Associative Embedding for Game-Agnostic Team Discrimination

Maxime Istasse, Julien Moreau, Christophe de Vleeschouwer

Associative Embedding for Game-Agnostic Team Discrimination

Maxime Istasse*, Julien Moreau*, Christophe De Vleeschouwer
UCLouvain ICTEAM, Belgium
*These authors contributed equally to this work
{maxime.istasse,julien.moreau,christophe.devleeschouwer}@uclouvain.be

Abstract

Assigning team labels to players in a sport game is not a trivial task when no prior is known about the visual appearance of each team. Our work builds on a Convolutional Neural Network (CNN) to learn a descriptor, namely a pixel-wise embedding vector, that is similar for pixels depicting players from the same team, and dissimilar when pixels correspond to distinct teams. The advantage of this idea is that no per-game learning is needed, allowing efficient team discrimination as soon as the game starts. In principle, the approach follows the associative embedding framework introduced in [22] to differentiate instances of objects. Our work is however different in that it derives the embeddings from a lightweight segmentation network and, more fundamentally, because it considers the assignment of the same embedding to unconnected pixels, as required by pixels of distinct players from the same team. Excellent results, both in terms of team labelling accuracy and generalization to new games/arenas, have been achieved on panoramic views of a large variety of basketball games involving players interactions and occlusions. This makes our method a good candidate to integrate team separation in many CNN-based sport analytics pipelines.

1. Introduction

Team sports analytics has numerous applications, ranging from broadcast content enrichment to game statistical analysis for coaches [6, 30, 40]. Assigning team labels to detected players is of particular interest when investigating the relationship between team positioning and sport action success/failure statistics [3, 12, 16], but also for some specific tasks such as offside detection in soccer [10] or ball ownership prediction in basketball [33].

Many previous works have investigated computer vision methods to detect and track team sport players [7, 10, 17, 18, 19, 24, 31]. They can detect individual players, but generally resort to unpractical manual intervention or to unreliable heuristics to adapt their processing pipeline to recognize the players' team. Specifically, they generally need human intervention to adjust the team discriminant features (*e.g.* RGB histogram in [18], or CNN features in [17]) to the game at hand [3, 16, 17, 18]. A few methods have attempted to derive game-specific team features in an automatic manner [10, 31]. They consider the unsupervised clustering of color histograms [10] or bags of color features [31] computed on the spatial support of the players that are detected in the game at hand. Those methods depend on how well color discriminates the two teams, but is also quite sensitive to occlusions and to the quality of player detection and segmentation [19]. This probably explains why those previous works have been demonstrated in outdoor and highly contrasted scenes, as encountered in soccer for example. We show in Section 4 that those methods fail to address real-life indoor cases.

As observed in [17], indoor sports analytics have to deal with lower color contrast between players and background, and more dynamic scenes, with more frequent occlusions. [23, 24] also point out the low illumination, the strong reflections induced by dynamic advertising boards, the severe shadows, the large player density and the lack of color discrimination in indoor scenes.

In our work, we do not arbitrarily select a handcrafted feature to discriminate the teams. We do not consider a framework that requires game-specific adjustment either. Instead we adopt a generic learning-based strategy that aims at predicting a feature vector in each pixel, in such a way that, independently of the game at hand, similar vectors are predicted in pixels lying in players from a same team, while distinct vectors are assigned to pairs of pixels that correspond to distinct teams. In other words, we train a neural network to separate, in an embedding space, the pixels of different teams and to group those in common team. A simple and efficient clustering algorithm can then be used to dissociate different teams in an image. Hence, we do not rely on explicit recognition of specific teams, but rather

This research is supported by the DeepSport project of the Walloon region, Belgium. C. De Vleeschouwer is funded by the F.R.S.-FNRS (Belgium).

learn how to map player pixels to a feature space that promotes team clustering, whatever the team appearance. Although teams change at each game, there is thus no need for fine tuning or specific manual annotation for new games. The approach has been inspired by the associative embedding strategy recently introduced to discriminate instances in object detection problems [21, 22]. However, differently from [21, 22], it is demonstrated using a lightweight ICNet convolutional neural network (opening broader deployment perspectives than the heavy stacked hourglass architecture promoted in [21, 22]) and, to our knowledge, is the first work assigning similar embeddings to unconnected pixels, thereby extending the field of application of pixel-wise associative embedding.

To validate our method, we have trained our network on a representative set of images captured in a variety of games and arenas. Since only a few player keypoints (head, pelvis, and feet) have been annotated in addition to the player team index, the player segmentation component of our network has been trained with approximate ground-truth masks, corresponding to ellipses connecting the key points. Our CNN model is validated on games (teams) and arenas that have not been seen during training. It achieves above 90% team recognition accuracy, despite the challenging scenes (indoor, dynamic background, low contrast) and the inaccurate segmentation ground-truth considered during training. Interestingly, the lightweight backbone makes the solution realistic for real-time deployment.

Our paper is organized as follow. Section 2 reviews the related works associated to CNN-based sport analysis, segmentation, and associative embedding. Section 3 then introduces our proposed method, using a ICNet variant to both segment the players and compute pixel-wise team discriminant embeddings. The experiments presented in Section 4 demonstrate the relevance of our approach, while conclusions and some perspectives are provided in Section 5.

2. Related works

Recent developments in computer vision make an extensive use of Convolutional Neural Networks [28]. This section reviews the specific type of CNNs, named Fully Convolutional Network (FCN), that is used for image segmentation. It then introduces the recent associative embedding methods considered to turn object class segmentation into object instance segmentation.

2.1. Fully Convolutional Network (FCN)

Fully Convolutional Networks are characterized by the fact that they output spatial feature maps, strictly computed by the recursive application of convolutional layers, generally completed with ReLu activation and batch-normalization or dropout regularization layers.

In recent works dealing with sport video analysis, FCNs have been considered for specific segmentation tasks, including player jersey number extraction [11], soccer field lines and players segmentation [7]. In [17], a two-steps architecture, inspired by [35] and [37], is even proposed to extract players bounding-boxes with team labels. The network however needs to be trained on a game-per-game basis, which is impractical for large scale deployment. None of these works is thus able to differentiate player teams without requiring a dedicated training for each game, as proposed in Section 3, where a real-time amenable FCN provides the player segmentation mask, as well as a pixel-wise team-discriminant feature vector.

There are two main categories of real-time FCNs: encoder-decoder networks and multi-scale networks.

Encoder-decoder architectures adopt the encoder structure of classification networks, but replace their dense classification layers by fully convolutional layers that upsample and convolve the coded features up to pixel-wise resolution. SegNet (Segmentation Network) [2] was the first segmentation architecture to reach near real-time inference. It is a symmetrical encoder-decoder network, with skip connection of pooling indices from encoder layers to decoder layers. ENet (Efficient Neural Network) [25] follows SegNet, but comes with various improvements, whose most prominant one is the use of a smaller decoder than the encoder.

Quite recently, several authors proposed to adopt multi-scale architectures to better balance accuracy and inference complexity. Considering multiple scales allows to exploit both a large receptive field and a fine image resolution, with a reduced number of network layers. Among those networks, ICNet (Image Cascade Network) [38] is based on PSPNet (Pyramid Scene Parsing Network) [39], a state-of-the-art network for non real-time segmentation. ICNet encodes the features at three scales. The coarsest branch is a PSPNet, while finer ones are lighter networks, allowing to infer segmentation in real-time. Two-columns network [34], BiSeNet (Bilateral Segmentation Network) [36], GUN (Guided Upsampling Network) [20] and ContextNet [27] are composed of two branches.

2.2. Associative embedding

An embedding vector denotes a local descriptor that characterizes a signal locally in a way that can support a task of interest. Embeddings are thus not defined a priori. Instead, they are defined in an indirect manner, to support the task of interest. In computer vision, FCNs have recently been considered to compute pixel-wise embeddings in a variety of contexts related to pixel clustering or pixel association tasks. In this context, FCN training is not supervised to output a specified value. Rather, FCN training supervises the relations between the embedded vectors, and checks that they are consistent with the task of interest.

In [32], the embedding vector is used to compute the similarity between two pixel neighborhoods from two distinct images, typically to support a tracking task. Interestingly, a proxy task that consists in predicting the (known) color of a target frame based on the color in a reference frame is used to supervise the training of the FCN computing the embeddings. Good embeddings indeed result in relevant pixel associations, and in accurate color predictions. This reveals that a FCN can be trained in an indirect manner to support various higher-level tasks based on richer pixel-wise embedding.

Of special interest with respect to our team discrimination problem, *associative* embeddings have been introduced in [21, 22] and used in [15, 21, 22] to associate pixels sharing a common semantic property, namely the fact that they belong to the same object instance. Authors in [22] introduced associative embedding in the context of multi-person pose estimation from joints detection and grouping, and extended it to instance segmentation. More recently, [15] proposed CornerNet, a new state-of-the art one-shot bounding box object detector, by using associative embedding to group top-left and bottom-right box corners. In all these publications, the network is trained to give close embeddings to pixels from the same instance and distant embeddings to pixels corresponding to different instances. All these works are based on the same heavy stacked hourglass architecture. However, [21] suggest that the approach is not strictly restricted to this architecture, as long as two important properties are fulfilled: first, the network should have access both to global and local information; second, pixel-wise prediction at fine resolution is recommended, in order to avoid that a vector is subject to concurrent instances. This makes ICNet a premium candidate to segment players and compute team-specific embeddings in real time, since it computes features at three scales instead of two for other lightweight multi-branch FCN architectures.

3. Team segmentation using pixel-wise associative embedding

Player team discrimination is not a conventional segmentation problem since the visual specificities of each class are not known in advance. This section explains how associative embedding can be combined with player segmentation to address this problem.

3.1. Team discrimination & player segmentation

We propose to adopt the associative embedding paradigm to support the team discrimination task. In short, we design a fully convolutional network so that, in addition to a player segmentation mask, it outputs for each pixel a D-dimensional feature vector that is similar for pixels that correspond to players of the same team, while being distinct for pixels associated to distinct teams. As explained in the previous section, embeddings learning is not based on an explicit supervision. Instead, embeddings are envisioned as a latent pixel-wise representation, trained to support a pixel-wise association task, typically to group [15] or match [32] pixels together. In the context of object detection, associative embedding has been applied with success in [15, 22] to group pixels corresponding to a same object instance. In these works, multiple hourglass-shaped networks are stacked recursively in order to progressively refine the 1-D embedding value that aims to differentiate object instances in a given class. Our work differs from [22, 21] and [15] in two main aspects.

First, and because we target real-time deployment, the stacked hourglass architecture is replaced by an ICNet [38] backbone, as illustrated in Figure 1. As stated in [38], ICNet reaches 30 FPS for images of 1024×2048 pixels on one Titan X GPU card. We use ICNet because its multi-scale encoders, along with a spatial pyramidal pooling, give access to a reasonably large receptive field (important to share embedding information spatially) while preserving the opportunity to exploit high-resolution image signal locally (important for a fine characterization of the content).

Second, our work deals with the problem of associating pixels of players that are scattered across the whole image. This is in contrast with the association of neighboring/connected pixels generally considered in traditional association tasks [15, 22].

3.2. Network architecture

The ICNet network architecture has mostly been left unchanged. Only the final convolution layer has been adapted to provide $D+1$ channels. Those comprise 1 channel for semantic segmentation, with a sigmoid activation, along with D channels for embeddings with linear activation. Figure 1 presents the player segmentation channel in blue while the D channels for embeddings are represented in orange. A number of loss functions are combined to train the network. Along with the multi-scale semantic segmentation loss from [38], composed by L_{124}, L_{24} and L_4, we add an embedding loss inspired by [22, 21, 15]. It comprises two components, L_{pull} and L_{push}, which respectively pull teammates embeddings together and push opponents embeddings away from each other. L_{pull} and L_{push} only apply to the finest resolution. We have defined all loss components based on mean square distances.
L_{124}, L_{24} and L_4 losses are defined as:

$$L_{s\in\{124,24,4\}} = \frac{1}{HW} \sum_{(i,j)}^{H\times W} (\hat{m}^s_{ij} - m^s_{ij})^2 \qquad (1)$$

with H and W being the layer height and width, while $\hat{m}^s$ and m^s respectively denote the predicted and ground-truth

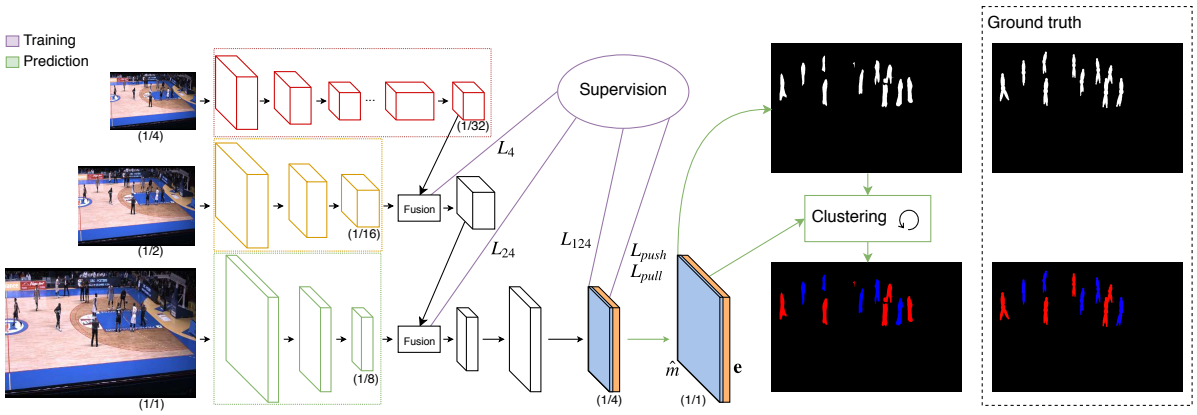

Figure 1: Overview of our architecture. ICNet [38] is used as backbone for following assets: pixel-wise segmentation, combination of three scales to encode global and local features, fast ([38] reaches 30 FPS at 1024×2048 resolution). Its last convolution is modified to output a segmentation mask along with vector embeddings in each pixel. We keep the multi-scale supervision for the segmentation and add L_{push} and L_{pull} to obtain similar embeddings in pixels of the same team and distant embeddings for pixels of different teams. After network inference, a simple clustering algorithm can effectively split in different teams the pixels from the segmentation mask $\hat{m}$ according to their embeddings $\mathbf{e}$.

player masks at scale s. Similarly, L_{pull} is formulated as:

$$L_{pull} = \frac{1}{HW} \sum_{n}^{\{1,2\}} \sum_{(i,j)}^{M_n} (t_{ij} - T_n)^2, \tag{2}$$

where T_n is the mean of the embeddings t_{ij} predicted across the pixels M_n of team n, *i.e.* $T_n = \sum_{(i,j) \in M_n} t_{ij}$.
In [22], the push loss is expressed as a mean over pairs of pixels of a cost function that is chosen to be high (low) when pixels that are not supposed to receive the same embedding have a similar (different) embedding. Recently, [21] and [15] employed a "margin-based penalty", and wrote that this is the most reliable formulation they tested. Hence, we also adopt a margin-based penalty loss. Formally, L_{push} is defined similarly to L_{pull}, except that rather than penalizing embeddings that are far away from their centroid, it penalizes embeddings that are too close from the centroid of another team:

$$L_{push} = \frac{1}{HW} \sum_{n}^{\{1,2\}} \sum_{(i,j)}^{M_n} \max(0; 1 - (t_{ij} - T_{3-n})^2) \tag{3}$$

Our global objective function finally becomes:

$$\begin{aligned} L = &\lambda_{124} L_{124} + \lambda_{24} L_{24} + \lambda_4 L_4 \\ &+ \lambda_{pull} L_{pull} + \lambda_{push} L_{push} \end{aligned} \tag{4}$$

with the lambda loss factors having to be tuned (chosen values are explained in Section 3.3).

At inference, upsampling of last layer is inserted before activation (respectively bilinear and nearest neighbor interpolations for segmentation and embedding channels). Then, a clustering algorithm is required to group pixels in teams. Fortunately, as observed in [22], the network does a great job at separating the embeddings for distinct teams, so that a simple and greedy method such as the one detailed in Algorithm 1 is able to handle the clustering properly. As appears from the pseudocode, our naive clustering algorithm relies on the assumption that a player pixel embedding surrounded by similar embeddings is representative of its team embedding. Given a team embedding vector, player pixels are likely to be assigned to that team if their embedding lies in a sphere of radius 1 around the team embedding. We incorporate a refinement step in which we compute the centroid of the selected pixels. Then, to resolve ambiguities, player pixels are associated to the closest of the centroids.

3.3. Implementation details and hyperparameters

Our network is trained to extract players only, and to estimate associative embeddings for team discrimination. Referees and other non-player persons are part of the background class. Our work is based on the PyTorch ICNet implementation [29]. Parameters have been empirically tuned. For the training, we employ Adam optimizer [14]. Losses factors defined in Equation 4 are: $\lambda_{124} = 1$ and $\lambda_{24} = \lambda_4 = 0.4$ as in original ICNet [38], $\lambda_{pull} = \lambda_{push} = 4$ and are thus very different than in [22, 21, 38] because our pull and push losses definitions are averaged over pixels rather than over instances. Our best found learning rate is $lr = 10^{-3}$, and has been implemented with the "poly" learning rate decay taken from [39, 38, 36] and their own sources. Compared to them, we apply the decay by epochs instead of iterations, but we keep the same power of 0.9. Hence, the learning rate at k^{th} epoch is

Input : $\hat{m}_{ij}$ the predicted segmentation mask
$\mathbf{e}_{ij}$ the predicted embedding in pixel (i,j)
Result: $O \in [\![0;2]\!]^{H \times W}$ teams occupancy map (0 is associated to the background)

$\mathcal{N}(i,j)$ neighborhood of pixel (i,j)
$V_{ij} \leftarrow \frac{1}{|\mathcal{N}(i,j)|} \sum_{(k,l) \in \mathcal{N}(i,j)} \|\mathbf{e}_{ij} - \mathbf{e}_{kl}\|_2^2$
$n \leftarrow 0$ the team counter
$\mathcal{R} \leftarrow \mathcal{S} \leftarrow \{(i,j) \mid \hat{m}_{ij} > 0.5\}$ the pixels to cluster
$O \leftarrow D_1 \leftarrow D_2 \leftarrow \mathbf{0}^{H \times W}$
while $\mathcal{R} \neq \{\}$ *and* $n < 2$ **do**
  $n \leftarrow n + 1$
  $(i,j) \leftarrow \text{argmin}_{(i,j) \in \mathcal{R}} V_{ij}$
  $\mathbf{c}_n \leftarrow \mathbf{e}_{ij}$ the centroid for team n
  $\mathcal{M}_n \leftarrow \{(i,j) \in \mathcal{R} \mid \|\mathbf{e}_{ij} - \mathbf{c}_n\|_2^2 < 1\}$
  $\mathbf{c}_n \leftarrow \frac{1}{|\mathcal{M}_n|} \sum_{(i,j) \in \mathcal{M}_n} \mathbf{e}_{ij}$
  $\mathcal{M}_n \leftarrow \{(i,j) \in \mathcal{R} \mid \|\mathbf{e}_{ij} - \mathbf{c}_n\|_2^2 < 1\}$
  $\mathcal{R} \leftarrow \mathcal{R} \setminus \mathcal{M}_n$
  for $(i,j) \in \mathcal{S}$ **do**
    $D_n(i,j) \leftarrow \|\mathbf{e}_{ij} - \mathbf{c}_n\|_2^2$
  end
end
for $(i,j) \in \mathcal{S}$ **do**
  if $n = 1$ **then**
    $O(i,j) \leftarrow 1$
  else if $n = 2$ **then**
    $O(i,j) \leftarrow \text{argmin}(\{D_1(i,j); D_2(i,j)\}) + 1$
  end
end

Algorithm 1: Simple clustering algorithm of pixels in the space of their associated embeddings. Up to two centroids are searched and refined, from the observation that for a team, embeddings of neighbour pixels can serve as initial prototype when they are similar. Embeddings similarity in the neighborhood of pixel (i,j), $\mathcal{N}(i,j)$, is called V_{ij}. After that, points are clustered according to their embedding vector's distance to centroids mapped in D_n arrays.

$lr.(1 - \frac{k}{max})^{power}$, with $max = 200$ denoting the total number of epochs, and lr being the base learning rate defined above. All but last layers of ICNet are initialized with pretrained Cityscapes ([8]) weights from [38], but a full training is done as the point of view adopted for sport field coverage is too different from the frontal point of view considered by cars in Cityscapes. Minibatch size is 16 and batch-normalization is applied. Neither weight decay regularization, nor dropout are added, but the following random data augmentation is considered: mirror flipping, central rotation of maximum 10 degrees, scaling such that $\min(\text{width}, \text{height}) = \text{random}([\frac{2}{3}, \frac{3}{2}]) \times 512$, color jitter in the perceptually uniform CIE L*C*h color space fixed to L ± 10, C ± 7 and h ± 30 degrees, to keep natural colors. We trained the network on crops of 512×512 pixels, located randomly in scaled images. Validation is performed on 512×512 pixels patches, extracted from images scaled such as its $\min(\text{width}, \text{height})$ equals 512. For each model, we select the parameters of the best epoch according to a validation score defined as the mean of intersection over union of the two teams, between prediction and our approximate reference masks. Inference for testing is done on court images downsampled to 1024×512 and padded to preserve the aspect ratio.

In our implementation, we adopted 5-D embeddings, mainly because more dimensions a priori get more ability to capture/encode visual team characteristics unambiguously. We expect this ability to become especially useful when the receptive field does not cover the whole scene. In that case, the embedding prediction in one pixel may not be able to rely on a teammate appearance or on the absence of collision with an opponent embedding when those players are far and disconnected from the pixel of interest. The embeddings have thus to be consistent across the scene, despite their relatively local receptive field. In other words, they have to capture local team characteristics unambiguously. In practice, ICNet builds a global receptive field, and our trials provided similar results with 1- to 5-D embeddings.

4. Experimental validation

To assess our method, this section first introduces an original dataset, and associated evaluation metrics. It then runs a K-fold cross-validation procedure, and compares the performance of our associative embedding team discrimination, with a conventional color histogram clustering, applied on top of instance segmentation.

4.1. Dataset characteristics

To demonstrate our solution, we have considered a proprietary basketball dataset. It involves a large variety of games and sport halls: images come from 40 different games and 27 different arenas. Images show innumerable situations: occlusions between teammates and/or opponents, regular player distribution, absence or presence of all the players, images from training sessions and professional games with public, various game actions, still and moving players, presence of referees, managers, mascots, dynamic led advertisements, photographers or other humans, various lighting conditions, different image sizes (smaller dimension is generally close or superior to 1000 pixels). This dataset is composed of 648 images covering a bit more than half of the sport field. Each player has been manually annotated. Annotations considered in our work

include a team label (Team A vs. Team B), and an approximate player mask. This mask has been derived from manual annotation of head, pelvis, and feet. It consists in seven ellipses approximately covering the head, the body (between head and pelvis), the pelvis, the legs (between pelvis and each foot), and the feet. Occlusions between ellipses of players located at different depth has been taken into account. Similarly to [7], our experiments reveal that the network can learn despite the coarseness of the masks. Players size in images feeding the network (scaling strategy in Section 3.3) is around $25 \times 75 \pm 15$ pixels.

4.2. Evaluation metrics

Our network enables player segmentation, as well as team discrimination. Evaluation metrics should thus reflect whether players have been properly detected, and whether teammates have received the same team label. Therefore, we consider the following counters and metrics, to be computed on a set of test images:

- N_{miss}: Number of missing players
- N_{corr}: Number of correct team associations
- N_{err}: Number of incorrect team associations
- Missed players rate,

$$R_{miss} = \frac{N_{miss}}{N_{corr} + N_{err} + N_{miss}} \tag{5}$$

- Correct team assignments rate,

$$R_{CTA} = \frac{N_{corr}}{N_{corr} + N_{err}} \tag{6}$$

We now explain how the outputs of our network, namely the player segmentation mask and the map of team labels derived from the embeddings clusters, are turned into those evaluation metrics[1]. Given a reference segmentation mask and a team label for each player instance, a simple majority vote strategy is adopted. A player is considered to be detected when the majority of pixels in the player instance segmentation mask are part of the segmentation mask predicted by the network. In that case, the majority label observed in the instance mask defines the team of the player. In practice, since our ground-truth mask only provides a rough approximation of the actual player instance silhouette, we resort to the part of the instance mask that is the most relevant for team classification, *i.e.* to the two ellipses that respectively cover the body and the pelvis area. Since pixels that are in the central part of the body and pelvis ellipses are less likely to be part of the background, only the pixels that are sufficiently close to the main principal axis of the body/pelvis shape are considered. (A distance threshold equal to one third of the maximal distance between ellipse border and principal axis has been adopted. Changing this threshold does not impact significantly the results.)

[1] Since accurate ground truth segmentation masks are not available from the dataset (see Section 4.1), the segmentation quality can not be assessed based on conventional intersection over union metrics.

Fold	1 .. 3	4	5 .. 9	10
Train	516	518	520	518
Val	66	64	64	66
Test	66	66	64	64

Table 1: Splits of the dataset used for cross-game validation. Each column corresponds to one (set of) folds, and lines define the number of training/validation/test samples. Validation and test sets contain the images from 4 games.

4.3. Results

In order to validate the proposed team discrimination method with available data, we consider a K-fold cross-validation framework. It partitions the 648-images dataset into K disjoint subsets, named folds. Each K-fold iteration preserves one fold for the test, and use the other folds for training and validation. Average and standard deviation metrics can then be computed based on the K iterations of the training/testing procedure. In our case, ten folds of approximately equal size have been considered. Moreover, to assess whether the model generalizes properly on new games and new arenas, we construct the folds so that each fold contains images from distinct games and/or arenas. Table 1 lists cross-game folds characteristics, and Table 2 cross-arena folds characteristics.

To estimate the value to give to our results, we compare them to a baseline reference. Since most previous methods recognize teams based on color histograms [10, 18, 31], generally after team-specific training, we compare associative embeddings to a method that collects color histograms on player instances, before clustering them into two sets. In practice, as for the associative embedding evaluation, only the player pixels that are sufficiently close to the body/pelvis principal axis are considered to build the histogram in RGB, with 8 bins per dimension (512-dimensional histogram). Adopted clustering is the [26] implementation of variational inference algorithm with a Dirichlet process prior [4], to fit at max two gaussians representing our two clusters (two teams). This method has the advantage of being able to automatically reduce the number of prototypes, it is useful when less than two teams are visible in an image.

Results of cross-game validation are presented in Table 3, while cross-validation on sport halls is presented in Table 4. Standard deviations are low, demonstrating the weak dependence to a specific set of training data. Rate of

Fold	1	2 .. 5	6	7	8 .. 9	10
Train	514	516	518	522	524	518
Val	66	66	64	62	62	68
Test	68	66	66	64	62	62

Table 2: Splits of the dataset used for cross-arena validation. Each column corresponds to one (set of) folds, and lines define the number of training/validation/test samples. Validation and test sets contain the images from 2 or 3 halls.

Method	R_{miss}	R_{CTA}
Associative Embedding	0.11 ± 0.04	0.91 ± 0.04
Color Histogram		0.62 ± 0.02

Table 3: Evaluation measures on cross-game K-fold: mean and standard deviation of missed player detection and of correct team assignment rates, for 10 folds.

Method	R_{miss}	R_{CTA}
Associative Embedding	0.11 ± 0.06	0.91 ± 0.03
Color Histogram		0.63 ± 0.02

Table 4: Evaluation measures on cross-arena K-fold: mean and standard deviation of missed player detection and of correct team assignment rates, for 10 folds.

missing detections is about 11%, which is an acceptable rate considering our backbone is the real-time ICNet model [38] with arduous indoor basketball images. It could probably be improved with a finer tuning of hyperparameters, as well as more accurate segmentation masks and a formulation that involves a class for referees (see failure cases analysis below). More recent and effective improved segmentation networks could also be considered as long as they are compatible with associative embedding.

In Figure 2, we observe that players are generally well detected but roughly segmented, probably due to our approximate training masks. However, segmentation masks are very clean compared to the background-subtracted foreground masks derived for such kind of scenes (see for example [24]). Therefore, they could advantageously replace those masks in algorithms using camera calibration to detect individual players from the segmentation mask [1, 5, 9].

In terms of team assignment, [17] mentions that they can not achieve good cross-game team assignment without fine-tuning. In comparison, our method reaches more than 90% of correct team assignments while testing on games and sport halls that are not seen during training. The baseline Bayesian color histogram clustering only reaches 62% of correct team assignments, which confirms that the team assignment task in the context of indoor sport is extremely difficult, as described in Section 1. We get near identical results for cross-arena evaluation.

Qualitative results are shown in Figure 2. As written in Section 3.3, we intend to extract players only, excluding referees and other humans. Images belong to testing folds, meaning that they originate from games or arenas not seen during training. Teams masks are drawn in red and blue.

The first five rows in Figure 2 illustrate how well the proposed method can deal with indoor basketball conditions. Players in fast movement and low contrast are detected and well grouped in teams. Occlusions, led advertisements, and artificial lighting are not a major problem. Associative embedding has a low sensitivity to high color similarities between background and foreground. Specific treacherous scenes with players of only one team and some other humans are correctly handled.

We estimate to 10% of the number of annotated players, the quantity of isolated regions that could fit humans, extracted in addition to reference instances. These detections come from referees and other unwanted persons on or close to the ground, and in certain cases from scenery elements. In basketball, the proportion of the number of referees related to the players is from 20 to 30% (we usually count 2 or 3 referees in a complete field, while players are $5 + 5$). Thus, it is interesting to see that our FCN trained on players generally avoids referees and other people. However, this is a challenging task, as can be seen in the two prominent failure cases shown in the last two rows of Figure 2, where referees shirts or pants are visually similar to a team. In the first example, a referee is detected as a player and included in a team (referee on the right, under the basket), and a player is filtered from predicted player class probably because it is seen as a referee by the network (background player in side of a referee). In the second example, the dark pants of a referee and a coach in the back of the court are assimilated to the team in black. This sample also presents a severe occlusion implying four players; inside and around this area, detection is inaccurate and team assignment of the orange player mixed with black teammates is lost.

5. Conclusion

Associative embedding is considered to address the team assignment problem in team sport competitions. It offers the advantage of discriminating teams in sport scenes, without requiring an unpractical per-game training. Promising results are obtained on a challenging basketball dataset, with few tuning and only approximate player mask annotations. In this work, the embeddings come with a player segmentation mask from a relatively simple multi-scale CNN, rather than the stacked hourglass network considered in previous works [15, 21, 22]. Our work could be extended to support instance segmentation, by using either instance embeddings [22] or projective geometry [1, 5, 9]. Future investigations of interest include the explicit recognition of referees, a deeper analysis of the embeddings distribution

Figure 2: Team discrimination with associative embedding. From left to right: test image, zoomed reference masks and prediction. The first five rows present success cases, while the last two show failure cases. From top to bottom: running players; strong shadows; occlusions; court and teams share the same colors; only one team; confusion between players and referees; extreme occlusions. Please refer to the numerical version of the paper for the colors and ability to zoom on details.

and a more careful weighting of losses [13].

References

[1] A. Alahi, L. Jacques, Y. Boursier, and P. Vandergheynst. Sparsity driven people localization with a heterogeneous network of cameras. *Journal of Mathematical Imaging and Vision*, 41(1):39–58, Sep 2011. 7

[2] V. Badrinarayanan, A. Kendall, and R. Cipolla. SegNet: a deep convolutional encoder-decoder architecture for image segmentation. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 39(12):2481–2495, Dec 2017. 2

[3] A. Bialkowski, P. Lucey, P. Carr, S. Sridharan, and I. Matthews. *Representing Team Behaviours from Noisy Data Using Player Role*, pages 247–269. Springer International Publishing, Cham, 2014. 1

[4] D. M. Blei and M. I. Jordan. Variational inference for dirichlet process mixtures. *Bayesian Anal.*, 1(1):121–143, 03 2006. 6

[5] P. Carr, Y. Sheikh, and I. Matthews. Monocular object detection using 3d geometric primitives. In *Computer Vision – ECCV 2012*, pages 864–878, Berlin, Heidelberg, 2012. Springer Berlin Heidelberg. 7

[6] J. Chen, H. M. Le, P. Carr, Y. Yue, and J. J. Little. Learning online smooth predictors for realtime camera planning using recurrent decision trees. In *The IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, June 2016. 1

[7] A. Cioppa, A. Deliege, and M. Van Droogenbroeck. A bottom-up approach based on semantics for the interpretation of the main camera stream in soccer games. In *The IEEE Conference on Computer Vision and Pattern Recognition (CVPR) Workshops*, June 2018. 1, 2, 6

[8] M. Cordts, M. Omran, S. Ramos, T. Rehfeld, M. Enzweiler, R. Benenson, U. Franke, S. Roth, and B. Schiele. The cityscapes dataset for semantic urban scene understanding. In *Proc. of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2016. 5

[9] D. Delannay, N. Danhier, and C. De Vleeschouwer. Detection and recognition of sports(wo)men from multiple views. In *2009 Third ACM/IEEE International Conference on Distributed Smart Cameras (ICDSC)*, pages 1–7, Aug 2009. 7

[10] T. D'Orazio, M. Leo, P. Spagnolo, P. L. Mazzeo, N. Mosca, M. Nitti, and A. Distante. An investigation into the feasibility of real-time soccer offside detection from a multiple camera system. *IEEE Transactions on Circuits and Systems for Video Technology*, 19(12):1804–1818, 2009. 1, 6

[11] S. Gerke, A. Linnemann, and K. Mller. Soccer player recognition using spatial constellation features and jersey number recognition. *Computer Vision and Image Understanding*, 159:105 – 115, 2017. Computer Vision in Sports. 2

[12] J. Hobbs, P. Power, L. Sha, and P. Lucey. Quantifying the value of transitions in soccer via spatiotemporal trajectory clustering. In *MIT Sloan Sports Analytics Conference*, 2018. 1

[13] A. Kendall, Y. Gal, and R. Cipolla. Multi-task learning using uncertainty to weigh losses for scene geometry and semantics. In *The IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, June 2018. 9

[14] D. P. Kingma and J. Ba. Adam: A method for stochastic optimization. *CoRR*, abs/1412.6980, 2015. 4

[15] H. Law and J. Deng. CornerNet: detecting objects as paired keypoints. In *The European Conference on Computer Vision (ECCV)*, September 2018. 3, 4, 7

[16] J. Liu and P. Carr. *Detecting and Tracking Sports Players with Random Forests and Context-Conditioned Motion Models*, pages 113–132. Springer International Publishing, Cham, 2014. 1

[17] K. Lu, J. Chen, J. J. Little, and H. He. Lightweight convolutional neural networks for player detection and classification. *Computer Vision and Image Understanding*, 172:77 – 87, 2018. 1, 2, 7

[18] W.-L. Lu, J.-A. Ting, J. J. Little, and K. P. Murphy. Learning to track and identify players from broadcast sports videos. *IEEE transactions on pattern analysis and machine intelligence*, 35(7):1704–1716, 2013. 1, 6

[19] M. Manafifard, H. Ebadi, and H. Abrishami Moghaddam. A survey on player tracking in soccer videos. *Computer Vision and Image Understanding*, 159:19 – 46, 2017. Computer Vision in Sports. 1

[20] D. Mazzini. Guided upsampling network for real-time semantic segmentation. In *The British Machine Vision Conference (BMVC)*, September 2018. 2

[21] A. Newell and J. Deng. Pixels to graphs by associative embedding. In *Advances in Neural Information Processing Systems 30*, pages 2171–2180. Curran Associates, Inc., 2017. 2, 3, 4, 7

[22] A. Newell, Z. Huang, and J. Deng. Associative embedding: End-to-end learning for joint detection and grouping. *CoRR*, abs/1611.05424, 2017. 1, 2, 3, 4, 7

[23] P. Parisot and C. De Vleeschouwer. Consensus-based trajectory estimation for ball detection in calibrated cameras systems. *Journal of Real-Time Image Processing*, Sep 2016. 1

[24] P. Parisot and C. De Vleeschouwer. Scene-specific classifier for effective and efficient team sport players detection from a single calibrated camera. *Computer Vision and Image Understanding*, 159:74 – 88, 2017. Computer Vision in Sports. 1, 7

[25] A. Paszke, A. Chaurasia, S. Kim, and E. Culurciello. Enet: A deep neural network architecture for real-time semantic segmentation. *CoRR*, abs/1606.02147, 2016. 2

[26] F. Pedregosa, G. Varoquaux, A. Gramfort, V. Michel, B. Thirion, O. Grisel, M. Blondel, P. Prettenhofer, R. Weiss, V. Dubourg, J. Vanderplas, A. Passos, D. Cournapeau, M. Brucher, M. Perrot, and E. Duchesnay. Scikit-learn: Machine learning in Python. *Journal of Machine Learning Research*, 12:2825–2830, 2011. 6

[27] R. P. K. Poudel, U. Bonde, S. Liwicki, and C. Zach. ContextNet: exploring context and detail for semantic segmentation in real-time. In *The British Machine Vision Conference (BMVC)*, September 2018. 2

[28] O. Russakovsky, J. Deng, H. Su, J. Krause, S. Satheesh, S. Ma, Z. Huang, A. Karpathy, A. Khosla, M. Bernstein, A. C. Berg, and L. Fei-Fei. ImageNet large scale visual recognition challenge. *International Journal of Computer Vision*, 115(3):211–252, Dec 2015. 2

[29] M. P. Shah. Semantic segmentation architectures implemented in pytorch. *https://github.com/meetshah1995/pytorch-semseg*, 2017. 4

[30] G. Thomas, R. Gade, T. B. Moeslund, P. Carr, and A. Hilton. Computer vision for sports: Current applications and research topics. *Computer Vision and Image Understanding*, 159:3 – 18, 2017. Computer Vision in Sports. 1

[31] X. Tong, J. Liu, T. Wang, and Y. Zhang. Automatic player labeling, tracking and field registration and trajectory mapping in broadcast soccer video. *ACM Trans. Intell. Syst. Technol.*, 2(2):15:1–15:32, Feb. 2011. 1, 6

[32] C. Vondrick, A. Shrivastava, A. Fathi, S. Guadarrama, and K. Murphy. Tracking emerges by colorizing videos. In V. Ferrari, M. Hebert, C. Sminchisescu, and Y. Weiss, editors, *Computer Vision – ECCV 2018*, pages 402–419, Cham, 2018. Springer International Publishing. 3

[33] X. Wei, L. Sha, P. Lucey, P. Carr, S. Sridharan, and I. Matthews. Predicting ball ownership in basketball from a monocular view using only player trajectories. In *The IEEE International Conference on Computer Vision (ICCV) Workshops*, December 2015. 1

[34] Z. Wu, C. Shen, and A. van den Hengel. Real-time semantic image segmentation via spatial sparsity. *CoRR*, abs/1712.00213, 2017. 2

[35] F. Yang, W. Choi, and Y. Lin. Exploit all the layers: Fast and accurate cnn object detector with scale dependent pooling and cascaded rejection classifiers. In *The IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, June 2016. 2

[36] C. Yu, J. Wang, C. Peng, C. Gao, G. Yu, and N. Sang. BiSeNet: bilateral segmentation network for real-time semantic segmentation. In *The European Conference on Computer Vision (ECCV)*, September 2018. 2, 4

[37] F. Yu and V. Koltun. Multi-scale context aggregation by dilated convolutions. In *ICLR*, 2016. 2

[38] H. Zhao, X. Qi, X. Shen, J. Shi, and J. Jia. ICNet for real-time semantic segmentation on high-resolution images. In *The European Conference on Computer Vision (ECCV)*, September 2018. 2, 3, 4, 5, 7

[39] H. Zhao, J. Shi, X. Qi, X. Wang, and J. Jia. Pyramid scene parsing network. In *The IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, July 2017. 2, 4

[40] S. Zheng, Y. Yue, and J. Hobbs. Generating long-term trajectories using deep hierarchical networks. In *Advances in Neural Information Processing Systems*, pages 1543–1551, 2016. 1