neural Information Processing Systems, vol. 30. Curran Associates, Inc. (2017). URL https://proceedings.neurips.cc/paper/2017/file/2650d6089a6d640c5e85b2b88265dc2b-Paper.pdf 5
76. Kendall, A., Gal, Y., Cipolla, R.: Multi-task learning using uncertainty to weigh losses for scene geometry and semantics. In: Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR) (2018) 5
77. Kingma, D.P., Welling, M.: Auto-encoding variational bayes. In: Y. Bengio, Y. LeCun (eds.) 2nd International Conference on Learning Representations, ICLR 2014, Banff, AB, Canada, April 14-16, 2014, Conference Track Proceedings (2014). URL http://arxiv.org/abs/1312.6114 11
78. Kipf, T.N., Welling, M.: Semi-supervised classification with graph convolutional networks. In: International Conference on Learning Representations (ICLR) (2017) 4
79. Kiureghian, A.D., Ditlevsen, O.: Aleatory or epistemic? does it matter? Structural Safety **31**(2), 105–112 (2009). DOI https://doi.org/10.1016/j.strusafe.2008.06.020. URL https://www.sciencedirect.com/science/article/pii/S0167473008000556. Risk Acceptance and Risk Communication 5
80. Kläser, A., Marszalek, M., Schmid, C.: A Spatio-Temporal Descriptor Based on 3D-Gradients. In: BMCV, pp. 1–10. BMVA, Leeds, UK (2008) 1, 3
81. Kondratyuk, D., Yuan, L., Li, Y., Zhang, L., Tan, M., Brown, M., Gong, B.: Movinets: Mobile video networks for efficient video recognition. In: Proceedings of the IEEE/CVF conference on computer vision and pattern recognition, pp. 16020–16030 (2021) 4, 17
82. Koniusz, P., Cherian, A., Porikli, F.: Tensor representations via kernel linearization for action recognition from 3D skeletons. In: ECCV, pp. 1–14. Springer Science+Business Media, Amsterdam, The Netherlands (2016) 4, 9
83. Koniusz, P., Mikolajczyk, K.: Soft Assignment of Visual Words as Linear Coordinate Coding and Optimisation of its Reconstruction Error. In: ICIP, pp. 2461–2464. IEEE, Brussels, Belgium (2011) 3
84. Koniusz, P., Wang, L., Cherian, A.: Tensor representations for action recognition. In: IEEE Transactions on Pattern Analysis and Machine Intelligence. IEEE (2020) 4, 9
85. Koniusz, P., Yan, F., Gosselin, P.H., Mikolajczyk, K.: Higher-order Occurrence Pooling on Mid- and Low-level Features: Visual Concept Detection. Technical Report **1**(1), 1–20 (2013) 5
86. Koniusz, P., Yan, F., Gosselin, P.H., Mikolajczyk, K.: Higher-order occurrence pooling for bags-of-words: Visual concept detection. TPAMI **39**(2), 313–326 (2017) 4, 5
87. Koniusz, P., Yan, F., Mikolajczyk, K.: Comparison of Mid-Level Feature Coding Approaches And Pooling Strategies in Visual Concept Detection. CVIU **117**, 479–492 (2012). DOI 10.1016/j.cviu.2012.10.010 3, 5, 6
88. Koniusz, P., Zhang, H.: Power normalizations in fine-grained image, few-shot image and graph classification. In: IEEE Transactions on Pattern Analysis and Machine Intelligence. IEEE (2020) 9
89. Koniusz, P., Zhang, H., Porikli, F.: A deeper look at power normalizations. In: CVPR, pp. 5774–5783. IEEE, Salt Lake City, UT, USA (2018) 5, 6
90. Korban, M., Li, X.: Ddgcn: A dynamic directed graph convolutional network for action recognition. In: A. Vedaldi, H. Bischof, T. Brox, J.M. Frahm (eds.) Computer Vision – ECCV 2020, pp. 761–776. Springer International Publishing, Cham (2020) 4
91. Kozlov, A., Andronov, V., Gritsenko, Y.: Lightweight Network Architecture for Real-Time Action Recognition, p. 2074–2080. Association for Computing Machinery, New York, NY, USA (2020). URL https://doi.org/10.1145/3341105.3373906 2, 4, 8
92. Kuehne, H., Jhuang, H., Garrote, E., Poggio, T., Serre, T.: HMDB: A large video database for human motion recognition. In: ICCV, pp. 2556–2563. IEEE, Barcelona, Spain (2011) 13
93. Kumar, D., Kumar, C., Seah, C., Xia, S., Shao, M.: Finding achilles' heel: Adversarial attack on multi-modal action recognition. In: C.W. Chen, R. Cucchiara, X. Hua, G. Qi, E. Ricci, Z. Zhang, R. Zimmermann (eds.) MM, pp. 3829–3837. ACM, Seattle, United States (2020). DOI 10.1145/3394171.3413531. URL https://doi.org/10.1145/3394171.3413531 4
94. Laptev, I.: On space-time interest points. IJCV **64**(2-3), 107–123 (2005). DOI 10.1007/s11263-005-1838-7. URL http:

//dx.doi.org/10.1007/s11263-005-1838-7 3
95. Li, C., Su, B., Wang, J., Zhang, Q.: Human action recognition using multi-velocity STIPs and motion energy orientation histogram. J. Inf. Sci. Eng. **30**, 295–312 (2014) 3
96. Li, J., Wei, P., Zhang, Y., Zheng, N.: A slow-i-fast-p architecture for compressed video action recognition. In: MM, pp. 2039–2047. ACM, Seattle, United States (2020) 4
97. Li, M., Chen, S., Chen, X., Zhang, Y., Wang, Y., Tian, Q.: Actional-structural graph convolutional networks for skeleton-based action recognition. In: The IEEE Conference on Computer Vision and Pattern Recognition (CVPR) (2019) 4
98. Lin, T.Y., Maire, M., Belongie, S., Hays, J., Perona, P., Ramanan, D., Dollár, P., Zitnick, C.L.: Microsoft coco: Common objects in context. In: D. Fleet, T. Pajdla, B. Schiele, T. Tuytelaars (eds.) Computer Vision – ECCV 2014, pp. 740–755. Springer International Publishing, Cham (2014) 5
99. Lingqiao, L., Wang, L., Liu, X.: In Defence of Soft-assignment Coding. In: ICCV, pp. 2486–2493. IEEE, Barcelona, Spain (2011) 3
100. Liu, Z., Gao, G., Qin, A.K., Wu, T., Liu, C.H.: Action recognition with bootstrapping based long-range temporal context attention. In: L. Amsaleg, B. Huet, M.A. Larson, G. Gravier, H. Hung, C. Ngo, W.T. Ooi (eds.) MM, pp. 583–591. ACM, Nice, France (2019). DOI 10.1145/3343031.3350916. URL https://doi.org/10.1145/3343031.3350916 4
101. Liu, Z., Ning, J., Cao, Y., Wei, Y., Zhang, Z., Lin, S., Hu, H.: Video swin transformer. arXiv preprint arXiv:2106.13230 (2021) 2, 4
102. Lu, C., Koniusz, P.: Few-shot keypoint detection with uncertainty learning for unseen species. CoRR **abs/2112.06183** (2021). URL https://arxiv.org/abs/2112.06183 5
103. Ma, N., Zhang, X., Zheng, H.T., Sun, J.: Shufflenet v2: Practical guidelines for efficient cnn architecture design. In: Proceedings of the European Conference on Computer Vision (ECCV) (2018) 4
104. Mairal, J., Koniusz, P., Harchaoui, Z., Schmid, C.: Convolutional kernel networks. In: NIPS, pp. 1–9. MIT Press, Montreal, Quebec, Canada (2014) 10
105. Matthies, H.G.: Quantifying uncertainty: Modern computational representation of probability and applications. In: A. Ibrahimbegovic, I. Kozar (eds.) Extreme Man-Made and Natural Hazards in Dynamics of Structures, pp. 105–135. Springer Netherlands, Dordrecht (2007) 5
106. McInnes, L., Healy, J., Saul, N., Grossberger, L.: Umap: Uniform manifold approximation and projection. The Journal of Open Source Software **3**(29), 861 (2018) 22
107. Neimark, D., Bar, O., Zohar, M., Asselmann, D.: Video transformer network. In: 2021 IEEE/CVF International Conference on Computer Vision Workshops (ICCVW), pp. 3156–3165 (2021). DOI 10.1109/ICCVW54120.2021.00355 2, 4
108. Owens, A., Efros, A.A.: Audio-visual scene analysis with self-supervised multisensory features. arXiv preprint arXiv:1804.03641 (2018) 5
109. Paoletti, G., Cavazza, J., Beyan, C., Del Bue, A.: Unsupervised Human Action Recognition with Skeletal Graph Laplacian and Self-Supervised Viewpoints Invariance. In: The 32nd British Machine Vision Conference (BMVC) (2021) 4
110. Papenberg, N., Bruhn, A., Brox, T., Didas, S., Weickert, J.: Highly accurate optic flow computation with theoretically justified warping. IJCV **67**, 141–158 (2006) 4
111. Patrick, M., Campbell, D., Asano, Y., Misra, I., Metze, F., Feichtenhofer, C., Vedaldi, A., Henriques, J.F.: Keeping your eye on the ball: Trajectory attention in video transformers. Advances in neural information processing systems **34**, 12493–12506 (2021) 2, 4, 19
112. Perronnin, F., Dance, C.: Fisher kernels on visual vocabularies for image categorization. In: CVPR, vol. 0, pp. 1–8. IEEE, Minneapolis, Minnesota, USA (2007) 1, 3, 6
113. Perronnin, F., Sánchez, J., Mensink, T.: Improving the Fisher Kernel for Large-Scale Image Classification. In: ECCV, pp. 143–156. Springer Science+Business Media, Heraklion, Crete (2010) 1, 3, 6
114. Pham, N., Pagh, R.: Fast and scalable polynomial kernels via explicit feature maps. In: ACM SIGKDD, pp. 239–247. ACM, Chicago, USA (2013). DOI 10.1145/2487575.2487591. URL http://doi.acm.org/10.1145/2487575.2487591 7
115. Piergiovanni, A., Angelova, A., Toshev, A., Ryoo, M.S.: Evolving space-time neural architectures for videos. In: ICCV, pp. 1793–1802. IEEE, Seoul, Korea (2019) 16
116. Piergiovanni, A., Kuo, W., Angelova, A.: Rethinking video vits: Sparse video tubes for joint image and video learning. In: Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition, pp. 2214–2224 (2023) 2, 4
117. Piergiovanni, A., Ryoo, M.S.: Recognizing actions in videos from unseen viewpoints. In: Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR), pp. 4124–4132 (2021) 18
118. Qin, Z., Liu, Y., Ji, P., Kim, D., Wang, L., McKay, R.I., Anwar, S., Gedeon, T.: Fusing higher-order features in graph neural networks for skeleton-based action recognition. IEEE Transactions on Neural Networks and Learning Systems **35**(4), 4783–4797 (2024). DOI 10.1109/TNNLS.2022.3201518 4
119. Redmon, J., Divvala, S., Girshick, R., Farhadi, A.: You only look once: Unified, real-time object detection. In: CVPR, pp. 779–788. IEEE, Boston, MA, USA (2015) 4
120. Ren, S., He, K., Girshick, R., Sun, J.: Faster R-CNN: Towards real-time object detection with region proposal networks. In: NIPS, pp. 91–99. MIT Press, Montreal, Canada (2015) 3, 4, 5, 13
121. Revaud, J., Weinzaepfel, P., Harchaoui, Z., Schmid, C.: EpicFlow: Edge-Preserving Interpolation of Correspondences for Optical Flow. In: CVPR, pp. 1164–1172. IEEE, Boston, MA, USA (2015) 4
122. Rohrbach, M., Amin, S., Andriluka, M., Schiele, B.: A database for fine grained activity detection of cooking activities. In: CVPR, pp. 1194–1201. IEEE, Providence, Rhode Island (2012) 13
123. Russakovsky, O., Deng, J., Su, H., Krause, J., Satheesh, S., Ma, S., Huang, Z., Karpathy, A., Khosla, A., Bernstein, M., Berg, A.C., Fei-Fei, L.: ImageNet large scale visual recognition challenge. IJCV **115**(3), 211–252 (2015). DOI 10.1007/s11263-015-0816-y 5
124. Ryoo, M.S., Piergiovanni, A., Kangaspunta, J., Angelova, A.: Assemblenet++: Assembling modality representations via attention connections. In: ECCV, pp. 1–19. Springer Science+Business Media, Glasgow, UK (2020) 2, 4, 8, 17
125. Ryoo, M.S., Piergiovanni, A., Tan, M., Angelova, A.: Assemblenet: Searching for multi-stream neural connectivity in video architectures. In: ICLR, pp. 1–15. ICLR, Addis Ababa, Ethiopia (2020) 2, 4, 8, 16, 17
126. Sandler, M., Howard, A., Zhu, M., Zhmoginov, A., Chen, L.C.: Mobilenetv2: Inverted residuals and linear bottlenecks. In: The IEEE Conference on Computer Vision and Pattern Recognition (CVPR) (2018) 8
127. Scovanner, P., Ali, S., Shah, M.: A 3-Dimentional SIFT Descriptor and its Application to Action Recognition. In: MM, pp. 357–356. ACM, Augsburg, Germany (2007) 1, 3
128. Seo, Y.M., Choi, Y.S.: Graph Convolutional Networks for Skeleton-Based Action Recognition with LSTM Using Tool-Information, p. 986–993. Association for Computing Machinery,

New York, NY, USA (2021). URL https://doi.org/10.1145/3412841.3441974 4
129. Shi, L., Zhang, Y., Cheng, J., Lu, H.: Two-stream adaptive graph convolutional networks for skeleton-based action recognition. In: 2019 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR), pp. 12018–12027 (2019). DOI 10.1109/CVPR.2019.01230 4
130. Shi, L., Zhang, Y., Cheng, J., Lu, H.: Two-stream adaptive graph convolutional networks for skeleton-based action recognition. In: CVPR (2019) 4
131. Shotton, J., Fitzgibbon, A., Cook, M., Sharp, T., Finocchio, M., Moore, R., Kipman, A., Blake, A.: Real-Time Human Pose Recognition in Parts from Single Depth Images. In: CVPR, pp. 1297–1304 (2011) 4
132. Sigurdsson, G.A., Varol, G., Wang, X., Farhadi, A., Laptev, I., Gupta, A.: Hollywood in homes: Crowdsourcing data collection for activity understanding. In: ECCV, pp. 1–17. Springer Science+Business Media, Amsterdam, The Netherlands (2016) 13
133. Simonyan, K., Zisserman, A.: Two-stream convolutional networks for action recognition in videos. In: NIPS, pp. 568–576. MIT Press, Montreal, Quebec, Canada (2014) 1, 4
134. Sivic, J., Zisserman, A.: Video Google: A text retrieval approach to object matching in videos. ICCV **2**, 1470–1477 (2003) 1, 3, 6
135. Srivastava, S., Sharma, G.: Omnivec2 - a novel transformer based network for large scale multimodal and multitask learning. In: 2024 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR), pp. 27402–27414. IEEE Computer Society, Los Alamitos, CA, USA (2024). DOI 10.1109/CVPR52733.2024.02588. URL https://doi.ieeecomputersociety.org/10.1109/CVPR52733.2024.02588 2, 4, 5
136. Stork, J.A., Spinello, L., Silva, J., Arras, K.O.: Audio-based human activity recognition using non-markovian ensemble voting. In: 2012 IEEE RO-MAN: The 21st IEEE International Symposium on Robot and Human Interactive Communication, pp. 509–514 (2012). DOI 10.1109/ROMAN.2012.6343802 5
137. Subedar, M., Krishnan, R., Meyer, P.L., Tickoo, O., Huang, J.: Uncertainty-aware audiovisual activity recognition using deep bayesian variational inference. In: Proceedings of the IEEE/CVF International Conference on Computer Vision (ICCV) (2019) 5
138. Szegedy, C., Ioffe, S., Vanhoucke, V., Alemi, A.A.: Inception-v4, inception-resnet and the impact of residual connections on learning. In: AAAI, pp. 4278–4284. AAAI Press, San Francisco,CA,USA (2017). URL http://dl.acm.org/citation.cfm?id=3298023.3298188 5
139. Szegedy, C., Vanhoucke, V., Ioffe, S., Shlens, J., Wojna, Z.: Rethinking the inception architecture for computer vision. In: CVPR, pp. 2818–2826. IEEE, Las Vegas, NV, USA (2016) 5
140. Tan, M., Le, Q.: EfficientNet: Rethinking model scaling for convolutional neural networks. In: K. Chaudhuri, R. Salakhutdinov (eds.) Proceedings of the 36th International Conference on Machine Learning, *Proceedings of Machine Learning Research*, vol. 97, pp. 6105–6114. PMLR (2019). URL https://proceedings.mlr.press/v97/tan19a.html 4
141. Tang, Y., Ma, L., Zhou, L.: Hallucinating optical flow features for video classification. In: IJCAI, pp. 926–932. IJCAI, Macao, China (2019) 2
142. Tomar, S.: Converting video formats with ffmpeg. Linux Journal **2006**(146), 10 (2006) 11
143. Tong, Z., Song, Y., Wang, J., Wang, L.: Videomae: Masked autoencoders are data-efficient learners for self-supervised video pre-training. Advances in neural information processing systems **35**, 10078–10093 (2022) 2, 4, 8
144. Tran, D., Bourdev, L., Fergus, R., Torresani, L., Paluri, M.: Learning Spatiotemporal Features with 3D Convolutional Networks. In: ICCV, pp. 4489–4497. IEEE, Santiago, Chile (2015) 1, 4
145. Tran, D., Wang, H., Torresani, L., Ray, J., LeCun, Y., Paluri, M.: A closer look at spatiotemporal convolutions for action recognition. In: Proceedings of the IEEE conference on Computer Vision and Pattern Recognition, pp. 6450–6459 (2018) 8
146. Uijlings, J.R., Duta, I.C., Rostamzadeh, N., Sebe, N.: Realtime Video Classification using Dense HOF/HOG. In: ICMR, pp. 145–152. ACM, New York, NY, USA (2014) 3
147. Varol, G., Laptev, I., Schmid, C.: Long-term temporal convolutions for action recognition. TPAMI **40**(6), 1510–1517 (2018) 4
148. Wang, H., Kläser, A., Schmid, C., Cheng-Lin, L.: Action Recognition by Dense Trajectories. In: CVPR, pp. 3169–3176. IEEE, Colorado Springs, CO, USA (2011) 1, 3
149. Wang, H., Kläser, A., Schmid, C., Liu, C.L.: Dense Trajectories and Motion Boundary Descriptors for Action Recognition. IJCV **103**, 60–79 (2013) 1, 3
150. Wang, H., Schmid, C.: Action Recognition with Improved Trajectories. In: ICCV, pp. 3551–3558. IEEE, Sydney, Australia (2013) 1, 2, 3, 10
151. Wang, J., Cherian, A.: Learning discriminative video representations using adversarial perturbations. In: ECCV, pp. 716–733. Springer Science+Business Media, Munich, Germany (2018). DOI 10.1007/978-3-030-01225-0_42. URL https://doi.org/10.1007/978-3-030-01225-0_42 2, 4, 16, 17
152. Wang, L.: Analysis and evaluation of Kinect-based action recognition algorithms. Master's thesis, School of the Computer Science and Software Engineering, The University of Western Australia (2017) 1
153. Wang, L.: Robust human action modelling. Ph.D. thesis, The Australian National University (Australia) (2023) 1, 2, 4
154. Wang, L., Huang, B., Zhao, Z., Tong, Z., He, Y., Wang, Y., Wang, Y., Qiao, Y.: Videomae v2: Scaling video masked autoencoders with dual masking. In: Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition, pp. 14549–14560 (2023) 2, 4, 8, 19
155. Wang, L., Huynh, D.Q., Koniusz, P.: A comparative review of recent kinect-based action recognition algorithms. TIP **29**(1), 15–28 (2019). DOI 10.1109/TIP.2019.2925285 1
156. Wang, L., Huynh, D.Q., Mansour, M.R.: Loss switching fusion with similarity search for video classification. In: IEEE ICIP, pp. 974–978 (2019). DOI 10.1109/ICIP.2019.8803051 1
157. Wang, L., Koniusz, P.: Self-Supervising Action Recognition by Statistical Moment and Subspace Descriptors, p. 4324–4333. Association for Computing Machinery, New York, NY, USA (2021). URL https://doi.org/10.1145/3474085.3475572 2, 4
158. Wang, L., Koniusz, P.: Temporal-viewpoint transportation plan for skeletal few-shot action recognition. In: Proceedings of the Asian Conference on Computer Vision, pp. 4176–4193 (2022) 4
159. Wang, L., Koniusz, P.: Uncertainty-dtw for time series and sequences. In: European Conference on Computer Vision, pp. 176–195. Springer (2022) 4
160. Wang, L., Koniusz, P.: 3mformer: Multi-order multi-mode transformer for skeletal action recognition. In: Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition, pp. 5620–5631 (2023) 4
161. Wang, L., Koniusz, P.: Flow dynamics correction for action recognition. In: ICASSP 2024-2024 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP), pp. 3795–3799. IEEE (2024) 4
162. Wang, L., Koniusz, P., Huynh, D.Q.: Hallucinating IDT descriptors and I3D optical flow features for action recognition with cnns. In: ICCV, pp. 8697–8707. IEEE, Seoul, Korea (2019) 2, 3, 4, 10, 16, 17, 22

163. Wang, L., Liu, J., Koniusz, P.: 3d skeleton-based few-shot action recognition with jeanie is not so naïve. arXiv preprint arXiv:2112.12668 (2021) 4
164. Wang, L., Liu, J., Zheng, L., Gedeon, T., Koniusz, P.: Meet jeanie: a similarity measure for 3d skeleton sequences via temporal-viewpoint alignment. International Journal of Computer Vision pp. 1–32 (2024) 4
165. Wang, L., Sun, K., Koniusz, P.: High-order tensor pooling with attention for action recognition. In: ICASSP 2024-2024 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP), pp. 3885–3889. IEEE (2024) 2, 4
166. Wang, L., Wang, L., Lu, H., Zhang, P., Ruan, X.: Saliency detection with recurrent fully convolutional networks. In: ECCV, pp. 825–841. Springer Science+Business Media, Amsterdam, The Netherlands (2016). DOI 10.1007/978-3-319-46493-0_50 5
167. Wang, L., Yuan, X., Gedeon, T., Zheng, L.: Taylor videos for action recognition. In: Forty-first International Conference on Machine Learning (2024) 1
168. Wang, M., Xing, J., Liu, Y.: Actionclip: A new paradigm for video action recognition. CoRR **abs/2109.08472** (2021). URL https://arxiv.org/abs/2109.08472 17
169. Wang, W., Seraj, F., Havinga, P.J.M.: A sound-based crowd activity recognition with neural network based regression models. In: Proceedings of the 13th ACM International Conference on PErvasive Technologies Related to Assistive Environments, PETRA '20. Association for Computing Machinery, New York, NY, USA (2020). DOI 10.1145/3389189.3389196. URL https://doi.org/10.1145/3389189.3389196 5
170. Wang, Y., Li, K., Li, X., Yu, J., He, Y., Chen, G., Pei, B., Zheng, R., Xu, J., Wang, Z., et al.: Internvideo2: Scaling video foundation models for multimodal video understanding. ECCV (2024) 2, 4, 5, 8, 19
171. Weinberger, K., Dasgupta, A., Langford, J., Smola, A., Attenberg, J.: Feature hashing for large scale multitask learning. In: ICML, pp. 1113–1120. ACM, Montreal, Canada (2009). DOI 10.1145/1553374.1553516. URL http://doi.acm.org/10.1145/1553374.1553516 7
172. Weinzaepfel, P., Revaud, J., Harchaoui, Z., Schmid, C.: DeepFlow: Large displacement optical flow with deep matching. In: ICCV, pp. 1–8. IEEE, Sydney, NSW, Australia (2013). URL http://hal.inria.fr/hal-00873592 4
173. Willems, G., Tuytelaars, T., Gool, L.V.: An efficient dense and scale-invariant spatio-temporal interest point detector. In: ECCV, pp. 650–663. Springer Science+Business Media, Marseille, France (2008). DOI 10.1007/978-3-540-88688-4_48. URL https://doi.org/10.1007/978-3-540-88688-4_48 3
174. Wu, C.Y., Feichtenhofer, C., Fan, H., He, K., Krahenbuhl, P., Girshick, R.: Long-term feature banks for detailed video understanding. In: CVPR, pp. 284–293. IEEE, Long Beach, California, USA (2019) 17, 18
175. Wu, Q., Wang, Z., Deng, F., Feng, D.D.: Realistic human action recognition with audio context. In: 2010 International Conference on Digital Image Computing: Techniques and Applications, pp. 288–293 (2010). DOI 10.1109/DICTA.2010.57 5
176. Xie, S., Sun, C., Huang, J., Tu, Z., Murphy, K.: Rethinking spatiotemporal feature learning: Speed-accuracy trade-offs in video classification. In: Proceedings of the European Conference on Computer Vision (ECCV) (2018) 7, 19
177. Yan, A., Wang, Y., Li, Z., Qiao, Y.: PA3D: Pose-action 3D machine for video recognition. In: CVPR, pp. 7922–7931. IEEE, Long Beach, California, USA (2019) 16
178. Yan, S., Xiong, X., Arnab, A., Lu, Z., Zhang, M., Sun, C., Schmid, C.: Multiview transformers for video recognition. In: Proceedings of the IEEE/CVF conference on computer vision and pattern recognition, pp. 3333–3343 (2022) 2, 4
179. Yan, S., Xiong, Y., Lin, D.: Spatial Temporal Graph Convolutional Networks for Skeleton-Based Action Recognition. In: AAAI (2018) 3, 4
180. Yang, D., Wang, Y., Dantcheva, A., Garattoni, L., Francesca, G., Bremond, F.: Unik: A unified framework for real-world skeleton-based action recognition. BMVC (2021) 17, 18
181. Yao, H., Wu, W., Li, Z.: Side4video: Spatial-temporal side network for memory-efficient image-to-video transfer learning. arXiv preprint arXiv:2311.15769 (2023) 2, 4
182. Yeffet, L., Wolf, L.: Local trinary patterns for human action recognition. In: ICCV, pp. 492–497. IEEE, Seoul, Korea (2009) 3
183. Zhang, C., Zou, Y., Chen, G., Gan, L.: PAN: persistent appearance network with an efficient motion cue for fast action recognition. In: L. Amsaleg, B. Huet, M.A. Larson, G. Gravier, H. Hung, C. Ngo, W.T. Ooi (eds.) MM, pp. 500–509. ACM, Nice, France (2019). DOI 10.1145/3343031.3350876. URL https://doi.org/10.1145/3343031.3350876 4
184. Zhang, H., Zhang, J., Koniusz, P.: Few-shot learning via saliency-guided hallucination of samples. In: CVPR, pp. 2770–2779. IEEE, Long Beach California (2019) 1, 5
185. Zhang, J., Zhang, T., Dai, Y., Harandi, M., Hartley, R.: Deep unsupervised saliency detection: A multiple noisy labeling perspective. In: CVPR, pp. 1–10. IEEE, Salt Lake City,UT,USA (2018) 1, 5
186. Zhang, X., Xu, C., Tao, D.: Context aware graph convolution for skeleton-based action recognition. In: 2020 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR), pp. 14321–14330 (2020). DOI 10.1109/CVPR42600.2020.01434 4
187. Zhang, Y., Li, X., Liu, C., Shuai, B., Zhu, Y., Brattoli, B., Chen, H., Marsic, I., Tighe, J.: Vidtr: Video transformer without convolutions. In: Proceedings of the IEEE/CVF International Conference on Computer Vision (ICCV), pp. 13577–13587 (2021) 17
188. Zhu, L., Sevilla-Lara, L., Tran, D., Feiszli, M., Yang, Y., Wang, H.: FASTER recurrent networks for video classification. CoRR **abs/1906.04226** (2019). URL http://arxiv.org/abs/1906.04226 4, 8
189. Zhu, L., Wang, L., Raj, A., Gedeon, T., Chen, C.: Advancing video anomaly detection: A concise review and a new dataset. In: The Thirty-eight Conference on Neural Information Processing Systems Datasets and Benchmarks Track (2024) 1
190. Zhu, W., Liang, S., Wei, Y., Sun, J.: Saliency optimization from robust background detection. In: CVPR, pp. 2814–2821. IEEE, Columbus, OH, USA (2014). DOI 10.1109/CVPR.2014.360 5
191. Zoph, B., Vasudevan, V., Shlens, J., Le, Q.V.: Learning transferable architectures for scalable image recognition. In: CVPR, pp. 1–14. IEEE, Salt Lake City, UT, USA (2018) 5