MV-Soccer: Motion-Vector Augmented Instance Segmentation for Soccer Player Tracking

Supplementary Material

Abstract

In these supplemental materials, we provide additional details of our approach. We do so not only for completeness and clarity but also for reproducibility.

Hyperparameters

The following hyperparameters were tweaked to generate good results.

1. **Learning rate:** We used stochastic gradient descent (SGD) and the Adam optimizer combined with an initial learning rate of $1e - 5$ and the final learning rate of 0.01 with momentum 0.6.
2. **Nominal Batch Size (NBS):** We tested our model on the nominal batch size 16, 4, 8, and 32 for instance segmentation. respectively.
3. **Number of epochs:** We trained our models for 100 epochs. All the results were obtained from all models with the same number of epochs.
4. **Image size:** We set the image size $608 \times 608 \times 3$ for input. We obtained better performance for larger input image sizes but required substantially more computational resources.
5. **Number of classes:** We focus on specific objects like players, referees, goalkeepers, and the ball, resulting in 4 classes.
6. **Confidence threshold:** We set the minimum confidence level of 0.25, but that is not a strict limitation as we also explored the range $[0.04, 0.4]$.
7. **Activation Function:** We use Sigmoid Linear Units (SiLU) [42] as an activation function instead of Rectified Linear Units (ReLU) in the hidden layers. Unlike ReLU, which clamps negative values to zero, SiLU applies sigmoid on all values. We also apply a nominal batch normalization (NBN) approach rather than sticking to any fixed value. We evaluated our method on different batch sizes, including $bs = 4, 8, 16, 32$ and 64. The SiLU activation is defined as follows.

$$\text{SiLU}(x) = x \cdot \sigma(x) \quad \text{where} \quad \sigma(x) = \frac{1}{1 + e^{-x}}, \tag{S.1}$$

where x represents the input to the SiLU activation function, e is the Euler number, and $\sigma(x) = \frac{1}{1+e^{-x}}$ the standard sigmoid function.

Loss Functions

Our approach uses a combination of several loss functions. The total loss is a weighted sum of the following individual losses.

Objectness loss: The objectness loss is computed using the difference between predicted and ground truth objectness scores for each bounding box. We use the binary cross-entropy loss,

$$\begin{aligned} L_{\text{obj}} = \lambda_{\text{obj}} \sum_{i=0}^{S} \sum_{j=0}^{B} \mathbb{1}_{ij}^{\text{obj}} \left(\text{IOU}_{ij} - \sigma \left(\hat{t}_{ij}^{\text{obj}} \right) \right)^2 \\ + \lambda_{\text{obj}} \sum_{i=0}^{S} \sum_{j=0}^{B} \mathbb{1}_{ij}^{\text{noobj}} \left(\text{IOU}_{ij} - \sigma \left(\hat{t}_{ij}^{\text{obj}} \right) \right)^2, \end{aligned} \tag{S.2}$$

where λ_{obj} is the weight for the objectness loss, S and B the grid size and number of bounding boxes predicted per grid cell, $\mathbb{1}_{ij}^{\text{obj}}$ is an indicator function that equals one if object j is assigned to cell i, and $\mathbb{1}_{ij}^{\text{noobj}}$ is an indicator if object j is not assigned to cell i. IOU$_{i,j}$ is the intersection-over-union (IOU) between the predicted bounding box and the ground truth box, and $\sigma(\hat{t}_{ij}^{\text{obj}})$ is the sigmoid activation of the objectness score $\hat{t}_{ij}^{\text{obj}}$, which is a prediction of the probability that object j is present in cell i.

Mask loss: We compute the difference between predicted and ground truth instance masks for each object,

$$L_{\text{mask}} = \lambda_{\text{mask}} \sum_{i=0}^{S} \sum_{j=1}^{n_a} \mathbb{1}_{ij}^{\text{obj}} \left(M_{ij} - \widehat{M}_{ij} \right)^2, \tag{S.3}$$

where S is the number of grid cells, n_a is the number of anchors per cell, $\mathbb{1}_{ij}^{obj}$ is the indicator function for the presence of an object in cell i with anchor j, M_{ij} is the predicted mask for the object in cell i with anchor j, and $\widehat{M}_{ij}$ is the ground truth mask for the object in cell i with anchor j. λ_{mask} is the weight given to the mask loss.

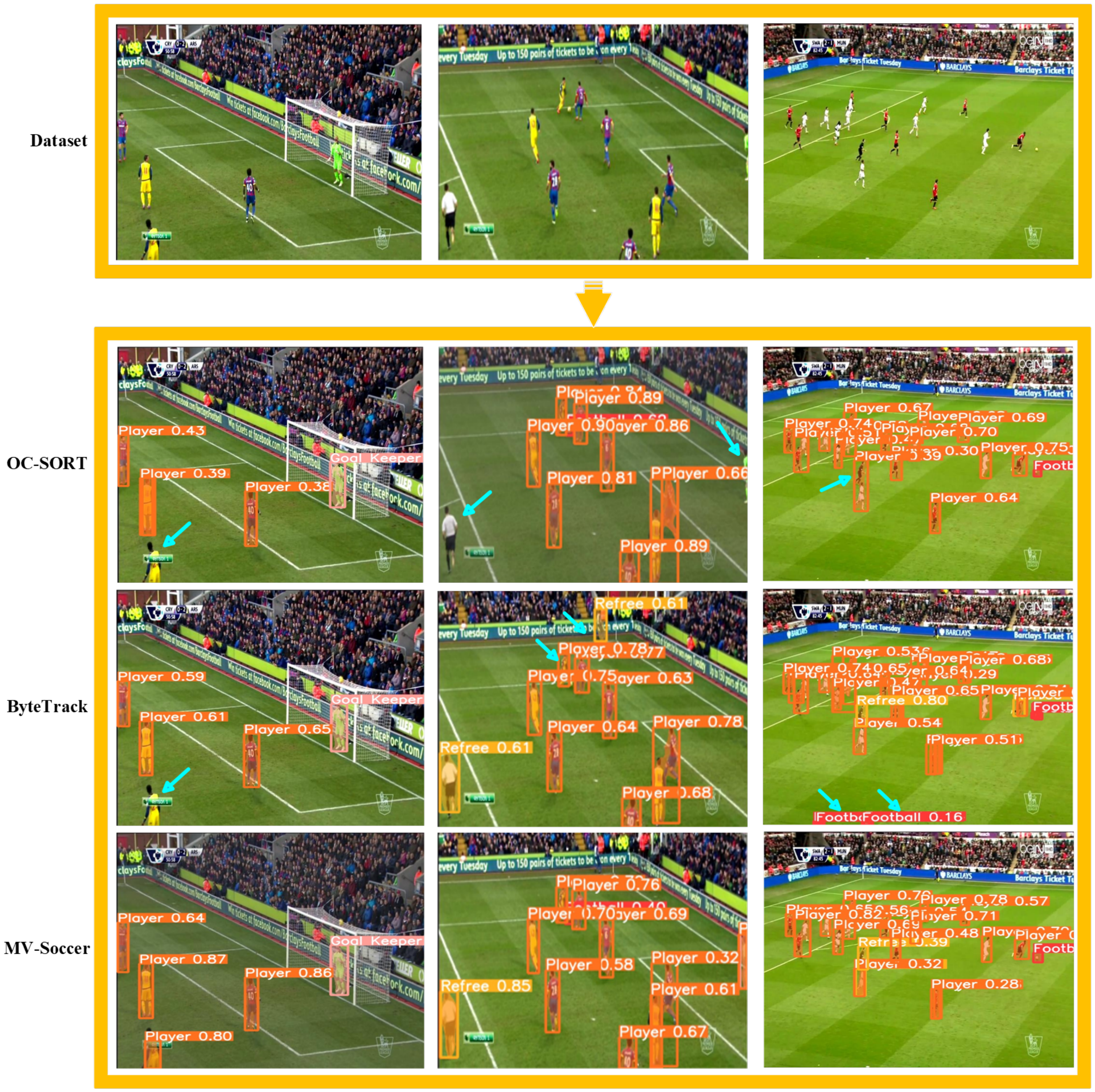


Figure S.1. **Qualitative results:** Obtained, for instance segmentation and tracking outputs generated by MV-Soccer on DFL - Bundesliga Data Shootout and the SoccerNet dataset using OC-SORT, ByteTrack and MV-Soccer (ours). Left to right column: three different camera scenarios. (1) near field, (2) midfield, (3) wide field. Cyan arrows indicate the localization, segmentation and tracking errors. Our approach (last row) consistently provides better results in all three perspectives.

Coordinate loss: This loss is defined as follows.

$$L_{\text{coord}} = \lambda_{\text{coord}} \sum_{i=0}^{S} \sum_{j=0}^{B} \mathbb{1}_{ij}^{\text{obj}} \left((x_i - \hat{x}_i)^2 + (y_i - \hat{y}_i)^2 \right) \tag{S.4}$$

$$+ \lambda_{\text{coord}} \sum_{i=0}^{S} \sum_{j=0}^{B} \mathbb{1}_{ij}^{\text{obj}} \left(w_i - \hat{w}_i)^2 + (h_i - \hat{h}_i)^2 \right),$$

where S is the grid size, B is the number of anchor boxes, $\mathbb{1}_{ij}^{\text{obj}}$ is an indicator function that is equal to 1 if the i^{th} grid cell and j^{th} anchor box are responsible for detecting the object, and 0 otherwise. (x_i, y_i) and $(\hat{x}_i, \hat{y}_i)$ are the predicted and ground truth centre coordinates of the bounding box in the i^{th} grid cell. (w_i, h_i) and $(\hat{w}_i, \hat{h}_i)$ are the predicted and

ground truth width and height of the bounding box in the i^{th} grid cell. λ_{coord} is a hyperparameter that controls the importance of the coordinate loss relative to the other losses.
The total loss is the weighted sum of these losses. Fig. S.2 summarizes the following different loss details while focusing on tracking for overall loss on all three datasets. (a) overall loss, (b) canonical loss, (c) depth loss, (d) distortion loss, (e) flow loss and (f) flow smoothness loss in our work.

Metrics

We use the following definitions of metrics in our work.

Instance Segmentation Metrics

Intersection over Union (IoU): We use the IoU metric for instance segmentation and tracking tasks using motion vectors. An IoU loss measures the gap between predictions and ground truth during training. We extend the notion of the classic IoU to include motion vectors as follows.

$$\mathrm{IoU_m} := \frac{A_{\mathrm{i}} \times \mathrm{MAF}}{A_u}. \tag{S.5}$$

This formulation assigns an overall score to motion vectors. $\mathrm{IoU_m}$ is the Intersection over Union metric with motion vectors (modifier). A_{i} is the area of spatial intersection between the segmented region and the tracked region. MAF is a motion vector alignment factor measuring the vector alignment between the segmented and tracked regions. A_{u} is the area of spatial union, considering both the segmented and the tracked regions. We performed a deep comparative analysis of our MV-Soccer with the current and previous versions of YOLO (v5, v7, and v8) for instance segmentation. Tab. 3 summarizes our segmentation results on benchmark and SoccerPro datasets.
Precision measures the statistical spread of stochastic observations for their true expected value. It is defined as

$$\text{Precision} := \frac{\mathrm{TP}}{\mathrm{TP} + \mathrm{FP}}$$

where TP (True Positives) is the number of correctly predicted instances and FP (False Positives) is the number of instances predicted by the model that are false positives.
Recall measures the fraction of relevant correct predictions. It is defined as

$$\text{Recall} := \frac{\mathrm{TP}}{\mathrm{TP} + \mathrm{FN}},$$

where TP are the true positives and FN (False Negatives) is the number of instances not detected.
Mean Average Precision (mAP) is used in instance segmentation to measure the anticipated object instances' correctness vs. ground truth annotations. It is defined and calculated as follows:

$$\mathrm{mAP} := \frac{1}{N}\sum_{i=1}^{N} \mathrm{AP}_i,$$

where N is the total number of classes and AP_i is the average Precision for class i.

Tracking Metrics

Higher Order Tracking Accuracy(HOTA): Higher Order Tracking Accuracy combines various tracking aspects, including localization and identity accuracy, into a single metric. HOTA combines various tracking accuracy aspects, including localisation and identity accuracy, into a single metric. HOTA provides a better understanding of a tracker's performance than traditional metrics like MOTA or IDF1. HOTA is defined as follows.

$$\mathrm{HOTA} := \mathrm{A_{ass}} - \mathrm{A_{loc}} - \mathrm{FP} - \mathrm{IDSW},$$

where $\mathrm{A_{ass}}$ is the Assignment Accuracy. It represents how accurately the algorithm assigns predicted bounding boxes to ground truth objects across different IoU thresholds. $\mathrm{A_{loc}}$ is the localization accuracy. It quantifies the average distance between the centres of predicted bounding boxes and their corresponding ground truth bounding boxes. FP counts the false positive and IDSW counts identity switches. Identity switches occur when the algorithm incorrectly associates a predicted object with an identity different from its ground truth identity.
Multiple Order Tracking Accuracy(MOTA): MOTA measures the overall tracking performance by considering false positives, negatives, and identity switches. It takes into account both localization errors and errors in maintaining object identities. MOTA is defined as follows.

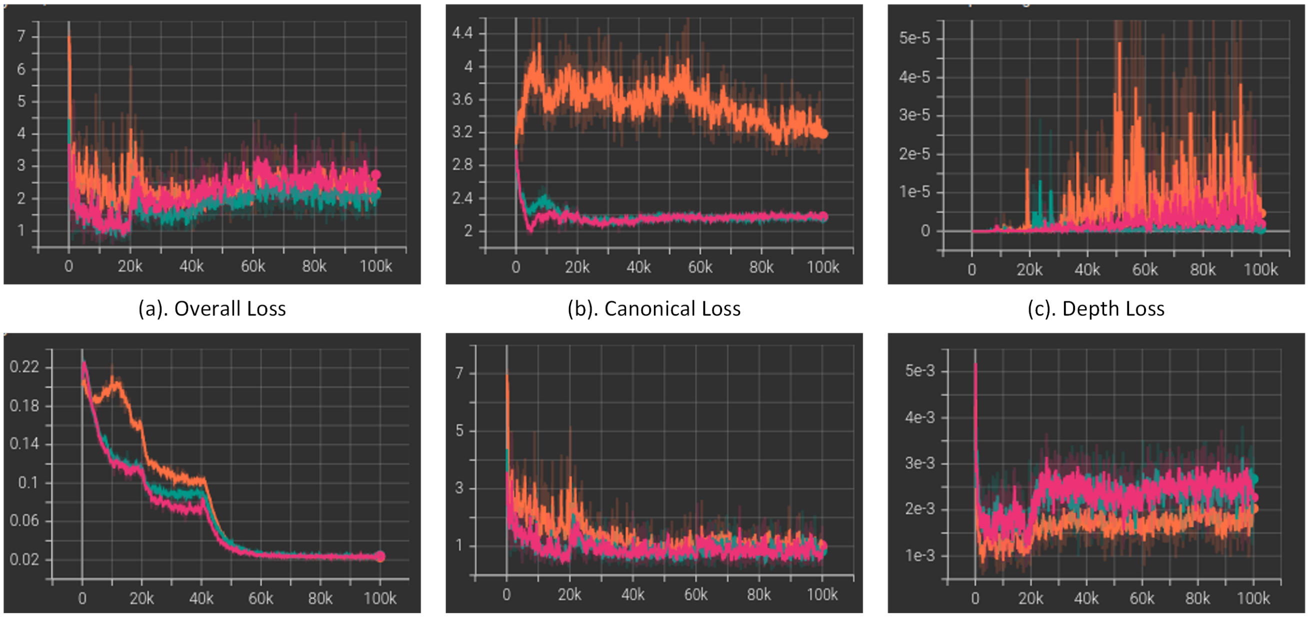
$$\mathrm{MOTA} := 1 - \frac{\mathrm{FN} + \mathrm{FP} + \mathrm{IDSW}}{\mathrm{GT}}$$

where FN is the number of false negatives, FP is the number of false positives, IDSW is the number of identity switches, and GT is the total number of ground truth objects.

Identification F1(IDF1): The IDF1 metric focuses on the identity aspect of tracking, measuring the harmonic mean of Precision and recall for object identities. IDF1 is defined as follows.

$$\mathrm{IDF1} := \frac{2 \times \mathrm{TP}}{2 \times \mathrm{TP} + \mathrm{FP} + \mathrm{FN}}$$

where, TP is the number of true positives, FP is the number of false positives, and FN is the number of false



(a). Overall Loss

(b). Canonical Loss

(c). Depth Loss

(d). Distortion Loss

(e). Flow Loss

(f). Flow Smoothness Loss

Figure S.2. The overall loss computed on DFL - Bundesliga Data Shootout dataset, SoccerNet-Tracking dataset and SoccerPro dataset, starting from left (a) overall loss, (b) canonical loss, (c) depth loss, (d) distortion loss (e) flow loss, and (f) loss of the smoothness of the flow on all three datasets, for instance segmentation and tracking.

Table 6. Comparative Analysis of class-based scores on YOLO (v5, v7, and v8) and MV-Soccer (ours) on all the combined datasets for Instance Segmentation and Tracking

			Train				**Val**			
Models	**Size**	**Classes**	**Precision**	**Recall**	**mAP$^{\text{box}}_{50-95}$**	**mAP$^{\text{mask}}_{50-95}$**	**Precision**	**Recall**	**mAP$^{\text{box}}_{50-95}$**	**mAP$^{\text{mask}}_{50-95}$**
YOLOv5s-seg	640	Player	0.61	0.84	0.75	0.33	0.57	0.81	0.69	0.30
YOLOv5m-seg	640	Player	0.66	0.77	0.70	0.33	0.56	0.74	0.62	0.25
YOLOv7-seg	640	Player	0.83	0.84	0.83	0.47	0.83	0.83	0.82	**0.41**
YOLOv8l-seg	640	Player	0.77	0.82	0.74	0.38	0.76	0.80	0.74	0.29
YOLOv8x-seg	640	Player	0.62	0.81	0.72	0.38	0.61	0.79	0.71	0.31
MV-Soccer	640	Player	**0.97**	**0.87**	**0.89**	**0.54**	**0.93**	**0.85**	**0.83**	**0.41**
YOLOv5s-seg	640	Goalkeeper	0.78	0.43	0.53	0.21	0.89	0.47	0.63	0.28
YOLOv5m-seg	640	Goalkeeper	**0.99**	0.44	0.75	0.32	**0.99**	0.50	0.74	0.37
YOLOv7-seg	640	Goalkeeper	**0.99**	0.79	0.88	0.40	**0.99**	0.79	0.88	0.39
YOLOv8l-seg	640	Goalkeeper	0.93	0.81	**0.96**	0.46	0.87	0.74	0.88	0.38
YOLOv8x-seg	640	Goalkeeper	0.93	0.77	0.87	0.46	0.93	0.77	0.83	0.42
MV-Soccer	640	Goalkeeper	**0.99**	**0.83**	0.87	**0.49**	**0.99**	**0.81**	**0.92**	**0.45**
YOLOv5s-seg	640	Referee	0.85	0.46	0.56	0.31	0.76	0.42	0.49	0.22
YOLOv5m-seg	640	Referee	0.90	0.58	0.67	0.36	0.85	0.58	0.67	0.30
YOLOv7-seg	640	Referee	0.86	0.69	0.77	0.49	0.82	0.69	0.77	0.43
YOLOv8l-seg	640	Referee	0.69	0.69	0.72	0.34	0.61	0.58	0.64	0.28
YOLOv8x-seg	640	Referee	0.67	0.70	0.70	0.32	0.67	0.70	0.71	0.35
MV-Soccer	640	Referee	**0.92**	**0.72**	**0.88**	**0.51**	**0.90**	**0.74**	**0.81**	**0.47**
YOLOv5s-seg	640	Football	0.36	0.35	0.30	0.10	0.31	0.30	0.24	0.12
YOLOv5m-seg	640	Football	0.54	0.39	0.32	0.14	0.56	0.52	0.46	0.14
YOLOv7-seg	640	Football	0.46	0.35	0.31	0.16	0.46	0.35	0.36	0.11
YOLOv8l-seg	640	Football	0.47	0.35	0.30	0.10	0.36	0.26	0.22	0.09
YOLOv8x-seg	640	Football	0.21	0.13	0.18	0.08	0.28	0.17	0.23	0.08
MV-Soccer	640	Football	**0.63**	**0.51**	**0.47**	**0.41**	**0.61**	**0.48**	**0.44**	**0.39**

negatives.

Fig. S.1 visually compares both near and far views for OC-Sort, ByteTack, and MV-Soccer (ours). As can be seen,

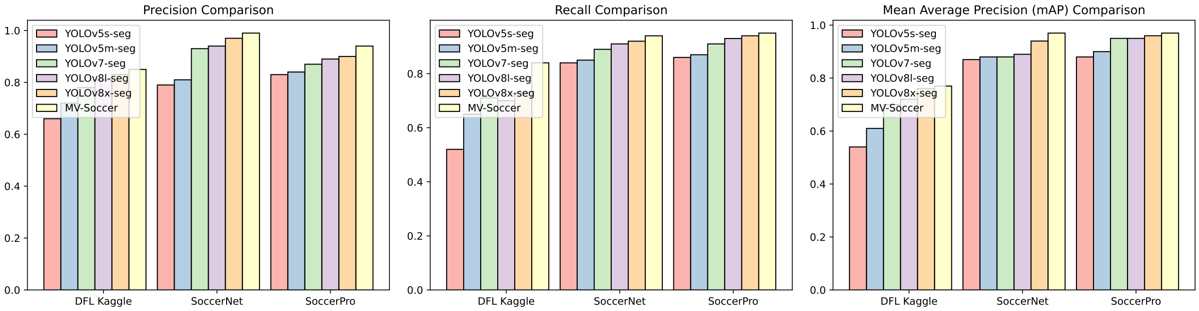


Figure S.3. Comparative Analysis of all five Instance Segmentation models with the proposed MV-Soccer and their results on the combined datasets. Left to right: **Precision**, **Recall**, and **mAP** for YOLO (v5, v7, and v8) and MV-Soccer (ours).

Table 7. Tracking Results on MOT17 Validation and MOT20 Training Datasets

Tracker	**MOT17 Validation Dataset**				**MOT20 Training Dataset**			
	(w) Motion Vectors				**(w) Motion Vectors**			
	HOTA	**MOTA**	**IDF1**	**FPS**	**HOTA**	**MOTA**	**IDF1**	**FPS**
Enhanced Motion:								
OC-SORT [9]	61.3	76.2	75.1	23.4	60.2	74.5	76.3	19.7
MotionTrack [38]	64.7	79.5	78.7	13.2	63.4	77.4	78.2	9.7
Embedding:								
StrongSORT [17]	—	—	—	—	—	—	—	—
IoU only:								
ByteTrack [51]	—	—	—	—	—	—	—	—
BoT-SORT [1]	—	—	—	—	—	—	—	—
MV-Soccer	**64.9**	**79.8**	**79.4**	**27.2**	**63.6**	**78.4**	**78.7**	**23.5**
	(w/o) Motion Vectors				**(w/o) Motion Vectors**			
Enhanced Motion:								
OC-SORT [9]	54.7	74.6	69.7	19.3	52.4	73.1	69.3	17.6
MotionTrack [38]	58.2	72.9	68.6	8.4	57.4	72.2	67.8	8.2
Embedding;								
StrongSORT [17]	56.3	71.5	70.2	6.7	54.9	70.6	68.4	6.1
IoU only:								
ByteTrack [51]	57.7	75.6	69.3	14.4	57.3	74.5	68.7	12.7
BoT-SORT [1]	61.6	76.2	74.7	7.6	61.3	75.4	74.3	5.3
MV-Soccer	**63.4**	**77.6**	**75.2**	**23.4**	**63.2**	**77.5**	**75.2**	**21.4**

our method provides consistently better results.