Hallucinating IDT Descriptors and I3D Optical Flow Features for Action Recognition with CNNs

Lei Wang[*,1,2] Piotr Koniusz[*,1,2] Du Q. Huynh[3]
[1]Data61/CSIRO, [2]Australian National University, [3]University of Western Australia
firstname.lastname@{data61.csiro.au[1], anu.edu.au[2], uwa.edu.au[3]}

Abstract

In this paper, we revive the use of old-fashioned handcrafted video representations for action recognition and put new life into these techniques via a CNN-based hallucination step. Despite of the use of RGB and optical flow frames, the I3D model (amongst others) thrives on combining its output with the Improved Dense Trajectory (IDT) and extracted with its low-level video descriptors encoded via Bag-of-Words (BoW) and Fisher Vectors (FV). Such a fusion of CNNs and handcrafted representations is time-consuming due to pre-processing, descriptor extraction, encoding and tuning parameters. Thus, we propose an end-to-end trainable network with streams which learn the IDT-based BoW/FV representations at the training stage and are simple to integrate with the I3D model. Specifically, each stream takes I3D feature maps ahead of the last 1D conv. layer and learns to 'translate' these maps to BoW/FV representations. Thus, our model can hallucinate and use such synthesized BoW/FV representations at the testing stage. We show that even features of the entire I3D optical flow stream can be hallucinated thus simplifying the pipeline. Our model saves 20–55h of computations and yields state-of-the-art results on four publicly available datasets.

1. Introduction

Action Recognition (AR) pipelines have transitioned from the use of handcrafted descriptors [13, 54, 31, 64, 65, 66] to CNN models such as the two-stream network [56], 3D spatio-temporal features [60], spatio-temporal ResNet [17] and the I3D network pre-trained on Kinetics-400 [4]. Such CNNs operate on RGB/optical flow videos thus failing to capture some domain-specific information which sophisticated low-level representations capture by design. One prominent example are Improved Dense Trajectory (IDT) descriptors [66] which are typically encoded with Bag-of-Words (BoW) [57, 12] or Fisher Vectors (FV) [47, 48] and fused with CNNs [21, 8, 9, 67, 10] at the classifier which improves results due to several sophisticated steps of IDT: (i) camera motion estimation, (ii) motion descriptor modeling along motion trajectories estimated by the optical flow, (iii) pruning inconsistent matches, (iv) focusing on human motions via a human detector, (v) combination of IDT with powerful and highly complementary to each other video descriptors such as Histogram of Oriented Gradients (HOG) [22, 31], Histogram of Optical Flow (HOF) [13] and Motion Boundary Histogram (MBH) [65] *e.g.*, HOF and MBH contain zero- and first-order motion statistics [66].

However, extracting dense trajectories and corresponding video descriptors is costly due to several off-line/CPU-based steps. Motivated by this shortcoming, we propose simple trainable CNN streams on top of a CNN network (in our case I3D [4]) which learn to 'translate' the I3D output into IDT-based BoW and FV global video descriptors. We can even 'translate' the I3D RGB output into I3D Optical Flow Features (OFF). At the testing stage, our so-called BoW, and FV and OFF streams (on top of I3D) are able to hallucinate such global descriptors which we feed into the final layer preceding a classifier. We show that IDT/OFF representations can be synthesized by our network thus removing the need of actually computing them which simplifies the AR pipeline. With a handful of convolutional/FC layers and basic CNN building blocks, our representation rivals sophisticated AR pipelines that aggregate features frame-by-frame *e.g.*, HOK [8] and rank-pooling [21, 9, 67, 10]. Below, we detail our contributions:

I. We are the first to propose that old-fashioned IDT-based BoW and FV global video descriptors can be learned via simple dedicated CNN-streams at the training stage and simply hallucinated for classification with a CNN action recognition pipeline during testing.

II. We show that even the I3D optical flow stream can be easily hallucinated from the I3D RGB stream.

III. We study various aspects of our model *e.g.*, the count sketch [49] of features to avoid overfitting when fusing several streams and Power Normalization [38, 37, 39] to prevent so-called burstiness in BoW, FV and CNNs, and we perform several experiments on four datasets.

Sections 2 and 3 introduce the background, notations and concepts. Sections 4 and 5 present our method and results.

*Both authors contributed equally.

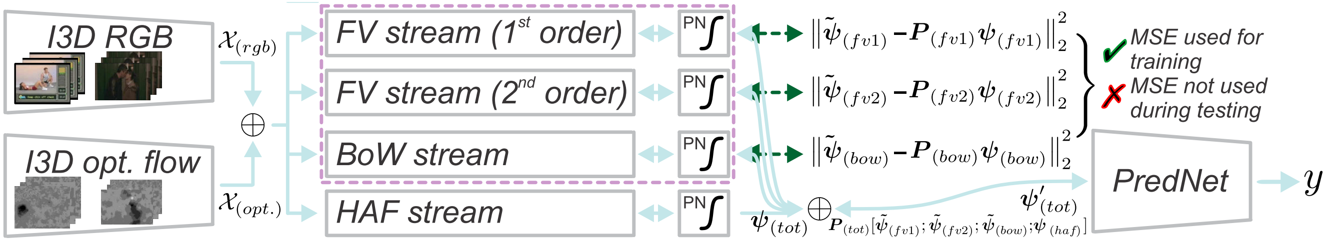

Figure 1: The overview of our pipeline. We remove the prediction and the last 1D conv. layers from I3D RGB and optical flow streams, concatenate ($\oplus$) the 1024×7 feature representations $\mathcal{X}_{(rgb)}$ and $\mathcal{X}_{(opt.)}$, and feed them into our *Fisher Vector* (*FV*), *Bag-of-Words* (*BoW*), and *the High Abstraction Features* (*HAF*) streams followed by the *Power Normalization* (*PN*) blocks. The resulting feature vectors $\tilde{\psi}_{(fv1)}$, $\tilde{\psi}_{(fv2)}$, $\tilde{\psi}_{(bow)}$ and $\psi_{(haf)}$ are concatenated ($\oplus$) and fed into our *Prediction Network* (*PredNet*). By ✓, we indicate that the three Mean Square Error (MSE) losses are only applied at the training stage to train our FV (first- and second-order components) and BoW hallucinating streams (indicated in dashed red). By ✗, we indicate that the MSE losses are switched off at the testing stage. Thus, we hallucinate $\tilde{\psi}_{(fv1)}$, $\tilde{\psi}_{(fv2)}$ and $\tilde{\psi}_{(bow)}$, and pass them to PredNet together with $\psi_{(haf)}$ to obtain labels y. The original training FV and BoW feature vectors (used only during training) are denoted by $\psi_{(fv1)}$, $\psi_{(fv2)}$ and $\psi_{(bow)}$, while P are count sketch projecting matrices (see text for details).

2. Related Work

Below, we describe handcrafted spatio-temporal video descriptors and their encoding strategies, optical flow, and deep learning pipelines for video classification.

Handcrafted video representations. Early AR relied on spatio-temporal interest point detectors [43, 14, 6, 73, 44, 64] and spatio-temporal descriptors [13, 54, 61, 64, 65, 66] which capture various appearance and motion statistics.

Spatio-temporal interest point detectors were developed for the task of identifying spatio-temporal regions of videos rich in motion patterns relevant to classification, thus providing sampling locations for local descriptors. The number of sampling points had a significant influence on the processing speed due to the volumetric nature of videos. Harris3D [43], one of the earliest detectors, performs a search for extreme points in the spatio-temporal domain via the so-called structure tensor and the determinant-to-trace ratio test. Cuboid [14], a faster detector, applies Gaussian and Gabor filters in spatial and temporal domains, respectively. Selective STIP [6] extracts initial key-point candidates with the Harris corner detector followed by the candidate suppression with a so-called surround suppression mask. Hes-STIP, a more recent detector, uses integral videos and Hessian matrix to search the scale-space for local maxima of the signal. Evaluations and further reading on spatio-temporal detectors can be found in surveys [23, 68, 69].

One drawback of spatio-temporal interest point detectors is the sparsity of key-points and inability to capture long-term motion patterns. Thus, a Dense Trajectory (DT) [64] 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 IDT [66, 65] 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 [22], HOF [13] and MBH [65]. HOG [22] contains statistics of the amplitude of image gradients w.r.t. the gradient orientation. Thus, it captures the static appearance cues while its close cousin, HOG-3D [31], is designed for spatio-temporal interest points. In contrast, HOF [13] captures histograms of optical flow while MBH [65] 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 SIFT3D [54], SURF3D [73] and LTP [75].

In this work, we follow the standard practice, that is, we use the Improved Dense Trajectories [64, 8, 10] and we encode them together with HOG, HOF, and MBH descriptors via BoW [57, 12] and FV [47, 48] which we describe below.

Descriptor encoding. BoW [57, 12], a global image representation, is likely the oldest encoding strategy for local descriptors. It consists of (i) clustering with k-means for a collection of descriptor vectors from the training set to build so-called visual vocabulary, (ii) assigning each descriptor to its nearest cluster center from the visual dictionary, and (iii) aggregating the one-hot assignment vectors via average pooling. Similar models such as Soft Assignment (SA) [62, 33] and Localized Soft Assignment (LcSA) [45, 38] use the Component Membership Probability (CMP) of GMM to assign each descriptor with some probability to visual words followed by average or non-linear pooling [38, 70].

In this paper, we chose the simplest BoW model [12]

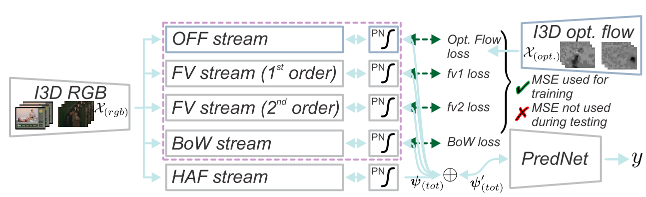

Figure 2: Hallucinating the Optical Flow Features (OFF).

with Power Normalization [38] detailed in Section 3. BoW can be seen as zero-order statistics of FV [47, 48], thus we also employ FV to capture first- and second-order statistics of local descriptors. FV builds a visual dictionary from training data via GMM. Then, a displacement/square displacement of each descriptor vector w.r.t. each GMM component center is taken, normalized by its GMM standard deviation/variance to capture the first/second-order terms, and then soft-assigned via CMP to each GMM component.

Optical flow. As a key concept in AR from videos, optical flow is the distribution of velocities of movement of brightness pattern across frames [26] such as the pattern of motion of objects, surfaces and edges in a visual scene caused by the relative motion between an observer and a scene [27]. Early optical flow coped with small displacements via energy minimization [26, 46]. However, to capture informative motions of subjects/objects, optical flow needs to cope with large displacements [1]. As energy-based methods suffer from the local minima, local descriptor matching is used in Large Displacement Optical Flow (LDOF) [3]. Recent methods use non-rigid descriptor matching [72], segment matching [2] or even edge-preserving interpolation [51].

In this work, we are not concerned with the use of the newest possible optical flow. Thus, we opt for LDOF [46].

CNN-based action recognition. The success of AlexNet [40] and ImageNet [53] sparked studies into AR with CNNs. Early models extracted per-frame representations followed by average pooling [30] which discards the temporal order. To fix such a shortcoming, frame-wise CNN scores were fed to LSTMs [15]. Two-stream networks [56] compute representations per RGB frame and per 10 stacked optical flow frames. However, a more obvious extension is to model spatio-temporal 3D CNN filters [29, 60, 17, 63].

The recent I3D model [4] draws on the two-stream networks, 'inflates' 2D CNN filters pre-trained on ImageNet to spatio-temporal 3D filters, and implements temporal pooling across the inception module. In this paper, we opt for the I3D network but our proposed layers are independent of the CNN design. We are concerned with 'absorbing' the old yet powerful IDT representations and/or optical flow features into CNN and hallucinating them at the test time.

Temporal aggregation. While two-stream networks [56] discard the temporal order and others use LSTMs [15], many AR pipelines address the spatio-temporal aggregation. Rank pooling [20, 21] projects frame-wise feature vectors onto a line such that the temporal order of vectors is preserved along the line. Subspace and kernel rank pooling [9, 67] use projections into the RKHS in which the temporal order of frames is preserved. Another aggregation family captures second- or higher-order statistics [8, 32, 37, 16].

In this paper, we are not concerned with temporal pooling. Thus, we use a 1D convolution (as in I3D [4]).

Power Normalization family. BoW, FV and even CNN-based descriptors have to deal with 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*' [28], a phenomenon also present in video descriptors. Power Normalization [38, 36] is known to suppress the burstiness, and it has been extensively studied in the context of BoW [38, 36, 37, 39]. Moreover, a connection to Max-pooling was found in survey [38] which also shows 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 papers [38, 39], many Power Normalization functions are closely related. We outline Power Normalizations used in our work in Section 3.

3. Background

In our work, we use BoW/FV (training stage), as well as Power Normalization [38, 37] and count sketches [71].

Notations. We use boldface uppercase letters to express matrices *e.g.*, $\boldsymbol{M}, \boldsymbol{P}$, regular uppercase letters with a subscript to express matrix elements *e.g.*, P_{ij} is the $(i,j)^{\text{th}}$ element of $\boldsymbol{P}$, boldface lowercase letters to express vectors, *e.g.* $\mathbf{x}, \boldsymbol{\phi}, \boldsymbol{\psi}$, and regular lowercase letters to denote scalars. Vectors can be numbered *e.g.*, $\boldsymbol{m}_1, ..., \boldsymbol{m}_K$ or $\mathbf{x}_n$, *etc.*, while regular lowercase letters with a subscript express an element of vector *e.g.*, m_i is the i^{th} element of $\boldsymbol{m}$. Operators ';' and $\oplus$ concatenate vectors *e.g.*, $\oplus_{i\in\mathcal{I}_K}\mathbf{v}_i = [\mathbf{v}_1; ...; \mathbf{v}_K]$ while $\mathcal{I}_d$ denotes an index set of integers $\{1, ..., d\}$.

3.1. Descriptor Encoding Schemes

Bag-of-Words [57, 12] assigns each local descriptor $\boldsymbol{x}$ to the closest visual word from $\boldsymbol{M} = [\boldsymbol{m}_1, ..., \boldsymbol{m}_K]$ built via k-means. In order to obtain mid-level feature $\boldsymbol{\phi}$, we solve:

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

Fisher Vector Encoding [47, 48] uses a Mixture of K Gaussians from a GMM used as a dictionary. It performs descriptor coding w.r.t. to Gaussian components $G(w_k, \boldsymbol{m}_k, \boldsymbol{\sigma}_k)$ which are parametrized by mixing probability, mean, and on-diagonal standard deviation. The first- and second-order features $\boldsymbol{\phi}_k, \boldsymbol{\phi}'_k \in \mathbb{R}^D$ are :

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

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

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

where p and θ are the component membership probabilities and parameters of GMM, respectively. For each descriptor $\boldsymbol{x}$ of dimensionality D (after PCA), its encoding $\boldsymbol{\phi}$ is of $2KD$ dim. as $\boldsymbol{\phi}$ contains first- and second-order statistics.

3.2. Pooling a.k.a. Aggregation

Traditionally, pooling is performed via averaging mid-level feature vectors $\phi(\mathbf{x})$ corresponding to (local) descriptors $\mathbf{x} \in \boldsymbol{\mathcal{X}}$ from a video sequence $\boldsymbol{\mathcal{X}}$, that is $\boldsymbol{\psi} = \text{avg}_{\mathbf{x}\in\boldsymbol{\mathcal{X}}}\, \boldsymbol{\phi}(\mathbf{x})$, and (optionally) applying the ℓ_2-norm normalization. In this paper, we work with either sequences $\boldsymbol{\mathcal{X}}$ (for which the above step is used) or subsequences.

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

$$\boldsymbol{\psi}_{s,t} = (\boldsymbol{\phi}'_t - \boldsymbol{\phi}'_{s-1}) / (\|\boldsymbol{\phi}'_t - \boldsymbol{\phi}'_{s-1}\|_2 + \epsilon), \tag{4}$$

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

3.3. Power Normalization

As alluded to in Section 2, we apply Power Normalizing functions to BoW and FV streams which hallucinate these two modalities (and HAF/OFF stream explained later). We investigate three operators $g(\boldsymbol{\psi}, \cdot)$ detailed by Remarks 1–3.

Remark 1. *AsinhE function [39] is an extension of a well-known Power Normalization (Gamma) [39] defined as $g(\boldsymbol{\psi}, \gamma) = \text{Sgn}(\boldsymbol{\psi})|\boldsymbol{\psi}|^\gamma$ for $0 < \gamma \leq 1$ to the operator with a smooth derivative and a parameter γ'. AsinhE is defined as the normalized Arcsin hyperbolic function:*

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

Remark 2. *Sigmoid (SigmE), a Max-pooling approximation [39], is an extension of the MaxExp operator defined as $g(\boldsymbol{\psi}, \eta) = 1 - (1 - \boldsymbol{\psi})^\eta$ for $\eta > 1$ to the operator with a smooth derivative, a response defined for real-valued $\boldsymbol{\psi}$ (rather than $\boldsymbol{\psi} \geq 0$), a parameter η' and a small const. ϵ':*

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

Remark 3. *AxMin, a piece-wise linear form of SigmE [39], is given as $g(\boldsymbol{\psi}, \eta'') = \text{Sgn}(\boldsymbol{\psi}) \min(\eta''\boldsymbol{\psi}/(\|\boldsymbol{\psi}\|_2 + \epsilon'), 1)$ for $\eta'' > 1$ and a small constant ϵ'.*

Despite the similar role of these three pooling operators, we investigate each of them as their interplay with end-to-end learning differs. Specifically, $\lim_{\boldsymbol{\psi}\to\pm\infty} g(\boldsymbol{\psi}, \cdot)$ for AsinhE and SigmE are $\pm\infty$ and ± 1, resp., thus their asymptotic behavior differs. Moreover, AxMin is non-smooth and relies on the same gradient re-projection properties as ReLU.

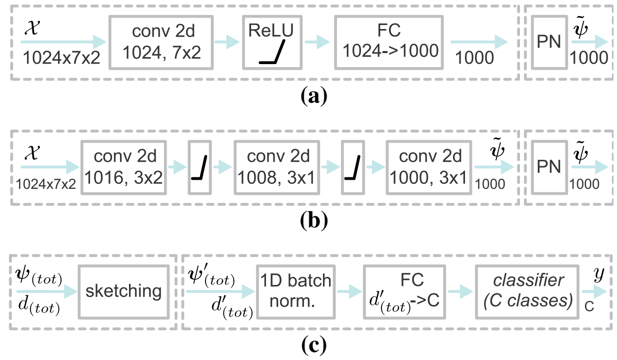

Figure 3: Stream types used in our network. Figures 3a and 3b show *Fully Connected* and *Convolutional* variants used for the practical realization of the FV, BoW, OFF and HAF streams. Figure 3c shows our PredNet. Note that we indicate the type of operation and its parameters in each block *e.g.*, *conv2d* and its number of filters/size, or *Power Normalization* (*PN*). Beneath arrows, we indicate the size of input, intermediate or output representation.

3.4. Count Sketches

Sketching vectors by the count sketch [11, 71] 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 $\boldsymbol{h} \in \mathcal{I}^d_{d'}$ contain d uniformly drawn integer numbers from $\{1, ..., d'\}$ and vector $\boldsymbol{s} \in \{-1, 1\}^d$ contain d uniformly drawn values from $\{-1, 1\}$. Then, the sketch projection matrix $\boldsymbol{P} \in \{-1, 0, 1\}^{d' \times d}$ becomes:*

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

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

Proof. It directly follows from the definition of the count sketch *e.g.*, see Definition 1 [71]. □

Remark 4. *Count sketches are unbiased estimators: $\mathbb{E}_{\boldsymbol{h},\boldsymbol{s}}(p(\boldsymbol{\psi}, \boldsymbol{P}(\boldsymbol{h}, \boldsymbol{s})), p(\boldsymbol{\psi}', \boldsymbol{P}(\boldsymbol{h}, \boldsymbol{s}))) = \langle \boldsymbol{\psi}, \boldsymbol{\psi}' \rangle$. As variance $\mathbb{V}_{\boldsymbol{h},\boldsymbol{s}}(p(\boldsymbol{\psi}), p(\boldsymbol{\psi}')) \leq \frac{1}{d'} \left(\langle \boldsymbol{\psi}, \boldsymbol{\psi}' \rangle^2 + \|\boldsymbol{\psi}\|_2^2 \|\boldsymbol{\psi}'\|_2^2 \right)$, we note that larger sketches are less noisy. Thus, for every modality we compress, we use a separate sketch matrix $\boldsymbol{P}$. As video modalities are partially dependent, this implicitly leverages the unbiased estimator and reduces the variance.*

Proof. For the first and second property, see Appendix A of paper [71] and Lemma 3 [49]. □

4. Approach

Our pipeline is illustrated in Figure 1. It consist of (i) the Fisher Vector and Bag-of-Words hallucinating streams denoted as FV and BoW (shown in dashed red), respectively,

(ii) the High Abstraction Features stream denoted as HAF, and (iii) the Prediction Network abbreviated as PredNet.

The role of BoW/FV streams is to take I3D intermediate representations generated from the RGB and optical flow frames and learn to hallucinate BoW/FV representations. For this purpose, we use the MSE loss between the ground-truth BoW/FV and the outputs of BoW/FV streams. The role of the HAF stream is to further process I3D intermediate representations before they are concatenated with hallucinated BoW/FV. PredNet fuses the concatenated BoW/FV/HAF and learns class concepts. Figure 2 shows our pipeline for hallucinating the OFF representation (I3D optical flow). Below, we describe each module in detail.

4.1. BoW/FV Hallucinating Streams

BoW/FV take as input the I3D intermediate representations $\mathcal{X}_{(rgb)}$ and $\mathcal{X}_{(opt.)}$ of size $1024{\times}7$ which were obtained by stripping the classifier and the last 1D conv. layer of I3D pre-trained on Kinetics-400. The latter dimension of $\mathcal{X}_{(rgb)}$ and $\mathcal{X}_{(opt.)}$ can be thought of as the temporal size. We concatenate $\mathcal{X}_{(rgb)}$ and $\mathcal{X}_{(opt.)}$ along the third mode and obtain $\mathcal{X}$ which has dimensionality $1024{\times}7{\times}2$. As FV contains the first- and second-order statistics, we use a separate stream per each type of statistics, and a single stream for BoW. For the practical choice of BoW/FV pipelines, we use either a Fully Connected (FC) unit shown in Figure 3a or a Convolutional (Conv) pipeline in Figure 3b. Thus, we investigate the following hallucinating stream combinations: (i) BoW-FC and FV-FC, (ii) BoW-Conv and FV-FC, or (iii) BoW-Conv and FV-Conv. Where indicated, we also equip each stream with Power Normalization (PN). For specific PN realizations, we investigate AsinhE, SigmE, and AxMin variants from Remarks 1, 2 and 3. Below we detail how we obtained ground-truth BoW/FV.

Ground-truth BoW/FV. To train Fisher Vectors, we computed 256 dimensional GMM-based dictionaries on descriptors resulting from IDT [66] according to steps described in Sections 2 and 3.1. We applied PCA to trajectories (30 dim.), HOG (96 dim.), HOF (108 dim.), MBHx (96 dim.) and MBHy (96 dim.), and we obtained the final 213 dim. local descriptors. We applied encoding as in Eq. (2) and (3), the aggregation from Section 3.2 and Power Normalization from Section 3.3. Thus, our encoded first- and second-order FV representations, each of size $256 \times 213 = 54528$, had to be sketched to 1000 dimensions. To this end, we followed Section 3.4, prepared matrices $\boldsymbol{P}_{(fv1)}$ and $\boldsymbol{P}_{(fv2)}$ as in Proposition 2, and fixed both of them throughout experiments. The sketched first- and second-order representations $\psi'_{(fv1)} = \boldsymbol{P}_{(fv1)}\psi_{(fv1)}$ and $\psi'_{(fv2)} = \boldsymbol{P}_{(fv2)}\psi_{(fv2)}$ can be readily combined next with the MSE loss functions detailed in Section 4.5.

For BoW, we followed Section 3.1 and applied k-means to build a 1000 dim. dictionary from the same descriptors which were employed to pre-compute FV. Then, the descriptors were encoded according to Eq. (1), aggregated according to steps described in Section 3.2 and normalized by Power Normalization from Section 3.3. Where indicated, we used 4000 dim. dictionary and thus applied sketching on such BoW to limit its vector size to 1000 dim.

We note that we use ground-truth BoW/FV descriptors only at the training stage to train our hallucination streams.

4.2. High Abstraction Features

High Abstraction Features (HAF) take as input the I3D intermediate representations $\mathcal{X}_{(rgb)}$ and $\mathcal{X}_{(opt.)}$. Practical realizations of HAF pipelines are identical to those of BoW/FV/OFF. Thus, we have a choice of either FC or Conv units illustrated in Figures 3a and 3b. We simply refer to those variants as HAF-FC and HAF-Conv, respectively. Similar to BoW/FV/OFF streams, the HAF representation also uses Power Normalization and it is of size 1000.

4.3. Optical Flow Features

For pipeline in Figure 2, the I3D intermediate representation $\mathcal{X}_{(rgb)}$ only is fed to hallucination/HAF streams. I3D Optical Flow Features $\mathcal{X}_{(opt.)}$ are pre-computed as the training ground-truth for the OFF layer (the MSE loss is used).

4.4. Combining Hallucinated BoW/FV/OFF and HAF

Figure 1 indicates that FV (first- and second-order), BoW and HAF feature vectors $\tilde{\boldsymbol{\psi}}_{(fv1)}$, $\tilde{\boldsymbol{\psi}}_{(fv2)}$, $\tilde{\boldsymbol{\psi}}_{(bow)}$ and $\boldsymbol{\psi}_{(haf)}$ are concatenated (via operator $\oplus$) to obtain $\boldsymbol{\psi}_{(tot)}$ and subsequently sketched (if indicated so during experiments), that is, $\boldsymbol{\psi}'_{(tot)} = \boldsymbol{P}_{(tot)}\boldsymbol{\psi}_{(tot)}$ which reduces the size of the total representation from $d = 4000$ to $500 \leq d' \leq 2000$. Matrix $\boldsymbol{P}_{(tot)}$ is prepared according to Proposition 2 and fixed throughout experiments. As sketching is a linear projection, we can backpropagate through it with ease. When also hallucinating OFF as in Figure 2, we additionally concatenate $\boldsymbol{\psi}_{(off)}$ with other feature vectors to obtain $\boldsymbol{\psi}_{(tot)}$.

PredNet. The final unit of our overall pipeline, PredNet, is illustrated in Figure 3c. On input, we take $\boldsymbol{\psi}_{(tot)}$ (no sketching) or ($\boldsymbol{\psi}'_{(tot)}$) if sketching is used, pass it via the batch normalization and then an FC layer which produces a C dim. representation passed to the cross-entropy loss.

4.5. Objective and its Optimization

During training, we combine MSE loss functions responsible for training hallucination streams with the class. loss:

$$\ell^*(\boldsymbol{\mathcal{X}}, \mathbf{y}; \bar{\boldsymbol{\Theta}}) = \frac{\alpha}{|\mathcal{H}|} \sum_{i \in \mathcal{H}} \left\|\tilde{\boldsymbol{\psi}}_i - \boldsymbol{\psi}'_i\right\|_2^2 + \ell\Big(f(\boldsymbol{\psi}'_{(tot)}; \boldsymbol{\Theta}_{(pr)}), \mathbf{y}; \boldsymbol{\Theta}_{(\ell)}\Big),$$

$$\text{where: } \forall i \in \mathcal{H}, \tilde{\boldsymbol{\psi}}_i = g(\hbar(\boldsymbol{\mathcal{X}}, \boldsymbol{\Theta}_i), \eta), \boldsymbol{\psi}'_i = \boldsymbol{P}_i \boldsymbol{\psi}_i,$$
$$\boldsymbol{\psi}_{(haf)} = g\big(\hbar(\boldsymbol{\mathcal{X}}, \boldsymbol{\Theta}_{(haf)}), \eta\big),$$
$$\boldsymbol{\psi}'_{(tot)} = \boldsymbol{P}_{(tot)} \Big[\oplus_{i \in \mathcal{H}} \tilde{\boldsymbol{\psi}}_i; \boldsymbol{\psi}_{(haf)}\Big]. \quad (8)$$

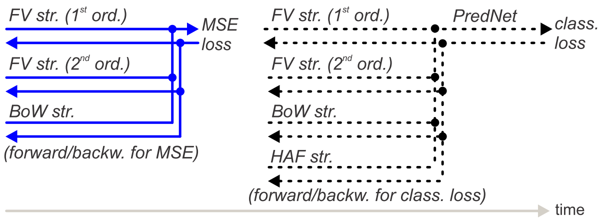

Figure 4: Optimization. In each step, we have (i) forward/backward passes via BoW/FV (optionally OFF) streams for the MSE loss followed by (ii) forward/backward passes via BoW/FV (opt. OFF), and HAF streams and PredNet for the classification loss.

The above equation is a trade-off between the MSE loss functions $\{\|\tilde{\psi}_i - \psi'_i\|_2^2, i \in \mathcal{H}\}$ and the classification loss $\ell(\cdot, \mathbf{y}; \boldsymbol{\Theta}_{(\ell)})$ with some label $\mathbf{y} \in \mathcal{Y}$ and parameters $\boldsymbol{\Theta}_{(\ell)} \equiv \{\boldsymbol{W}, \mathbf{b}\}$. The trade-off is controlled by a constant $\alpha \geq 0$ while MSE is computed over hallucination streams $i \in \mathcal{H}$, and $\mathcal{H} \equiv \{(fv1), (fv2), (bow), (off)\}$ is our set of hallucination streams which can be modified to multiple/few such streams depending on the task at hand. Moreover, $g(\cdot, \eta)$ is a Power Normalizing function chosen from the family described in Section 3.3, $f(\cdot; \boldsymbol{\Theta}_{(pr)})$ is the PredNet module with parameters $\boldsymbol{\Theta}_{(pr)}$ which we learn, $\{\hbar(\cdot, \boldsymbol{\Theta}_i), i \in \mathcal{H}\}$ are the hallucination streams while $\{\tilde{\psi}_i, i \in \mathcal{H}\}$ are the corresponding hallucinated BoW/FV/OFF representations. Moreover, $\hbar(\cdot, \boldsymbol{\Theta}_{(haf)})$ is the HAF stream with the output denoted by $\psi_{(haf)}$. For the hallucination streams, we learn parameters $\{\boldsymbol{\Theta}_i, i \in \mathcal{H}\}$ while for HAF, we learn $\boldsymbol{\Theta}_{(haf)}$. The full set of parameters we learn is defined as $\bar{\boldsymbol{\Theta}} \equiv (\{\boldsymbol{\Theta}_i, i \in \mathcal{H}\}, \boldsymbol{\Theta}_{(haf)}, \boldsymbol{\Theta}_{(pr)}, \boldsymbol{\Theta}_{(\ell)})$. Furthermore, $\{\boldsymbol{P}_i, i \in \mathcal{H}\}$ are the projection matrices for count sketching of the ground-truth BoW/FV/OFF feature vectors $\{\psi_i, i \in \mathcal{H}\}$ while $\{\psi'_i, i \in \mathcal{H}\}$ are the corresponding sketched/compressed representations. Finally, $\boldsymbol{P}_{(tot)}$ is the projection matrix for hallucinated BoW/FV/OFF representations concatenated with each other and HAF, that is, for $\psi_{(tot)} = \left[\oplus_{i \in \mathcal{H}} \tilde{\psi}_i; \psi_{(haf)}\right]$ which results in the sketched counterpart $\psi'_{(tot)}$ that goes into the PredNet module f. Section 3.4 details how to select matrices $\boldsymbol{P}$. If sketching is not needed, we simply set a given $\boldsymbol{P}$ to be the identity projection $\boldsymbol{P} = \mathbb{I}$. In our experiments, we simply set $\alpha = 1$.

Optimization. We minimize $\ell^*(\boldsymbol{\mathcal{X}}, \mathbf{y}; \bar{\boldsymbol{\Theta}})$ w.r.t. parameters of each stream, that is $\{\boldsymbol{\Theta}_i, i \in \mathcal{H}\}$ for hallucination streams, $\boldsymbol{\Theta}_{(haf)}$ for the HAF stream, $\boldsymbol{\Theta}_{(pr)}$ for PredNet and $\boldsymbol{\Theta}_{(\ell)}$ for the classification loss. In practice, we perform a simple alternation over two minimization steps shown in Figure 4. In each iteration, we perform one forward and backward pass regarding the MSE losses to update the parameters $\{\boldsymbol{\Theta}_i, i \in \mathcal{H}\}$ of the hallucination streams. Then, we perform one forward and backward pass regarding the classification loss ℓ. We update all network streams during this pass. Thus, one can think of our network as multi-task learning with BoW/FV/OFF and label learning tasks.

	sp1	*sp2*	*sp3*	mean acc.
HAF only	81.83%	80.78%	80.45%	81.02%
HAF+BoW/FV exact	83.00%	82.80%	81.70%	**82.50%**
HAF+BoW halluc.	82.29%	81.24%	80.98%	81.50%
HAF+FV halluc.	82.68%	81.05%	79.93%	81.22%
HAF+BoW/FV halluc.	82.88%	82.74%	81.50%	**82.37%**

Table 1: Evaluations of pipelines on the HMDB-51 dataset. We compare (*HAF only*) and (*HAF+BoW/FV exact*) which show the lower- and upper bound on the accuracy, and our (*HAF+BoW/FV halluc.*), (*HAF+BoW halluc.*) and (*HAF+FV halluc.*).

Furthermore, we use the Adam minimizer with 10^{-4} initial learning rate which we halve every 10 epochs. We run our training for 50–100 epochs depending on the dataset.

Sketching the Power Normalized vectors.

Proposition 3. *Sketching PN vectors increases the sketching variance (ℓ_2-normalized by vec. norms) by* $1 \leq \kappa \leq 2$.

Proof. Normalize variance $\mathbb{V}$ from Remark 4 by the norms $\|\psi\|_2^2 \|\psi'\|_2^2$. Consider $\mathbb{V}^{(\gamma)}$ which is the variance for d dimensional vectors $\{(\psi^\gamma, \psi'^\gamma) : \psi \geq 0, \psi' \geq 0\}$ power normalized by Gamma from Remark 1, and divide it accordingly by $\|\psi^\gamma\|_2^2 \|\psi'^\gamma\|_2^2$. For extreme PN ($\gamma \to 0$), we have:

$$\lim_{\gamma \to 0} \mathbb{V}^{(\gamma)} = \frac{1}{d'} \lim_{\gamma \to 0} \left(\frac{\langle \psi^\gamma, \psi'^\gamma \rangle^2}{\|\psi^\gamma\|_2^2 \|\psi'^\gamma\|_2^2} + 1 \right) = \frac{2}{d'}. \tag{9}$$

Now, assume that d dimensional ψ and ψ' are actually ℓ_2-norm normalized. Then, we have the following ratio of variances:

$$\kappa = \mathbb{V} / \mathbb{V}^{(\gamma)} = 2 / (\langle \psi, \psi' \rangle^2 + 1), \tag{10}$$

Thus, $1 \leq \kappa \leq 2$ depends on (ψ, ψ'), and κ varies smoothly between $[1; 2]$ for $1 \leq \gamma \leq 0$ of Gamma, a monotonically increasing function. For typical $\gamma = 0.5$, we measured for the actual data that $\kappa \approx 1.3$. □

5. Experiments

5.1. Datasets and Evaluation Protocols

HMDB-51 [41] consists of 6766 internet videos over 51 classes; each video has ∼20–1000 frames. Following the protocol, we report the mean accuracy across three splits.

YUP++ [19] dataset contains so-called video textures. It has 20 scene classes, 60 videos per class, and its splits contain scenes captured with the static or moving camera. We follow the standard splits (1/9 dataset for training).

MPII Cooking Activities [52] consist of high-resolution videos of people cooking various dishes. The 64 distinct 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 Faster-RCNN [50] to crop video around humans.

Charades [55] consist of of 9848 videos of daily indoors activities, 66500 temporal annotations and 157 classes.

	static	*dynamic*	*mixed*	mean acc.
HAF only	92.03%	81.67%	89.07%	87.59%
HAF+BoW/FV exact	93.30%	89.82%	92.41%	91.84%
HAF+BoW halluc.	92.69%	85.93%	92.41%	90.34%
HAF+FV halluc.	92.69%	88.15%	91.48%	90.77%
HAF+BoW/FV halluc.	93.15%	89.63%	92.31%	**91.69%**

Table 2: Eval. of pipelines on YUP++. See Table 1 for the legend.

	sp1	*sp2*	*sp3*	mean acc.
HAF-Conv+BoW/FV-FC halluc.	81.96%	80.39%	80.52%	80.95%
HAF-FC+BoW/FV-Conv halluc.	82.42%	81.30%	81.50%	81.74%
HAF-FC+BoW/FV-FC halluc.	82.88%	82.74%	81.50%	**82.37%**

Table 3: Evaluations of pipelines on the HMDB-51 dataset. We compare (*HAF+BoW/FV halluc.*) approach on different architectures used for HAF and BoW/FV streams such as (*FC*) and (*Conv*).

5.2. Evaluations

We start our experiments by investigating various aspects of our pipeline and then we present our final results.

HAF, BoW and FV streams. Firstly, we ascertain the gains from our HAF and BoW/FV streams. We evaluate the performance of (i) the HAF-only baseline pipeline without IDT-based BoW/FV information (*HAF only*), (ii) the HAF-only baseline with exact ground-truth IDT-based BoW/FV added at both training and testing time (*HAF+BoW/FV exact*), and (iii) the combined HAF plus IDT-based BoW/FV streams (*HAF+BoW/FV halluc.*). We also perform evaluations on (iv) HAF plus IDT-based BoW stream (*HAF+BoW halluc.*) and HAF plus IDT-based FV stream (*HAF+FV halluc.*) to examine how much gain IDT-based BoW and FV bring, respectively. As Section 4.1 suggests that each stream can be based on either the Fully Connected (FC) or Convolutional (Conv.) pipeline, we firstly investigate the use of FC unit from Figure 3a, that is, we use HAF-FC, BoW-FC and HAF-FC streams. PredNet also uses FC. For ground-truth FV, we use 1000 dim. sketches.

Table 1 presents results on the HMDB-51 dataset. As expected, the (*HAF only*) is the poorest performer while (*HAF+BoW/FV exact*) is the best performer determining the upper limit on the accuracy. Hallucinating (*HAF+BoW halluc.*) outperforms (*HAF+FV halluc.*) marginally. We expect FV to perform close to BoW due to the significant compression with sketching by factor $\sim 52.5\times$. Approaches (*HAF+FV/BoW halluc.*) and (*HAF+BoW/FV exact*) achieve the best results, and outperform (*HAF only*) by 1.35% and 1.48% accuracy. These result show that our hallucination strategy (*HAF+FV/BoW halluc.*) can mimic (*HAF+BoW/FV exact*) closely. Our 82.37% accuracy is the new state of the art. Below we show larger gains on YUP++.

Table 2 presents similar findings on the YUP++ dataset. Our (*HAF+FV halluc.*) brings the improvement of ~ 2.2 and $\sim 6.5\%$ over (*HAF+BoW halluc.*) and (*HAF only*) on scenes captured with the moving camera (*dynamic*). Our (*HAF+BoW/FV halluc.*) yields $\sim 8.0\%$ over (*HAF only*) thus demonstrating again the benefit of hallucinating BoW/FV descriptors. The total gain for (*HAF+BoW/FV halluc.*) over (*HAF only*) equals 4.1%. Our (*HAF+FV/BoW halluc.*) matches results of (*HAF+BoW/FV exact*) without explicitly computing BoW/FV during testing. Below, we investigate different architectures of our streams.

Fully Connected/Convolutional streams. Figures 3a and 3b show two possible realizations of HAF, BoW and FV streams. While FC and Conv. architectures are not the only possibilities, they are the simplest ones. Table 3 shows that using FC layers (*FC*) for HAF and BoW/FV streams, denoted as (*HAF-FC+BoW/FV-FC halluc.*) outperforms Convolutional (*Conv*) variants by up to $\sim 1.5\%$ accuracy. Thus, we use only the FC architecture in what follows.

Sketching and Power Normalization. As PredNet uses FC layers (see Figure 3c), we expect that limiting the input size to this layer via count sketching from Section 3.4 should benefit the performance. Moreover, as visual and video representations suffer from so-called burstiness, we investigate AsinhE, SigmE and AxMin from Remarks 1, 2 and 3.

Figure 5a investigates the classification accuracy on the HMDB-51 dataset (split 1) when our HAF and BoW/FV feature vectors $\{\tilde{\psi}_i, i \in \mathcal{H}\}$ and $\psi_{(haf)}$ (described in Sections 4.4 and 4.5) are passed via Power Normalizing functions AsinhE, SigmE and AxMin prior to concatenation (see Figure 1). From our experiment it appears that all PN functions perform similarly and improve results from the baseline 82.29% to $\sim$ 83.20% accuracy. We observe a similar gain from 93.15% to 94.44% acc. on YUP++ (*static*). In what follows, we simply use AsinhE for PN.

Figure 5b illustrates on the HMDB-51 dataset (split 1) that applying count sketching on concatenated HAF and BoW/FV feature vectors $\psi_{(tot)}$, which produces $\psi'_{(tot)}$ (see Section 4.5 for reference to symbols), improves results from 82.88% to 83.92% accuracy for $d' = 2000$. This is expected as reduced size of $\psi'_{(tot)}$ results in fewer parameters of the FC layer of PredNet and less overfitting. Similarly, for the YUP++ dataset and the split (*static*), we see the performance increase from 93.15% to 94.81% accuracy.

Comparisons with other methods. Below we present our final results and we contrast them against the state of the art. Table 4 shows results on the HMDB-51 dataset. For

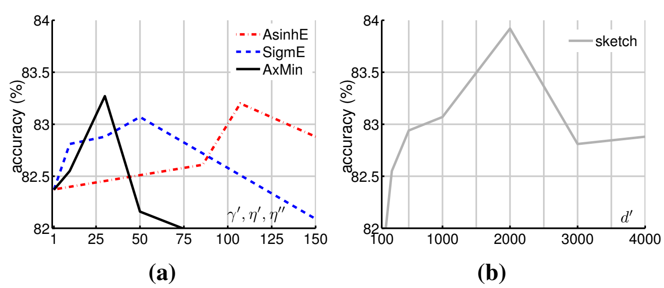

Figure 5: Evaluations of (fig. 5a) Power Normalization and (fig. 5b) sketching on the HMDB-51 dataset (split 1 only).

	sp1	*sp2*	*sp3*	mean acc.
HAF only	81.83%	80.78%	80.45%	81.02%
HAF+BoW/FV halluc.	83.46%	82.61%	81.37%	**82.48%**

ADL+ResNet+IDT 74.3% [67]	STM Network+IDT 72.2% [18]
ADL+I3D 81.5% [67]	Full-FT I3D 81.3% [4]

Table 4: Evaluations of (*top*) our (*HAF+BoW/FV halluc.*) and (*bottom*) comparisons to the state of the art on HMDB-51.

	static	*dynamic*	*mixed*	mean stat/dyn	mean all
HAF only	92.03%	81.67%	89.07%	86.8%	87.6%
HAF+BoW/FV halluc.	94.81%	**89.63%**	**93.33%**	**92.2%**	**92.6%**
T-ResNet [19]	92.41%	81.50%	89.00%	87.0%	87.6%
ADL I3D [67]	**95.10%**	88.30%	-	91.7%	-

Table 5: Evaluations of (*top*) our (*HAF+BoW/FV halluc.*) and (*bottom*) comparisons to the state of the art on YUP++.

	sp1	*sp2*	*sp3*	*sp4*	*sp5*	*sp6*	*sp7*	mAP
HAF+BoW halluc.	73.9	71.6	76.2	70.7	76.3	71.9	63.4	71.9%
HAF+BoW halluc.+SK/PN	73.9	75.8	72.2	73.9	77.0	73.6	68.8	**73.6%**
HAF* only	74.6	73.2	77.0	75.1	76.1	75.6	71.9	74.8%
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$^\bullet$+BoW hal.+MSK/PN	80.8	80.9	85.0	83.9	82.0	79.8	79.6	**81.7%**
ditto+OFF hal.	81.2	81.2	84.9	83.4	84.2	78.9	79.1	**81.8%**
I3D+BoW MTL$^\bullet$	79.1	78.1	83.6	78.7	79.1	78.6	76.5	79.1%

KRP-FS 70.0% [9]	KRP-FS+IDT 76.1% [9]	GRP 68.4% [7]	GRP+IDT 75.5% [7]

Table 6: Evaluations of (*top*) our (*HAF+BoW halluc.*) pipeline without sketching/PN, with sketching/PN (*SK/PN*). The (*HAF* only*) is our baseline without the BoW stream, (*) denotes human-centric pre-processing while (*MSK/PN*)in pipeline (*HAF*+BoW hal.+MSK/PN*) denotes multiple sketches per BoW followed by Power Norm (*PN*). (*bottom*) Other methods on the MPII dataset.

HAF only	HAF+BoW/FV exact	HAF+BoW/FV/OFF halluc. +MSK×2/PN	HAF+BoW/FV/OFF halluc. +MSK×4/PN	HAF+BoW/FV/OFF halluc. +MSK×8/PN
37.2	41.9	42.0	42.2	**43.1**

Table 7: Evaluations of our methods on the Charades dataset.

our method, we used sketching of $\psi_{(tot)}$ with $d'=2000$ and PN. Our (*HAF+BoW/FV halluc.*) model yields 82.48% acc. which beats results in the literature to the best of our knowledge. If we tune PN per split, our results reach 82.78% accuracy. However, we do not advise such tuning due to danger of overfitting. We note that we outperform more complex methods such as Adversarial Discriminative Learning (ADL) with I3D [67] and Fully Fine-Tuned I3D [4].

Table 5 shows results on the YUP++ dataset. Our (*HAF+BoW/FV halluc.*) model yields very competitive results on the static protocol and outperforms competitors on the dynamic and mixed protocols. With 92.2% mean accuracy over static and dynamic scores (*mean stat/dyn*), we outperform more complex ADL+I3D [67] and T-ResNet [19].

Table 6 shows results for the MPII dataset for which we use HAF with/without the BoW (4000 dim.) hallucination stream (no FV stream). As MPII contains subsequences, we use integral pooling from Prop. 1. Our basic model (*HAF+BoW halluc.*) scores $\sim 71.9\%$ mAP. Applying sketching and PN (*HAF+BoW halluc.+SK/PN*) yields 73.6% mAP. Unlike GRP+IDT [7] and KRP-FS+IDT [9], our first two experiments do not use any human- or motion-centric pre-processing. With human-centric crops, denoted with (*), our baseline without BoW (*HAF* only*) achieves 74.8% mAP. The model with BoW (*HAF+BoW halluc.*) scores 77.8% mAP. By utilizing 4 sketches for BoW and 4 BoW streams with Power Normalization (*HAF*+BoW hal.+MSK/PN*), we obtain 80.4% mAP.

Hallucinating Optical Flow. For (*HAF$^\bullet$+BoW hal.+MSK/PN*) in Table 6, we increased the resolution of RGB frames $2\times$ to obtain larger human-centric crops and $2\times$ larger optical flow res., which yielded 81.7% mAP. In the same setting, hallucinating optical flow feat. (*ditto+OFF hal.*) yielded 81.84% mAP, the new state of the art.

Charades. In Table 7, baselines (*HAF only*) and (*HAF+BoW/FV exact*) score 37.2% and 41.9% mAP. Moreover, our best pipeline (*HAF+BoW/FV/OFF halluc.+MSK×8/PN*) that hallucinates IDT BoW/FV and I3D optical flow features (OFF) with 8 sketches per BoW/FV/OFF and PN yielded 43.1% (a much more complex feature banks [74] yield 43.4%). Finally, if 25% of this dataset was dedicated to testing, ∼55h of computations would be saved.

Discussion. There exist several reasons explaining why our pipeline works well *e.g.*, sophisticated IDT trajectory modeling is unlikely to be captured by off-the-shelf CNNs unless a CNN is enforced to learn IDT. We perform translation of the I3D output into IDT-based BoW/FV descriptors thus enforcing I3D to implicitly learn IDT which co-regularizes I3D which resembles Domain Adaptation (DA) methods: a source network co-regularizes a target network [34, 58, 35, 25, 24, 42] by the alignment of feature statistic of both streams. Related to DA is Multi-task Learning (MTL) known for boosting generalization/preventing overfitting of CNNs due to task specific losses [5]. MTL training on related tasks is known to boost individual task accuracies beyond a vanilla feature fusion [59]. Finally, our pipeline uses self-supervision *e.g.*, IDT BoW/FV and OFF descriptors represent easy to obtain self-information about videos. We train our halluc./last I3D layers via task-specific losses (similar to MTL). However, our halluc. layers distill the domain specific cues which are fed back into the network (PredNet) which boosts our results by further ∼2.7% compared to vanilla (*I3D+BoW MTL$^\bullet$*) in Table 6.

6. Conclusions

We have proposed a simple yet powerful strategy that learns IDT-based descriptors (and even optical flow features) and hallucinates them in a CNN pipeline for AR at the test time. With state-of-the-art results, we hope our method will spark a renewed interest in IDT-like descriptors.

References

[1] Luis Alvarez, Joachim Weickert, and Javier Sánchez. Reliable estimation of dense optical flow fields with large displacements. *IJCV*, 39(1):41–56, Aug. 2000. 3

[2] Jim Braux-Zin, Romain Dupont, and Adrien Bartoli. A general dense image matching framework combining direct and feature-based costs. In *ICCV*, pages 185–192, 2013. 3

[3] Thomas Brox and Jitendra Malik. Large displacement optical flow: Descriptor matching in variational motion estimation. *TPAMI*, 33(3):500–513, Mar. 2011. 3

[4] João Carreira and Andrew Zisserman. Quo Vadis, Action Recognition? A New Model and the Kinetics Dataset. *CVPR*, pages 1–10, 2018. 1, 3, 8

[5] Rich Caruana. Multitask learning. *Machine Learning*, 28(1):41–75, July 1997. 8

[6] Bhaskar Chakraborty, Michael B. Holte, Thomas B. Moeslund, and Jordi Gonzàlez. Selective spatio-temporal interest points. *CVIU*, 116(3):396–410, 2012. 2

[7] Anoop Cherian, Basura Fernando, Mehrtash Harandi, and Stephen Gould. Generalized rank pooling for action recognition. In *CVPR*, 2017. 6, 8

[8] Anoop Cherian, Piotr Koniusz, and Stephen Gould. Higher-order pooling of CNN features via kernel linearization for action recognition. In *WACV*, pages 130–138, 2017. 1, 2, 3

[9] Anoop Cherian, Suvrit Sra, Stephen Gould, and Richard Hartley. Non-linear temporal subspace representations for activity recognition. In *CVPR*, pages 2197–2206, 2018. 1, 3, 6, 8

[10] Vasileios Choutas, Philippe Weinzaepfel, Jérôme Revaud, and Cordelia Schmid. PoTion: Pose motion representation for action recognition. In *CVPR*, pages 7024–7033, 2018. 1, 2

[11] Graham Cormode and Marios Hadjieleftheriou. Finding frequent items in data streams. *Proc. VLDB Endow.*, 1(2):1530–1541, Aug. 2008. 4

[12] Gabriella Csurka, Christopher R. Dance, Lixin Fan, Jutta Willamowski, and Cédric Bray. Visual categorization with bags of keypoints. *ECCV Workshop*, pages 1–22, 2004. 1, 2, 3

[13] Navneet Dalal, Bill Triggs, and Cordelia Schmid. Human Detection Using Oriented Histogram of Flow and Appearance. *ECCV*, pages 428–441, 2006. 1, 2

[14] Piotr Dollár, Vincent Rabaud, Garrison Cottrell, and Serge Belongie. Behavior recognition via sparse spatio-temporal features. In *Proceedings of the 14th International Conference on Computer Communications and Networks*, pages 65–72, 2005. 2

[15] Jeff Donahue, Lisa Anne Hendricks, Sergio Guadarrama, Marcus Rohrbach, Subhashini Venugopalan, Trevor Darrell, and Kate Saenko. Long-term recurrent convolutional networks for visual recognition and description. In *CVPR*, pages 2625–2634, 2015. 3

[16] Pengfei Fang, Jieming Zhou, Soumava Kumar Roy, Lars Petersson, and Mehrtash Harandi. Bilinear attention networks for person retrieval. In *ICCV*, 2019. 3

[17] Christoph Feichtenhofer, Axel Pinz, and Richard P. Wildes. Spatiotemporal residual networks for video action recognition. In *NIPS*, pages 3468–3476, 2016. 1, 3

[18] Christoph Feichtenhofer, Axel Pinz, and Richard P. Wildes. Spatiotemporal multiplier networks for video action recognition. In *CVPR*, 2017. 8

[19] Christoph Feichtenhofer, Axel Pinz, and Richard P. Wildes. Temporal residual networks for dynamic scene recognition. In *CVPR*, 2017. 6, 8

[20] Basura Fernando, Efstratios Gavves, José Oramas M., Amir Ghodrati, and Tinne Tuytelaars. Modeling video evolution for action recognition. In *CVPR*, pages 5378–5387, 2015. 3

[21] Basura Fernando and Stephen Gould. Learning end-to-end video classification with rank-pooling. In *ICML*, volume 48, pages 1187–1196, 2016. 1, 3

[22] William T. Freeman and Michal Roth. Orientation histograms for hand gesture recognition. Technical Report TR94-03, MERL - Mitsubishi Electric Research Laboratories, Cambridge, MA 02139, Dec. 1994. 1, 2

[23] Steffen Gauglitz, Tobias Höllerer, and Matthew Turk. Evaluation of interest point detectors and feature descriptors for visual tracking. *IJCV*, 94(3):335, Mar 2011. 2

[24] Mehrtash Harandi, Mathieu Salzmann, and Richard Hartley. Dimensionality reduction on SPD manifolds: The emergence of geometry-aware methods. *TPAMI*, 2018. 8

[25] Samitha Herath, Mehrtash Harandi, Basura Fernando, and Richard Nock. Min-max statistical alignment for transfer learning. *CVPR*, 2019. 8

[26] Berthold K. P. Horn and Brian G. Schunck. Determining optical flow. *Artificial Intelligence*, 17:185–203, 1981. 3

[27] Ian M. L. Hunter. Book review: Thinking in perspective: Critical essays in the study of thought processes. *Quarterly Journal of Experimental Psychology*, 32(2):358–359, 1980. 3

[28] Hervé Jégou, Matthijs Douze, and Cordelia Schmid. On the Burstiness of Visual Elements. *CVPR*, pages 1169–1176, 2009. 3

[29] Shuiwang Ji, Wei Xu, Ming Yang, and Kai Yu. 3D convolutional neural networks for human action recognition. *TPAMI*, 35:221–231, 2010. 3

[30] Andrej Karpathy, George Toderici, Sanketh Shetty, Thomas Leung, Rahul Sukthankar, and Li Fei-Fei. Large-scale video classification with convolutional neural networks. In *CVPR*, pages 1725–1732, 2014. 3

[31] Alexander Kläser, Marcin Marszalek, and Cordelia Schmid. A Spatio-Temporal Descriptor Based on 3D-Gradients. *BMCV*, pages 1–10, 2008. 1, 2

[32] Piotr Koniusz, Anoop Cherian, and Fatih Porikli. Tensor representations via kernel linearization for action recognition from 3D skeletons. *ECCV*, 2016. 3

[33] Piotr Koniusz and Krystian Mikolajczyk. Soft Assignment of Visual Words as Linear Coordinate Coding and Optimisation of its Reconstruction Error. *ICIP*, 2011. 2

[34] Piotr Koniusz, Yusuf Tas, and Fatih Porikli. Domain adaptation by mixture of alignments of second- or higher-order scatter tensors. *CVPR*, 2017. 8

[35] Piotr Koniusz, Yusuf Tas, Hongguang Zhang, Mehrtash Harandi, Fatih Porikli, and Rui Zhang. Museum exhibit identification challenge for the supervised domain adaptation and beyond. In *ECCV*, 2018. 8

[36] Piotr Koniusz, Fei Yan, Philippe-Henri Gosselin, and Krystian Mikolajczyk. Higher-order Occurrence Pooling on Mid- and Low-level Features: Visual Concept Detection. *Technical Report*, 2013. 3

[37] Piotr Koniusz, Fei Yan, Philippe-Henri Gosselin, and Krystian Mikolajczyk. Higher-order occurrence pooling for bags-of-words: Visual concept detection. *TPAMI*, 39(2):313–326, 2017. 1, 3
[38] Piotr Koniusz, Fei Yan, and Krystian Mikolajczyk. Comparison of Mid-Level Feature Coding Approaches And Pooling Strategies in Visual Concept Detection. *CVIU*, 2012. 1, 2, 3
[39] Piotr Koniusz, Hongguang Zhang, and Fatih Porikli. A deeper look at power normalizations. In *CVPR*, pages 5774–5783, 2018. 1, 3, 4
[40] Alex Krizhevsky, Ilya Sutskever, and Geoffrey E. Hinton. ImageNet classification with deep convolutional neural networks. *NIPS*, pages 1106–1114, 2012. 3
[41] Hildegard Kuehne, Hueihan Jhuang, Estíbaliz Garrote, Tomaso Poggio, and Thomas Serre. HMDB: A large video database for human motion recognition. In *ICCV*, pages 2556–2563, 2011. 6
[42] Soumava Kumar Roy, Mehrtash Harandi, Richard Nock, and Richard Hartley. Siamese networks: The tale of two manifolds. In *ICCV*, 2019. 8
[43] Ivan Laptev. On space-time interest points. *IJCV*, 64(2-3):107–123, Sept. 2005. 2
[44] Chuanzhen Li, Bailiang Su, Jingling Wang, and Qin Zhang. Human action recognition using multi-velocity STIPs and motion energy orientation histogram. *J. Inf. Sci. Eng.*, 30:295–312, 2014. 2
[45] Liu Lingqiao, Lei Wang, and Xinwang Liu. In Defence of Soft-assignment Coding. *ICCV*, 2011. 2
[46] Nils Papenberg, Andrés Bruhn, Thomas Brox, Stephan Didas, and Joachim Weickert. Highly accurate optic flow computation with theoretically justified warping. *IJCV*, 67:141–158, 2006. 3
[47] Florent Perronnin and Christopher Dance. Fisher kernels on visual vocabularies for image categorization. *CVPR*, 0:1–8, 2007. 1, 2, 3
[48] Florent Perronnin, Jorge Sánchez, and Thomas Mensink. Improving the Fisher Kernel for Large-Scale Image Classification. *ECCV*, pages 143–156, 2010. 1, 2, 3
[49] Ninh Pham and Rasmus Pagh. Fast and scalable polynomial kernels via explicit feature maps. In *Proceedings of the 19th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, pages 239–247. ACM, 2013. 1, 4
[50] Shaoqing Ren, Kaiming He, Ross Girshick, and Jian Sun. Faster R-CNN: Towards real-time object detection with region proposal networks. In *NIPS*, pages 91–99, 2015. 6
[51] Jérôme Revaud, Philippe Weinzaepfel, Zaid Harchaoui, and Cordelia Schmid. EpicFlow: Edge-Preserving Interpolation of Correspondences for Optical Flow. In *CVPR*, 2015. 3
[52] Marcus Rohrbach, Sikandar Amin, Mykhaylo Andriluka, and Bernt Schiele. A database for fine grained activity detection of cooking activities. In *CVPR*, 2012. 6
[53] Olga Russakovsky, Jia Deng, Hao Su, Jonathan Krause, Sanjeev Satheesh, Sean Ma, Zhiheng Huang, Andrej Karpathy, Aditya Khosla, Michael Bernstein, Alexander C. Berg, and Li Fei-Fei. ImageNet large scale visual recognition challenge. *IJCV*, 115(3):211–252, 2015. 3
[54] Paul Scovanner, Saad Ali, and Mubarak Shah. A 3-Dimentional SIFT Descriptor and its Application to Action Recognition. *CRCV*, pages 1–4, 2007. 1, 2
[55] Gunnar A. Sigurdsson, Gül Varol, Xiaolong Wang, Ali Farhadi, Ivan Laptev, and Abhinav Gupta. Hollywood in homes: Crowdsourcing data collection for activity understanding. In *ECCV*, 2016. 6
[56] Karen Simonyan and Andrew Zisserman. Two-stream convolutional networks for action recognition in videos. In *NIPS*, pages 568–576, 2014. 1, 3
[57] Josef Sivic and Andrew Zisserman. Video Google: A text retrieval approach to object matching in videos. *ICCV*, 2:1470–1477, 2003. 1, 2, 3
[58] Yusuf Tas and Piotr Koniusz. CNN-based action recognition and supervised domain adaptation on 3D body skeletons via kernel feature maps. *BMVC*, 2018. 8
[59] Sebastian Thrun. Is learning the n-th thing any easier than learning the first? *NIPS*, pages 640–646, 1996. 8
[60] Du Tran, Lubomir Bourdev, Rob Fergus, Lorenzo Torresani, and Manohar Paluri. Learning Spatiotemporal Features with 3D Convolutional Networks. *ICCV*, pages 4489–4497, 2015. 1, 3
[61] Jasper R.R. Uijlings, Ionut Cosmin Duta, Negar Rostamzadeh, and Nicu Sebe. Realtime Video Classification using Dense HOF/HOG. *ICMR*, 2014. 2
[62] Jan C. van Gemert, Cor J. Veenman, Arnold W. M. Smeulders, and Jan-Mark Geusebroek. Visual word ambiguity. *TPAMI*, 32(7):1271–1283, July 2010. 2
[63] Gül Varol, Ivan Laptev, and Cordelia Schmid. Long-term temporal convolutions for action recognition. *TPAMI*, 40(6):1510–1517, 2018. 3
[64] Heng Wang, Alexander Kläser, Cordelia Schmid, and Liu Cheng-Lin. Action Recognition by Dense Trajectories. *CVPR*, pages 3169–3176, 2011. 1, 2
[65] Heng Wang, Alexander Kläser, Cordelia Schmid, and Cheng-Lin Liu. Dense Trajectories and Motion Boundary Descriptors for Action Recognition. *IJCV*, 2013. 1, 2
[66] Heng Wang and Cordelia Schmid. Action Recognition with Improved Trajectories. *ICCV*, pages 3551–3558, 2013. 1, 2, 5
[67] Jue Wang and Anoop Cherian. Learning discriminative video representations using adversarial perturbations. In *ECCV*, pages 716–733, 2018. 1, 3, 8
[68] Lei Wang. 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, Nov 2017. 2
[69] Lei Wang, Du Q. Huynh, and Piotr Koniusz. A comparative review of recent kinect-based action recognition algorithms. *IEEE Transactions on Image Processing*, 2019. 2
[70] Lei Wang, Du Q. Huynh, and Moussa Reda Mansour. Loss switching fusion with similarity search for video classification. *ICIP*, 2019. 2
[71] Kilian Weinberger, Anirban Dasgupta, John Langford, Alex Smola, and Josh Attenberg. Feature hashing for large scale multitask learning. In *ICML*, pages 1113–1120, 2009. 3, 4
[72] Philippe Weinzaepfel, Jérôme Revaud, Zaid Harchaoui, and Cordelia Schmid. DeepFlow: Large displacement optical flow with deep matching. In *ICCV*, 2013. 3
[73] Geert Willems, Tinne Tuytelaars, and Luc Van Gool. An efficient dense and scale-invariant spatio-temporal interest point detector. In *ECCV*, pages 650–663, 2008. 2
[74] Chao-Yuan Wu, Christoph Feichtenhofer, Haoqi Fan, Kaiming He, Philipp Krahenbuhl, and Ross Girshick. Long-term

feature banks for detailed video understanding. In *CVPR*, June 2019. 8
[75] Lahav Yeffet and Lior Wolf. Local trinary patterns for human action recognition. *ICCV*, pages 492–497, 2009. 2