HAL
open science

COARSE-TO-FINE AGGREGATION FOR CROSS-GRANULARITY ACTION RECOGNITION

Ahmed Mazari, Hichem Sahbi

To cite this version:

Ahmed Mazari, Hichem Sahbi. COARSE-TO-FINE AGGREGATION FOR CROSS-GRANULARITY ACTION RECOGNITION. IEEE ICIP, 2020, Abu-Dhabi, United Arab Emirates. hal-03089394

HAL Id: hal-03089394

https://hal.science/hal-03089394v1

Submitted on 28 Dec 2020

HAL is a multi-disciplinary open access archive for the deposit and dissemination of scientific research documents, whether they are published or not. The documents may come from teaching and research institutions in France or abroad, or from public or private research centers.

L'archive ouverte pluridisciplinaire **HAL**, est destinée au dépôt et à la diffusion de documents scientifiques de niveau recherche, publiés ou non, émanant des établissements d'enseignement et de recherche français ou étrangers, des laboratoires publics ou privés.

COARSE-TO-FINE AGGREGATION FOR CROSS-GRANULARITY ACTION RECOGNITION

Ahmed Mazari *Hichem Sahbi*

Sorbonne University, UPMC, CNRS, LIP6, F-75005 Paris, France

ABSTRACT

In this paper, we introduce a novel hierarchical aggregation design that captures different levels of temporal granularity in action recognition. Our design principle is coarse- to-fine and achieved using a tree-structured network; as we traverse this network top-down, pooling operations are getting less invariant but timely more resolute and well localized. Learning the combination of operations in this network — which best fits a given ground-truth – is obtained by solving a constrained minimization problem whose solution corresponds to the distribution of weights that capture the contribution of each level (and thereby temporal granularity) in the global hierarchical pooling process. Besides being principled and well grounded, the proposed hierarchical pooling is also video-length agnostic and resilient to misalignments in actions. Extensive experiments conducted on the challenging UCF-101 database corroborate these statements.

***Index Terms*—** Hierarchical pooling, deep multiple representation learning, action recognition

1. INTRODUCTION

Many applications such as video surveillance [1, 2], scene captioning and understanding [3–10] as well as robotics [11–15] require automatic recognition of human actions. This task is one of the most challenging problems in video analysis which consists in assigning action categories to image sequences. The difficulty of this task stems from the intrinsic properties of actions (human appearance and motion, articulation, velocity, etc.) and also their extrinsic acquisition conditions (camera motion and resolution, illumination, occlusion, cluttered background, etc). Existing action recognition solutions process videos in order to extract (handcrafted or learned) representations [16–22, 36] prior to their classification using shallow [19, 20, 23, 25–28, 42] or deep models [24, 29–32, 32–35]. The latter are particularly powerful in visual recognition [39,40] (and other neighboring fields [37, 38]) and successful methods include 2D/3D two-stream convolutional neural networks (CNNs) [29–31, 41]. This success, which comes at the expense of a substantial increase in the number of training parameters, is tributary to the availability of large *labeled* video datasets that capture all the intrinsic and the extrinsic properties of scenes and actions. However, labeled videos are scarce and existing ones are at least an order of magnitude smaller compared to the datasets used in other related tasks (such as image classification) while action recognition is inherently far more challenging. As a result, deep networks used for action recognition become more exposed to over-fitting.

Deep convolutional networks have nonetheless the ability to attenuate the high dependency on labeled data by introducing pooling (a.k.a aggregation) operators which gradually reduce the dimensionality, the number of training parameters and thereby the risk of over-fitting. However, pooling (such as averaging) may dilute the relevant information especially when action categories exhibit strong variations in their temporal granularity. Indeed, while coarsely-grained actions could still remain easy to discriminate using average pooling, fine-grained ones become more confound; hence, one should design a pooling mechanism which conveys multiple levels of granularity across categories.

In this paper, we introduce a novel hierarchical pooling (aggregation) design that captures different levels of temporal granularity in action recognition. Our design principle is "coarse-to-fine" and achieved using a tree-structured network; as we parse this hierarchy top-down, pooling operations are getting less invariant but timely more resolute and well dedicated to fine-grained action categories. Given a hierarchy of aggregation operations, our goal is to learn weighted (linear and nonlinear) combinations of these pooling operations that best fit a given action recognition ground-truth. We solve this problem by minimizing a constrained objective function whose parameters correspond to the distribution of weights through multiple aggregation levels; each weight measures the contribution of its granularity in the global learned video representation. Besides being able to handle aggregations at different levels, the particularity of our solution resides in its ability to handle misaligned[1] and variable duration videos (without any explicit alignment or up/down-sampling) and thereby makes it possible to fully benefit from the whole frames in videos. Extensive experiments conducted on the challenging UCF-101 benchmark show the validity and the out-performance of our hierarchical aggregation design w.r.t the related work.

[1] misalignments are usually due to imprecise detection and trimming of actions in videos (which is also known to be a cumbersome task when achieved manually and error-prone when achieved automatically [47–51]) and this adds spurious details/context in the analyzed actions.

2. FRAME-WISE DESCRIPTION AT A GLANCE

We consider a collection of videos $\mathcal{S} = \{\mathcal{V}_i\}_{i=1}^n$ with each one being a sequence of frames $\mathcal{V}_i = \{f_{i,t}\}_{t=1}^{T_i}$ and a set of action categories (a.k.a classes or categories) denoted as $\mathcal{C} = \{1, \dots, C\}$. In order to describe the visual content of a given video $\mathcal{V}_i$, we rely on a two-stream process; the latter provides a complete description of appearance and motion that characterizes the spatio-temporal aspects of moving objects and their interactions. The output of the appearance stream (denoted as $\{\phi_a(f_{i,t})\}_{t=1}^{T_i} \subset \mathbb{R}^{2048}$) is based on the deep residual network (ResNet-101) trained on ImageNet [46] and fine-tuned on UCF-101 while the output of the motion stream (denoted as $\{\phi_m(f_{i,t})\}_{t=1}^{T_i} \subset \mathbb{R}^{2048}$) is also based on ResNet-101 but trained on optical flow input frames [41, 46]; in the appearance stream, the number of input channels in the underlying ResNet is kept fixed (equal to 3) while in the motion stream, the number of channels is reset to 20 (instead of 3). When training the latter, the initial weights of these 20 channels are obtained by averaging the 3 original (appearance) channel weights and by replicating their values through the 20 new motion channels. Considering these frame-wise representations, our goal is to introduce an alternative to usual frame aggregation schemes (namely sampling and global average pooling) which instead learns a hierarchical aggregation that models *coarse as well as fine grained action categories*.

3. MULTIPLE AGGREGATION LEARNING

Given a video $\mathcal{V}$ with T frames, we define $\mathcal{N}$ as a tree-structured network with depth up to D levels and width up to 2^{D-1}. Let $\mathcal{N} = \cup_{k,l}\mathcal{N}_{k,l}$ with $\mathcal{N}_{k,l}$ being the kth node of the lth level of $\mathcal{N}$; all nodes belonging to the lth level of $\mathcal{N}$ define a partition of the temporal domain $[0, T]$ into 2^{l-1} equally-sized subdomains. A given node $\mathcal{N}_{k,l}$ in this hierarchy aggregates the frames that belong to its underlying temporal interval. Each node $\mathcal{N}_{k,l}$ also defines an appearance and a motion representation respectively denoted as $\psi^a_{k,l}(\mathcal{V}_i)$, $\psi^m_{k,l}(\mathcal{V}_i)$ and set as $\psi^a_{k,l}(\mathcal{V}_i) = \frac{1}{|\mathcal{N}_{k,l}|}\sum_{t\in\mathcal{N}_{k,l}} \phi_a(f_{i,t})$, $\psi^m_{k,l}(\mathcal{V}_i) = \frac{1}{|\mathcal{N}_{k,l}|}\sum_{t\in\mathcal{N}_{k,l}} \phi_m(f_{i,t})$. Depending on the level in $\mathcal{N}$, each representation captures a particular temporal granularity of motion and appearance into a given scene; it is clear that top-level representations capture coarse visual characteristics of actions while bottom-levels (including leaves) are dedicated to fine-grained and timely-resolute sub-actions. Knowing a priori which levels (and nodes in these levels) capture the best – a given action category – is not trivial. In the remainder of this section, we introduce a novel learning framework which achieves multiple aggregation design and finds the best combination of levels and nodes in these levels that fits different temporal granularities of action categories.

3.1. Multiple aggregation learning

Considering the motion stream, we define – for each node $\mathcal{N}_{k,l}$ – a set of variables $\beta_m = \{\beta^m_{k,l}\}_{k,l}$ (with $\beta^m_{k,l} \in [0, 1]$ and $\sum_{k,l} \beta^m_{k,l} = 1$) which measures the importance (and hence the contribution) of $\psi^m_{k,l}(\mathcal{V})$ in the global motion representation of $\mathcal{V}$ (denoted as $\psi_m(\mathcal{V})$). Precisely, two variants are considered for ψ_m

$$\begin{aligned} (*) \quad \psi_m(\mathcal{V}) &= \left(\beta^m_{1,1}\psi^m_{1,1}(\mathcal{V}) \dots \beta^m_{k,l}\psi^m_{k,l}(\mathcal{V}) \dots\right)^\top \\ (**) \quad \psi_m(\mathcal{V}) &= \sum_{k,l} \beta^m_{k,l}\psi^m_{k,l}(\mathcal{V}). \end{aligned} \tag{1}$$

As shown above, the variant in (*) corresponds to a concatenation scheme while (**) corresponds to averaging; the *former* relies on the hypothesis that nodes in $\mathcal{N}$ (and hence sub-actions in different videos) are well aligned whereas the *latter* relaxes this hypothesis (see later Eq. 2). Similarly to motion, we define the aggregations and the set of variables $\beta_a = \{\beta^a_{k,l}\}_{k,l}$ associated to appearance stream. In the remainder of this paper, and unless explicitly mentioned, the symbols m, a are omitted in the notation and all the subsequent formulation is applicable to motion as well as appearance streams.

In order to weight the impact of nodes in the hierarchy $\mathcal{N}$ and put more emphasis on the most relevant granularity of the learned aggregation, we consider multiple representation learning that generalizes [43, 44] both to linear and nonlinear combinations. Its main idea consists in finding a kernel $\mathcal{K}$ as a combination of positive semi-definite (p.s.d) elementary kernels $\{\kappa(.,.)\}$ associated to $\{\mathcal{N}_{k,l}\}_{k,l}$. Considering the two maps in Eq. (1), we define the two variants of $\mathcal{K}$ as

$$\begin{aligned} \mathcal{K}(\mathcal{V}, \mathcal{V}') &= \sum_l \sum_k \beta_{k,l}\, \kappa(\psi_{k,l}(\mathcal{V}), \psi_{k,l}(\mathcal{V}')) \\ \mathcal{K}(\mathcal{V}, \mathcal{V}') &= \sum_{l,l'} \sum_{k,k'} \beta_{k,l}\beta_{k',l'}\, \kappa(\psi_{k,l}(\mathcal{V}), \psi_{k',l'}(\mathcal{V}')). \end{aligned} \tag{2}$$

As $\beta_{k,l} \in [0, 1]$, the kernel $\mathcal{K}$ is p.s.d resulting from the closure of the p.s.d of κ w.r.t the sum and the product. Let $\mathcal{C} = \{1, \dots, C\}$ be a set of action categories and let $\{(\mathcal{V}_i, y_{ic})\}_i$ be a training set of actions associated to $c \in \mathcal{C}$ with $y_{ic} = +1$ if $\mathcal{V}_i$ belongs to the category c and $y_{ic} = -1$ otherwise. Using $\mathcal{K}$, we train multiple max margin classifiers (denoted $\{g_c\}_{c\in\mathcal{C}}$) whose kernels (in Eq. 2) correspond to level-wise linear (resp. cross-wise nonlinear) combinations of elementary kernels dedicated to $\{\mathcal{N}_{k,l}\}_{k,l}$. A classifier associated to an action category c is given by $g_c(\mathcal{V}) = \sum_i \alpha^c_i y_{ic}\mathcal{K}(\mathcal{V}, \mathcal{V}_i) + b_c$, here b_c is a shift, $\{\alpha^c_i\}_i$ is a set of positive parameters found (together with $\beta = \{\beta_{k,l}\}_{k,l}$) by minimizing the following constrained quadratic programming (QP) problem

$$\begin{aligned} \min_{0\le\beta\le 1, \|\beta\|_1=1, \{\alpha^c_i\}} \quad & \frac{1}{2}\sum_c \sum_{i,j} \alpha^c_i\alpha^c_j y_{ic}y_{jc}\mathcal{K}(\mathcal{V}_i, \mathcal{V}_j) - \sum_i \alpha^c_i \\ \text{s.t.} \quad & \alpha^c_i \ge 0, \quad \sum_i y_{ic}\alpha^c_i = 0, \quad \forall i, c. \end{aligned} \tag{3}$$

As the problem in Eq. 3 is not convex w.r.t β, $\{\alpha_i^c\}$ taken jointly and convex when taken separately, an EM-like iterative optimization procedure can be used: first, parameters in β are fixed and the above problem is solved w.r.t $\{\alpha_i^c\}$ using QP, then $\{\alpha_i^c\}$ are fixed and the resulting problem is solved w.r.t β using either linear programming for (*) and QP for (**). This iterative process stops when the values of all these parameters remain unchanged or when it reaches a maximum number of iterations. However, in spite of being relatively effective (see later Table 1), this EM-like procedure is computationally expensive as it requires solving multiple instances of constrained quadratic problems[2] and the number of necessary iterations to reach convergence is large in practice.

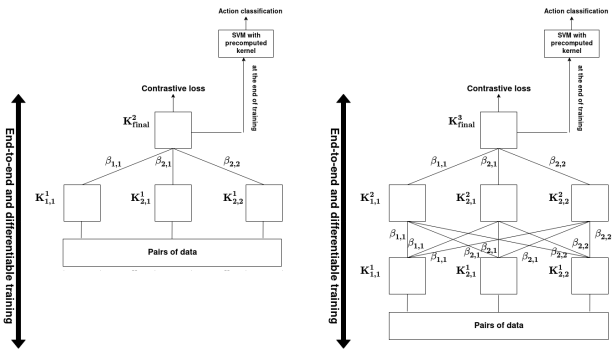

Fig. 1. Examples of networks used to train a 2-level hierarchical aggregation with "concatenation" (left) and "averaging" (right). These two networks correspond to the two equations in (2); their inputs correspond to the elementary kernels evaluated on pairwise nodes in $\mathcal{N}$: constrained to be aligned for "concatenation" and unconstrained for "averaging" **(Better to zoom the PDF version).**

3.2. Deep contrastive loss design

In what follows, we consider a procedure that decouples the learning of β from $\{\alpha_i^c\}$ resulting into more efficient and also still effective training process. In this procedure, we first model the kernels in Eq. 2 using two networks (see Fig. 1), and we learn their parameters using a contrastive loss criterion (that benefits from larger training data pairs), then we plug the resulting $\mathcal{K}$ into Eq. 3 in order to learn the parameters $\{\alpha_i^c\}$ in one step. We consider an end-to-end framework which learns the parameters β of these networks (that capture the importance of nodes in the hierarchical aggregation) by minimizing

$$\min_{0\leq\beta\leq 1, \|\beta\|_1=1} \quad E(\beta, \mathcal{S}, \mathcal{K}, Y), \tag{4}$$

here E models the disagreement between the predicted kernel values on video pairs $\{\mathcal{K}(\mathcal{V}_i, \mathcal{V}_j)\}_{\mathcal{V}_i,\mathcal{V}_j\in\mathcal{S}}$ and their ground-truth $\{Y(\mathcal{V}_i, \mathcal{V}_j)\}_{\mathcal{V}_i,\mathcal{V}_j\in\mathcal{S}}$ with $Y(\mathcal{V}_i, \mathcal{V}_j) = +1$ iff $\mathcal{V}_i$ and $\mathcal{V}_j$ belong to the same class and $Y(\mathcal{V}_i, \mathcal{V}_j) = -1$ otherwise. This objective function can still be minimized using gradient descent and back-propagation. However, some constraints should be carefully tackled; indeed, whereas the forward step can be achieved, gradient back-propagation (through our multiple aggregation shown mainly in Fig. 1-right) should be achieved while sharing parameters in the same layers and across layers. Besides, constraints on $\beta's$ should also be handled.

[2] whose complexity scales quadratically w.r.t the size of training data and the number of nodes in the hierarchy $\mathcal{N}$.

3.3. Constraint implementation

Considering $\frac{\partial E}{\partial \mathcal{K}}$ available, the gradients $\frac{\partial E}{\partial \beta}$ cannot be straightforwardly obtained using a direct application of the chain rule (as already available in PyTorsh); on the one hand, any step following the gradient $\frac{\partial E}{\partial \beta}$ should preserve equality and inequality constraints in Eq. (4) while a direct application of the chain rule provides us with a surrogate gradient which ignores these constraints. On the other hand, as the parameters β are shared across layers (when using "averaging" in Fig. 1), this requires a careful update of $\frac{\partial E}{\partial \beta}$ as shown subsequently.

In order to implement the equality and inequality constraints in Eq. 4, we consider a re-parametrization as $\beta_{k,l} = h(\hat{\beta}_{k,l})/\sum_{k',l'} h(\hat{\beta}_{k',l'})$ for some $\{\hat{\beta}_{k,l}\}_{k,l}$ with h being strictly monotonic real-valued (positive) function and this allows free settings of the parameters $\{\hat{\beta}_{k,l}\}_{k,l}$ during optimization while guaranteeing $\beta_{k,l} \in [0, 1]$ and $\sum_{k,l} \beta_{k,l} = 1$. During back-propagation, the gradient of the loss E (now w.r.t $\hat{\beta}$'s) is updated using the chain rule as

$$\begin{aligned} \frac{\partial E}{\partial \hat{\beta}_{k,l}} &= \sum_{p,q} \frac{\partial E}{\partial \beta_{p,q}} . \frac{\partial \beta_{p,q}}{\partial \hat{\beta}_{k,l}} \\ \text{with} \quad & \frac{\partial \beta_{p,q}}{\partial \hat{\beta}_{k,l}} = \frac{h'(\hat{\beta}_{k,l})}{\sum_{k',l'} h(\hat{\beta}_{k',l'})} . (\delta_{p,q,k,l} - \beta_{p,q}), \end{aligned} \tag{5}$$

and $\delta_{p,q,k,l} = 1_{\{(p,q)=(k,l)\}}$. In practice $h(.) = \exp(.)$ and $\frac{\partial E}{\partial \beta_{p,q}}$ is obtained from layerwise gradient backpropagation (as already integrated in standard deep learning tools including PyTorch). Hence, $\frac{\partial E}{\partial \hat{\beta}_{k,l}}$ is obtained by multiplying the original gradient $\left[\frac{\partial E}{\partial \beta_{p,q}}\right]_{p,q}$ by the Jacobian $\left[\frac{\partial \beta_{p,q}}{\partial \hat{\beta}_{k,l}}\right]_{p,q,k,l}$ which simply reduces to $\left[\beta_{k,l}(\delta_{p,q,k,l} - \beta_{p,q})\right]_{p,q,k,l}$ when $h(.) = \exp(.)$.

As the parameters $\{\hat{\beta}_{k,l}\}_{k,l}$ are not totally independent across layers (see again Fig. 1-right), we consider a further step that accumulates (averages) the gradients $\{\frac{\partial E}{\partial \hat{\beta}_{k,l}}\}_{k,l}$ with shared indices and replaces these gradients by the averaged ones. It is easy to see that these accumulated (shared) gradients (when used to update $\hat{\beta}$'s using gradient descent) also preserve the equality and inequality constraints in Eq. 4.

4. EXPERIMENTS

We evaluate the performance of our action recognition method on the challenging UCF-101 (split-2) dataset [45]. The latter includes 13,320 videos belonging to 101 action categories of variable duration, cluttered background and

misaligned content[3]. As discussed previously, we first extract 2D two-stream frame-wise representations, then we combine them using our hierarchical aggregation design prior to achieve action recognition. We follow the exact protocol in [45] to evaluate and compare our method w.r.t different settings as well as the related work.

Settings. Different settings are considered in order to assess the performance of our method: i) multiple depths of our hierarchical aggregation network ranging from 2 to 6, ii) two streams (motion and appearance) as well as their fusion, and iii) the two types of aggregations namely "concatenation" and "averaging". In order to learn the weights of our hierarchical aggregations for all the aforementioned settings, we conducted experiments using both the EM-like procedure as well as the deep multiple kernel learning (DMKL) shown in section 3. In the latter, we achieve DMKL for 4,000 iterations using PyTorch Adam optimizer[4] and we set the learning rate to 0.0005 and the batch-size to 2048. As already discussed, we use a contrastive loss for DMKL and we plug the resulting kernel into multi-class SVMs for training and testing; given a test video, its category corresponds to the SVM with the highest score.

	Depth (D)	Appearance	Motion	Fusion
Concatenat.	2	82.78	80.12	89.49
	3	82.91	80.59	89.68
	4	83.04	80.73	89.72
	5	**83.17**	**80.80**	**89.76**
	6	82.76	80.62	89.63
Averaging	2	82.96	80.53	89.67
	3	83.16	80.78	89.74
	4	83.28	81.00	89.87
	5	83.36	81.00	89.89
	6	**83.36**	**81.07**	**89.91**

Table 1. This table shows the behavior of our multiple aggregation learning using the EM procedure w.r.t the depth of the network $\mathcal{N}$. These results are reported for both motion and appearance streams as well as their combination and also for concatenation and averaging (note that RBF is used as an elementary kernel for DMKL). The drop in the performances of the "concatenation" scheme (from $D = 5$ to $D = 6$, i.e., the most resolute nodes) is mainly due to the sensitivity of "concatenation" to misalignments in the most resolute nodes of $\mathcal{N}$ while "averaging" enhances the performances steadily.

Performances and comparison. Table 1 shows the performances of the different configurations (described earlier); from these results, we observe a consistent gain as the depth of our hierarchy increases with an advantage of "averaging" w.r.t "concatenation". This gain is observed on both motion and appearance streams with a significant leap when fusing them. These gains also reflect the importance of node crossbreeding ("averaging" vs. "concatenation") especially when videos are subject to cluttered context and when their actions are misaligned as frequently observed in the UCF-101.

We also show (in Table. 2) a comparison of our hierarchical aggregation against two other aggregation methods: global average pooling and also spectrograms [24]; the *former* produces a global representation that averages all the frame descriptions while the *latter* keeps all the frame representations and concatenate them (as an image) prior to their classification using 2D CNNs [24]. Note that these two comparative methods are interesting as they correspond to two extreme cases of our hierarchy, namely the root and the leaf levels; in particular, the spectrogram (of a video $\mathcal{V}$ with T frames) is obtained when the number of leaf nodes, in the hierarchy $\mathcal{N}$, is exactly equal to T (see again [24]). We also compare our method against another aggregation method based on colorized heatmaps [32] as a variant of the global average pooling; these heatmaps correspond to timely-stamped and averaged frame-wise probability distributions of human keypoints. Finally, we compare the classification performances of our method against two closely related 2D CNN action recognition works: 2D two-streams CNNs in [29] and [46] (respectively based on VGG and ResNet) as well as the method in [52]. From these results, we observe a consistent gain of our hierarchical aggregation design w.r.t these related methods.

Methods	Appearance	Motion	Fusion
Our HA+C (EM)	82.76	80.62	89.63
Our HA+C (DMKL)	82.82	80.69	89.66
Our HA+A (EM)	83.36	81.07	89.91
Our HA+A (DMKL)	**83.44**	81.17	**89.95**
GAP in [24]	66.15	✗	✗
Spectrogram [24]	64.41	✗	✗
Colorized heatmaps [32]	✗	64.38	✗
C3D [52]	82.3	✗	✗
Temporal Pyramid [24]	68.58	✗	✗
2D 2-stream VGG [29]	73	**83.7**	86.9
2D 2-stream ResNet [46]	82.1	79.4	88.5

Table 2. Comparison w.r.t state-of-the-art methods. In this table HA, C, A, GAP stand for "Hierarchical Aggregation", "Concatenation", "Averaging" and "Global Average Pooling" respectively. Note that our HA results are obtained with $D = 6$ and RBF is used for both EM and DMKL settings. In the related work, the symbol "✗" means that the configuration either "does not apply" or "not tested" in the related paper.

5. CONCLUSION

In this paper, we introduced a hierarchical aggregation design for cross-granularity action recognition. Our method is based on the minimization of a constrained objective function whose solution corresponds to the distribution of weights in a hierarchy of pooling operations that best fits the granularity of action categories. Besides being able to handle videos with multiple granularities, the strength of our method resides also in its ability to handle videos with variable duration and misalignment. Experiments conducted on UCF-101 dataset show the validity of our approach *w.r.t* the related work. As future work, we are currently investigating the extension of our hierarchical crossbreeding aggregation method in order to handle longer videos as a part of the more challenging problem of activity recognition.

[3] Many actions are misaligned as their videos are endowed with large context while others are precisely trimmed and contain only the actions of interest

[4] We run experiments on single GPU; GeForce RTX 2080 Ti (with 11 GB).

6. REFERENCES

[1] A. Ben Mabrouk, E. Zagrouba. Abnormal behavior recognition for intelligent video surveillance systems: A review. In Expert Systems with Applications Volume 91, Pages 480-491, 2018

[2] Y. Han, P. Zhanga, T. Zhuob, W. Huang, Y. Zhanga. Going deeper with two-stream ConvNets for action recognition in video surveillance. In PRL Volume 107, Pages 83-90, 2018

[3] B. Wang, L. Ma, W. Zhang, W. Liu. Reconstruction network for video captioning. In IEEE CVPR, 2018

[4] J. Wang, W. Jiang, L. Ma, W. Liu, Y. Xu. Bidirectional attentive fusion with context gating for dense video captioning. In IEEE CVPR, 2018

[5] N. Aafaq, N. Akhtar, W. Liu, S. Zulqarnain Gilani, A. Mian. Spatio-Temporal Dynamics and Semantic Attribute Enriched Visual Encoding for Video Captioning. In CVPR, 2019

[6] Minlong Lu, Ze-Nian Li, Yueming Wang, Gang Pan. Deep Attention Network for Egocentric Action Recognition. In IEEE TIP, Volume 28, Issue 8, 2019

[7] T. Mahmud, M. Billah, M. Hasan, Am. K. Roy-Chowdhury. Captioning Near-Future Activity Sequences. In arXiv:1908.00943, 2019

[8] T. Bagautdinov, A. Alahi, F. Fleuret, P. Fua, S. Savarese. Social Scene Understanding: End-To-End Multi-Person Action Localization and Collective Activity Recognition. In CVPR, 2017

[9] J. Shao, K. Kang, C. Change Loy, X. Wang. Deeply Learned Attributes for Crowded Scene Understanding. In CVPR, 2015

[10] M. Pantic, A. Pentland, A. Nijholt, T.S. Huang. Human Computing and Machine Understanding of Human Behavior: A Survey. In Human Computing and Machine Understanding of Human Behavior, 2007

[11] H. Meng, N. Pears, C. Bailey. A Human Action Recognition System for Embedded Computer Vision Application. In CVPR, 2007

[12] T. Theodoridis, A. Agapitos, H. Hu, S.M. Lucas. Ubiquitous robotics in physical human action recognition: A comparison between dynamic ANNs and GP. In IEEE ICRA, 2008

[13] Y. Demiris. Prediction of intent in robotics and multi-agent systems. Cogn Proc (2007) 8: 151. https://doi.org/10.1007/s10339-007-0168-9

[14] M. Nan, A. Stefania Ghi, A. Gavril, M. Trascau, A. Sorici, B. Cramariuc, A. Magda Florea. Human Action Recognition for Social Robots. In Int Conf on Control Systems and Computer Science, 2019

[15] E. Coupet, F. Moutarde, S. Manitsaris. Multi-users online recognition of technical gestures for natural humanrobot collaboration in manufacturing. Robot (2019) 43: 1309

[16] W. Lu and James J. Little. Simultaneous tracking and action recognition using the pca-hog descriptor. In ECCV, 2006

[17] I. Laptev. On Space-Time Interest Points. In IJCV, Volume 64, Issue 23, pp 107123, 2005

[18] B. K.P.Horn, B. G.Schunck. Determining optical flow. Artificial Intelligence, Volume 17, Issues 13, Pages 185-203, 1981

[19] H. Wang, A. Klaser, C. Schmid. Action Recognition by Dense Trajectories. In CVPR, 2011

[20] H. Wang, C. Schmid. Action Recognition with Improved Trajectories. In ICCV, 2013

[21] G. Csurka, C. R. Dance, L. Fan, J. Willamowski, C. Bray. Visual Categorization with Bags of Keypoints. In ECCV, 2004

[22] G. Csurka, F. Perronnin. Fisher Vectors : Beyond Bag-of-Visual-Words Image Representations. In International Conference on Computer Vision, Imaging and Computer Graphics, 2010

[23] S. Lazebnik, C. Schmid, J. Ponce. Beyond Bags of Features: Spatial Pyramid Matching for Recognizing Natural Scene Categories. In CVPR, 2006

[24] A. Mazari, H. Sahbi. Deep Temporal Pyramid Design for Action Recognition. In ICASSP, 2019

[25] H. Pirsiavash, D. Ramanan. Detecting Activities of Daily Living in First-person Camera Views. In CVPR, 2012

[26] L. Chen, L. Duan, D. Xu. Event Recognition in Videos by Learning From Heterogeneous Web Sources. In CVPR, 2013

[27] H. Wang, C. Yuan, W. Hu, C. Sun. Supervised class-specific dictionary learning for sparse modeling in action recognition. PR, Volume 45, Issue 11, Pages 3902-3911, 2012

[28] C. Schuldt, I. Laptev, B. Caputo. Recognizing human actions: a local SVM approach. In ICPR, 2004

[29] K. Simonyan, A. Zisserman. Two-Stream Convolutional Networks for Action Recognition in Videos. In NeurIPS, 2014

[30] C. Feichtenhofer, A. Pinz, A. Zisserman. Convolutional Two-Stream Network Fusion for Video Action Recognition. In CVPR, 2016

[31] J. Carreira, A. Zisserman. Quo Vadis, Action Recognition? A New Model and the Kinetics Dataset. In CVPR, 2017

[32] V. Choutas, P. Weinzaepfel, J. Revaud, C. Schmid. PoTion: Pose MoTion Representation for Action Recognition. In CVPR, 2018

[33] C. Feichtenhofer, A. Pinz, R-P. Wildes. Spatiotemporal Residual Networks for Video Action Recognition. In NeurIPS, 2016

[34] C. Feichtenhofer, A. Pinz, R-P. Wildes. Spatiotemporal Multiplier Networks for Video Action Recognition. In CVPR, 2017

[35] C. Feichtenhofer, A. Pinz, R-P. Wildes, Andrew Zisserman. What have we learned from deep representations for action recog? In CVPR, 2018

[36] M. Attique khan, M. Sharif, T. Akram, M. Raza, T. Saba, A. Rehmane. Hand-crafted and deep convolutional neural network features fusion and selection strategy: An application to intelligent human action recognition. In Applied Soft Computing, Volume 87, February 2020, 105986

[37] A. Graves, A. Mohamed, G. Hinton. Speech recognition with deep recurrent neural networks. In ICASSP, 2013

[38] G.y Hinton, L. Deng, D. Yu, G. Dahl, A. Mohamed, N. Jaitly, A. Senior, V. Vanhoucke, P. Nguyen, T. Sainath and B. Kingsbury. Deep Neural Networks for Acoustic Modeling in Speech Recognition. In IEEE Signal Processing Magazine, Vol 29: pp. 82-97, 2012

[39] K. He, X. Zhang, S. Ren, J. Sun. Delving Deep into Rectifiers: Surpassing Human-Level Performance on ImageNet Class. In ICCV, 2015

[40] T. Xiao, Y. Xu, K. Yang, J. Zhang, Y. Peng, Z. Zhang. The Application of Two-Level Attention Models in Deep Convolutional Neural Network for Fine-Grained Image Classification. In CVPR, 2015

[41] L. Wang, Y. Xiong, Z. Wang, Y. Qiao, D. Lin, X. Tang, L. Van Gool. Temporal Segment Networks: Towards Good Practices for Deep Action Recognition. In ECCV, 2016

[42] D. Xu, S-F. Chang. Visual Event Recognition in News Video using Kernel Methods with Multi-Level Temporal Alig. In IEEE CVPR, 2007

[43] M. Gnen, E. Alpaydn. Multiple Kernel Learning Algorithms. In JMLR : 2211-2268, 2011

[44] C.cortes, M. Mohri, A. Rostamizadeh. Algorithms for learning Kernels based on Centered Alignement. In JMLR : 795-828, 2012

[45] K. Soomro, A-R. Zamir and M. Shah. UCF101: A Dataset of 101 Human Action Classes From Videos in The Wild, CRCV-TR-12-01, 2012.

[46] J. Yihuang. Pretrained 2D two streams network for action recognition on UCF-101 based on temporal segment network. https://github.com/jeffreyyihuang/two-stream-action-recognition, 2017

[47] Pointly-Supervised Action Localization. In IJCV, Volume 127, Issue 3, 263281, 2019

[48] W. Xu, Z. Miao, J. Yu, Q. Ji. Action recognition and localization with spatial and temporal contexts. Neurocomputing Vol 333, 351-363, 2019

[49] H. Zhao, A. Torralba, L. Torresani, Z. Yan. HACS: Human Action Clips and Segments Dataset for Recognition and Temporal Localization. In ICCV, 2019

[50] P. Weinzaepfel, Z. Harchaoui, C. Schmid. Learning to track for spatio-temporal action localization. In ICCV, 2015

[51] G. Yu, J. Yuan. Fast Action Proposals for Human Action Detection and Search. In CVPR, 2015

[52] D. Tran, L. Bourdev, R. Fergus, L. Torresani, M. Paluri . Learning Spatiotemporal Features with 3D Convolutional Networks. In ICCV, 2015