Temporally-Aware Feature Pooling for Action Spotting in Soccer Broadcasts

Silvio Giancola Bernard Ghanem
King Abdullah University of Science and Technology, Saudi Arabia
{silvio.giancola,bernard.ghanem}@kaust.edu.sa

Abstract

Toward the goal of automatic production for sports broadcasts, a paramount task consists in understanding the high-level semantic information of the game in play. For instance, recognizing and localizing the main actions of the game would allow producers to adapt and automatize the broadcast production, focusing on the important details of the game and maximizing the spectator engagement. In this paper, we focus our analysis on action spotting in soccer broadcast, which consists in temporally localizing the main actions in a soccer game. To that end, we propose a novel feature pooling method based on NetVLAD, dubbed ***NetVLAD++****, that embeds temporally-aware knowledge. Different from previous pooling methods that consider the temporal context as a single set to pool from, we split the context* before *and* after *an action occurs. We argue that considering the contextual information around the action spot as a single entity leads to a sub-optimal learning for the pooling module. With NetVLAD++, we disentangle the context from the* past *and* future *frames and learn specific vocabularies of semantics for each subsets, avoiding to blend and blur such vocabulary in time. Injecting such prior knowledge creates more informative pooling modules and more discriminative pooled features, leading into a better understanding of the actions. We train and evaluate our methodology on the recent large-scale dataset SoccerNet-v2, reaching* 53.4% *Average-mAP for action spotting, a* $+12.7\%$ *improvement w.r.t the current state-of-the-art.*

1. Introduction

The volume of sports TV broadcast available worldwide increased at a fast pace over the last years. The amount of hours of sports TV broadcasted in the United States from 2002 to 2017 has grown $>4\times$ [11], with similar trends in European countries [13, 14]. Consequently, the market size for the sports media rights is booming [12], revealing a ludicrous market estimated to worth >25B USD by 2023 in US alone. Yet, creating broadcast contents still requires a tremendous manual effort from the producers who could heavily benefit from automated processes.

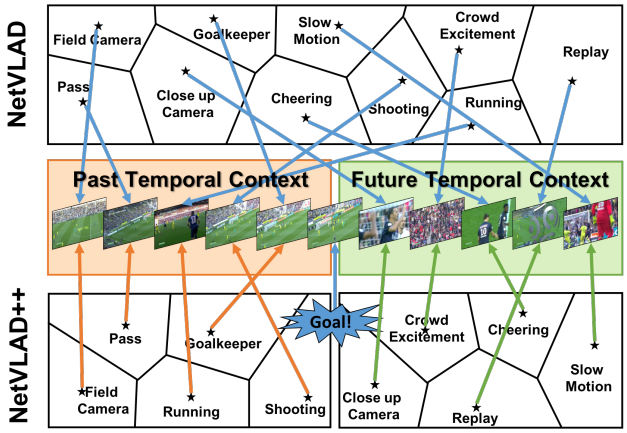

Figure 1. **NetVLAD** (top) *vs*. **NetVLAD++** (bottom) pooling modules for action spotting. Our temporally-aware NetVLAD++ pooling module learns specific vocabularies for the *past* and *future* semantics around the action to spot.

Autonomous broadcast production requires an understanding of the sports it focuses on. In particular, it needs to be aware of *where* to look at, *e.g*. focusing the camera on spatial areas of the field, but also *when* to look at, *e.g*. focusing on a given actions of interest temporally anchored in the broadcast. While most literature focuses on *where* to look at, less effort were put on *when* to look at. As an example, player or ball detection and tracking algorithms reach excellent performances [28, 34, 8, 4] and are commonly used as priors to identify *where* to look at, by simply regressing the extrinsic parameters of the cameras to center those objects [36]. Yet, the level of semantics involved in this task is rather low and does not require higher understanding of the game. On the other hand, identifying *when* to look at requires understanding higher semantics level, closer to the game, focusing on abstract action concepts rather than well defined actor's or object's priors. To that end, we believe that understanding actions rather than actors is a more challenging task, yet to explore in sports videos.

In this work, we propose to tackle the task of action spotting, *i.e*. localizing well-defined actions in time, an-

chored with single timestamps along a video. Such task is commonly solved by looking at a temporal context around a given timestamp and regressing the actionness in time. The class-aware actionness is then reduced to a singularly anchored spot using non maximum suppression techniques [19]. Previous works on that realm consider temporal pooling techniques [19, 41], spatio-temporal encoding [32], multi-tower temporal CNN [42] or context-aware regression modules [7]. Inspired by those related works, we propose a temporally-aware pooling technique that push forward the previous state-of-the-art. In particular, we developed a pooling module that consider the near *past* and *future* context around the action, independently. Our novel temporal module, dubbed **NetVLAD++**, is based on two NetVLAD pooling layer across the frames *before* and *after* the action occurs, respectively. Such temporal awareness brings a significant boost in the performances in the SoccerNet-v2 benchmark, leading into state-of-the-art performances for action spotting.

Contributions. We summarize them as follow: **(i)** We introduce NetVLAD++, a novel pooling module for action spotting that learns a temporally-aware vocabulary for past and future temporal context. **(ii)** We implement a more efficient architecture for action spotting, in term of memory and computational complexity, leading to state-of-the-art performances for action spotting on SoccerNet-v2. **(iii)** We propose a comprehensive ablation that points out the contributions of each architectural block design.

2. Related Work

Computer Vision in Soccer. The literature in soccer-related computer vision mainly focuses on low-level understanding of a soccer broadcast [29], *e.g*. localizing a field and its lines [9, 17, 23], detecting players [8, 44], their motion [18, 28], their pose [5, 46], their team [24], the ball [34, 35] or a pass feasibility [33]. Understanding frame-wise information is useful to enhance the visual experience of sports viewers [30] and to gather player statistics [36], but it falls short of higher-level game understanding needed for automatic editing purposes. With the appearance of large scale datasets such as SoccerNet [19], Yu *et al*. [45] and SoccerDB [26], higher level tasks started to appear. SoccerNet [19] introduced the task of action spotting, *i.e*. localizing every action with its timestamp in a large corpus of TV broadcasts. They introduced a dataset of 500 games from the European leagues, annotated with 6637 actions of goals, cards and substitutions. Yu *et al*. [45] released a novel dataset of 222 broadcast videos of 45 min each. They introduced interesting annotations of camera shots, players position, events and stories, yet do not provide any task nor baseline on how to use those annotations. SoccerDB [26] merged a subset of 270 games from SoccerNet with 76 soccer games from the Chinese Super League. They proposed several tasks, ranging from object detection, action recognition, temporal action localization and replay segmentation. Lastly, SoccerNet-v2 [10] extended SoccerNet [19] with more than $300k$ extra annotations and propose novel tasks that would support the automatic production of soccer broadcast. In particular, SoccerNet-v2 [10] extended the task of action spotting to 17 classes to understand the fine-grained details of a soccer game. They also introduced two novel tasks: camera shot segmentation for broadcast editing purposes and replay grounding for highlight and summarization purposes. In this work, we leverage the fine-grained annotations from SoccerNet-v2 [10] and compete in the task of action spotting.

Action spotting. Action spotting was introduced in SoccerNet [19] and defined as the localization of a instantaneous event anchored with a single timestamp, namely an *action*, in contrast with *activities*, defined with a start and an end [22]. It draws similarities with the concept of action completion [21] where an action is defined with a single anchor in time, but serve a different purpose of predicting the future completion of that action. Giancola *et al*. [19] introduced a first baseline on SoccerNet [19] based on different pooling techniques. Yet, their code is hardly optimized, leading into an strong under-estimation of the pooling performances for action spotting. Vanderplaetse *et al*. [41] later improved that baseline by merging visual and audio features in a multi-modal pooling approach. Rongved *et al*. [32] trained a 3D ResNet encoder [39] directly from the video frames. Although the performances were far from the baseline, mostly due to the difficulty of training an encoder from scratch, the technical prowess lied in training end-to-end for action spotting with 16 V100 GPU combining 512GB of memory. Vats *et al*. [42] leveraged a multi-tower CNN to process information at various temporal scales to account for the uncertainty of the action locations. Cioppa *et al*. [7] proposed a method based on a context-aware loss function that model the temporal context surrounding the actions. They propose an alternative approach that predicts multiple spots from each chunk of video, by regressing multiple temporal offsets for the actions. Most recently, Tomei *et al*. [37] introduced a regression and masking approach with RMS-Net, inspired by common detection pipeline [31] and self-supervised pre-training [16]. It is worth noting that they reach impressive performance gains by fine tuning the last ResNET block of the video frame encoder. In our work, we improve the original temporal pooling mechanism proposed in SoccerNet [19] by introducing temporally-aware bag-of-words pooling modules. Unlike Tomei *et al*. [37], we refrain from fine-tuning the pre-extracted ResNET frame features for a fair comparison with the related work, but simply allow for a learnable projection (similar to PCA) to reduce the feature dimensionality.

3. Methodology

In this section, we first recall the definition of NetVLAD and propose a more computationally efficient implementation (3.1), we present our novel temporally-aware NetVLAD pooling module learning the *past* and *future* temporal context independently (3.2) and its implementation in a more comprehensive pipeline for action spotting (3.3).

3.1. Recall on NetVLAD

NetVLAD [2] is a differentiable pooling technique inspired by VLAD [25]. In particular, VLAD learns clusters of features descriptors and defines an aggregation of feature as the average displacement of each features with respect to the center of its closer cluster. NetVLAD generalizes VLAD by **(i)** softening the assignment for full-differentiable capability, and **(ii)** disentangling the definition of the cluster and the assignment of the samples.

VLAD. Formally, given a set of N D-dimensional features $\{\mathbf{x}_i\}_{i=1..N}$ as input, a set of K clusters centers $\{\mathbf{c}_k\}_{k=1..K}$ with same dimension D as VLAD parameters, the output of the VLAD descriptor V is defined by:

$$V(j,k) = \sum_{i=1}^{N} a_k(\mathbf{x}_i)(\mathbf{x}_i(j) - \mathbf{c}_k(j)) \tag{1}$$

where $\mathbf{x}_i(j)$ and $\mathbf{c}_k(j)$ are respectively the j-th dimensions of the i-th descriptor and k-th cluster center. $a_k(\mathbf{x}_i)$ denotes the hard assignment of the sample $\mathbf{x}_i$ from its closer center, *i.e.* $a_k(\mathbf{x}_i) = 1$ if $\mathbf{c}_k$ is the closest center of $\mathbf{x}_i$, 0 otherwise. The matrix V is then L2-normalized at the cluster level, flatten into a vector of length $D \times K$ and further L2-normalized globally.

NetVLAD. The VLAD module is non-differentiable due to the hard assignment $a_k(\mathbf{x}_i)$ of the samples $\{\mathbf{x}_i\}_{i=1}^N$ to the clusters $\{\mathbf{c}_k\}_{i=1}^K$. Those hard-assignment creates discontinuities in the feature space between the clusters, impeding gradients to flow properly. To circumvent this issue, NetVLAD [2] introduces a soft-assignment $\tilde{a}_k(\mathbf{x}_i)$ for the samples $\{\mathbf{x}_i\}_{i=1}^N$, based on their distance to each cluster center. Formally:

$$\tilde{a}_k(\mathbf{x}_i) = \frac{e^{-\alpha\|\mathbf{x}_i - \mathbf{c}_k\|^2}}{\sum_{k'=1}^{K} e^{-\alpha\|\mathbf{x}_i - \mathbf{c}_{k'}\|^2}} \tag{2}$$

$\tilde{a}_k(\mathbf{x}_i)$ ranges between 0 and 1, with the highest value assigned to the closest center. α is a temperature parameter that controls the softness of the assignment, a high value for α (*e.g.* $\alpha \to +\infty$) would lead to a hard assignment like in VLAD. Furthermore, by expanding the squares and noticing that $e^{-\alpha\|\mathbf{x}_i\|^2}$ will cancel between the numerator and the denominator, we can interpret Equation (2) as the softmax of a convolutional layer for the input features parameterized by $\mathbf{w}_k = 2\alpha\mathbf{c}_k$ and $b_k = -\alpha\|\mathbf{c}_k\|^2$. Formally:

$$\tilde{a}_k(\mathbf{x}_i) = \frac{e^{\mathbf{w}_k^T\mathbf{x}_i + b_k}}{\sum_{k'} e^{\mathbf{w}_{k'}^T\mathbf{x}_i + b_{k'}}} \tag{3}$$

Finally, by plugging the soft-assignment from (3) into the VLAD formulation in (1), the NetVLAD features are defined as in Equation (4), later L2-normalized per cluster, flattened and further L2-normalized in its entirety.

$$V(j,k) = \sum_{i=1}^{N} \frac{e^{\mathbf{w}_k^T\mathbf{x}_i + b_k}}{\sum_{k'} e^{\mathbf{w}_{k'}^T\mathbf{x}_i + b_{k'}}} (\mathbf{x}_i(j) - \mathbf{c}_k(j)) \tag{4}$$

Note that the original VLAD optimizes solely the cluster centers $\mathbf{c}_k$, while NetVLAD optimizes for $\{\mathbf{w}_k\}$, $\{b_k\}$ and $\{\mathbf{c}_k\}$ independently, dropping the constraint of $\mathbf{w}_k = 2\alpha\mathbf{c}_l$ and $b_k = -\alpha\|\mathbf{c}_k\|^2$. These constraints were similarly dropped in [3], arguing for further freedom in the training process.

Efficient implementation. Implementing NetVLAD with libraries such as Tensorflow or Pytorch could lead to several memory challenges in mini-batch training. In particular, the formulation in (4) would lead to a 4-dimensional tensor, in particular due to the residuals $(\mathbf{x}_i(j) - \mathbf{c}_k(j))$, defined with a batch size (B), a set size (N), a number of clusters (K) and a features dimension (D). With small considerations in Equation (4), in particular splitting the residual in the two operands like in Equation (5), leads to a difference of two 3D tensors only, reducing the memory footprint as well as the computational complexity. Empirically, we experienced a $\sim 5\times$ speed up in computation (backward and forward) and a similar reduction for the memory footprint.

$$\begin{aligned} V(j,k) &= \sum_{i=1}^{N} \tilde{a}_k(\mathbf{x}_i)(\mathbf{x}_i(j) - \mathbf{c}_k(j)) \\ &= \sum_{i=1}^{N} \tilde{a}_k(\mathbf{x}_i)\mathbf{x}_i(j) - \Big(\sum_{i=1}^{N} \tilde{a}_k(\mathbf{x}_i)\Big)\mathbf{c}_k(j) \end{aligned} \tag{5}$$

Pooling for Action Spotting. We follow a similar architecture structure proposed in SoccerNet [19]. In particular, we learn to classify whether specific actions occurs within a temporal window. For inference, we densely slide the temporal window along the video to produce a class-aware actionness, on top of which we apply a non-maximum suppression (NMS). The frame features are pre-computed and pre-reduced in dimension with PCA, then pooled along the sliding window to predict the actionness of the central frame. Yet, SoccerNet [19] does not optimize the dimensionality reduction for the end-task and the pooling method is not aware of the temporal order of the frame features, nor consider *past* and *future* context independently.

3.2. NetVLAD++: Temporally-aware pooling

We propose a temporally-aware pooling module dubbed NetVLAD++ as our primary contribution. The VLAD and NetVLAD pooling methods are permutation invariant, as a consequence, do not consider the order of the frames, but only aggregates the features as a set. In the particular case of action spotting, the frames features from the videos are temporally ordered in time, and can be categorized between *past* and *future* context.

As noted by Cioppa *et al*. [7], the amount of context embedded before and after an action occurs is different, yet complementary. In addition, we argue that different actions might share similar vocabulary either before or after those actions occur, but usually not both. As an example, the semantic information contained *before* a "goal" occurs and *before* a "shot on/off target" occurs are similar, representing a lower level semantic concept of a player shooting on a target and a goalkeeper trying to catch that ball. Yet, those two action classes depict different contextual semantics *after* it occurs, with the presence of cheering (for "goal") or frustration (for simple "shot") in the players. Following a similar logic, the spotting of a "penalty" would benefit more from the knowledge of what happened *before* that penalty was shot, as the follow-up cheering would look similar to any other goal. Without loss of generality, it appears that the amount of information to pool among the features *before* and *after* an action occurs might contain different low-level semantics, helping identifying specific fine-grained actions.

To that end, we propose a novel temporally-aware pooling module, dubbed **NetVLAD++**, as depicted in Figure 1. In particular, we learn 2 different NetVLAD pooling modules for the frame features from *before* and *after* an action occurs. We define the *past* context as the frame feature with a temporal offset in $[-T_b, 0[$ and the *future* context as the frame feature with a temporal offset in $[0, T_a]$. Each pooling module aggregates different clusters of information from the 2 subsets of features, using K_a and K_b clusters, respectively for the *after* and *before* subsets. Formally:

$$V = \square(V_b, V_a) \tag{6}$$

with $\square$ an aggregation of V_b and V_a that represent the NetVLAD pooled features for the sample *before* and *after* the action occurs, parameterized with K_b clusters for the *past* context and K_a clusters for the *future* context.

3.3. Architecture for Action Spotting

We integrated our novel pooling module into a larger architecture depicted in Figure 2, follows a similar structure presented in SoccerNet [19]. In particular, it is based on a pre-trained frame feature encoder, a dimensionality reduction, a pooling module from a temporally sliding window and a per-frame classifier that depicts a class-aware

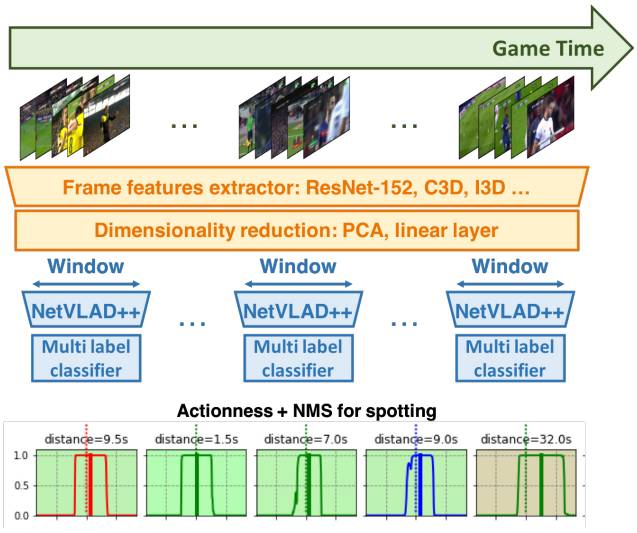

Figure 2. Action spotting architecture based on our novel temporally aware pooling module.

actionness. The action spotting is then performed using a non-maximum suppression (NMS). The main difference with SoccerNet [19] are twofold: an end-to-end learnable dimensionality reduction layer different from PCA and a temporally-aware pooling module.

Video encoding. We use the features extracted by SoccerNet-v2, based on ResNet-152 [20] pre-trained on ImageNet [15]. The weights are frozen and the frame features are pre-extracted at 2fps with a resolution of 224x224, scaled down in height then cropped on the sides for the width. The features correspond to the activation of the last layer of the ResNet-152 architecture, after the max pooling across the 2D feature map and before the classification layer, resulting in features of dimension 2048. We consider those features as input for the remaining of the architecture.

Dimensionality reduction. The dimension of the features are reduced from 2048 to 512, following SoccerNet [19] that learned a PCA reduction to that dimension. We argue that a linear layer would learn a better linear combination of the frame features, by removing the orthogonality constraint introduced by PCA. We refer to the experiments to appreciate the boost in performances. Moreover, learning a PCA reduction is feasible offline only, hence not practical for online training as it require the feature to be pre-extracted.

Temporally-aware pooling. We consider window chunks of time T s along the video. The temporally contiguous set of features are split equally *before* and *after* the center of the window and pooled accordingly. We normalize the features along the feature dimension and apply the 2 NetVLAD module for each subset of features. The 2 output NetVLAD features are concatenated along the feature dimension, leading into a feature of dimension $(K_b + K_a) \times D$.

Table 1. **State-of-the-art comparison.** We report the results of NetVLAD++ for action spotting (Average-mAP %) on SoccerNet-v2 [10]. We report the performances of our best model over 5 runs and detail its performances for each action class.

	SoccerNet-v2	visible	unshown	Ball out	Throw-in	Foul	Ind. free-kick	Clearance	Shots on tar.	Shots off tar.	Corner	Substitution	Kick-off	Yellow card	Offside	Dir. free-kick	Goal	Penalty	Yel.→Red	Red card
MaxPool [19]	18.6	21.5	15.0	38.7	34.7	26.8	17.9	14.9	14.0	13.1	26.5	40.0	30.3	11.8	2.6	13.5	24.2	6.2	0.0	0.9
NetVLAD [19]	31.4	34.3	23.3	47.4	42.4	32.0	16.7	32.7	21.3	19.7	55.1	51.7	45.7	33.2	14.6	33.6	54.9	32.3	0.0	0.0
AudioVid [41]	39.9	43.0	23.3	54.3	50.0	55.5	22.7	46.7	26.5	21.4	66.0	54.0	52.9	35.2	24.3	46.7	69.7	52.1	0.0	0.0
CALF [7]	40.7	42.1	29.0	63.9	56.4	53.0	41.5	51.6	26.6	27.3	71.8	47.3	37.2	41.7	25.7	43.5	**72.2**	30.6	0.7	0.7
NetVLAD++	**53.4**	**59.4**	**34.8**	**70.3**	**69.0**	**64.2**	**44.4**	**57.0**	**39.3**	**41.0**	**79.7**	**68.7**	**62.1**	**56.7**	**39.3**	**57.8**	71.6	**79.3**	**3.7**	**4.0**

Video Chunk Classification. In training, we consider non-overlapping window chunks with a sliding window of stride T. We build a classifier on top of the pooled feature, composed of a single neural layer with sigmoid activation and dropout. Since multiple actions can occur in the same temporal window, we consider a multi-label classification approach. A video chunk is labeled with all classes that appear on the chunk with a multi-label one-hot encoding. Similar to SoccerNet [19], we optimize for a multi-label binary cross-entropy loss as defined in Equation (7).

$$\mathcal{L} = \frac{1}{N} \sum_{i=1}^{N} y_i \log(x_n) + (1 - y_i) \log(1 - x_n) \qquad (7)$$

Inference. We run the sliding window of time T along unseen videos with a temporal stride of 1 to report the class-aware actionness scores in time. Similar to [19, 7, 10], we use a Non Maximum Suppression (NMS) module to reduce positive spots closer than a given temporal threshold T_{NMS}.

4. Experiments

Architectural design. We set our temporally-aware NetVLAD pooling to have as many parameters and similar complexity as a traditional NetVLAD pooling. We refrain on adapting the size of the vocabulary for the context *before* and *after* the action occurs, nor share the clusters between the 2 pooling layers. As a result, we not only enforce $K = K_a + K_b$, but also set $K_a = K_b = K/2$. Similarly, we set $T_a = T_b = T/2$ and consider the same amount of temporal context from *before* and *after* the actions. We reached highest performances by setting a temporal window $T = 15$s and $K = 64$ clusters. Finally, we suppress duplicate spottings around the highest confidence score with a NMS considering a centered window of $T_{NMS} = 30$s.

Training details. We use the Adam [27] optimizer with default β parameters from PyTorch and a starting learning rate of 10^{-3} that we decay from a factor of 10 after the validation loss does not improve for 10 consecutive epochs. We stop the training once the learning rate decays below 10^{-8}. Typically, a training converges in $\sim$100 epochs corresponding to $\sim$2h on a GTX1080Ti with a memory footprint of $\sim$1GB. Note that such footprint does not account for the extraction of the ResNet-152 frame features, that were pre-extracted. The code is available at https://soccer-net.org/.

Dataset and metrics. We train our novel architecture with the learnable dimensionality reduction and the temporally-aware pooling module on the SoccerNet dataset using the recent annotations with 17 classes from SoccerNet-v2 [10] and the recommended train/val/test split (300/100/100 games). We consider the action spotting Average-mAP introduced by SoccerNet [19], that considers the average precision (AP) for the spotting results per class within a given tolerance δ, averaged per class (mAP). The mAP are further averaged over tolerances ranging from 5s to 60s using a step size of 5s as per common practice [19, 7, 10, 37].

4.1. Main Results

The main performances of our action spotting architecture based on NetVLAD++ are compared in Table 1 with the current state-of-the-art for action spotting on SoccerNet-v2. For our method, we report the best model over 5 runs, as per common practice in video understanding [1], yet report a standard deviation contained within 0.2%. The main metric Average-mAP exhibits a boost of 12.7% w.r.t the previous state-of-the-art method CALF [7]. The improvement is consistent across 16 over the 17 classes of actions, where only the class *Goal* displayed worst performances. All the methods reported in Table 1 leverage ResNet-152 features extracted at 2fps (in addition of VGGish audio features for AudioVid [41]). Each of those baselines have different ways to deal with the frame features to solve for action spotting.

Table 2. **Ablation Studies. (Top) Main Contributions:** We highlight the improvement of each component of our novel **[NetVLAD++]** module and architecture on top of **[NetVLAD]**, with optimal NMS parameters **[NMS*]** and optimal window size T **[NMS*/T*]**. We highlight the contribution of the learnable linear layer **[w/o lin.layer]** and the temporally-aware feature pooling **[w/o tmp-aware]**. All performances are averaged over 5 runs. **(Bottom) Temporal context:** We report the Average-mAP for temporal window size T ranging from 5 to 30, averaged over 5 runs. Best performances per class are reported in bold. $T = 15s$ appears to be optimal.

	SoccerNet-v2	visible	unshown	Ball out	Throw-in	Foul	Ind. free-kick	Clearance	Shots on tar.	Shots off tar.	Corner	Substitution	Kick-off	Yellow card	Offside	Dir. free-kick	Goal	Penalty	Yel.→Red	Red card
Ablation	**Main Contributions**																			
NetVLAD	31.4	34.2	23.5	46.9	41.2	31.3	17.4	34.2	18.5	19.1	55.6	50.9	46.7	31.4	17.8	34.2	54.5	33.9	0.0	0.0
+ NMS*	47.1	52.3	34.8	60.2	57.5	52.8	38.8	54.5	36.2	36.0	72.4	66.7	**63.4**	49.6	33.0	50.6	66.3	55.0	2.4	4.5
+ NMS*/T*	48.4	54.2	32.5	62.8	60.0	53.8	38.8	56.2	36.6	37.7	76.7	67.2	62.5	51.8	30.8	51.2	67.0	60.2	3.8	5.2
NetVLAD++	**53.3**	**59.1**	35.1	**70.2**	**68.9**	**64.1**	**45.2**	**56.6**	**38.2**	**40.4**	**79.8**	68.9	61.1	56.1	**38.0**	**58.2**	**71.6**	**79.1**	**5.5**	3.5
w/o tmp-aware	50.2	56.6	32.2	64.2	61.3	54.4	39.4	**56.6**	37.3	39.6	77.6	66.1	60.7	**56.4**	32.7	55.6	66.4	64.3	4.5	**16.9**
w/o lin.layer	50.7	55.8	**37.3**	68.2	65.3	62.4	43.4	56.0	37.1	38.3	78.9	**70.3**	59.6	50.0	35.3	55.2	70.2	67.7	1.7	1.5
Window Size	**Temporal context**																			
T = 05s	46.0	52.8	28.6	66.9	68.6	46.5	36.1	50.2	34.9	**41.1**	81.2	58.9	54.7	51.7	9.5	**58.6**	45.2	64.3	12.5	1.2
T = 10s	50.7	57.1	34.5	**70.2**	**70.1**	61.5	42.8	53.7	37.3	39.3	**81.9**	66.6	59.1	55.1	26.8	58.3	63.1	68.6	5.8	1.7
T = 15s	**53.3**	**59.1**	**35.1**	70.2	68.9	**64.1**	**45.2**	**56.6**	38.2	40.4	79.8	**68.9**	**61.1**	**56.1**	38.0	58.2	71.6	**79.1**	5.5	3.5
T = 20s	53.0	58.3	35.1	67.5	66.1	62.2	44.5	56.4	38.5	39.5	77.2	68.9	59.1	54.8	39.0	57.0	**73.4**	78.6	10.3	7.1
T = 25s	50.7	55.6	34.9	64.2	62.8	59.4	45.0	55.4	**38.9**	37.3	71.0	65.2	59.6	54.5	39.1	54.0	70.9	63.7	**16.5**	4.6
T = 30s	49.4	53.9	35.1	60.1	57.5	54.7	43.2	51.6	38.3	35.9	65.7	62.1	59.3	55.2	**39.4**	53.8	70.8	75.5	9.4	**7.5**

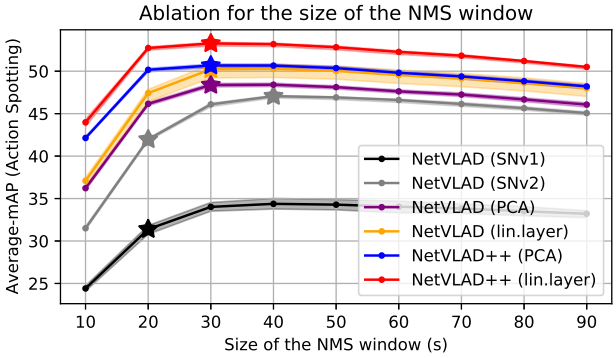

Figure 3. **Ablation on the size of the NMS window.** We report the performances of the 4 models presented in Table 2, with different size for the NMS window. Each entry point is averaged over 5 runs overlaid with max and min performances.

4.2. Main Ablation Study

Our improvement originates from 3 main differences w.r.t NetVLAD: **(i)** an optimized NMS head to extract spotting results, **(ii)** a linear layer for the frame features *vs*. a PCA reduction and **(iii)** a temporally-aware pooling module. We ablate each component in Table 2 with performances averaged over 5 runs. Figure 3 ablates the window size for the NMS and illustrates in transparency the variation between best and worst performances (yet contained).

Optimal setup for NetVLAD. First, we optimized the hyper-parameters for NetVLAD-like pooling methods. In particular, we identified 2 components that boost further the performances reported in SoccerNet-v2 [10]: the NMS head and the size of the sliding window T. SoccerNet [19] only considered spotting predictions with confidence scores higher than 0.5, depicting a lower-bound estimation of the performances (31.4%), as shown in Figure 3 (**black**) and Table 2 (**NetVLAD**). Considering all action spots without the threshold constraint on the confidence score leads to 42.0% Avg-mAP (+10.6%). Yet, further finetuning the size of the NMS window leads to 47.1% Avg-mAP (+5.1%), as shown in Figure 3 (**grey**) and Table 2 (+**NMS***). Finally, optimizing the size of the window T from $20s$ to $15s$ leads to an Avg-mAP of 48.4% (+1.3%), as shown in Figure 3 (**purple**) and Table 2 (+**NMS*/T***).

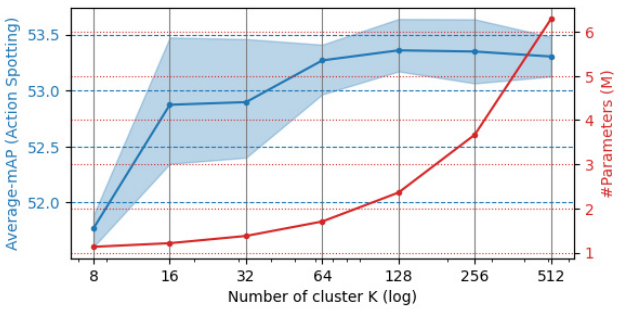

Figure 4. **Ablation for the Vocabulary Size.** The spotting performances are plotted in blue, averaged over 5 runs. Light blue illustrate the range of Average-mAP (min/max). Red indicated the number of parameters. More clusters increase the vocabulary but also the number of parameters, with a saturation after $K = 64$.

NetVLAD++. The learnable linear layer exhibits in Figure 3 (**yellow**) and Table 2 (**w/o tmp-aware**) a $+1.8\%$ boost w.r.t the best optimized NetVLAD (50.2% Avg-mAP). Similarly, the temporally-aware pooling exhibit in Figure 3 (**blue**) and Table 2 (**w/o lin.layer**) a $+2.3\%$ boost w.r.t the best optimized NetVLAD (50.7% Avg-mAP). Our final **NetVLAD++** displays an Avg-mAP of 53.4%, a boost of 4.9% w.r.t to the best optimized NetVLAD (Figure 3 (**red**)).

Ablation per class. Table 2 further depicts the performances per class. The linear layer mostly improves the performances for the visible instances of actions w.r.t the PCA dimensionality reduction, yet displays a drop for the unshown instances. We argue the unshown actions are diverse hence challenging to learn from. In contrast, the PCA reduction is generic and unsupervised, hence does not not suffer from the challenging input data. Regarding the temporal-awareness, the improvement is consistent regardless of the action visibility. Most classes appears to provide optimal results with NetVLAD++. Yet, *Clearances*, *Kick-Off*, *Yellow* and *Red cards* appear not to benefit from the temporal-awareness. We hypothesize that that those actions are already discriminative enough without it. Similarly, *Substitution* and *Kick-Off* do not benefit from the linear projection. Here, we believe the visual cue for those actions are global enough to not require a trainable projection.

4.3. Clusters and Temporal Windows

We further investigate the choice of hyper-parameters for NetVLAD++, in particular the size of the temporal windows T (Table 2) and the number of cluster K (Figure 4).

For the temporal windows, $T = 15$s appears to be optimal for the Average-mAP, yet specific action classes could benefit from different temporal boundaries. We believe that considering a different temporal window per class could lead to optimal results, yet impractical in our architecture that consider a single window to pool features from.

Table 3. **More video encoder:** Spotting performance using I3D, C3D and ResNET-152 video encoders, averaged over 5 runs (means $\pm$ std). NetVLAD++ with linear layer for dimensionality reduction results in best performances for all encoders.

Pooling	NetVLAD	NetVLAD++	NetVLAD++
Encoder	(PCA)	(PCA)	(lin layer)
I3D	34.9 ± 0.3	38.1 ± 0.1	$\mathbf{41.5 \pm 0.1}$
C3D	46.1 ± 0.3	47.2 ± 0.2	$\mathbf{48.6 \pm 0.8}$
ResNet	48.4 ± 0.2	50.7 ± 0.2	$\mathbf{53.3 \pm 0.2}$

As for the number of clusters, the more the better, yet the performances appears to saturate after 64 clusters. In fact, K define the size of the vocabulary to cluster the pooled features but the vocabulary can only improve up to a certain extent. Furthermore, note that an increase in the number of cluster irrevocably leads into an increase in the number of parameters. The classifier process the NetVLAD features of dimension $K * D$ in a fully connected layer parameterized with $(K * D + 1)$ weight and biases per class. Practically, we refrain on using a large vocabulary K as it leads to large number of parameters and chose $K = 64$ in the design of our architecture.

4.4. More Video Encoders

Most related works consider ResNet-152 for the video feature encoder [19, 7, 10]. SoccerNet [19] provides alternative I3D [6] and C3D [38] video features, yet showed worst performances [19]. In Table 3, we show that our temporally-aware pooling NetVLAD++ transfers well to I3D and C3D, boosting NetVLAD with 6.6% and 2.5%, respectfully. More recent video encoders such as R3D [43] and R(2+1)D [40] could lead to higher performances, but due to the computational complexity of pre-extracting frame features, we leave that for future works.

4.5. More Temporally-Aware Pooling Modules

We further transfer our temporally-aware pooling to further pooling modules. In particular, we implement temporally-aware Max, Average and NetRVLAD pooling modules as reported in Table 4. We refrained in optimizing the performances for each module, and only highlight the relative improvement brought by the temporal awareness.

We developed MaxPool++ and AvgPool++ based on MaxPool and AvgPool with an extra temporal awareness. We considered the PCA-reduced ResNet features as the low number of parameters for those models (9234 parameters each) impeded a stable learning on top of higher dimensionality features. Note that MaxPool++ and AvgPool++ concatenates the *past* and *future* context (twice the dimensionality) which inevitably leads to a similar increase in parameters (18450 parameters each). MaxPool++ and AvgPool++

Table 4. **More pooling modules.** Spotting performances using Max, Avg and NetRVLAD pooling modules. All temporally-aware pooling method outperforms the original pooling.

Pooling	Original	Tmp.-Aware
MaxPooling (PCA)	23.7 ± 0.4	$\mathbf{31.6 \pm 0.7}$
AvgPooling (PCA)	32.5 ± 0.1	$\mathbf{40.6 \pm 0.2}$
NetRVLAD (lin.layer)	48.0 ± 0.2	$\mathbf{50.9 \pm 0.3}$
NetVLAD (lin.layer)	50.2 ± 0.6	$\mathbf{53.3 \pm 0.2}$

displayed impressive boosts in performances ($+7.9\%$ and $+8.1\%$ resp.) now flirting with performances similar to previous baselines proposed in SoccerNet-v2 [10], yet leveraging $\sim 20\times$ less parameters for the spotting head.

Following previous experiments on residual-less NetVLAD [19], we developed NetRVLAD++ on top of NetRVLAD, which drops the cluster parameters $\mathbf{c}_k(j)$ in (4), leading to slightly less parameters to learn. We build NetRVLAD++ on top of full ResNet feature with our learnable feature projection. The relative improvement here is similar ($+3.1\%$), yet the performances are not on par (-2.2%) with NetVLAD++, highlighting the importance of the NetVLAD residuals.

5. Discussion

Temporal-awareness *vs*. CALF [7] *vs*. RMS-Net [37]. NetVLAD++ is not the first approach that considers temporal semantic regions around the action to spot. CALF [7] defines a high-level semantic context from different temporal regions *far distant*, *just before* and *just after* an action occurs. They introduce a hand-crafted loss function that weights the contextual information. Still, they leverage the same features for the context *before* and *after* the actions occurs. In contrast, we drop the *far distant* semantic context in NetVLAD++ and learn specific features from different vocabulary (NetVLAD clusters) for the *past* and *future* temporal context. RMS-Net [37] propose a similar contextual approach borrowed from the NLP literature that masks out part of the temporal context. In particular, they drop the *past* information during training, expecting the model to focus exclusively on the future frames. In contrast, we learn both *past* and *future* temporal context independently on NetVLAD++, and merge both learned context.

More temporal regions. We considered extending the temporal region beyond the close *past* and *future* contexts, following insights from CALF [7] that considered *far distant* temporal segments. Our experiments with *far before* and *far after* temporal contexts did not lead to any improvement for the learning of the pooling module and inevitably increases the number of hyper-parameters defining those temporal regions. We believe the temporal context *before* and *after* are discriminative enough, while the *far distant* equivalent are more blurry in time with the *close* context. Also, each action class might consider different temporal context for the *far distant*, which would lead to more confusion for the learnable pooling layers. We believe a global video feature or a better temporal aggregation of the features across the complete video could lead to a better temporal understanding and would take care of the *far distant* temporal context.

Table 5. **SoccerNet-v2 Challenge.** Our NetVLAD++ approach reach best performances on the SoccerNet

Method	Avg-mAP	Visible	Unshown
NetVLAD [19]	30.74	32.99	23.27
CALF [7]	42.22	43.51	37.91
RMS-Net [37]	49.66	53.11	38.92
NetVLAD++	**52.54**	**57.12**	**46.15**

Spotting Regression. Both CALF [7] and RMS-Net [37] learn to regress action spots. We decided not to regress the actions spot but rather rely on a dense sliding windows with an NMS to discard non-optimal action spots. A dense sliding window inevitably leads to slower inference, yet NetVLAD++ takes <1 second to infer a complete 90min soccer game from pre-extracted features.

SoccerNet-v2 Challenge. We tested our approach on the segregated challenge set of SoccerNet-v2. For the competition, we trained on the *train+val* sets, validated on the *test* set and inferred on the *challenge* set, that we submitted on the evaluation server. At submission time (Table 5), we reached SOTA performances with 52.54% Avg-mAP.

6. Conclusion

In this work, we proposed a temporally-aware learnable pooling module for the task of action spotting on soccer videos. We first showed that NetVLAD can further be optimized on SoccerNet-v2. We further improve the pooling-based action spotting architecture by learning a linear projection that reduce the features dimension and split the past and future features to pool, leading to state-of-the-art performances on the SoccerNet-v2 benchmark. We show a complete ablation and transfer capability of our contribution to any pooling layer and input features, paving the road for more temporally-aware learning in video. We believe future works should focus on integrating local frame features from low-level semantics (player, ball, field, etc...) and consider complete videos rather than temporally-bounded clips as input. Future works should learn to accumulate knowledge in time or based on attention models in order to reach higher-level of understanding in soccer broadcasts.

Acknowledgments: This work is supported by the KAUST Office of Sponsored Research under Award No. OSR-CRG2017-3405.

References

[1] Humam Alwassel, Silvio Giancola, and Bernard Ghanem. Tsp: Temporally-sensitive pretraining of video encoders for localization tasks. *arXiv preprint arXiv:2011.11479*, 2020. 5

[2] Relja Arandjelovic, Petr Gronat, Akihiko Torii, Tomas Pajdla, and Josef Sivic. NetVLAD: CNN architecture for weakly supervised place recognition. In *CVPR*, pages 5297–5307, 2016. 3

[3] Relja Arandjelovic and Andrew Zisserman. All about VLAD. In *CVPR*, pages 1578–1585, 2013. 3

[4] Lewis Bridgeman, Marco Volino, Jean-Yves Guillemaut, and Adrian Hilton. Multi-person 3d pose estimation and tracking in sports. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition Workshops*, pages 0–0, 2019. 1

[5] Lewis Bridgeman, Marco Volino, Jean-Yves Guillemaut, and Adrian Hilton. Multi-Person 3D Pose Estimation and Tracking in Sports. In *IEEE Conference on Computer Vision and Pattern Recognition (CVPR) Workshops*, pages 2487–2496, June 2019. 2

[6] Joao Carreira and Andrew Zisserman. Quo vadis, action recognition? a new model and the kinetics dataset. In *proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 6299–6308, 2017. 7

[7] Anthony Cioppa, Adrien Deliège, Silvio Giancola, Bernard Ghanem, Marc Van Droogenbroeck, Rikke Gade, and Thomas B. Moeslund. A context-aware loss function for action spotting in soccer videos. In *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 13126–13136, 2020. 2, 4, 5, 7, 8

[8] Anthony Cioppa, Adrien Deliège, Maxime Istasse, Christophe De Vleeschouwer, and Marc Van Droogenbroeck. ARTHuS: Adaptive Real-Time Human Segmentation in Sports Through Online Distillation. In *IEEE Conference on Computer Vision and Pattern Recognition (CVPR) Workshops*, pages 2505–2514, June 2019. 1, 2

[9] Anthony Cioppa, Adrien Deliège, and Marc Van Droogenbroeck. A bottom-up approach based on semantics for the interpretation of the main camera stream in soccer games. In *IEEE Conference on Computer Vision and Pattern Recognition (CVPR) Workshops*, pages 1846–1855, June 2018. 2

[10] Adrien Deliège, Anthony Cioppa, Silvio Giancola, Meisam J Seikavandi, Jacob V Dueholm, Kamal Nasrollahi, Bernard Ghanem, Thomas B Moeslund, and Marc Van Droogenbroeck. Soccernet-v2: A dataset and benchmarks for holistic understanding of broadcast soccer videos. *arXiv preprint arXiv:2011.13367*, 2020. 2, 5, 6, 7, 8

[11] Deloitte. Length of sports tv broadcast hours in the united states from 2002 to 2017. In *Statista - The Statistics Portal*, 2021. Retrieved January 11, 2021, from https://www.statista.com/statistics/290110/length-sports-tv-programming-available-usa/. 1

[12] Deloitte. Sports media rights market size in north america from 2006 to 2023 (in billion u.s. dollars)*. In *Statista - The Statistics Portal*, 2021. Retrieved January 11, 2021, from https://www.statista.com/statistics/194225/sports-media-rights-revenue-in-north-america/. 1

[13] Deloitte. Volume of sports programs on free television in france between 2010 and 2018. In *Statista - The Statistics Portal*, 2021. Retrieved January 11, 2021, from https://www.statista.com/statistics/1016830/sports-television-hourly-volume-france/. 1

[14] Deloitte. Volume of sports programs on pay television in france between 2000 and 2018. In *Statista - The Statistics Portal*, 2021. Retrieved January 11, 2021, from https://www.statista.com/statistics/1025759/hourly-volume-sports-pay-tv-france/. 1

[15] Jia Deng, Wei Dong, Richard Socher, Li-Jia Li, Kai Li, and Li Fei-Fei. Imagenet: A large-scale hierarchical image database. In *CVPR*, pages 248–255. IEEE, 2009. 4

[16] Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. Bert: Pre-training of deep bidirectional transformers for language understanding. *arXiv preprint arXiv:1810.04805*, 2018. 2

[17] Dirk Farin, Susanne Krabbe, Peter de With, and Wolfgang Effelsberg. Robust camera calibration for sport videos using court models. In *Storage and Retrieval Methods and Applications for Multimedia*, pages 80–91, December 2003. 2

[18] Panna Felsen, Pulkit Agrawal, and Jitendra Malik. What will happen next? Forecasting player moves in sports videos. In *IEEE International Conference on Computer Vision (ICCV)*, pages 3362–3371, October 2017. 2

[19] Silvio Giancola, Mohieddine Amine, Tarek Dghaily, and Bernard Ghanem. SoccerNet: A Scalable Dataset for Action Spotting in Soccer Videos. In *IEEE Conference on Computer Vision and Pattern Recognition (CVPR) Workshops*, pages 1711–1721, June 2018. 2, 3, 4, 5, 6, 7, 8

[20] Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. Deep residual learning for image recognition. In *IEEE International Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 770–778, June 2016. 4

[21] Farnoosh Heidarivincheh, Majid Mirmehdi, and Dima Damen. Detecting the moment of completion: temporal models for localising action completion. *arXiv preprint arXiv:1710.02310*, 2017. 2

[22] Fabian Caba Heilbron, Victor Escorcia, Bernard Ghanem, and Juan Carlos Niebles. ActivityNet: A Large-Scale Video Benchmark for Human Activity Understanding. In *IEEE International Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 961–970, June 2015. 2

[23] Namdar Homayounfar, Sanja Fidler, and Raquel Urtasun. Sports field localization via deep structured models. In *IEEE International Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 4012–4020, July 2017. 2

[24] Maxime Istasse, Julien Moreau, and Christophe De Vleeschouwer. Associative Embedding for Team Discrimination. In *IEEE Conference on Computer Vision and Pattern Recognition (CVPR) Workshops*, pages 2477–2486, June 2019. 2

[25] Hervé Jégou, Matthijs Douze, Cordelia Schmid, and Patrick Pérez. Aggregating local descriptors into a compact image representation. In *CVPR*, pages 3304–3311. IEEE, 2010. 3

[26] Yudong Jiang, Kaixu Cui, Leilei Chen, Canjin Wang, and Changliang Xu. Soccerdb: A large-scale database for comprehensive video understanding. In *International Workshop on Multimedia Content Analysis in Sports*, pages 1–8, October 2020. 2

[27] Diederik P Kingma and Jimmy Ba. Adam: A method for stochastic optimization. *arXiv preprint arXiv:1412.6980*, 2014. 5

[28] Mehrtash Manafifard, Hamid Ebadi, and Hamid Abrishami Moghaddam. A survey on player tracking in soccer videos. *Computer Vision and Image Understanding*, 159:19–46, June 2017. 1, 2

[29] Thomas B. Moeslund, Graham Thomas, and Adrian Hilton. *Computer Vision in Sports*. Springer, 2014. 2

[30] Konstantinos Rematas, Ira Kemelmacher-Shlizerman, Brian Curless, and Steve Seitz. Soccer on your tabletop. In *IEEE International Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 4738–4747, June 2018. 2

[31] Shaoqing Ren, Kaiming He, Ross Girshick, and Jian Sun. Faster r-cnn: towards real-time object detection with region proposal networks. *IEEE transactions on pattern analysis and machine intelligence*, 39(6):1137–1149, 2016. 2

[32] Olav A. Nergård Rongved, Steven A. Hicks, Vajira Thambawita, Håkon K. Stensland, Evi Zouganeli, Dag Johansen, Michael A. Riegler, and Pål Halvorsen. Real-time detection of events in soccer videos using 3D convolutional neural networks. In *IEEE International Symposium on Multimedia (ISM)*, December 2020. In press. 2

[33] Adrià Arbués Sangüesa, A. Martín, J. Fernández, C. Ballester, and G. Haro. Using player's body-orientation to model pass feasibility in soccer. In *IEEE Conference on Computer Vision and Pattern Recognition (CVPR) Workshops*, pages 3875–3884, June 2020. 2

[34] Saikat Sarkar, Amlan Chakrabarti, and Dipti Prasad Mukherjee. Generation of Ball Possession Statistics in Soccer Using Minimum-Cost Flow Network. In *IEEE Conference on Computer Vision and Pattern Recognition (CVPR) Workshops*, pages 2515–2523, June 2019. 1, 2

[35] Rajkumar Theagarajan, Federico Pala, Xiu Zhang, and Bir Bhanu. Soccer: Who has the ball? Generating visual analytics and player statistics. In *IEEE Conference on Computer Vision and Pattern Recognition (CVPR) Workshops*, pages 1830–1838, June 2018. 2

[36] Graham Thomas, Rikke Gade, Thomas B. Moeslund, Peter Carr, and Adrian Hilton. Computer vision for sports: Current applications and research topics. *Computer Vision and Image Understanding*, 159:3–18, June 2017. 1, 2

[37] Matteo Tomei, Lorenzo Baraldi, Simone Calderara, Simone Bronzin, and Rita Cucchiara. Rms-net: Regression and masking for soccer event spotting. *arXiv preprint arXiv:2102.07624*, 2021. 2, 5, 8

[38] Du Tran, Lubomir Bourdev, Rob Fergus, Lorenzo Torresani, and Manohar Paluri. Learning spatiotemporal features with 3d convolutional networks. In *Proceedings of the IEEE international conference on computer vision*, pages 4489–4497, 2015. 7

[39] Du Tran, Heng Wang, Lorenzo Torresani, Jamie Ray, Yann LeCun, and Manohar Paluri. A closer look at spatiotemporal convolutions for action recognition. In *Proceedings of the IEEE conference on Computer Vision and Pattern Recognition*, pages 6450–6459, 2018. 2

[40] Du Tran, Heng Wang, Lorenzo Torresani, Jamie Ray, Yann LeCun, and Manohar Paluri. A closer look at spatiotemporal convolutions for action recognition. In *Proceedings of the IEEE conference on Computer Vision and Pattern Recognition*, pages 6450–6459, 2018. 7

[41] Bastien Vanderplaetse and Stephane Dupont. Improved soccer action spotting using both audio and video streams. In *IEEE Conference on Computer Vision and Pattern Recognition (CVPR) Workshops*, pages 3921–3931, June 2020. 2, 5

[42] Kanav Vats, Mehrnaz Fani, Pascale Walters, David A Clausi, and John Zelek. Event detection in coarsely annotated sports videos via parallel multi-receptive field 1d convolutions. In *IEEE Conference on Computer Vision and Pattern Recognition (CVPR) Workshops*, pages 882–883, June 2020. 2

[43] Huijuan Xu, Abir Das, and Kate Saenko. R-c3d: Region convolutional 3d network for temporal activity detection. In *Proceedings of the IEEE international conference on computer vision*, pages 5783–5792, 2017. 7

[44] Ying Yang and Danyang Li. Robust player detection and tracking in broadcast soccer video based on enhanced particle filter. *Journal of Visual Communication and Image Representation*, 46:81–94, July 2017. 2

[45] Junqing Yu, Aiping Lei, Zikai Song, Tingting Wang, Hengyou Cai, and Na Feng. Comprehensive dataset of broadcast soccer videos. In *2018 IEEE Conference on Multimedia Information Processing and Retrieval (MIPR)*, pages 418–423. IEEE, 2018. 2

[46] Dan Zecha, Moritz Einfalt, and Rainer Lienhart. Refining joint locations for human pose tracking in sports videos. In *IEEE Conference on Computer Vision and Pattern Recognition (CVPR) Workshops*, pages 2524–2532, June 2019. 2