Real-time CNN-based segmentation architecture for ball detection in a single view setup

Anonymous Author(s)
Submission Id: 3

Figure 1: Rutgers vs. Fordham at The RAC, 2017 — Spot the ball.

Abstract

This paper considers the task of detecting the ball from a single viewpoint in the challenging but common case where the ball interacts frequently with players while being poorly contrasted with respect to the background. We propose a novel approach by formulating the problem as a segmentation task solved by an efficient CNN architecture. To take advantage of the ball dynamics, the network is fed with a pair of consecutive images. Our inference model can run in real time without the delay induced by a temporal analysis. We also show that test-time data augmentation allows for a significant increase the detection accuracy. As an additional contribution, we publicly release the dataset on which this work is based.

Keywords

CNN, ball detection, basketball, single viewpoint, low-latency, real-time, dataset, neural networks

ACM Reference Format:
Anonymous Author(s). 2019. Real-time CNN-based segmentation architecture for ball detection in a single view setup. In *Proceedings of ACM Conference (Conference'17).* ACM, New York, NY, USA, 8 pages. https://doi.org/10.1145/nnnnnnn.nnnnnnn

Conference'17, July 2017, Washington, DC, USA

ACM ISBN 978-x-xxxx-xxxx-x/YY/MM…$15.00
https://doi.org/10.1145/nnnnnnn.nnnnnnn

1 Introduction

Locating the ball in team sports is of major interest both to feed sport analytics [33] and to enrich broadcasted content [11], but in the context of real-time automated production of team sports events [5, 6, 9, 16], knowing the ball position with *accuracy* and *without delay* is even more critical.

In the past ten years, the task of detecting the ball has been largely investigated, leading to industrial products like the automated line calling system in tennis or the goal-line technology in soccer [12]. However, the ball detection problem remains unsolved for cases of important practical interest because of two main issues.

First, the task becomes quite challenging when the ball is partially occluded due to frequent interactions with players, as often encountered in team sports. Earlier works have generally addressed this problem by considering a calibrated multi-view acquisition setup to increase the candidates detection reliability by checking their consistency across the views [12, 17, 19, 25, 30]. To better deal with instants at which the ball is held by players, some works have even proposed to enrich multiview ball detection with cues derived from players tracking [23, 36, 37]. However, those solutions require the installation of multiple cameras around the field, which is significantly more expensive than single viewpoint acquisition systems [19] and appears difficult to deploy in many venues[1].

Second, the weakly contrasted appearance of the ball, as generally encountered in indoor environments, leads to many false detections. This problem is often handled by exploiting a ballistic trajectory prior in order to discriminate between true and false detected candidates [4, 7, 8, 15, 25, 39, 44]. In addition to this prior, some works also use the cues derived from the players tracking [23, 41], or even use the identity and team affiliation of every player in

[1] From our personal experience of 100+ installations in professional basketball arenas.

every frame [38]. However, this prior induces a significant delay, making those solutions inappropriate for real-time applications. Furthermore, they require detecting the ball at high frame rates, inducing hardware and computational constraints and making them unusable for single instants applications.

Overall, for scenarios considering instant images in indoor scenes captured from a single view point, the ball detection problem remains largely unsolved. This is especially true for basketball (involving fast dynamics and many players interactions) where state-of-the-art solutions saturates around 40% detection rate, restricted in the basket area [26]. Those limitations force current real-time game analysis solutions to rely on a connected ball [34], which requires to insert a transmitter within the ball ; or use multiple cameras and multiple servers to process the different data streams [12, 19, 27].

In this paper, we propose a learning-based method reaching close to 70% ball detection rate on unseen venues, with very few false positives, demonstrated on basketball images. In order to train and validate our method, we created a new dataset of single view basketball scenes featuring many interactions between the ball and players, and low-contrasted backgrounds.

Similar to most modern image analysis solutions, our method builds on a Convolutional Neural Network (CNN) [18]. In earlier works, several CNNs have been designed to address the object detection problem, and among the most accurate, *Mask R-CNN* [13] has been trained to detect a large variety of objects, including balls. The experiments presented in Section 4.2 reveal that applying the universal *Mask R-CNN* detector to our problem largely fails. However, fine tuning the pre-trained weights to deal with our dataset significantly improves the detection performance (despite staying $\sim$ 5% below our method). However, *Mask R-CNN* is too complex to run in real-time on an affordable architecture.

To provide a computationally simple alternative to *Mask R-CNN*, a few works have designed a CNN model to specifically detect the ball in a sports context. [31] adopts a *classification* strategy by splitting the image in a grid of overlapping patches, each of which is fed to a CNN that assesses whether or not it contains a ball. However, the model is far from real-time[2], and only two different games are considered to train and validate the method. In the Robocup Soccer context, [32] formulates the detection problem as a *regression* task aiming at predicting the coordinates of the ball in the image. The performance remains relatively poor despite the reasonable contrast between the white ball and the green field. Furthermore, regression of object coordinates is known to be poorly addressed by CNNs [22]. Hence, this strategy is expected to poorly generalize to real team sport scenes.

In comparison to those initial attempts to exploit CNNs to detect the ball, the contributions of our work are multiple and multifaceted.

Primarily, we propose to formulate the ball detection problem as a *segmentation* problem, for which CNNs are known to be quite effective [10]. Such formulation is especially relevant in our team sport ball detection context since there is only one object-of-interest that we aim to detect. Hence, the CNN does not need to handle the object instantiation problem. In addition, we use a pair of consecutive images coming from a fixed (or motion compensated) viewpoint to take advantage of the ball dynamics without the delay caused by a temporal regularization, allowing low-latency applications. We show that this approach allows fast and reliable ball detection in weakly contrasted or cluttered scenes.

Furthermore, we show that the use of test-time data augmentation[3] permits a significant increase in the detection accuracy at small false positive rates. This is in the continuation of recent works showing that test-time data augmentation can be used to estimate the prediction uncertainty of a CNN model [3, 35].

Finally, as we believe it can help research in sport scene analysis, we plan to release our dataset with the non-anonymous version of this paper. Indeed, beyond providing pairs of consecutive images, to our knowledge, it is the only dataset to offer a representative sample of professional basketball images gathered on no less than 30 different arenas around the world. It features a large variety of game actions and lighting conditions, cluttered scenes and complex backgrounds. This solves a weakness of several previous works suffering from a limited set of validation data, having very few different games considered [26, 31, 32].

2 Method

Our proposed solution is illustrated in Figure 2. It formulates the ball detection problem as a segmentation problem where a fully convolutional neural network is trained to output a heatmap predicting the ball segmentation mask. At inference, a detection rule is used to

[2]We measured 3.6fps on an Nvidia RTX 2080 Ti with an overlap of 10 pixels between the patches. See their paper for implementation details.

[3]Generating multiple transformed versions of an input sample, in order to predict an ensemble of outputs for that input sample

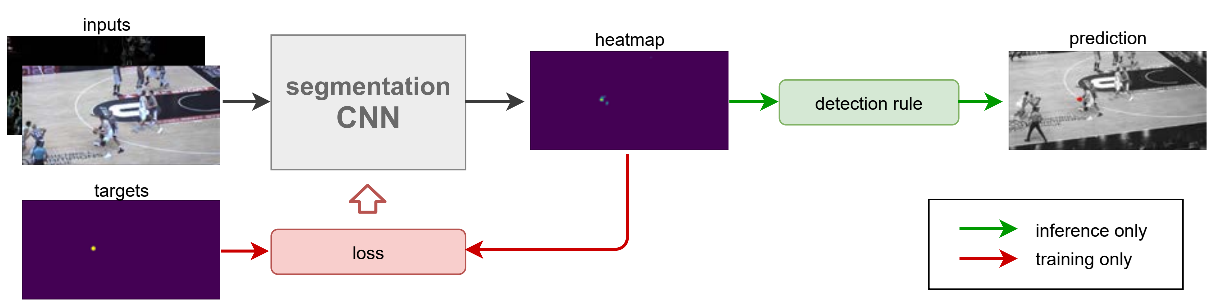

Figure 2: Our detector is based on a segmentation task performed by a fully convolutional network that outputs a heatmap of the ball position. At inference, a detection rule is used to predict the ball location from the heatmap.

extract a ball candidate from the heatmap. The fully convolutional workflow makes it possible to work with any input sizes.

CNN implementation. Among fully convolutional networks, multi-scale architectures are known to better trade-off complexity and accuracy on large images by combining wide shallow branches that manipulate fine image details through a reduced number of layers, with deep narrow branches that access smaller images and can thus afford a deeper sequence of layers [24, 28, 40, 43]. The major benefit of this tradeoff is a fast processing allowing real-time applications.

In practice, the CNN used to evaluate our method is the ICNet implementation available at [14] except that the 3 input resolutions were changed from $\{(1), (1/2), (1/4)\} \times \text{input size}$ to $\{(1), (1), (1/2)\} \times \text{input size}$ in order to better handle the small size of the ball at the lowest resolutions. The output heatmap is produced by applying a softmax on the last layer and removing the channel relative to the background.

From a computational point of view, the selected CNN offers satisfying real-time performances (see Section 4.2 Table 1) but any segmentation network could be used instead; and we could expect even higher detection speed using recent fast segmentation networks [24, 28, 40].

Detection rule. We infer ball candidates as points lying above a threshold τ in the predicted heatmap. Since we know *a priori* there is only one ball in the scene, the detection rule is further constrained by limiting the number of ball candidates per scene. This approach, named top-k, detects (up to) the k highest spots in the heatmap, as long as they are higher than the threshold τ. In practice, to avoid multiple detections for the same heatmap hotspot, highest points in the heatmap are selected first, and their surrounding pixels are ignored for subsequent detections, in a greedy way. Note that the computational cost associated with this detection rule is negligible compared to the CNN inference.

Exploiting the ball dynamics. When not held by a player, the ball generally moves rapidly, either due to a pass between players, a shot, or dribbles. To provide the opportunity to exploit motion information, we propose to feed the network with the information carried by two consecutive images denoted $\mathcal{I}_a$ and $\mathcal{I}_b$, where each image is composed of the 3 conventional channels in the *RGB* space. Two strategies were considered. The first one, consisting in feeding the network with the concatenation of $\mathcal{I}_a$ and $\mathcal{I}_b$ on the channels axis, gave poor results and is not presented in this work. The second strategy consists in concatenating the image of interest together with its difference with the previous image. Hence:

$$\text{Input} = \left(R_{\mathcal{I}_a}, G_{\mathcal{I}_a}, B_{\mathcal{I}_a}, |R_{\mathcal{I}_a} - R_{\mathcal{I}_b}|, |G_{\mathcal{I}_a} - G_{\mathcal{I}_b}|, |B_{\mathcal{I}_a} - B_{\mathcal{I}_b}|\right) \tag{1}$$

Training. The training is done using the Stochastic Gradient Descent optimization algorithm applied on the mean cross-entropy loss, at the pixel level, between the output heatmap and the binary segmentation mask of the ball. The meta parameters were selected based on grid searches. The learning rate has been set to 0.001 (decay by a factor of 2 every 40 epochs), batch size to 4, and number of epochs to 150 in all our experiments. The weights obtained at the iteration with the smallest error on the validation set were kept for testing. The network is fed with 1024×512 pixels inputs obtained by a data augmentation process including mirroring (around vertical axis), up- and down- scaling (maintaining the ball size between 15 and 45 pixels, which corresponds to the size range of balls observed by the cameras used to acquire the dataset) and cropping (keeping the ball within the crop).

Terminology. We will use the term **scene** when referring to the original image captured by the camera, and **random-crop** to denote a particular instance of random cropping, scaling, and mirroring parameters. Multiple different random crops can thus be extracted from a single scene.

3 Validation Methodology

Dataset. The experiments were conducted on a rich dataset of 280 basketball scenes coming from professional games that occurred in 30 different arenas on multiple continents. The cameras captured half of a basketball court and have a resolution between 2Mpx and 5Mpx. The resulting images have a definition varying between 65px/m (furthest point on court in the arena with the lowest resolution cameras) and 265px/m (closest point on court in the arena with the highest resolution cameras). For each scene, two consecutive images were captured. The delay between those two captures is 33ms or 40ms, depending on the acquisition frame rate. As shown in Figure 6, the dataset presents a large variety of game configurations and various lighting conditions. The ball was manually annotated in the form of a binary segmentation mask. In all images, at least half of the ball is visible.

K-fold testing. To validate our work, a K-fold training/testing strategy has been adopted in all our experiments. It partitions the dataset into K subsets, named folds, and runs K iterations of the training/testing procedure. Each iteration preserves one fold for testing, and shuffle the other folds before splitting the samples in 90% for the training set and 10% for the validation set. To assess the generalization capabilities of our model to unseen games and arenas, the K folds were defined so that each fold only contains images from arenas that are not present in the other folds. In practice, K has been set to 7, which means that each fold contains about 40 scenes coming from 3 or 4 different arenas.

ROC curves and metrics. For a given top-k detection rule, the accuracy is assessed based on ROC curves. Each ROC plots the detection rate (TPR) as a function of the false positive rate (FPR), while progressively changing the detection threshold τ. The detection rate measures the fraction of scenes for which the ball has been detected, while the false positive rate measures the mean number of false candidates that are detected per scene. In practice, a detection is considered as being a true (false) ball detection if it lies inside (outside) the surface covered by the ball in the annotated segmentation mask.

Test-time data augmentation. Multiple different random crops from the same scene were considered at test time by aggregating their output heatmap. As the ball location is unknown at inference, the random crops that were combined were similar (IoU > 0.9). It is like providing small variations of an input in which the ball is expected to be detected (see Figure 8 (left)).

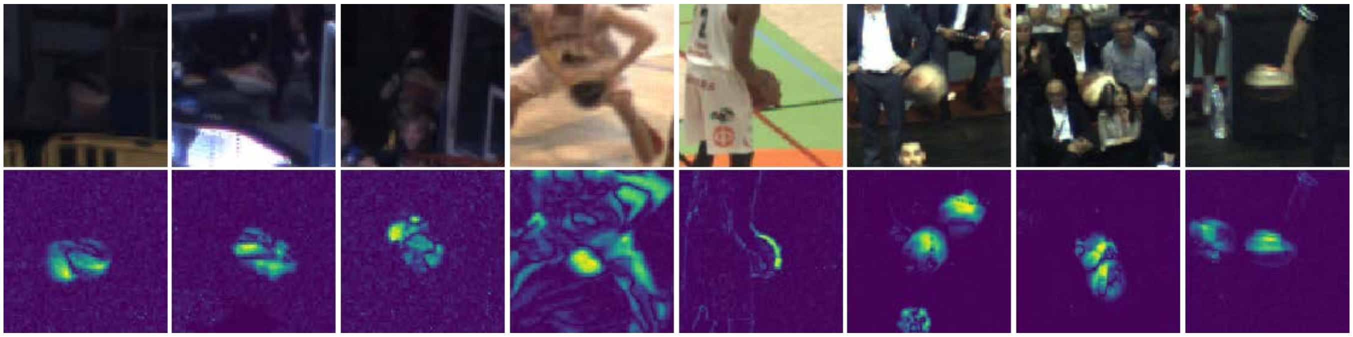

Figure 3: Image samples (top) and corresponding mean differences to previous image (bottom) for scenes in which the ball is detected when providing the difference to the previous image to the network, but remains undetected by the model that ignores the difference.

4 Experimental results

This section assesses our Ball Segmentation method (named *BallSeg*) based on the dataset introduced in Section 3. First, we validate the CNN input and the detection rule. Then, we compare *BallSeg* with the *Mask R-CNN* fine tuned with our dataset. Finally, we use test-time data augmentation to analyze how different random crops impact the detection. In addition, we take advantage of test-time data augmentation to improve detection accuracy at the cost of increased computational complexity, by merging the heatmaps associated with multiple random crops of the same scene.

4.1 Inputs choice and detection rule validation

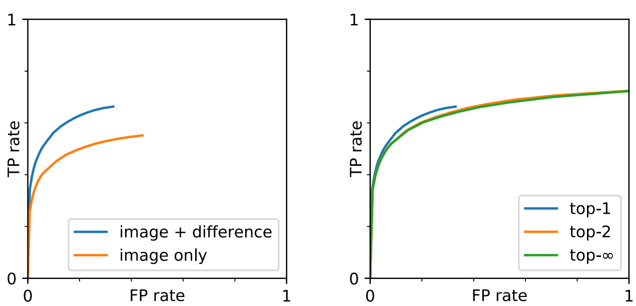

Figure 4: *BallSeg* ROC curves. Left: feeding the network with the difference to the previous image in addition to the image of interest improves accuracy (top-1 detection rule). Right: top-1 detection rule achieves better true/false positives trade-offs, than top-k strategies.

We compared the performances of our *BallSeg* when feeding the network with and without the difference between consecutive images. Figure 4 (left) shows that presenting the input as described in Eq. 1 significantly improves the detection accuracy. Figure 3 reveals that using the difference with the previous image allows detecting the ball in low contrasted scenes.

Different top-k detection rules are compared in Figure 4 (right). A top-1 detection rule achieves better true/false positive trade-offs at low false positive rates (for $\tau = 0.01$: TPR=0.66 and FPR=0.33). This is far better than all previous methods presented in the introduction, especially given that our dataset includes a large variety of scenes, presenting the ball in all kinds of game situations.

In the rest of the paper, unless specified otherwise, results are thus provided for a top-1 detection rule, using the input described by Eq. 1.

4.2 Comparison with state-of-the-art object detection method

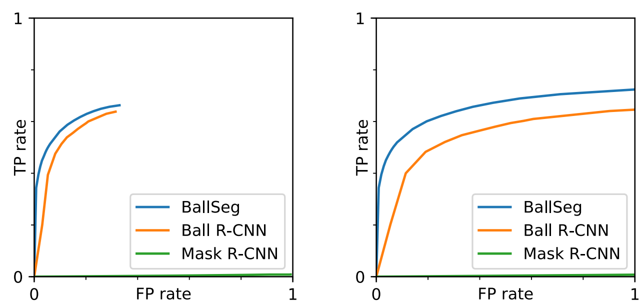

Figure 5: Comparison between *BallSeg* (our proposed method, using a fast segmentation network) and two computationally complex *Mask R-CNN* instances: *Mask R-CNN* pre-trained on MS-COCO, and the same network fine tuned on our dataset (*Ball R-CNN*). Left: top-1 detection rule; Right: top-2 detection rule.

To better evaluate the value of our *BallSeg* model, we compared it with the results obtained with an universal *Mask R-CNN* model. We used the implementation of [1] that provides weights trained on the MS-COCO dataset [21] and already has a class for the ball. For a fairest comparison, the *Mask R-CNN* network has also been fine tuned to deal with our dataset. The resulting model (denoted *Ball R-CNN*) was obtained by running a conventional optimization with stochastic gradient descent optimizer with a learning rate lr. First, we updated the front part of the network for n_f epochs, then we updated the whole network for n_w epochs. A grid search has been performed to select the (lr, n_f, n_w) triplet. Best K-fold mean test

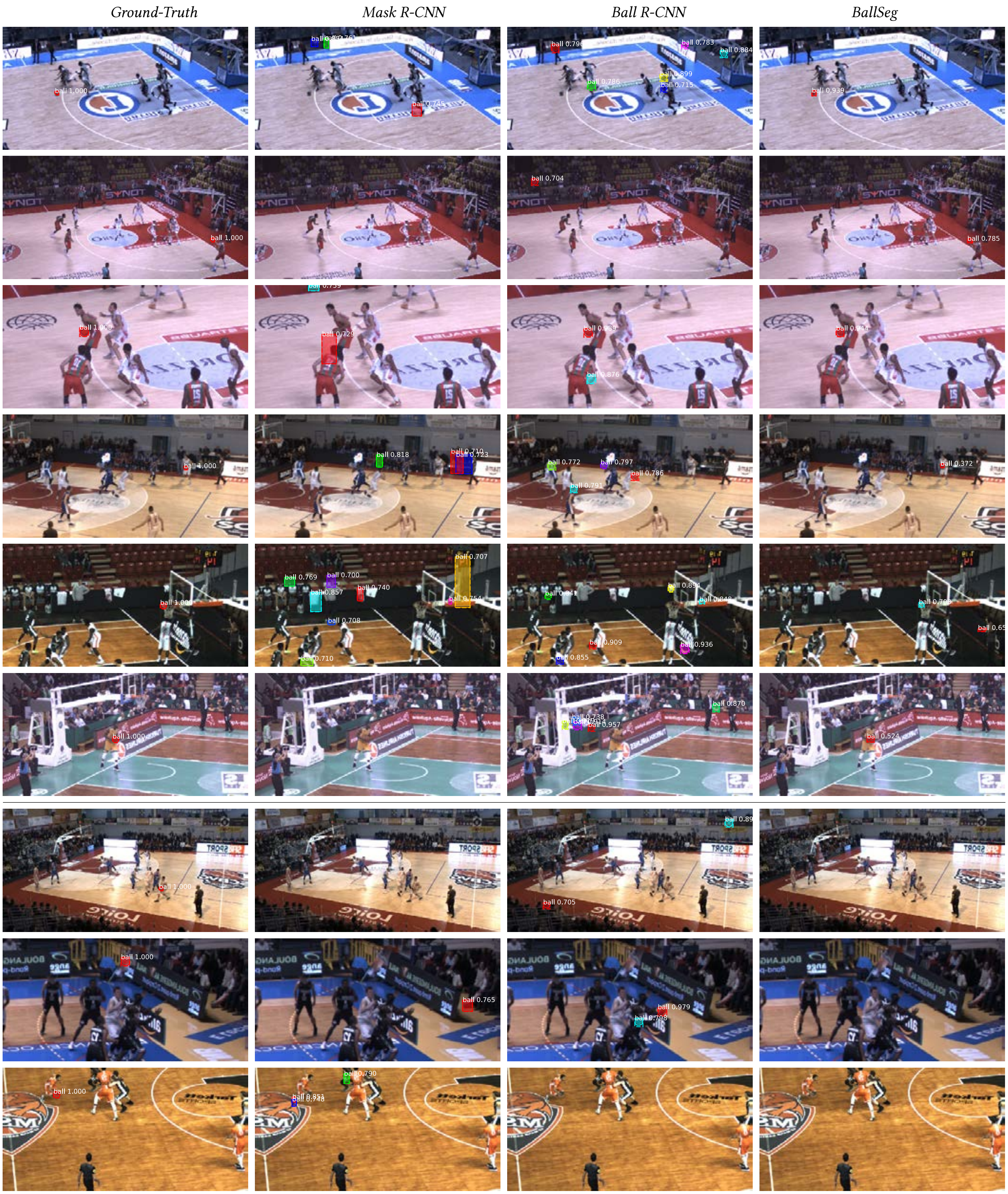

Figure 6: Our *BallSeg* gives accurate results in many different configurations. Top 6 rows: success cases. Bottom 3 rows: failure cases. First column: *Ground-Truth*. Second column: the class "ball" from the out-of-the-shelf *Mask R-CNN*. Third column: fined-tuned *Mask R-CNN* on our dataset (*Ball R-CNN*). Fourth column: our method, using a segmentation CNN (*BallSeg*).

set accuracy was obtained with $lr = 10^{-3}$, $n_f = 10$ and $n_w = 20$.

$$lr \in \{10^{-2}, 10^{-3}, 10^{-4}, 10^{-5}\}$$
$$n_f \in \{0, 1, 10\}$$
$$n_w \in \{0, 1, 10, 20, 100\}$$

In Figure 5, we observe that *BallSeg* outperforms *Ball R-CNN*. Besides, the fact that *Mask R-CNN* largely fails reveals the need to have sport-specific datasets for the complex task of ball detection in a team sport context. Figure 6 shows a visual comparison between *BallSeg*, the out-of-the-shelf *Mask R-CNN* and *Ball R-CNN* on different image samples from our dataset.

In addition to this qualitative analysis, Table 1 presents the measured computational complexity of the two models: *BallSeg* using the *ICNet* implementation from [14] and *Ball R-CNN* being *Mask R-CNN* from [1]. We observe that, besides reaching a better accuracy than *Ball R-CNN*, *BallSeg* is significantly faster. Note that those numbers highly depends on the implementation. Indeed, *ICNet* can benefit from a 6× speed improvement once compressed with filter-pruning, while keeping the same accuracy [2, 20, 42].

	single image $1024 \times 512 \times 3$	image + diff $1024 \times 512 \times 6$	image + diff $1280 \times 720 \times 6$
Mask R-CNN	4.33 fps	N/A	N/A
BallSeg (our method)	38.39 fps	24.67 fps	12.08 fps

Table 1: Framerate of the two methods compared on an Nvidia GTX 1080 Ti, with a batch size of 2. The segmentation approach is significantly faster than the state-of-the-art *Mask R-CNN*.

4.3 Test-time data augmentation

This section investigates how different random crops of the same scene impacts the detection. It first observes that the ball segmentation errors rarely affect all random crops. In other words, prediction errors correspond to high entropy output distributions, which by definition correspond to a high uncertainty. This is in line with the observation made by [3, 35] that wrong predictions are associated with high uncertainty levels, i.e. to large diversity of predictions for transformed inputs. More interestingly, our experiments also reveal that false positives are not spatially consistent across the ensemble of heatmaps. This is an important novel observation, since it gives the opportunity to significantly increase (~ 10%) the detection accuracy at small false positive rate by aggregating the ensemble of heatmaps obtained by multiple different random-crops.

4.3.1 Random crops consistency and failure case analysis Figure 7 (Left) presents the distribution, over our 280 scenes, of the percentage of random crops in which the ball is detected for top-1, top-2, and top-3 detection rules and $\tau = 0$. We observe that when the ball is not detected in all random crops (about half of the scenes), it is generally detected in some random crops. This suggests that the detection rate could be improved by combining multiple random crops (see Section 4.3.2). This opportunity is supported by Figure 7 (Right), which presents, for a top-1 detection rule, the ball detection rate as a function of the number of random crops considered for a scene. We observe that for 93% of the dataset scenes, when 20 different random crops are considered for each scene, the ball is detected as top-1 in at least one of those random crops.

4.3.2 Accuracy/complexity trade-off This section investigates how to improve the true/false positive trade-off based on the computation of more than one heatmap per scene. It builds on the fact that the ball has a higher chance to be detected if more random crops are considered (see Figure 7 (Right)), and on the observation that the ball generally induces a significant spot in the heatmap of every random crop, even if this spot is not always the highest one (see in Figure 7 (Left) where a top-2 or top-3 detection rule allows to increase the number of scenes in which the ball is detected for all random crops).

Figure 8 shows an example where the heatmaps predicted for distinct random crops of the same scene are generally more consistent at the actual ball location than for the false candidates. From this observation, we propose to merge the heatmaps of different random image samples by averaging their intensity. Figure 9 shows that applying a top-1 detection rule on the averaged heatmap significantly improves the true vs. false positive rate tradeoff (~ 10%

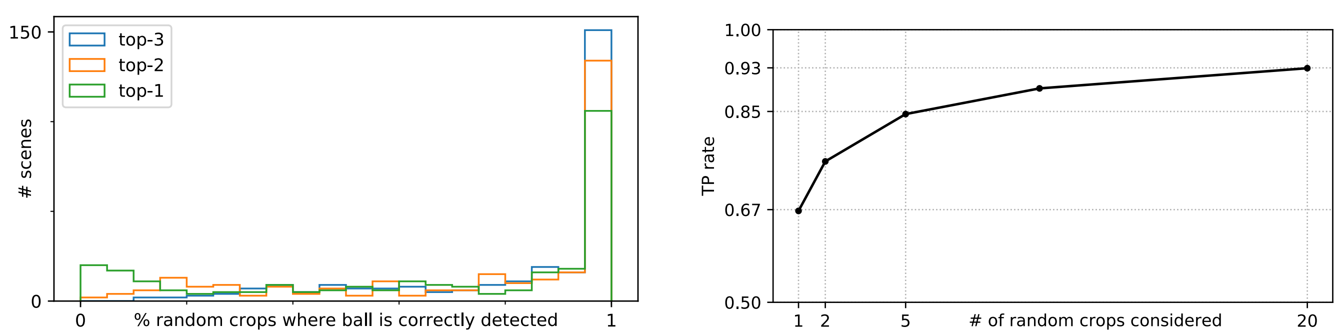

Figure 7: 1000 random-crops were created for each scene. Left: Distribution, over the 280 dataset scenes, of the percentage of random crops in which the ball is detected for top-1, top-2, and top-3 detection rules. Right: Ball detection rate as a function of the number of random crops considered for a scene, for a top-1 detection rule. The ball is detected in the scene if it is detected in at least one of the random crops considered.

Figure 8: Averaging the heatmaps over random image samples helps to discriminate the ball among candidates. Left: Five random crops having similar IoU in a given scene are used; Middle: Scatter of their heatmaps most salient points; Right: Mean heatmap intensity using the five input heatmaps.

detection rate increase at small false positive rate), already when only two random image samples are averaged.

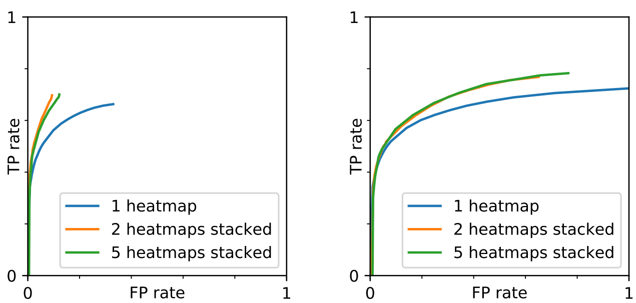

Figure 9: Left: top-1 and Right: top-2 ROC curves obtained when heatmaps coming from 1, 2 or 5 similar random crops of the same scene are averaged.

This gives the opportunity to improve accuracy at the cost of additional complexity, which in turn can support accurate automatic annotation of unlabelled data for more accurate models training, thereby following the data distillation paradigm [29].

5 Conclusion

This paper proposes a new approach for ball detection using a simple camera setup by (i) adopting a CNN-based segmentation paradigm, taking advantage of the ball uniqueness in the scene, and (ii) using two consecutive frames to give cues about the ball motion while keeping a very low latency.

Furthermore, the approach benefits from recent advances made by fast segmentation networks, allowing real-time implementation.

In particular, the segmentation network used to demonstrate the approach reduces drastically the computational complexity compared to the conventional *Mask R-CNN* detector, while achieving better results: close to 70% detection rate (with a very small false positive rate) on unseen games and arenas, based on an arduous dataset.

References

[1] Waleed Abdulla. 2017. Mask R-CNN for object detection and instance segmentation on Keras and TensorFlow. https://github.com/matterport/Mask_RCNN.

[2] Oles Andrienko. 2019. ICNet and PSPNet-50 in Tensorflow for real-time semantic segmentation. https://github.com/oandrienko/fast-semantic-segmentation.

[3] Murat Seckin Ayhan and Berens P. . 2018. Test-time Data Augmentation for Estimation of Heteroscedastic Aleatoric Uncertainty in Deep Neural Networks. In *MIDL*.

[4] B. Chakraborty and S. Meher. 2013. A real-time trajectory-based ball detection-and-tracking framework for basketball video. *Journal of Optics* (2013).

[5] F. Chen and C. De Vleeschouwer. 2010. Personalized production of basketball videos from multi-sensored data under limited display resolution. *CVIU* (2010).

[6] F. Chen, D. Delannay, and C. De Vleeschouwer. 2011. An autonomous framework to produce and distribute personalized team-sport video summaries: a basket-ball case study. *IEEE Trans. on Multimedia* 13, 6 (2011).

[7] H.-T. Chen, M.-C. Tien, Y.-W. Chen, W.-J. Tsai, and S.-Y. Lee. 2009. Physics-based ball tracking and 3D trajectory reconstruction with applications to shooting location estimation in basketball video. *Journal VCIR* (2009).

[8] Hua-Tsung Chen, Wen-Jiin Tsai, Suh-Yin Lee, and Jen-Yu Yu. 2012. Ball tracking and 3D trajectory approximation with applications to tactics analysis from single-camera volleyball sequences. *Multimedia Tools and Applications* (2012).

[9] J. Chen, H. Le, P. Carr, Y. Yue, and J.J. Little. 2016. Learning online smooth predictors for realtime camera planning. In *CVPR*.

[10] L.-C. Chen, G. Papandreou, F. Schroff, and H. Adam. 2018. Rethinking atrous convolution for semantic image segmentation. In *ECCV*.

[11] I.A. Fernandez, F. Chen, F. Lavigne, X. Desurmont, and C. De Vleeschouwer. 2010. Browsing Sport Content through an Interactive H.264 Streaming Session. In *MMEDIA*.

[12] Hawk-Eye. 2012. Hawk-Eye ball tracking system. *http://www.hawkeyeinnovations.co.uk/page/sports-officiating* (2012).

[13] K. He, G. Gkioxari, P. Dollár, and R. Girshick. 2017. Mask R-CNN. In *ICCV*.

[14] Hellochick. 2018. ICNet-tensorflow implementation. https://github.com/hellochick/ICNet-tensorflow

[15] Amit Kumar K.C., P. Parisot, and C. De Vleeschouwer. 2011. Demo: Spatio-temporal template matching for ball detection. In *ICDSC*.

[16] Keemotion. 2019. Keemotion production technology. *http://www.keemotion.com* (2019).

[17] C. Lampert and J. Peters. 2012. Real-time detection of colored objects in multiple camera streams with off-the-shelf hardware components. *Journal of Real-Time Image Processing* (2012).

[18] Y. LeCun, Y. Bengio, and G. Hinton. 2015. Deep learning. *Nature* (2015).

[19] Eric Li. 2019. "Intel TrueView" presented at CVSports19 (a CVPR 2019 Workshop) on the 17th of June, 2019.

[20] H. Li, A. Kadav, I. Durdanovic, H. Samet, and H.P. Graf. 2017. Pruning filters for efficient convnets. In *ICLR*.

[21] Tsung-Yi Lin, Michael Maire, Serge J. Belongie, Lubomir D. Bourdev, Ross B. Girshick, James Hays, Pietro Perona, Deva Ramanan, Piotr Dollár, and C. Lawrence Zitnick. 2014. Microsoft COCO: Common Objects in Context. *CoRR* abs/1405.0312 (2014). arXiv:1405.0312 http://arxiv.org/abs/1405.0312

[22] R. Liu, J. Lehman, P. Molino, F. Petroski Such, E. Frank, A. Sergeev, and J. Yosinski. 2018. An intriguing failing of convolutional neural networks and the CoordConv solution. In *NIPS*.

[23] A. Maksai, X. Wang, and P. Fua. 2016. What Players do with the Ball: A Physically Constrained Interaction Modeling. In *CVPR*.

[24] Davide Mazzini. 2018. Guided Upsampling Network for Real-Time Semantic Segmentation. In *BMVC*.

[25] P. Parisot and C. De Vleeschouwer. 2011. Graph-based filtering of ballistic trajectory. In *ICME*.

[26] Pascaline Parisot and Christophe De Vleeschouwer. 2016. Consensus-based trajectory estimation for ball detection in calibrated cameras systems. *Journal of Real-Time Image Processing* (22 Sep 2016).

[27] Gopal Sarma Pingali, Agata Opalach, and Yves Jean. 2000. Ball Tracking and Virtual Replays for Innovative Tennis Broadcasts. In *ICPR*.

[28] Rudra P. K. Poudel, Ujwal Bonde, Stephan Liwicki, and Christopher Zach. 2018. ContextNet: Exploring Context and Detail for Semantic Segmentation in Real-time. In *BMVC*.

[29] I. Radosavovic, P. Dollar, R. Girshick, G. Gkioxari, and K. He. 2018. Data Distillation: Towards Omni-Supervised Learning. *CVPR* (2018).

[30] J. Ren, J. Orwell, G.A. Jones, and M. Xu. 2008. Real-Time modeling of 3D soccer ball trajectories from multiple fixed cameras. *IEEE Transactions on Circuits and Systems for Video Technology* (2008).

[31] Vito Reno, Nicola Mosca, Roberto Marani, Massimiliano Nitti, Tiziana D'Orazio, and Ettore Stella. 2018. Convolutional Neural Networks Based Ball Detection in Tennis Games. In *CVPR*.

[32] Daniel Speck, Pablo Barros, Cornelius Weber, and Stefan Wermter. 2017. Ball Localization for Robocup Soccer Using Convolutional Neural Networks. In *RoboCup 2016: Robot World Cup XX*.

[33] G. Thomas, R. Gade, T. Moeslund, and A. Hilton. 2017. Computer vision for sports: current applications and research topics. *CVIU* (2017).

[34] R/GA Ventures. 2017. ShotTracker and Keemotion Case Study. https://www.youtube.com/watch?v=hT98F2pxMYo

[35] Guotai Wang, Wenqi Li, M. Aertsen, J. Deprest, S. Ourselin, and T. Vercauteren. 2019. Aleatoric uncertainty estimation with test-time augmentation for medical image segmentation with convolutional neural networks. *Neurocomputing* (2019).

[36] X. Wang, V. Ablavsky, H. Ben Shitrit, and P. Fua. 2014. Take your Eyes off the Ball: Improving Ball-Tracking by Focusing on Team Play. *CVIU* (2014).

[37] X. Wang, E. Türetken, F. Fleuret, and P. Fua. 2014. Tracking interacting objects optimally using integer programming. In *ECCV*.

[38] X. Wei, L. Sha, P. Lucey, P. Carr, S. Sridharan, and I. Matthews. 2016. Predicting Ball Ownership in Basketball from a Monocular View Using Only Player Trajectories. In *CVPR*.

[39] F. Yan, A. Kostin, W. Christmas, and J. Kittler. 2006. A Novel Data Association Algorithm for Object Tracking in Clutter with Application to Tennis Video Analysis. In *CVPR*.

[40] Changqian Yu, Jingbo Wang, Chao Peng, Changxin Gao, Gang Yu, and Nong Sang. 2018. BiSeNet: Bilateral Segmentation Network for Real-time Semantic Segmentation. In *ECCV*.

[41] Y. Zhang, H. Lu, and C. Xu. 2008. Collaborate ball and player trajectory extraction in broadcast soccer video. In *International Conference on Pattern Recognition*.

[42] Hengshuang Zhao. 2019. ICNet for Real-Time Semantic Segmentation on High-Resolution Images. https://github.com/hszhao/ICNet.

[43] H. Zhao, X. Qi, X. Shen, J. Shi, and J. Jia. 2018. ICNet for Real-Time Semantic Segmentation on High-Resolution Images. In *ECCV*.

[44] X. Zhou, Q. Huang, L. Xie, and S. J. Cox. 2013. A two layered data association approach for ball tracking. In *ICASSP*.