Self-supervising Action Recognition by Statistical Moment and Subspace Descriptors

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

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

ABSTRACT

In this paper, we build on a concept of self-supervision by taking RGB frames as input to learn to predict both action concepts and auxiliary descriptors e.g., object descriptors. So-called hallucination streams are trained to predict auxiliary cues, simultaneously fed into classification layers, and then hallucinated at the testing stage to aid network. We design and hallucinate two descriptors, one leveraging four popular object detectors applied to training videos, and the other leveraging image- and video-level saliency detectors. The first descriptor encodes the detector- and ImageNet-wise class prediction scores, confidence scores, and spatial locations of bounding boxes and frame indexes to capture the spatio-temporal distribution of features per video. Another descriptor encodes spatio-angular gradient distributions of saliency maps and intensity patterns. Inspired by the characteristic function of the probability distribution, we capture four statistical moments on the above intermediate descriptors. As numbers of coefficients in the mean, covariance, coskewness and cokurtosis grow linearly, quadratically, cubically and quartically w.r.t. the dimension of feature vectors, we describe the covariance matrix by its leading \( n' \) eigenvectors (so-called subspace) and we capture skewness/kurtosis rather than costly coskewness/cokurtosis. We obtain state of the art on five popular datasets such as Charades and EPIC-Kitchens.

CCS CONCEPTS

- Computing methodologies → Activity recognition and understanding; Supervised learning by regression; Computer systems organization → Neural networks.

KEYWORDS

self-supervision; hallucination; subspaces; action recognition

ACM Reference Format:
Lei Wang and Piotr Koniusz*. 2021. Self-supervising Action Recognition by Statistical Moment and Subspace Descriptors. In Proceedings of the 29th ACM Int’l Conference on Multimedia (MM ’21), Oct. 20–24, 2021, Virtual Event, China. ACM, New York, NY, USA, 10 pages. https://doi.org/10.1145/3474085.3475572

*Corresponding author. The code will be at http://users.crcs.anu.edu.au/~koniusz/.

Permission to make digital or hard copies of all or part of this work for personal or classroom use is granted without fee provided that copies are not made or distributed for profit or commercial advantage and that copies bear this notice and the full citation on the first page. Copyrights for components of this work owned by others than ACM must be honored. Abstracting with credit is permitted. To copy otherwise, or republish, to post on servers or to redistribute to lists, requires prior specific permission and/or a fee. Request permissions from permissions@acm.org.

MM ’21, October 20–24, 2021, Virtual Event, China.
© 2021 Association for Computing Machinery.
ACM ISBN 978-1-4503-8651-7/21/10...
$15.00
https://doi.org/10.1145/3474085.3475572

1 INTRODUCTION

Action Recognition (AR) has progressed from hand-crafted video representations [13, 37, 67, 78–80, 82–84] to Convolutional Neural Networks (CNN) [5, 19, 69, 74]. The two-stream networks [69], 3D spatio-temporal features [74], spatio-temporal ResNet model [19] and the new Inflated 3D (3D) convolutions network pre-trained on Kinetics-400 [5]. Often, AR combine the RGB and optical flow inputs, and benefit from a late fusion (next to the classifier) with low-level representations such as Improved Dense Trajectory (IDT) descriptors [80] due to their highly complementary nature [8–10, 22, 81]. Recently, AssembleNet and AssembleNet++ [66], learnt with the Neural Architecture Search (NAS) have yielded superb results.

A recent AR pipeline [85], called DEEP-HAL, used IDT descriptors encoded with Bag-of-Words (BoW) [12, 70] and Fisher Vectors (FV) [56, 57] to learn them by so-called hallucination streams and generate at the testing stage to boost results beyond a naive fusion of modalities. DEEP-HAL and approach [73] have shown that even optical flow frames encoded by a network can be learnt by another network trained on RGB frames only, thus pointing at redundancy in training both RGB and optical flow network streams. DEEP-HAL [85] has attained the state of the art on several AR benchmarks by learning to hallucinate IDT-based BoW/FV and Optical Flow Features (OFF) from a single RGB-based 3D network stream.

DEEP-HAL opens up an exciting opportunity to investigate what other representations can co-regularize/self-supervise a backbone network for AR with the goal of learning to hallucinate costly representations at the training stage and simply leveraging outputs of hallucination streams at the testing time. We build on DEEP-HAL which already includes IDT-based BoW/FV and OFF streams. However, we investigate the self-supervisory ability of object/saliency
AssembleNet variant, we obtain the 2048 (iii) embedded confidence scores, (iv) embedded bounding box coordinates. Such detected objects together with their relevance to the task of action recognition. Figure 1a shows a few of the ODF descriptors, we concatenate together per bounding box per frame (i) the one-hot detection and (ii) ImageNet scores, (iii) embedded confidence scores, (iv) embedded bounding box coordinates, and (v) embedded normalized frame index. For all bounding boxes, we stack such features into a matrix. Inspired by the characteristic function of the probability density function, we extract the mean, leading eigenvectors of covariance, skewness and kurtosis.

Figure 2: We build on DEEP-HAL [85] which includes I3D RGB and Optical Flow networks (the latter net. is only used during training). For AssembleNet and AssembleNet++, the backbone encodes both RGB and the optical flow, which is synthesized on the fly from RGB frames. For the I3D variant, we remove the prediction and the last 1D conv. layers from I3D RGB and optical flow streams, we feed the feature representations \( \mathbf{X}_{\text{opt.}} \) into Bag-of-Words (BoW), Fisher Vector (FV), the Optical Flow Features (OFF) and the High Abstraction Features (HAF) streams (dashed black) followed by the Power Normalization (PN) and Sketching (SK) blocks. The OFF stream is supervised by \( \mathbf{X}_{\text{opt.}} \). For the AssembleNet variant, we obtain the 2048 feature representations \( \mathbf{X}_{\text{rgb}} \) and do not use the OFF stream/optical flow backbone. Moreover, we introduce \( \text{DET1,...,DET4, SAL1 and SAL2 streams corresponding to our detector-} \) and saliency-based descriptors (dashed blue). The resulting feature vectors \( \mathbf{\psi}_{(i)} \), where \( \cdot \) denotes the stream name e.g., \( \text{det1} \), \( \text{det2} \) etc., are reweighted by corresponding weights \( w_i \) (magenta lines) and aggregated (sum) by \( \oplus \). All \( \mathbf{\psi}_{(i)} \) are reweighted, aggregated (sum) and fed to Prediction Network (PredNet). By \( \mathbf{\psi} \), we indicate that the Mean Square Error (MSE) losses are used during training to supervise all streams outputting \( \mathbf{\psi}_{(i)} \) by the ground-truth \( \mathbf{\gamma} \). By \( \mathbf{\bar{X}} \), we indicate that the MSE losses are switched off for testing and \( \mathbf{\bar{\psi}}_{(i)} \) are hallucinated/fed into PredNet to obtain labels \( \mathbf{\gamma} \).

In this paper, we design and hallucinate two kinds of descriptors, namely Object Detection Features (ODF) and Saliency Detection Features (SDF). The ODF descriptor leverages faster R-CNN detector [61] based on backbones such as (i) Inception V2 [72], (ii) Inception ResNet V2 [71], (iii) ResNet101 [30] and (iv) NASNet [97]. The Inception V2, Inception ResNet V2 and NASNet are pre-trained on the COCO dataset [51] (91 object classes), whereas the ResNet101 is pre-trained on the AVA v2.1 dataset [27] (80 human AR classes). The above detectors are applied to training videos to identify humans and objects. Such detected objects together with their relevance and class labels summarized with our descriptor encourage the AR pipeline to focus on semantically important regions and actors relevant to the task of action recognition. Figure 1a shows a few of bounding boxes detected by these four detectors.

The SDF leverages image- and video-level saliency detectors such as MLN [95] and ACLNet [94] with the goal of identifying salient regions correlating with the human gaze in spatial and temporal sense. Saliency maps extracted from training videos and summarized by our descriptor help the AR pipeline learn spatial and temporal regions correlating with actions. Figures 1b and 1c show saliency maps from region-wise and temporal saliency detectors.

IDT detectors are fused with the majority of modern AR pipelines [8–10, 22, 81] at the classifier level for the best performance while DEEP-HAL [85] learns to hallucinate, and feeds them into the classification branch called PredNet. In this paper, we go further and prepare two compact descriptors, ODF and SDF, and hallucinate them within DEEP-HAL. We equip each hallucination branch with a weighting mechanism adjusted per epoch to attain the best results. Figure 2 illustrates DEEP-HAL at the conceptual level.

For ODF descriptors, we concatenate together per bounding box per frame (i) the one-hot detection and (ii) ImageNet [64] scores, (iii) embedded confidence scores, (iv) embedded bounding box coordinates, and (v) embedded normalized frame index. For all bounding boxes, we stack such features into a matrix. Inspired by the characteristic function of the probability density function, we extract the mean, leading eigenvectors of covariance, skewness and kurtosis. For SDF descriptors, per frame, we encode saliency via (i) kernelized descriptor on spatio-angular gradient distributions of saliency maps and (ii) intensity patterns. We obtain an ODF per detector and an SDF per saliency detector. Our contributions are as follows:

i. We propose to utilize the object and human detectors to enhance the performance of AR pipelines.

ii. We design two types of statistically motivated high-order compact descriptors, Object Detection Features and Saliency Detection Features, for the use in AR pipelines.

iii. We build on the recent DEEP-HAL pipeline [85] but we introduce AssembleNet and AssembleNet++ apart from I3D backbone. Moreover, we introduce a weight learning mechanism for hallucinated feature vectors, and ODF and SDF are hallucinated which leads to the state-of-the-art performance.

2 RELATED WORK

Below, we describe handcrafted spatio-temporal video descriptors, their encoding strategies and the optical flow used by DEEP-HAL [85]. We also describe deep learning pipelines for video classification. Finally, we discuss the object category and human detectors followed by the spatial and temporal saliency detectors used by us.

Early video descriptors. Early AR used on spatio-temporal interest point detectors [6, 15, 48, 49, 78, 89] and spatio-temporal descriptors [13, 67, 75, 78–80] which capture various appearance and motion statistics. As spatio-temporal interest point detectors are unable to capture long-term motion patterns, a Dense Trajectory (DT) [78] approach densely samples feature points in each frame to track them in the video (via optical flow). Then, multiple descriptors are extracted along trajectories to capture shape, appearance and motion cues. As DT cannot compensate for the camera motion, the DT [79, 80] estimates the camera motion to remove the global background motion. IDT also removes inconsistent matches via a human detector. For spatio-temporal descriptors, IDT
employs HOG [23], HOF [13] and MBH [79]. HOG [23] contains statistics of the amplitude of image gradients w.r.t. the gradient orientation, thus it captures the static appearance cues. In contrast, HOF [13] captures histograms of optical flow while MBH [79] 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 spatio-temporal descriptors include HOG-3D [37], SIFT3D [67], SURF3D [89] and LTP [92].

We use the DEEP-HAL [85] setup. We encode HOG, HOF, and MBH descriptors on the Improved Dense Trajectories [8, 10, 78] via BoW [12, 70] and FV [56, 57].

BoW/FV encoding. BoW [12, 70] uses a k-menas vocabulary to which local descriptors are assigned. Variants include Soft Assignment (SA) [39, 76] and Localized Soft Assignment (LcSA) [43, 52]. As we use DEEP-HAL [85], we use BoW [12] with Power Normalization [43], and FV [56, 57] which capture first- and second-order statistics of local descriptors assigned to GMM clusters. DEEP-HAL [85] setup describes how to obtain the BoW/FV global descriptors.

Optical flow. Older optical flow methods cope with small displacements [31, 55] while newer methods cope with larger displacements e.g., Large Displacement Optical Flow (LDOF) [4]. Recent methods use non-rigid descriptor or segment matching [3, 88], or edge-preserving interpolation [62]. We use LDOF [55].

Object detectors. Modern deep learning methods include Region-based Convolutional Neural Networks (R-CNN) [26], its faster variants [25, 61], its mask-based variants [29], and YOLO [60], YOLO v2, YOLO v3 etc., which use a single network for efficiency.

In this paper, we use the faster R-CNN detector [61] with backbones such as (i) Inception V2 [72], (ii) Inception ResNet V2 [71], (iii) ResNet101 [30] and (iv) NASNet [97]. As the Inception V2, Inception ResNet V2 and NASNet are pre-trained on the COCO dataset [51], they detect from 91 object classes good at summarizing e.g., indoor environments and helping us associate the scene context with actions. The ResNet101 model is pre-trained on the AVA v2.1 dataset [27] with 80 different human actions, thus directly helping human-centric action recognition problems.

In addition to detection scores, we describe each bounding box with ImageNet [64] scores from pre-trained Inception ResNet V2 [71].

Saliency detectors. Image regions correlating with human visual attention are detected by saliency detectors in the form of saliency maps. Deep saliency models [32, 86] outperform conventional saliency detectors [96] but they require pixel-wise annotations. Recent models include MNL [95] (weakly-supervised model), RFCN [86] (a fully-supervised model) and a cheap non-CNN Robust Background Detector (RBD) [96] (see survey [2] for more details).

For the spatial saliency, we use MNL [95] trained on multiple noisy labels from weak/unsupervised handcrafted saliency models. For temporal saliency, we use a CNN-LSTM ACLNet [94].

Deep learning AR. Early AR CNN models use frame-wise features and average pooling [36] discarding the temporal order. Thus, frame-wise CNN scores were fed to LSTMs [16] while the two-stream networks [69] compute representations per RGB frame and per 10 stacked optical flow frames. Finally, spatio-temporal 3D CNN filters [19, 35, 74, 77] model spatio-temporal patterns.

As two-stream networks [69] discard the temporal order, rank pooling [9, 21, 22, 81] and higher-order pooling [8, 17, 38, 40, 42] are popular. A recent 1D model [5] ‘inflates’ 2D CNN filters pre-trained on ImageNet to spatio-temporal 3D filters, and implements temporal pooling. PAN [93] proposes a motion cue called Persistence of Appearance that enables the network to distill the motion information directly from adjacent RGB frames. Approach [53] uses bootstrapping with long-range temporal context attention while approach [47] proposes a graph attention model to explore the semantics. Slow-I-Fast-P (SIFP) [50] for compressed AR contains the slow and fast pathways I and P, resp., receiving a sparse sampling 1-frame clip and a dense sampling pseudo optical flow clip.

AssembleNet [66] automatically finds a neural architecture with a good connectivity to capture spatio-temporal interactions for AR through NAS. AssembleNet++ [65] further learns the interactions between raw appearance and/or motion features and spatial object information through learning dynamic attention weights and search through the inter-block attention connectivity.

We use DEEP-HAL [85] which employs a 1D convolution for temporal pooling (1D net.). We also investigate the use of AssembleNet and AssembleNet++ as backbones to show that our proposed object and saliency descriptors are independent of the backbone. We focus on the design/ability of ODF/SDF to supervise DEEP-HAL.

Power Normalization. For BoW/FV and CNN-based streams, the so-called burstiness defined as the property that a given visual element appears more times in an image than a statistically independent model would predict [34] has to be tackled. Thus, we employ Power Normalization [41–45] which suppresses the burstiness via the so-called MaxExp pooling [43] given in Section 3.

3 BACKGROUND

Below, we present Power Normalization [42, 43], count sketches [87], and the RBF feature maps which we use in our pipeline with the goal of the burstiness and dimensionality reduction, and Cartesian coordinate/frame positional encoding.

Notations. We use boldface uppercase letters to express matrices e.g., M, P, regular uppercase letters with a subscript to express matrix elements e.g., Pij is the (i, j)th element of P, boldface lowercase letters to express vectors e.g., x, φ, ψ and regular lowercase letters to denote scalars. Vectors can be numbered e.g., x_i while regular lowercase letters with a subscript express an element of vector e.g., x_i is the ith element of x. Operators ’;’ and ’;’ concatenate vectors along the first and second mode, respectively, \( \overline{v}_i \in \mathcal{I}_d \) \( v_i = [v_1, ..., v_k] \) and \( \overline{v}_i \in \mathcal{I}_d \) \( v_i = [v_1, ..., v_k] \) concatenate a group of vectors in the first and second mode, respectively, \( \overline{v}_i \) denotes the aggregation (sum) while \( \mathcal{I}_d \) denotes an index set of integers \( \{1, ..., d\} \).

3.1 Power Normalization

Proposition 1. Sigmoid (SigmE), a Max-pooling approximation [45], is an extension of the MaxExp operator defined as \( g(\psi, \eta) = 1 - (1 - \eta)\psi \) for \( \eta > 1 \) to the operator with a smooth derivative, a response defined for real-valued \( \psi \) (rather than \( \psi \geq 0 \)), a parameter \( \eta' \) and a small constant \( \epsilon' \):

\[
g(\psi, \eta') = \frac{2}{1 + e^{-\eta'\psi(\|\psi\|_1 + \epsilon')}} - 1.
\]

Proof. See papers [43, 45] for extensive considerations.
As papers [43, 85] show that various pooling operators perform similarly, we equip our hallucination streams with SigmE followed by count sketching described below.

3.2 Count Sketches

Sketching vectors by the count sketch [11, 87] is used for their dimensionality reduction which we use in this paper.

**Proposition 2.** Let \( d \) and \( d' \) denote the dimensionality of the input and sketched output vectors, respectively. Let vector \( h \in \mathbb{R}^d \) contain \( d \) uniformly drawn integer numbers from \( \{1, \ldots, d'\} \) and vector \( s \in (-1, 1)^d \) contain \( d \) uniformly drawn values from \( (-1, 1) \). Then, the sketch projection matrix \( P \in \{-1, 0, 1\}^{d' \times d} \) becomes:

\[
P_{ij} = \begin{cases} 
s_i & \text{if } h_i = j, \\
0 & \text{otherwise},
\end{cases}
\]

and the sketch projection \( p : \mathbb{R}^d \rightarrow \mathbb{R}^{d'} \) is a linear operation given as \( p(\psi) = P\psi \) (or \( p(\psi, P) = P\psi \) to highlight \( P \)).

**Proof.** It directly follows from the definition of the count sketch e.g., see Definition 1 [87].

**Remark 1.** Count sketches are unbiased estimators:

\[
\mathbb{E}_{h,s}(p(\psi, P(h, s)), p(\psi'), P(h, s))) = \langle \psi, \psi' \rangle.
\]

As variance \( \mathbb{V}_{h,s}(p(\psi), p(\psi')) \leq \frac{1}{d'} \langle \psi, \psi' \rangle^2 + \|\psi\|_2^2 \|\psi'\|_2^2 \), the larger sketches are less noisy. Thus, for every modality, we use a separate sketch matrix \( P \).

**Proof.** For the first and second property, see Appendix A of paper [87] and Lemma 3 [58].

3.3 Positional Embedding

Let \( G_{\sigma}(x-x') = \exp(-\|x-x'\|^2 / 2\sigma^2) \) denote a standard Gaussian RBF kernel centered at \( x' \) and having a bandwidth \( \sigma \). Kernel linearization refers to rewriting this \( G_{\sigma} \) as an inner product of two infinite-dimensional feature maps. To obtain these maps, we use a fast approximation method based on product probability kernels [33]. Specifically, we employ the inner product of \( d'' \)-dimensional isotropic Gaussians given \( x, x' \in \mathbb{R}^{d''} \). Thus, we have:

\[
G_{\sigma}(x-x') = \left( \frac{2}{\sigma^2} \right)^{d''/2} \int_{\xi \in \mathbb{R}^{d''}} G_{\sigma/\sqrt{2}}(x-\xi) G_{\sigma/\sqrt{2}}(x'-\xi) \, d\xi.
\]

Eq. (3) is then approximated by replacing the integral with the sum over \( Z \) pivots \( \xi_1, \ldots, \xi_Z \), thus yielding a feature map \( \phi \) as:

\[
\phi(x; \{\xi_i\}_{i \in L_Z}) = \left[ G_{\sigma/\sqrt{2}}(x-\xi_1), \ldots, G_{\sigma/\sqrt{2}}(x-\xi_Z) \right]^T, \quad \text{and } G_{\sigma}(x-x') \approx \left( \sqrt{\sigma} \phi(x), \sqrt{\sigma} \phi(x') \right),
\]

(4) is the feature map. \( \{\xi_i\}_{i \in L_Z} \) are pivots. As we use 1 dim. signals, we simply cover interval \([0, 1]\) with \( Z \) equally spaced pivots. For clarity, we drop \( \{\xi_i\}_{i \in L_Z} \) and write \( \phi(x) \), etc.

4 APPROACH

Our pipeline is illustrated in Figure 2. It consists of (i) streams already present in DEEP-HAL [85] such as the FV/BoW streams (black), the High Abstraction Features (HAF) stream and the Optical Flow Features (OFF) which are fed into (ii) the Prediction Network abbreviated as PredNet. In this paper we focus on two non-trivial streams, that is the Object Detection Features and Saliency Detection Features (dashed blue) (ODF and SDF for short).

BoW/FV/OFF streams take the backbone intermediate representations generated from the RGB frames and learn to hallucinate BoW/FV and the optical flow (ISD only) representations via the MSE loss between the ground-truth BoW/FV/OFF and the outputs of BoW/FV/OFF streams. For AssembleNet/AssembleNet++, RGB and optical flow are combined by the backbone, thus we remove the OFF stream. The same MSE loss is applied to the ODF and SDF streams. However, the design of compact ground-truth ODF and SDF descriptors is one of our main contributions.

The HAF stream processes the backbone representations prior to combining them with the hallucinated streams. PredNet fuses the combined BoW/FV/OFF/HAF and our new ODF and SDF to learn actions on videos. Below, we start by describing how we obtain our ODF and SDF descriptors before we describe modules of DEEP-HAL [85] and our modifications. One change is that we learn weights for the weighted mean pooling (i.e., \( \sum_i w_i' \psi / \sum_i w_i \)) of each stream to avoid concatenation of streams (prevent overparametrization).

4.1 Statistical Motivation

Before we outline our ODF and SDF descriptors, we motivate the use of higher-order statistics. To compare videos, we want to capture a distribution of local features/descriptors e.g., detection scores. The characteristic function \( \varphi_f(\omega) = \mathbb{E}_{\mathbf{v} \sim \gamma} \left( \exp(i\omega^T \mathbf{v}) \right) \) describes the probability density \( f_j(v) \) of some video features (local features \( \mathbf{v} \sim \gamma \)). We obtain the Taylor expansion of the characteristic function:

\[
\mathbb{E}_{\mathbf{v} \sim \gamma} \left( \sum_{i=0}^{\infty} \frac{i!}{r!} \langle \mathbf{v}, \omega \rangle^r \right) = \frac{1}{N} \sum_{n=0}^{N} \sum_{r=0}^{\infty} \frac{1}{r!} \langle \mathbf{v}_n, \mathbf{v}_n \rangle^r = \sum_{r=0}^{\infty} \sum_{n=0}^{N} \langle \mathbf{X}(r), r! \otimes \mathbf{v}_n \rangle,
\]

where \( i \) is the imaginary number, and a tensor descriptor \( \mathbf{X}(r) = \frac{1}{N} \sum_{n=0}^{N} \mathbf{v}_n \otimes \mathbf{v}_n \). In principle, with infinite data and infinite moments, one can fully capture \( f_j(v) \). In practice, first-, second- and third-order moments are typically sufficient, however, second- and third-order tensors grow quadratically and cubically w.r.t. the size of \( \mathbf{v} \).

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 \( \mathbf{X}(r) \) corresponds to the notion of order \( r \) cumulants used in calculations of skewness (\( s = 3 \)) and kurtosis (\( s = 4 \)) but it grows linearly w.r.t. the size of \( \mathbf{v} \). Thus, in what follows, we use the \( \ell_2 \) 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:

\[ v = \delta(y_{(det)}; \phi(\delta)); \phi(\omega) \in E_{\delta}, \phi(u); \phi(\frac{t_{j}}{r_{j}}) \in \mathbb{R}^{d} \]  \hspace{1cm} (7)

where \( \delta = [0, ..., 1, ..., 0]^{T} \) is a vector with all zeros but a single 1 placed at the location \( y \). As we have 91 object classes for detectors trained on the COCO dataset and 80 classes for a detector trained on the AVA v2.1 dataset, we simply assume \( y_{(det)} \in \mathbb{I}^{90+80} \), that is, the labels 0, ..., 91 describe classes from COCO while classes 92, ..., 8091 describe classes from AVA v2.1. Moreover, \( y_{(det)} \in \mathbb{R}^{1001} \) is an f1 norm normalized ImageNet classification score, \( 0 \leq \zeta \leq 1 \) is the detector confidence score, \( u_{0}, ..., u_{d} \) are the top-left and bottom-right Cartesian coordinates of a bounding box normalized in range \([0, 1]\), and \( (t − 1)/(τ − 1) \) is the frame index normalized w.r.t. the video sequence length \( t \). For feature maps \( \phi(\cdot) \) defined in Eq. (4), we simply use \( Z = 7 \) pivots and the \( \sigma \) of RBF is set to 0.5. Finally, for all detections per video from a given detector, we first compute the mean \( \mu([u_{1}, ..., u_{N}]) \in \mathbb{R}^{d} \) (we write \( \mu \)) where \( N \) is the total number of detections. Then, we form a matrix \( Y \in \mathbb{R}^{d \times N} \):

\[ Y = \frac{1}{K_{j}} \sum_{i=1}^{K_{j}} \phi_{i}(u_{j} - \mu), \quad \frac{1}{K_{j}} \sum_{i=1}^{K_{j}} \phi_{i}(u_{j} - \mu) \]  \hspace{1cm} (8)

where \( K_{j} \) denotes a number of detections per frame \( j \) from \( J \) frames from which we extract higher-order statistical moments as described below. As \( N \) is large and its size varies from video to video, hallucinating \( Y \) directly is not feasible (nor it has invariance properties).

Firstly, we obtain \( U \exists V = svd(\tilde{Y}) \) rather than \( U \tilde{X} \tilde{X}^{T} = eig(\tilde{Y} \tilde{Y}^{T}) \) as \( N \ll d \), where \( U = [u_{1}, u_{2}, ...] \). Take \( X^{(r)} = (v - \mu)^{N \times n} \) (which we abbreviate to \( X^{(r)} \)) and \( k^{(r)} = diag(X^{(r)}) \) defined in Section 4.1. We form our multi-moment descriptor \( \psi_{(det)} \in \mathbb{R}^{d(4+n)} \), \( n \geq 1 \):

\[ \psi_{(det)} = \frac{\mu}{||\mu||_{2}}, \sum_{i=1}^{K_{j}} \phi_{i}(u_{i}) \left( \sum_{i}^{K_{j}} \phi_{i}(u_{i}) \right) \frac{k^{(r)}(\phi)}{1}, \frac{k^{(r)}(\phi)}{1}, \frac{k^{(r)}(\phi)}{1}, \frac{k^{(r)}(\phi)}{1}, \frac{k^{(r)}(\phi)}{1}, \frac{k^{(r)}(\phi)}{1} \]  \hspace{1cm} (9)

The composition of Eq. (9) is described in Section 4.1. It is easy to verify that \( \frac{k^{(r)}(\phi)}{1} \) and \( \frac{k^{(r)}(\phi)}{1} \) are the empirical versions of skewness and kurtosis given by \( \frac{\Sigma_{i,j} \phi_{i}(\phi) \phi_{j}(\phi)}{var^{2}((\phi))} \) and \( \frac{\Sigma_{i,j} \phi_{i}(\phi) \phi_{j}(\phi)}{var^{2}((\phi))} \).

4.3 Saliency Detection Features

We extract directional gradients from saliency frames by discretised gradient operators \([-1, 0, 1] \) and \([−1, 0, 1]^{T} \) and obtain gradient amplitude and orientation maps \( A \) and \( \theta \), where \( \phi(\cdot) \) per frame encoded by:

\[ u'(sal) = \sum_{l=1}^{N} A_{l}(\theta_{l}j/(2\pi)) \otimes \phi \left( \frac{i+1}{W+1} \right) \otimes \phi \left( \frac{j+1}{H+1} \right) \]  \hspace{1cm} (10)

where \( \otimes \) is the Kronecker product and \( \phi(\cdot) \) follows Eq. (4) with the exception that the assignment to Gaussians is realized in the modulo ring to respect the periodical nature of \( \theta \). We encode \( \phi(\cdot) \) with 12 pivots which encode the orientation of gradients. The remaining maps \( \phi(\cdot) \) are encoded with 5 pivots each, which correspond to spatial binning. Note that \( u'(sal) \) (we write \( u' \)) is similar to a single CKN layer \([54]\) but is simpler: for one-dimensional variables we sample pivots \( c, f \), learn \( \phi(\cdot) \) for maps \( \phi(\cdot) \). Each saliency frame is described as a feature vector \( u' = [u' / ||u'||_{2}, 1/ ||u'||_{2}] \in \mathbb{R}^{d} \), where \( I \) is a vectorized low-resolution saliency map. Thus, \( u' \) captures the directional gradient statistics and the intensity-based gist of saliency maps. Subsequently, we compute the mean \( \mu([u'_{1}, ..., u'_{J}]) \in \mathbb{R}^{d} \) (we simply write \( \mu \)) where \( J \) is the total number of frames per video. Then, we obtain \( \gamma_{t} = [u'_{1}, ..., u'_{J}] \in \mathbb{R}^{d(J)} \) which is compactly described by the multi-moment Eq. (9) resulting in \( \psi_{(sal)} \in \mathbb{R}^{d(4+n)} \).

4.4 Hallucinating Streams/High Abstr. Features

Each hallucinating stream takes as input the backbone intermediate representation \( X_{(rgb)} \), of size \( 1024 \times 7 \) by removing the classifier and the last 1D conv. layer of 3D pre-trained on Kinetics-400. For AssembleNet/AssembleNet++, instead of the classification layer (FC layer), we use a 2048 dimensional output from 3D Average Pooling layer. For the BoW/FV/OFF and HAL streams, we follow the steps described in the DEEP-HAL approach \([85]\). For all streams, we use a Fully Connected (FC) unit shown in Figure 3a. Each stream uses Power Normalization (PN) realized via SigmE and Sketching (SK) from 1000 to 512 dim. via \( \tilde{\psi}_{(det)} = P_{(det)} \tilde{\psi}_{(det)} \). Outputs \( \tilde{\psi}_{(det)} \) can be now aligned with ground-truth \( \tilde{\psi}_{(det)} \) described below. The same steps are applied to High Abstraction Features (HAF), combined with other streams, and also fed into PredNet (see Fig. 2). While hallucinating streams co-supervise the backbone via external ground-truth tasks, HAF simply passes the backbone features into PredNet.

**Ground-truth BoW/FV/OFF**. We follow the DEEP-HAL setup \([85]\) and apply PCA to a concatenation of IDT trajectories (30 dim.), HOG (96 dim.), HOF (108 dim.), MBHx (96 dim.) and MBHy (96 dim.). The resulting 213 dim. local descriptors are encoded by FV and BoW with a 256 and a 1000 dim. GMM and k-means dictionaries. For the OFF stream (not used with AssembleNet or AssembleNet++), we pre-computed 1D with SDLF \( X_{(opt)} \) (Fig. 2). All ground-truth representations were Power Normalized by SigmE/sketched to 512 dim. each via \( \tilde{\psi}_{(det)} = P_{(det)} \tilde{\psi}_{(det)} \) and fed into the MLE loss. No ground-truth testing data is used in training/testing.

**Ground-truth DET1, ... DET4/SAL1/SAL2**. The ODF ground-truth training representations are of size \( 1214 \times N \), where \( N \) is the total number of bounding boxes per video (50–10000). The feature dim. 1214 is composed of 80+91 dim. one-hot detection classes, 6×7 are the \( \phi(\cdot) \)-embedded confidence, bounding box coordinates and the

---

**Figure 3**: Stream details. Figure 3a shows the stream architecture used by us for the FV, BoW, OFF, HAF, DET1, ..., DET4, SAL1 and SAL2 streams. Figure 3b shows our PredNet. Operation and their parameters are in each block e.g., \( conv2d \) and its number of filters/size, Power Normalization (PN) and Sketching (SK). We indicate the size of input and/or output under arrows.
frame number, 1001 is the ImageNet score. We also consider a variant without the RBF embedding: \( \Phi(x) = x \) (1178xN size). The SDF ground-truth training repr. are of size 556xJ, where J is the number of frames per video. 300 dim. (12x5x5) concern spatio-angular gradient distributions and 256 dim. (16x16) concern the luminance of saliency maps. Each ODF/SDF is encoded per video with the multi-moment descriptor in Eq. (9) yielding 1178x(4+n') and 556x(4+n') compact representations (we vary n' and n'' between 1 and 5). ODF and SDF are Power Normalized by SigmE/sketched to 512 dim. each via \( \Phi'_i = P_i(\Phi_i) \) and fed into the MSE loss. No ground-truth testing representations were used for training/testing.

4.5 Objective Function

During training, we combine MSE loss functions which co-supervise hallucination streams with the classifier:

\[
\ell'(X, y; \Theta) = \frac{1}{|H|} \sum_{i \in H} \left| \Phi_i^T - \Phi_i^* \right|^2_F + \ell'_{(\text{det})}(\cdot; \Theta_{(t)}) \cdot \gamma; \Theta_{(t)}),
\]

where: \( \forall i \in H, \Phi_i = \mathbb{P}_i(g(h(X, \Theta)), \eta), \eta'_i = \mathbb{P}_i \eta_i, \)

\[
\Phi_{(haf)} = \mathbb{P}_i(h(X, \Theta_{(haf)})), \eta',
\]

\[
\Phi_{(tot)} = \frac{1}{|H|+1} \sum_{i \in H} w_i \tilde{\Phi}_i',
\]

\[
\tilde{\Phi}_{(det)} = \frac{1}{|D|} \sum_{i \in D} w_i \tilde{\Phi}_i', \tilde{\Phi}_{(sal)} = \frac{1}{|S|} \sum_{i \in S} w_i \tilde{\Phi}_i'.
\]

The above equation is a trade-off between the MSE loss functions (\( \left| \tilde{\Phi}_i - \tilde{\Phi}_i^* \right|^2_F, i \in H \)) and the classification loss (\( f(\cdot, y; \Theta_{(t)}) \)) with some label \( y \in \mathcal{Y} \) and parameters \( \Theta_{(t)} \equiv [W, b] \). The trade-off is controlled by \( \alpha \geq 0 \) while MSE is computed over halluc. streams \( i \in H \), and \( H = \{(f01), (f02), (bow), (off), (det1), \ldots (det4), (sal1), (sal2)\} \) are our set of hallucination streams. Moreover, \( g(\cdot, \eta) \) is a Power Norm. in Eq. (1), \( f(\cdot; \Theta_{(pr)}) \) which we learn, \( h(X, \Theta), i \in H \) are the hallucination streams while \( \tilde{\Phi}_i, i \in H \) are resulting hallucinated BoW/FV/OFF/ODF/SDF representations. We set \( \alpha = 1 \). Moreover, \( h(X, \Theta_{(haf)}) \) is the HAF stream with the sketched output \( \Phi_{(haf)} = \mathbb{P}_i(h(X, \Theta_{(haf)})) \). For the hallucination streams, we learn parameters \( \Theta_{(haf)} \) while for HAF, we learn \( \Theta_{(haf)} \). The full set of parameters we learn is defined as \( \Theta \equiv ((\Theta_{i}, i \in H), \Theta_{(haf)}, \Theta_{(pr)}, \Theta_{(t)}) \). Furthermore, \( \{P_i, i \in H\} \) and \( \{P_i, i \in H\} \) are the projection matrices for count sketching of streams \( \{\Phi_i, i \in H\} \) and the ground-truth feature vectors \( \{\phi_i, i \in H\} \). Finally, for \( \Phi_{(det)} \), we are a weighted average of several streams fed into the PredNet module. Moreover, \( H = \{(f01), (f02), (bow), (off), (det1), (det4), (sal1), (sal2)\} \) and \( S = \{(sal1), (sal2)\} \). Section 3.2 details how to select matrices \( P \). Let \( T \) be set to either \( H \), D or S, then our weights are:

\[
w_i = \frac{1}{|T|} \max_{\rho} \left( \frac{w_i}{\rho} \right),
\]

Prior to CNN training, we train an SVM on each ground-truth stream separately (using a manageable training subset), and we set weights \( w' \) proportionally to the accuracies obtained on the validation set. For the HAF stream, we simply set \( w'_{(haf)} = \frac{1}{|H|} \max_{\rho} \left( \frac{w_i}{\rho} \right) \) and \( \rho = 0.1 \). For the first few epochs (i.e., 10), we set \( \beta = 0 \) so that all streams receive equal weights. Subsequently, in each epoch, we run the Golden-section search to find the best \( \beta \geq 0 \). We start from initial boundary values \( \beta \in [0, 50] \), we train an SVM on a manageable subset of training data and evaluate \( \beta \) on the validation set, and we update boundary values for the next epoch accordingly. Eq. (12) has a nice property: for \( \beta = 0 \), we have \( w_i = 1/|T| \). For \( \beta \to \infty \), we have \( w_i = 1 \) if \( w_i = \max_{\rho} \left( \frac{w_i}{\rho} \right) \), otherwise \( w_i = 0 \). Thus, \( \beta \) interpolates between equalizing all weights and the winner-takes-all solution.

5 EXPERIMENTS

5.1 Datasets and Evaluation Protocols

HMDB-51 [46] has 6766 internet videos/51 classes; each video has \( \sim 20 \sim 1000 \) frames. We report the mean accuracy across three splits. YUP++ [20] has 20 scene classes of video textures, 60 videos per class. Splits contain scenes captured by the static or moving camera. We use standard splits (1/9 dataset for training) for evaluation. MPII Cooking Activities [63] contains high-resolution videos of people cooking dishes. The 64 activities from 3748 clips include coarse actions e.g., opening refrigerator, and fine-grained actions e.g., peel, slice, cut apart. We use the mean Average Precision (mAP) over 7-fold cross validation. For human-centric protocol [7, 9], we use the faster RCNN [61] to crop video around human subjects. Charades [68] consists of 9848 videos of daily indoor activities, 66900 clip annotations and 157 classes.

EPIC-Kitchens [14] is a multi-class egocentric dataset with 28K training videos associated with 352 noun and 125 verb classes. The dataset consists of 39,594 segments in 432 videos. We follow protocol [1]. We evaluate our model on validation, standard seen (S1: 8047 videos), and unseen (S2: 2929 videos) test sets.

5.2 Evaluations

Below, we show the effectiveness of our method. For smaller datasets, we use the I3D backbone. For large Charades and EPIC-Kitchens, we additionally investigate AssembleNet and AssembleNet++ backbones. Firstly, we evaluate various design components.

Ground-truth ODF+SVM. Firstly, we evaluate our ODF on SVM given the HMDB-51 dataset. We set \( n' = 3 \) for Eq. (9) and compare various detector backbones and pooling strategies. Table 1 shows that all detectors perform similarly with \( (det3) \) being slightly better than other methods. Moreover, max-pooling on ODFs from all four detectors is marginally better than the average-pooling. However, only the weighted mean \( (\text{all}-\text{w}) \) according to Eq. (12) outperforms \( (det3) \) which highlights the need for the robust aggregation of ODFs. Similarly, when we combine pre-trained DEEP-HAL with all detectors, the weighted mean \( (\text{DEEP-HAL+all- w}) \) performs best. Table 2 shows the similar trend on YUP++. We trained SVM only on videos for which at least one detection occurred, thus a 75.74% accuracy is much lower than the main results reported on the full pipeline. Figure 4 shows that \( \beta \neq 1 \) has a positive impact on reweighting.

Ground-truth SDF. The SDF on HMDB-51 and YUP++ yielded 24.35% and 32.68% accuracy. This is expected as SDFs do not capture a discriminative information per se but they locate salient spatial and temporal regions to focus the main network on them.

Multi-moment descr. Figure 5 shows that the concat. of the mean and three eigenvectors according to Eq. (9) yields good results
0.1
0.1
0.1
0.1
0.1
1
0.01 0.1 0.5 1 2 4 8 2057
59
61
63
β
mean accuracy (%)
0.01 0.1 0.5 1 2 4 8 2071
73
75
77
0.01 0.1 0.5 1 2 4 8 2040
40.5
41
41.5
42
β
mean accuracy (%)0.01 0.1 0.5 1 2 4 8 2080
81
82
83
84
1
4
13x215
40.5
21x40
mean accuracy (%)
mean accuracy (%)
mean accuracy (%)
mean accuracy (%)

Table 1: Evaluations of ODF on HMDB-51. (top) We evaluate backbones such as (det1) Inception V2, (det2) Inception ResNet V2, (det3) ResNet101 and (det4) NASNet. (middle) The average-pooled, max-pooled and the weighted mean combination of all detectors are given by (all-avg), (all-max) and (all+wei). (bottom) Pre-trained DEEP-HAL combined with all four detectors by the average-pooling, max-pooling and the weighted mean.

| avg    | max   | wei   |
|--------|-------|-------|
| det1   | 35.12%| 42.34%| 60.32%|
| DEEP-HAL+all | 74.22%| 71.85%| 75.74%|

Table 2: Pooling on YUP++. Results for the average-pooled (avg), max-pooled (max) and the weighted mean (wei) of all detectors (all) vs. pre-trained DEEP-HAL combined with all detectors by the average-pooling, max-pooling and the weighted mean.

but adding further vectors deteriorates the performance. Adding skewness and kurtosis (ς and ϕ) further improves results, while adding eigenvalues has a limited impact.

HMDB-51. Table 3 shows several DEEP-HAL variants, which all hallucinate BoW/FV/ODF. DEEP-HAL with our reweighting mechanism. (DEEP-HAL+W) outperforms the original DEEP-HAL denoted as (HAF/BoW/FV hal.) [85] by ~0.8. DEEP-HAL with our ODF and SDF descriptors (DEEP-HAL+ODF) and (DEEP-HAL+SDF) outperform (HAF/BoW/FV hal) by ~1.8% and ~1.4%, resp. This shows that both ODF and SDF are effective. Combining DEEP-HAL, ODF and SDF outperform DEEP-HAL by ~2.7% demonstrating the complementary nature of ODF and SDF. Utilizing our weighting mechanism with DEEP-HAL, ODF and SDF denoted as (DEEP-HAL+W+ODF+SDF) outperform (HAF/BoW/FV hal) by ~4.6%. Finally, DEEP-HAL with weighting, and ODF and SDF with RBF feature maps from Eq. (4) outperform (HAF/BoW/FV hal) by ~5.1.

YUP++. Table 4 shows that ODF is better than SDF, that is (DEEP-HAL+ODF) and (DEEP-HAL+SDF) outperform (HAF/BoW/FV hal) by ~0.6% and ~0.2%, resp. This is expected as YUP++ contains dynamic scenes without objects/specific saliency regions correlating with class concepts. However, a combination of detectors/saliency (DEEP-HAL+SD) plus weighting (DEEP-HAL+W+ODF+SDF) plus the RBF maps (DEEP-HAL+W+G+ODF+SDF) outperform (HAF/BoW/FV hal) by ~0.7%, ~1.6% and ~1.8% accuracy, resp.

MPII. Table 5 shows a ~3.0% mAP gain over (HAF/BoW/FV hal) due to detectors capturing the human interaction with objects.

Charades. Table 6 (top) presents relative gains of our hallucination pipeline (DEEP-HAL) with weighted mean pooling (W) and the RBF maps (G) denoted as (DEEP-HAL+W+G). We evaluate Object Detection Features (ODF) and Saliency Detection Features (SDF) with 512 dim. sketching (SK512) and note that (DEEP-HAL+W+G+ODF+SDF (SK512)) outperforms (DEEP-HAL+W+G+SDF (SK512)), and both methods outperform the baseline (HAF/BoW/FV hal) [85].

| sp1 | sp2 | sp3 | sp4 | sp5 | sp6 | sp7 | mAP |
|-----|-----|-----|-----|-----|-----|-----|-----|
| DEEP-HAL+ODF | 95.00% | 90.93% | 93.52% | 93.0% | 93.2% |
| DEEP-HAL+SDF | 95.10% | 91.11% | 93.61% | 93.1% | 93.3% |
| DEEP-HAL+W+ODF+SDF | 96.30% | 92.22% | 94.17% | 94.3% | 94.2% |
| DEEP-HAL+W+G+ODF+SDF | 96.30% | 92.40% | 94.35% | 94.4% | 94.4% |

Table 4: Evaluations of (top) our methods and (bottom) comparisons to the state of the art on HMDB-51.

| sp1 | sp2 | sp3 | sp4 | sp5 | sp6 | sp7 | mAP |
|-----|-----|-----|-----|-----|-----|-----|-----|
| KRP-FS+IDT 76.1% [9] | GRP+IDT 75.5% [7] |
| ADL+3D [81] | HAF/BoW/FV hal. [85] |
| MSOE-two-stream [26] | YUP++ [85] |

Table 5: Evaluations of (top) our methods and (bottom) comparisons to the state of the art on MPII.

Figure 4: The impact of β in the weighted mean on the classification results. Figure 4a shows results for HMDB-51 on (top) four detectors combined+SVM and (bottom) DEEP-HAL with four detectors combined+SVM. Figure 4b shows results for YUP++.

Figure 5: ODF eval. on SVM on four detectors (the weighted mean). Fig. 5a and 5b show results on HMDB-51 and YUP++. μ, ς, · · · , ϕ, ζ, ς correspond to the entries in Eq. (9).
Table 6 (bottom) shows that combining ODF and SDF into (DEEP-HAL+W+G+ODF+SDF (SK512)) yields 49.06% mAP which constitutes on a ~6% gain over the baseline (HAL/Bow/FV hal.) [85]. This demonstrates that ODF and SDF are highly complementary. Applying a larger sketch (DEEP-HAL+W+G+ODF+SDF (SK1024)) yields 50.14% mAP which matches the use (DEEP-HAL+W+G+ODF+SDF (exact)) that denotes a late fusion by concatenation of ODF and SDF with the stream resulting from DEEP-HAL fed into Pred-Net. Note that (exact) indicates that ODF and SDF are not hallucinated at the test time but they are computed; the results matching between (DEEP-HAL+W+G+ODF+SDF (SK1024)) and (DEEP-HAL+W+G+ODF+SDF (exact)) show that we can hallucinate ODF and SDF at the test time while regaining the full performance. We save computational time and hallucinate the detection and saliency features which boost results on Charades by ~6% over the baseline.

Table 7 shows that our idea applied to AssembleNet and AssembleNet++ yields state of the art e.g., we outperform these two networks by 4.5% and 5.6% mAP, respectively. We note that our detectors do not need to be computed at all at the test time.

In contrast, the best currently reported papers such as SlowFast networks [18] and AssembleNet [66] achieve 45.2% and 51.6% on Charades. As SlowFast networks and AssembleNet backbones can be used in place of 3D in our experimental setup, our approach is ‘orthogonal’ to these latest developments which focus on heavy mining for combinations of neural blocks/dataflow between them to obtain an ‘optimal’ pipeline. We achieve similar results with a simple approach based on self-supervised learning. Our pipeline is lightweight by comparison (no need for computations of the optical flow, or detections or segmentation masks at test time).

**ImageNet (global score) vs. object detectors.** Various scores from the object and saliency detectors which we use cannot be plugged directly into the DEEP-HAL due to the varying number of objects detected and the varying number of frames, thus we propose and use ODF and SDF descriptors. We also note that using a simplified variant of ODF which stacks up ImageNet scores per frame into a matrix (no detectors) to which we apply our multi-modal detector yielded ~4% worse results on Charades than our DEEP-HAL+ODF (detectors-based approach) which yields 48.0% mAP. This is expected as ImageNet is trained in a multi-class setting (one object per image) while detectors let us model robustly distributions of object classes and locations per frame.

**EPIC-Kitchens.** Table 8 shows the experimental results. 3D and AssembleNet/AssembleNet++ learn human-like semantic features due to ODF/SDF, and there is no evidence a backbone can discover these without a guidance. By comparing MPII (3748 clips) with large EPIC-Kitchens (39594 clips) (both about cooking), SDF+ODF boost MPII from 81.8 to 84.8%, and SDF+ODF boost EPIC-Kitchens from 32.51% (DEEP-HAL) to 35.88% (on seen classes protocol), and from 22.33% (DEEP-HAL) to 27.32% (on unseen classes protocol). The boost is 3% on both MPII and EPIC-Kitchens (nearly 10× more clips than MPII).

### 6 CONCLUSIONS

We have introduced two simple yet effective object and saliency descriptors, which perform self-supervision of an AR hallucination-based network. We have shown that modeling high-order statistical moments can result in small representations that can self-supervise our AR pipeline. The findings are in line with recent multi-task learning papers which argue that related tasks can co-supervise the main task. We are the first to hallucinate object and saliency detection descriptors with clear cut improvements in accuracy, and state-of-the-art results on the large-scale Charades and EPIC-Kitchens. More importantly, we demonstrate that hallucinating object and saliency detections is an attractive proposition even for the state-of-the-art AR backbones such as AssembleNet and AssembleNet++.

### REFERENCES

[1] Fabien Baradel, Natalia Neverova, Christian Wolf, Julien Mille, and Greg Mori. 2018. Object Level Visual Reasoning in Videos. In ECCV. Springer Science+Business Media, Munich, Germany, 1–16.

[2] Ali Borji, Ming-Ming Cheng, Huazhu Jiang, and Jia Li. 2015. Salient Object Detection: A Benchmark. TIP 24, 12 (2015), 5706–5722. https://doi.org/10.1109/TIP.2015.2487833

| Method                                      | 
|---------------------------------------------|
| HAL/Bow/FV hal.                             | 41.5 |
| DEEP-HAL+W+G+ODF+SDF (SK512)               | 47.42 |
| DEEP-HAL+W+G+ODF+SDF+DEEP-HAL+W+G+ODF+SDF (SK1024) | 43.30 |

Table 6: Evaluations of our methods on Charades (3D backbone).

| Method                                      | 
|---------------------------------------------|
| AssembleNet++ 50 (Kinetics-400 pre-training) | 55.8 |
| AssembleNet 50 (without pre-training)       | 57.98 |

Table 7: Evaluations of our methods on the Charades dataset (AssembleNet and AssembleNet++ backbones). Note that we do not use segmentation masks for AssembleNet and AssembleNet++, thus baseline results reported by us are slightly lower compared to authors’ results of 55.0% and 59.8% mAP, respectively.

| Method                                      | 
|---------------------------------------------|
| DEEP-HAL+ 50 (Kinetics-400 pre-training)    | 35.8 |
| DEEP-HAL+ 50 (without pre-training)         | 36.7 |

Table 8: Experimental results on the EPIC-Kitchens.
