Computer Methods and Programs in Biomedicine 209 (2021) 106302

Contents lists available at ScienceDirect

Computer Methods and Programs in Biomedicine

journal homepage: www.elsevier.com/locate/cmpb

DeepSperm: A robust and real-time bull sperm-cell detection in densely populated semen videos

Priyanto Hidayatullah[a,b,*], Xueting Wang[c], Toshihiko Yamasaki[c], Tati L.E.R. Mengko[a], Rinaldi Munir[a], Anggraini Barlian[d], Eros Sukmawati[e], Supraptono Supraptono[e]

[a] School of Electrical Engineering and Informatics, Institut Teknologi Bandung, Bandung, 40132, Indonesia
[b] Computer Engineering and Informatics Department, Politeknik Negeri Bandung, Kabupaten Bandung Barat, 40559, Indonesia
[c] Department of Information and Communication Engineering, The University of Tokyo, Tokyo, 113-8656, Japan
[d] School of Life Sciences and Technology, Institut Teknologi Bandung, Bandung, 40132, Indonesia
[e] Balai Inseminasi Buatan Lembang, Bandung, 40391, Indonesia

ARTICLE INFO

Article history:
Received 19 February 2020
Accepted 17 July 2021

Keywords:
Sperm-cell detection
Small-object detection
YOLO
Computer-aided sperm analysis

ABSTRACT

Background and Objective: Object detection is a primary research interest in computer vision. Sperm-cell detection in a densely populated bull semen microscopic observation video presents challenges that are more difficult than those presented by other general object-detection cases. These challenges include partial occlusion, vast number of objects in a single video frame, tiny size of the object, artifacts, low contrast, low video resolution, and blurry objects because of the rapid movement of the sperm cells. This study proposes a deep neural network architecture, called DeepSperm, that solves the aforementioned problems and is more accurate and faster than state-of-the-art architectures.
Methods: In the proposed architecture, we use only one detection layer, which is specific for small object detection. For handling overfitting and increasing accuracy, we set a higher input network resolution, use a dropout layer, and perform data augmentation on saturation and exposure. Several hyper-parameters are tuned to achieve better performance. Mean average precision (mAP), confusion matrix, precision, recall, and F1-score are used to measure accuracy. Frame per second (fps) is used to measure speed. We compare our proposed method with you only look once (YOLO) v3 and YOLOv4.
Results: In our experiment, we achieve 94.11 mAP on the test dataset, F1-score of 0.93, and a processing speed of 51.9 fps. In comparison with YOLOv4, our proposed method is 2.18 x faster on testing, and 2.9 x faster on training with a small dataset, while achieving comparative detection accuracy. The weights file size was also reduced significantly, with one-twentieth that of YOLOv4. Moreover, it requires a 1.07 x less graphical processing unit (GPU) memory than YOLOv4.
Conclusions: This study proposes DeepSperm, which is a simple, effective, and efficient deep neural network architecture with its hyper-parameters and configuration to detect bull sperm cells robustly in real time. In our experiments, we surpass the state-of-the-art in terms of accuracy, speed, and resource needs.

1. Introduction

Due to its massive population, Indonesia has a big beef demand. However, current beef production levels are insufficient to meet the demand. Indonesian government imported beef from foreign countries at a cost of approximately $900 million in 2019 [1]. The Indonesian government built and mandated some artificial insemination centers, such as The Lembang Institute for Artificial Insemination (*Balai Inseminasi Buatan/BIB*, Lembang, Indonesia), to provide high quality frozen bull semen as the main substance for artificial insemination, in order to address the lack of beef production. Artificial insemination is the most widely used reproductive technology in the country for improving beef production [2]. Currently, The Lembang Institute for Artificial Insemination carries out sperm evaluation manually.

When a manual sperm evaluation is performed, there is a risk of intra-variabilities and inter-variabilities. When the same ob-

* Corresponding author.
E-mail addresses: priyanto@polban.ac.id (P. Hidayatullah), xt_wang@hal.t.u-tokyo.ac.jp (X. Wang), yamasaki@cvm.t.u-tokyo.ac.jp (T. Yamasaki), tati@stei.itb.ac.id (T.L.E.R. Mengko), rinaldi@informatika.org (R. Munir), aang@sith.itb.ac.id (A. Barlian).

https://doi.org/10.1016/j.cmpb.2021.106302

server delivers different manual measurement findings at different times, even though the sample is the same, this is known as intra-variability. When two (or more) observers produce different manual measurement results on the same sample, this is known as inter-variability. Unlike manual measurement, computer-assisted sperm analysis produces consistent measurement results for each sample. For more details, the intra-variabilities and inter-variabilities [3], its subjectivity [4], high time and human-resource consumptions, and exhaustion to the observer's eyes have been the main drawbacks of manual evaluation. Automatic sperm evaluation has been a critical demand for the institution. A computer-aided sperm analysis (CASA) robust sperm detection capability is urgently required.

Dott and Foster reported that sperm motility assessment by CASA is well correlated with manual sperm measurement of various animals [5]. They also mentioned some studies that came to similar conclusions. Several studies also reported the comparison between manual and CASA based sperm evaluation [6,7]. The advantage of sperm measurement using CASA compared to manual measurement is that it is more accurate [8], high reproducibility [8], and objective [7].

Sperm quality is measured with different parameters such as concentration, morphology, and viability. However, sperm motility is the parameter that is the most related to fertilization [9–11]. The most common approach to measure sperm motility, in sequential order, is sperm detection, sperm tracking, computing CASA motility parameters, and sperm motility classification based on CASA motility parameters. Sperm detection is unquestionably important for accurate and timely sperm tracking, which has an impact on the entire sperm motility measuring process. Therefore, we focus on improving sperm-cell detection.

Several studies have been conducted for sperm-cell detection. However, some problems remain unsolved. In this study, we focus on solving these two problems: (1) limited accuracy and (2) high computational cost, which are detailed in the following section.

Observers in many artificial insemination centers are constantly confronted with densely populated sperm videos. In a single densely populated video frame, there can be more than 500 sperm cells.

According to a survey paper [12], the vast majority of previous sperm cell detectors achieved high accuracy because the density was small (only 10-20 sperm cells presents in the video). When the density increases, the accuracy decreases significantly. For example, Hamilton Thorne, a commercial computer-based automated system, yields measurement errors in densely populated sperm suspensions due to many colliding sperms as reported in [7,13].

Previous methods mostly used conventional image-processing approaches [14–16]. Some of the researchers employed mainly image binary morphological operations [17–19]. In contrast, Nissen et al. [20] used some sets of convolutional neural network (CNN) architectures for performing sperm detection. In those studies, they claimed to have achieved 84–94% sperm detection on low concentrated semen. However, once again, this percentage significantly degrades on highly concentrated semen.

Deep learning-based methods have gained popularity due to their performance. However, another problem arises with these methods, which is the need for a large number of valid annotated training data to prevent overfitting. Unfortunately, performing annotation in this study case is very laborious.

To summarize, the limited accuracy was caused by these specific challenges of sperm-cell detection on densely populated semen: frequent partial occlusion, vast number of objects in a single video frame, tiny size of the object, artifacts, low contrast, low video resolution, and blurry object because of the rapid movement of the sperm cell. In addition, having many annotated training data was laborious. These were the first problems we addressed.

Table 1
First dataset extractions.

Video number	Number of video frames extracted
1	50
2	50
3	2
4	2
5	2
6	2
7	2
8	2
9	2
10	2
11	2
12	2

Sperm-cell detection is a kind of object detection. Deep learning-based approaches, YOLOv3 [21] and YOLOv4 [22], have achieved state-of-the-art performance for solving general object-detection problems. Unfortunately, owing to the fairly large size of the architecture, large weights files were produced that needed considerable amounts of training and test time, graphical processing unit (GPU) memory, and considerable amount of storage space. The high computational cost was the second problem that we addressed.

2. Materials and methods

2.1. Dataset

We collected twelve bull-sperm observation videos from twelve different bulls at the *Balai Inseminasi Buatan Lembang*, Indonesia. The samples were not stained. The length of the videos varied from 1 s to 124 s. In each video frame, the average number of sperms was 370 with a 139.53 standard deviation. The least dense video frame contains 114 sperm cells whereas the densest video frame contains 568 sperm cells. An Olympus BX 51 phase-contrast microscope with 10 x magnification of the objective lens was employed to capture the sperms. We recorded the sperm using Sony Exwave-HAD digital camera with 10 x magnification which results in 100 x total magnification. The video resolution was 640 $\times$ 480 pixels recorded at 25 fps and stored in MPEG format. Each video was taken in moderately different lighting conditions. We extracted video frames from each video in jpeg format with three channels (RGB) and 24-bit depth. Despite the fact that the video frames appear to be monochromatic, they actually comprise three channels, as specified by the digital camera.

To know the characteristics of the dataset, we converted the frame extracted from each video sample into a greyscale image and plotted its histogram, as shown in Fig. 1. We also calculated the pixel mean value and the standard deviation (*std*) of the image pixels. There were two conclusions drawn from the samples; firstly, all the histograms were very narrow, representing that all the samples had low contrast; secondly, the histograms of the dataset vary from left-skewed, intermediate, and right-skewed.

We constructed the dataset using three different dataset splits to adequately test the models. The first split involves randomly extracting 50 frames from two videos with different histogram characteristics as a training dataset. We shuffled them; 80% was used for the training dataset and the remainder for the validation dataset. We randomly extracted two video frames from each of the remaining ten videos and used them as the test dataset. This split was suggested in [23] and was beneficial to test how well the model generalizes to new data [24]. This split is particularly useful for determining which models perform well with minimal data,

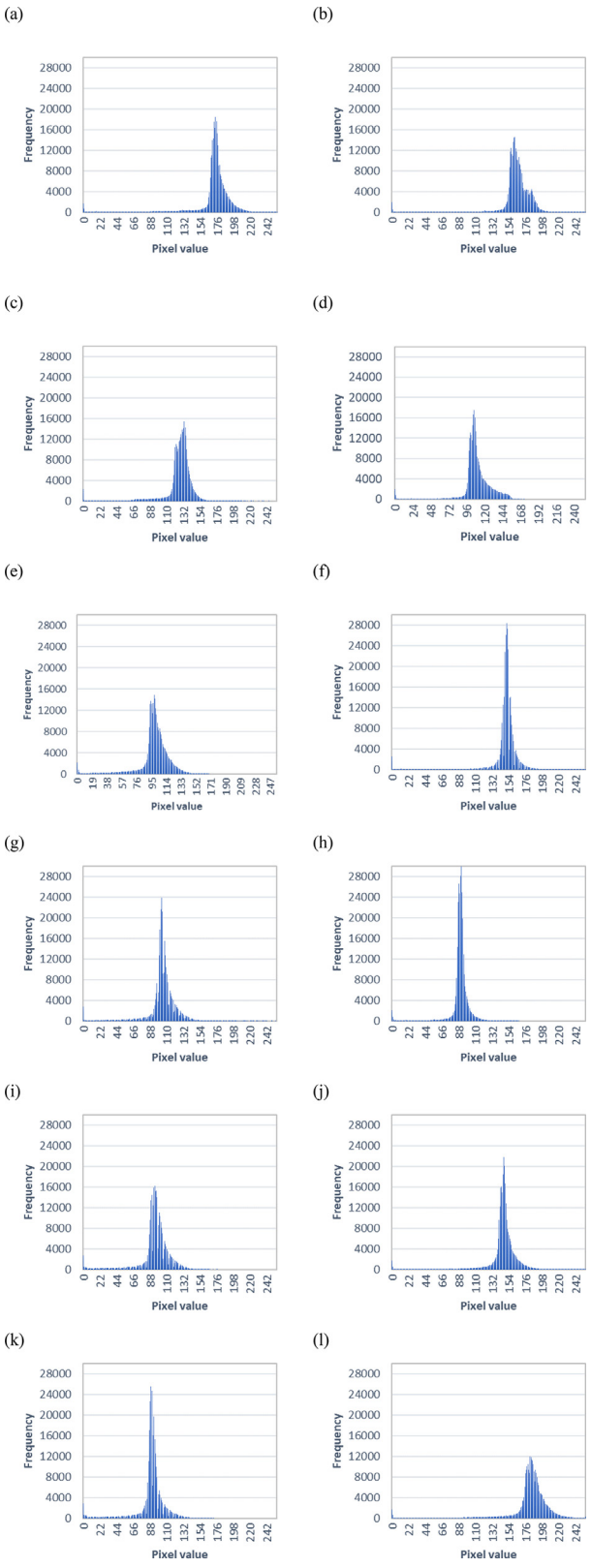


Fig. 1. (a) V1, mean = 171.19 and std = 27.92; (b) V2, mean = 163.22 and std = 24.38; (c) V3, mean = 125,45 and std = 21.27; (d) V4, mean = 108.84 and std = 19.87; (e) V5, mean = 97.82 and std = 21.34; (f) V6, mean = 148.86 and std = 22.81; (g) V7, mean = 102.86 and std = 21.46; (h) V8, mean = 88.93 and std = 15.51; (i) V9, mean = 93.09 and std = 22.49; (j) V10, mean = 144.55 and std = 25.16; (k) V11, mean = 88.56 and std = 21.17; (l) V12, mean = 179.95 and std = 32.25.

Table 2
First dataset proportions.

Dataset role	Source	Number of video frames
Training	Video 1 – 2	80
Validation	Video 1 – 2	20
Test	Video 3 – 12	20

Table 3
Second and third dataset extractions.

Video number	Number of video frames extracted
1	10
2	10
3	10
4	10
5	10
6	10
7	10
8	10
9	10
10	10
11	10
12	10

Table 4
Second dataset proportions.

Dataset role	Source	Number of video frames
Training	Video 1 – 9	75
Validation	Video 1 – 9	15
Test	Video 10 – 12	30

Table 5
Third dataset proportions.

Dataset role	Source	Number of video frames
Training	Video 1 - 7	60
Validation	Video 1 - 7	10
Test	Video 8 – 12	50

which is common in sperm detection. Tables 1 and 2 illustrate the dataset extractions and proportions.

The second and third splits are intended to take distinct data variations into account. It begins by randomly extracting 10 frames from each video. The second split begins with the selection of three videos for the test dataset with distinct histogram characteristics (left-skewed, intermediate, and right-skewed). For the validation dataset, we shuffled the remaining nine videos and randomly selected 15 frames. This split is also suggested in [23]. The dataset extractions and proportions are shown in Tables 3 and 4.

The third splits try to take different variations of the data while also providing greater test variance. It works in a similar way to the second split, but with a larger test dataset. For the test dataset, we take five videos with various histogram characteristics. For the validation dataset, we shuffled the remaining seven videos and randomly selected 10 frames. The dataset extractions are shown in Table 3 and the proportions are shown in Table 5.

2.2. Ground truth and dataset annotation

Ground truth data were obtained from manual detection of sperms by two experienced veterinarians who have more than 14 years of experience. The dataset in this study was relatively small because it was fairly laborious to annotate hundreds of sperm cells in a video frame. Several days were required to annotate 120 video frames. We adopted YOLOmark [25] to annotate the dataset manually. It is an annotation tool developed by Bochkovsky which is designed specifically for YOLO based model. It is the most user-

Table 6
Number of annotated sperm cells.

No of split scheme	Role	Number of annotated sperm cells
1	Training	28,735
	Validation	7,301
	Test	6,519
2	Training	28,048
	Validation	4,978
	Test	11,382
3	Training	21,390
	Valiation	4,117
	Test	18,901

friendly tool to annotate small objects such as sperm cells. It will create text annotation files with format

$$\langle object-class\rangle \langle x\_center\rangle \langle y\_center\rangle \langle width\rangle \langle height\rangle$$

where $\langle object - class\rangle$ is the object identity, integer number ranging from 0 to (classes-1) and $\langle x\_center\rangle$ $\langle y\_center\rangle$ $\langle width\rangle$ $\langle height\rangle$ is the bounding box specification, float number relative to width and height of image ranging from 0.0 to 1.0.

As an additional note, we evaluated sperms that reached the frame border. If 50% of their heads were visible, they were marked and counted in. As we have three dataset splits, the number of samples in each dataset role is different which is presented in Table 6. In total, there were 44,048 annotated sperm cells. Therefore, we had a considerable number of annotated objects.

2.3. Neural network architecture

The proposed architecture is based on YOLO. Sperm observed at a magnification of 100 x is considered a tiny object in this paper. Some survey papers compare different state-of-the-art methods for detecting small objects. Faster R-CNN was slower, achieving 33.6 mAP, whereas YOLOv3 was faster, achieving 32.5 mAP on the Wider Face dataset [26]. In tiny object samples from the PASCAL VOC 2007 dataset, YOLOv3 often outperformed Faster RCNN [27]. YOLOv3 (40 fps) is 10 times faster than Faster RCNN (4 fps) in both studies [26,27]. We chose a YOLO-based object detector since it has a high level of accuracy and is significantly faster than other methods. We plan to put the findings of these studies to use in practical application, therefore speed is as important as accuracy. As a result, we use YOLOv3 as a benchmark because it was known as being very good at recognizing small objects [26,27], and YOLOv4 as the current state-of-the-art.

To improve the detection accuracy, the input network resolution was increased up to 640 × 640. The network contained 42 layers in total. All the convolutional layers used batch normalization. The first seven layers were designed to downsample the image until a sufficient resolution for the architecture to detect small objects accurately (80 × 80).

In the following layers, there were 24 deep convolutional layers with leaky RELU [28] activation function which formula presented in Eq. (1) [28], i.e.,

$$\emptyset(x) = \begin{cases} x, & if\ x > 0 \\ 0.1\,x, & otherwise. \end{cases} \quad (1)$$

To prevent vanishing/exploding gradient problems and to increase the detection accuracy, a shortcut connection was added to utilize the residual layers for every two convolutional layers [29]. The last layer, the YOLO layer, gave detection prediction on each anchor box along with its confidence score. The number of filters of the YOLO layer was set according to Eq. (2) [28]. We used three anchor boxes, 1 × 1 grid size, and one class (sperm class).

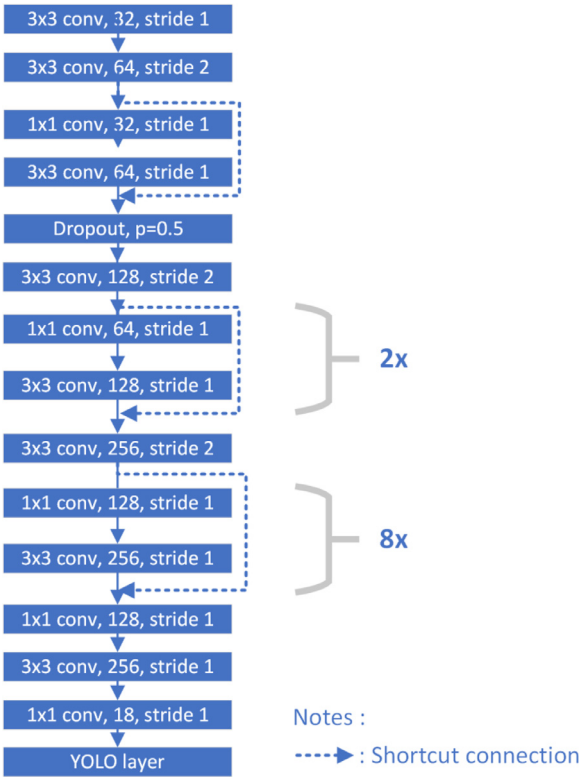


Fig. 2. The proposed method's architecture.

Therefore, the number of the filters at the YOLO layer according to Eq. (2) was 18.

$$n = Gs \times Gs \times (B * 5 + C), \quad (2)$$

where

$n = number\ of\ filters$
$Gs \times Gs = number\ of\ the\ grids$
$B = number\ of\ anchor\ boxes$
$C = number\ of\ classes.$

At the end of the network, the logistic function is used. YOLOv3 and YOLOv4 have three YOLO layers, whereas the proposed architecture has one YOLO layer only. This was because the proposed method is meant to detect only small objects; hence, a multi-scale prediction is not required. A more detailed schematic of the architecture is presented in Fig. 2.

The reason why the number of layers used in the proposed architecture is less than that of YOLOv3 and YOLOv4 is to enhance the training and testing speeds. The maximum filter size is 3 × 3 because the objects are very close to one another. This small filter size improves the speed while maintaining its accuracy. On the other hand, to enhance the accuracy, the input network resolution is increased to 640 × 640. It is higher than the input network resolution YOLOv3 and YOLOv4 which use 608 × 608 resolution.

We put a dropout layer just after the first shortcut layer. The threshold is set to 0.5, which means any node with a weight less than half is dropped. This layer is crucial to prevent overfitting in the case of limited data and to increase speed while maintaining accuracy. Our code can be accessed at https://github.com/pHidayatullah/DeepSperm.

We compare our proposed method with YOLOv3 and YOLOv4 which have been the state-of-the-art model for object detection. Table 7 shows the architecture comparison. We do not compare to YOLOv5 which has no official publication for its architecture,

Table 7
Architecture comparison.

Method	YOLOv3	YOLOv4	DeepSperm
Input network resolution	608 × 608	608 × 608	640 × 640
Number of layers	78	109	30
Number of nodes	61,150,304	60,246,720	3,530,592
Number of parameters	61,176,662	60,279,958	3,535,026
Number of detection (YOLO) layers	3	3	1
Activation function	Leaky ReLU	Mish	Leaky ReLU
Dropout	No	No	1
Randomized input network resolution	Yes	Yes	No

Table 8
Hyper-parameters comparison.

Method	YOLOv3	YOLOv4	DeepSperm
Momentum	0.9	0.949	0.9
Decay	0.0005	0.0005	0.0005
Learning rate	0.001	0.00261	0.001
Burn in	1000	4000	250
Batch size	64	64	64
Subdivision	16	64	16
Pretrained weights	darknet53.conv.74	yolov4.conv.137	darknet53.conv.74

hyper-parameter, and configuration that makes comparison difficult.

2.4. Hyper-parameters

Besides neural network architecture, the network parameters were also critical to obtain a faster training speed and a higher accuracy. We set the batch size to 64 and subdivisions to 16. To prevent overfitting, in addition to the dropout layer, the momentum parameter is used to penalize a substantial weight change from one iteration to another, whereas the decay parameter is used for penalizing enormous weights. We set the momentum to 0.9 and the decay parameter to 0.0005.

For the same purpose, we fine-tuned the learning rate. In this study, we set the default learning rate to 0.001, with burn-in (warming up) until 250 iterations, which is 6.5% of the total iterations: 4000. We also set the learning rate decay, with a factor of 0.1, after 1000 iterations, which is 25% of the total number of iterations.

Owing to the relatively small dataset, we generated more data from the existing data using data augmentation. We augmented the data by varying the exposure and saturation of each sample by specific parameters once. Both parameters were random numbers between 0.5 to 1.5. The augmentation formula can be found in Eq. (3).

$$I' = k_e V k_s S I \tag{3}$$

$$k_e = rand\ (0.5,\ 1.5) \tag{4}$$

$$k_s = rand\ (0.5,\ 1.5) \tag{5}$$

where

I' = *augmented image sample*
I = *original image sample*
k_e = *exposure parameter*
V = *value/exposure of the original image*
k_s = *saturation parameter .*
S = *saturation of the original image*

For clarity, we summarize the comparison between YOLOv3, YOLOv4, and the proposed method's hyper-parameters and configurations in Table 8.

2.5. Training and testing environment

We used two different system environments for training and testing. Both systems used Ubuntu 16.04 LTS operating system. Table 9 shows the specification comparison.

2.6. Training

As Tajbakhsh et al. [30] claimed that using a pre-trained model consistently outperformed training from scratch, we used a pre-trained model called darknet53.conv.74. It contained convolutional weights trained on ImageNet available in the YOLOv3 repository [31]. For comparison, we trained YOLOv3 with the same pre-trained model as ours and YOLOv4 with darknet.conv.137. YOLOv3 was trained on the original darknet framework [31], our model was trained using Bochkovskiy darknet implementation [32], and YOLOv4 was trained using the newest Bochkovskiy darknet implementation [33]. We trained YOLOv3 and YOLOv4 with their recommended original parameters except for the image augmentation. We used the same image augmentation parameters for YOLOv3 and our model specified in Eqs. (3), (4), and (5). On the other hand, YOLOv4 used its augmentation parameters. This training scheme was chosen to give the best possible result for each model.

Bochkovskiy [32] recommended using the number of iterations as many as 2000 times the number of classes. Because one class was used, the recommended number of iterations was 2000. However, we trained the network 4000 times (twice the recommendation) just in case we found the best weights after 2000 iterations.

2.7. Inference/testing

We performed the testing of all the methods on the validation dataset as well as on the test dataset. We used mAP@50 as the metric of accuracy so that we can directly compare the proposed method's accuracy with those of others. In mAP@50, 50 is used as an intersection over union parameter. We used mAP all-point-interpolation implementation [32], which was calculated according to Eqs. (6) and (7). Mean average precision (mAP) with all point interpolation calculates the area under the precision-recall curve for each unique recall. As described above, we use three different dataset splits, we calculate mAP for each split and averaging them to have mAP. We also use precision, recall, and F1-score as the metric to further analyze the result. Frame per second (fps) was

Table 9
Hardware specification comparison.

Model	Training Server	Testing PC
Processor	Intel(R) Xeon(R) Gold 6136 @ 3.00GHz	Intel Core i7 8700 @3.2 GHz
RAM	385 GB	16 GB
GPU	NVIDIA Titan V 12GB GPU RAM	NVIDIA GeForce RTX 2070 8GB GPU RAM

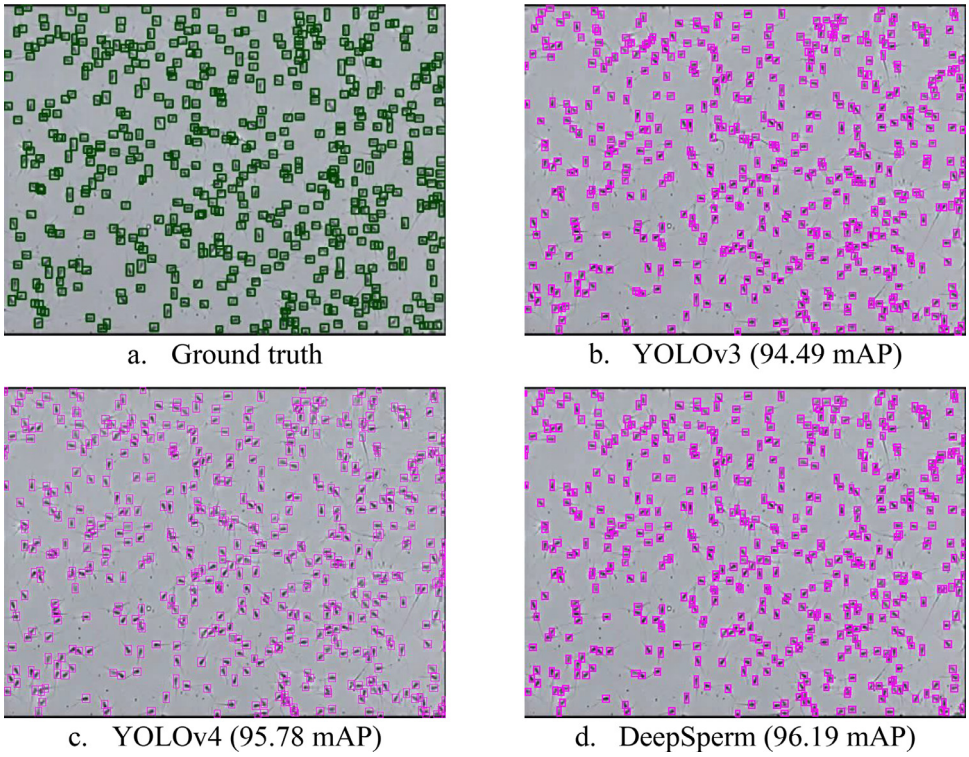


Fig. 3. Results comparison on one video frame from the validation dataset.

used as a metric for speed.

$$AP = \sum_{n=0}^{1} (r_{n+1} - r_n) p_{interp}(r_{n+1}) \tag{6}$$

$$p_{interp}(r_{n+1}) = \max_{\tilde{r}:\tilde{r} \geq r_{n+1}} p(\tilde{r}) \tag{7}$$

where

$p(\tilde{r}) = measured\ precision\ at\ recall\ \tilde{r}.$

In the testing phase, all models were tested on their original framework. On the Bochkovskiy implementation, we turned on the CUDNN_HALF option. This option allowed for the use of Tensor Cores of the GPU card to speed up the testing process.

3. Results and discussion

3.1. Accuracy

The proposed method achieved 96.19 mAP on the validation dataset and 94.11 mAP on the test dataset, which is comparable to YOLOv4. Figs. 3, 4, 5, 6, and 7 present the comparison of the results.

In a more detailed quantitative investigation (see Table 10), YOLOv4 has the highest recall (0.97). However, it has the largest number of false positives, lowering its precision to the lowest (0.81) of all the methods. Our proposed method has nearly the same recall (0.96) as YOLOv4 while maintaining precision (0.89). As a result, we have the highest F1-score of any method (0.93). YOLOv3 is a method that falls between YOLOv4 and our proposed method.

YOLOv3 used a 608 × 608 input network resolution which achieved 91.28 mAP on the test dataset. YOLOv4 used the same input network resolution and achieved 94.12 mAP average test accuracy. To increase the accuracy, we used a higher input network resolution (640 × 640). With this resolution, the input video frames were divided into a 640 × 640 grid, which reduced the grid size. It was the main key to our proposed method's accuracy.

We may deduce from Figs. 5, 6, and 7 that the second split provides the best average test accuracy (95.03 mAP) whereas the first split provides the poorest (89.49 mAP). It is reasonable because the second split has the largest data variation in the training dataset while the first split has the least. There is no significant difference in speed amongst models trained with different dataset splits.

3.2. Partial occlusion handling

Partial occlusion is one of the main challenges that causes limited detection accuracy. However, compared to other methods, our proposed method had been excellent at handling partial occlusion. Fig. 8 presents the comparison details.

In the first case, YOLOv3 failed to handle the partial occlusion of the bottom sperm cell. However, YOLOv4 and our proposed method was able to detect all the sperms correctly.

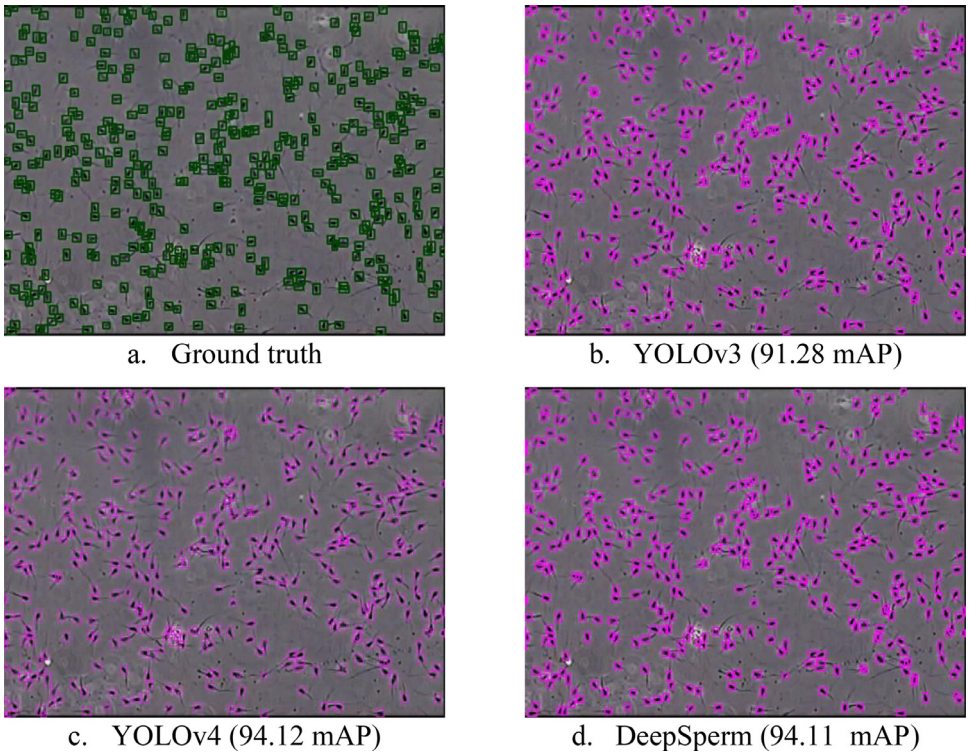


Fig. 4. Results comparison on one video frame from the test dataset.

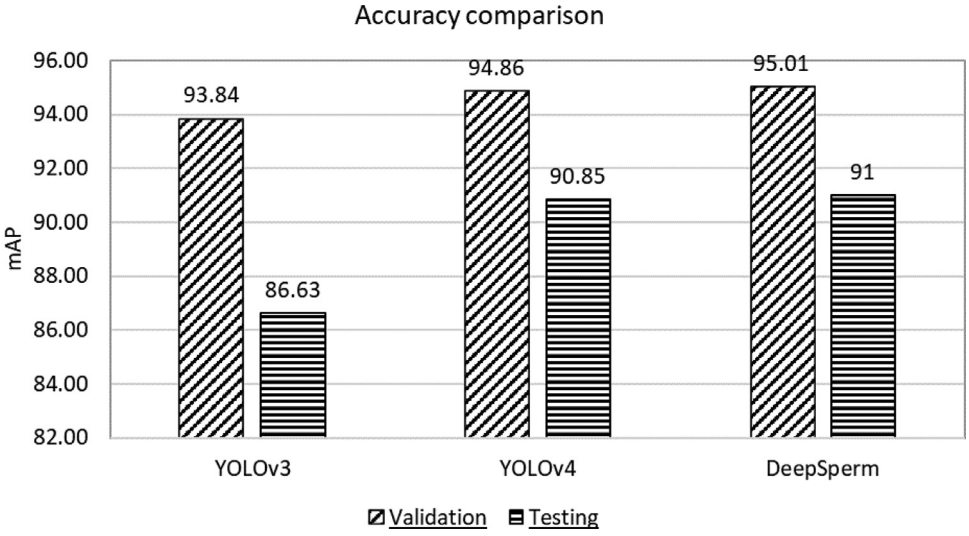


Fig. 5. Accuracy comparison of all the methods on the first dataset split.

In the second and third cases, our proposed method was able to detect the sperms accurately. The other methods still failed. The key factor for handling partial occlusion is the higher input network resolution.

3.3. Artifacts handling

Artifacts are also the main challenges that lead to limited detection accuracy. In one of the test samples, there were tiny marks. They had a similar grayscale as the sperm cells but smaller in size.

In YOLO9000 [34], the authors increased the recall of YOLO [28] by using anchor boxes. However, they obtained a small decrease in accuracy (mAP) [34]. The reason was YOLO based detector produces a significant amount of false positives. Fig. 9 indicates that YOLOv4 regarded two artifacts as sperm cells. Our proposed method and YOLOv3 were able to ignore these artifacts successfully. We believe that the failure of YOLOv4 ignoring those tiny marks is because of the use of max pooling layers in the architecture. Pooling layers do not preserve all the spatial information well by applying spatial resolution reduction.

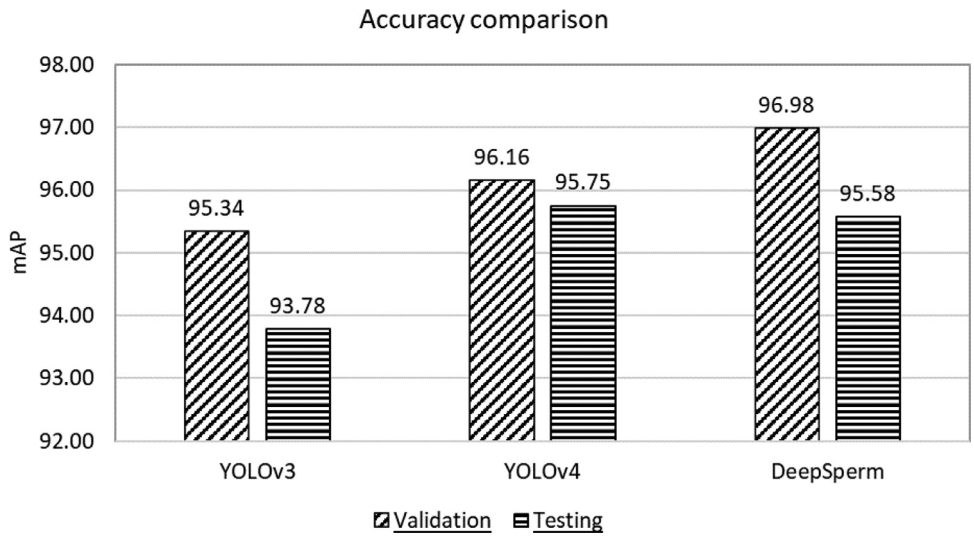


Fig. 6. Accuracy comparison of all the methods on the second dataset split.

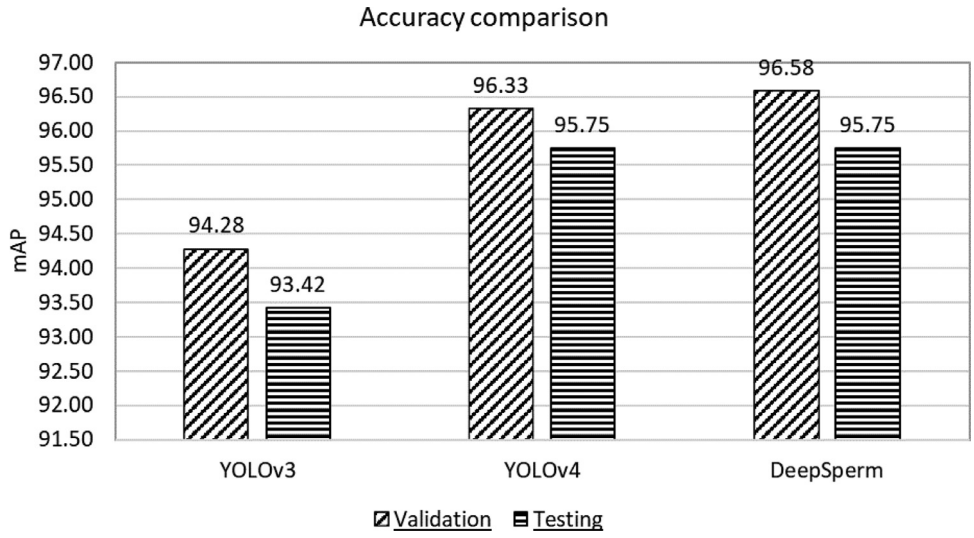


Fig. 7. Accuracy comparison of all the methods on the third dataset split.

Case No	Ground truth	YOLOv3	YOLOv4	DeepSperm
1				
2				
3				

Fig. 8. Partial occlusion handling comparison.

Table 10
Overall results comparison.

Model	YOLOv3			YOLOv4			DeepSperm		
Dataset split scheme	1	2	3	1	2	3	1	2	3
Number of layers	78			109			**30**		
Weights file size (MB)	249.9			250			**13.8**		
Best weights epoch	4000	4000	3900	2800	2100	2300	**1300**	**900**	**800**
Training time/epoch (s)	6.50	7.88	7.20	13.37	13.11	12.92	**3.98**	**4.74**	**4.85**
GPU RAM need (GB)	7.2			6.5			**6.1**		
Validation set accuracy (mAP@50)	93.84	95.34	94.28	94.86	96.16	96.33	**95.01**	**96.33**	**96.58**
Test set accuracy (mAP@50)	86.63	93.78	93.42	90.85	**95.75**	**95.75**	**91.00**	**95.75**	**95.75**
TP	6242	10923	18212	**6302**	**11046**	18302	6245	10965	**18393**
FP	1375	1101	**1808**	2706	1960	2608	**984**	**1027**	1903
FN	274	459	689	**214**	**336**	599	271	417	**508**
Precision	0.82	**0.91**	**0.91**	0.7	0.85	0.88	0.86	**0.91**	**0.91**
Recall	0.96	0.96	0.96	**0.97**	**0.97**	**0.97**	0.96	0.96	**0.97**
F1-score	0.88	0.93	**0.94**	0.81	0.91	0.92	0.91	**0.94**	**0.94**
Average Fps	18.1			23.8			**51.9**		
Average testing time/image (s)	0.049			0.042			**0.017**		

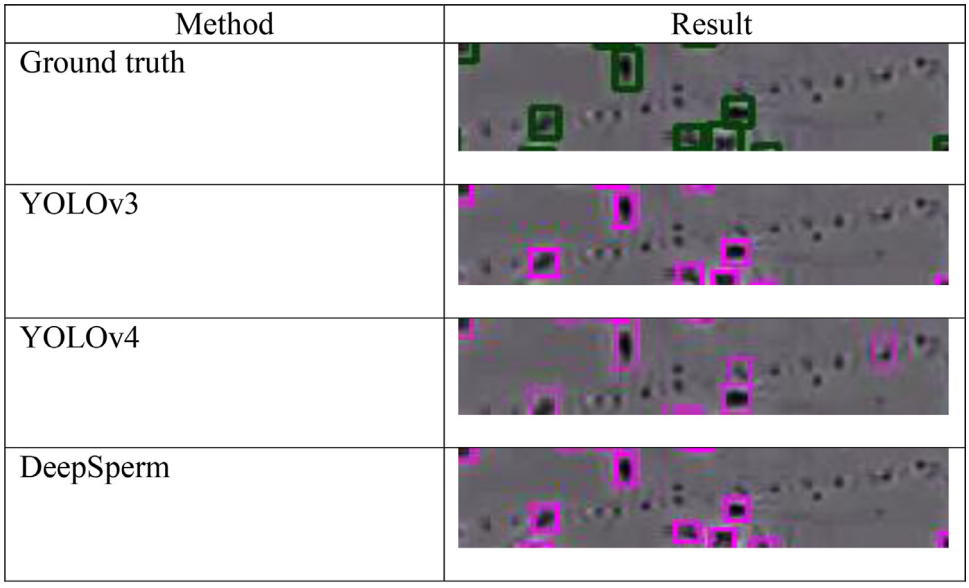


Fig. 9. Artifacts handling comparison.

We have managed to reduce false positives by increasing input network resolution. Compared to YOLOv4, we can reduce the number of false positives so that only two of these artifacts were regarded as sperm cells.

3.4. Overfitting handling

If the training dataset's samples have too little variation, the detectors may result in overfitting. We included samples in the dataset that we believe have sufficient variation to make the model resistant to overfitting. When detecting sperm with a magnification of 100x, the sperm cells appear relatively small, having annotated samples are often limited. Therefore, we add a single dataset split with low variation in the training data to examine the influence on accuracy and determine which model has the best generalization capabilities.

With three different dataset splits, the detection performed by YOLOv3 dropped by 3.21 mAP points. YOLOv3 used batch normalization in every convolutional layer, as well as in ours. Batch normalization was considered sufficient without any form of other regularizations. Therefore, the dropout layers were removed since YOLO9000 [34]. However, we observed that using only batch normalization was not sufficient for reducing overfitting. It was because of the small number of samples in the training dataset. As a solution, we utilized the dropout layer with a threshold of 0.5. There were two recommended positions of the dropout layer: the first at every layer and the second at the first layer only [35]. We observed that putting the dropout layer at the first shortcut connection yielded much better results than putting it at every shortcut connection.

In addition, based on the dataset histogram analysis, we performed data augmentation to increase the number of training data

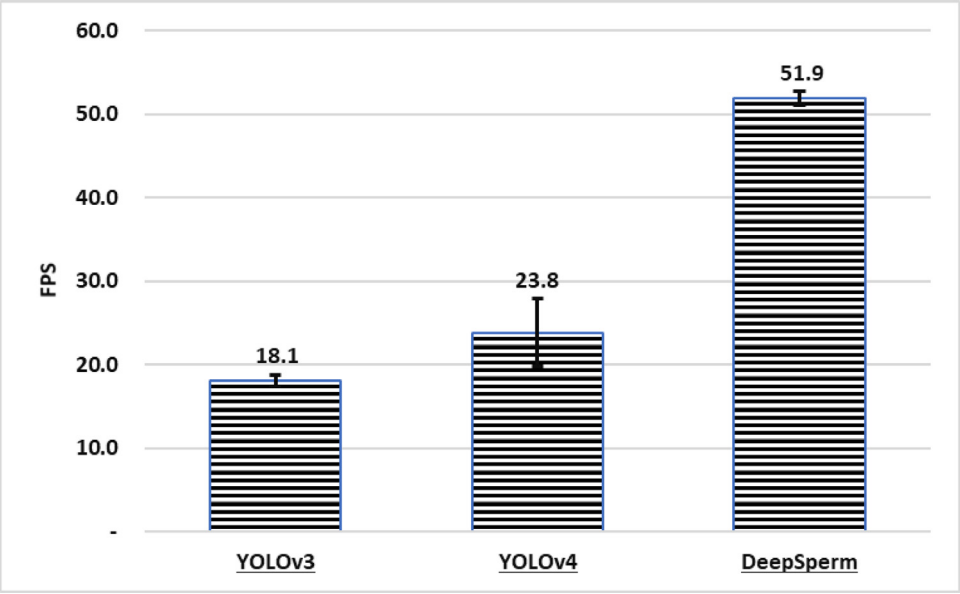


Fig. 10. Speed comparison of all the methods during the test phase.

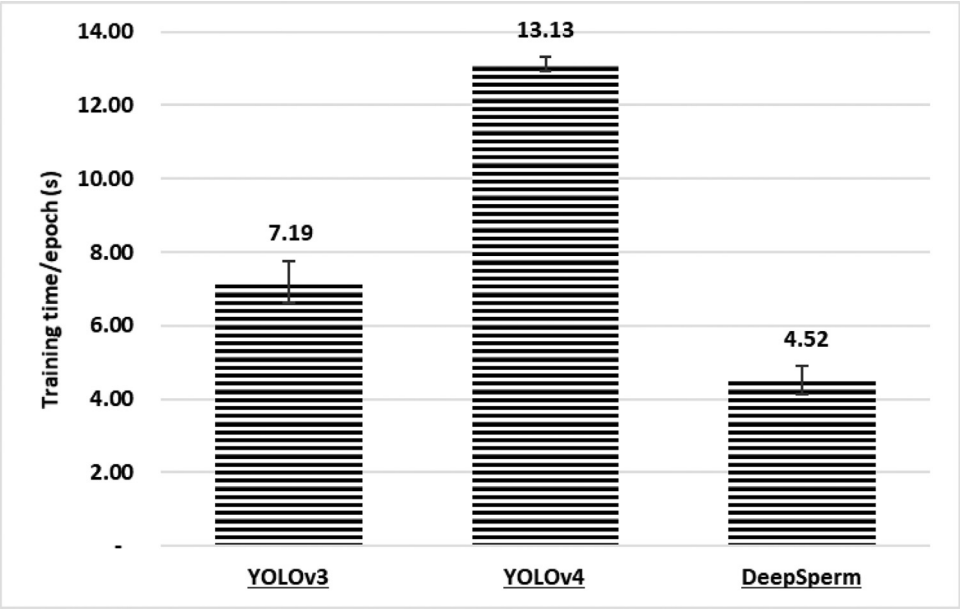


Fig. 11. Speed comparison of all the methods during the training phase.

by varying the data according to its exposure and saturation. The result indicated that the detection accuracy on the test dataset, as well as on the validation dataset, could be increased. With a 2.08 mAP gap, we achieved up to 96.19 mAP and 94.11 mAP on the validation and test datasets, respectively.

The narrowest discrepancy between validation and test average accuracy is found in YOLOv4 (1.67 mAP). The employment of a variety of augmentation techniques to enrich the training dataset is a critical aspect. The test accuracy, on the other hand, was very near to the proposed method accuracy, with only a 0.01 mAP difference.

YOLOv4 and the proposed method have the same accuracy gap of 4.01 mAP in the split when the training dataset only contains video frames from two videos, however YOLOv3 has a 6.21 mAP gap. The proposed method has the highest accuracy in this dataset split, with 95.01 mAP validation accuracy and 91.00 mAP test accuracy. It is obvious from this finding that including distinct variation samples in the training dataset is critical to avoid. It is also crucial to apply regularization and augmentation.

3.5. Speed

Speed is another highly critical criterion for object detection. For example, the observers in artificial insemination centers need the detection to be performed fast so that they can process as many semen evaluations as possible. Secondly, object detection (in this case, sperm cell detection) is often used as the first step for larger applications such as sperm cell tracking. Sperm cell tracking is used to measure sperm cell motility, which also needed to be in real time. We need a real-time sperm cell detector to have a real-time sperm cell tracker.

We employed a smaller network (one-twentieth of YOLOv3 and YOLOv4 convolutional layer parameters) to boost the test speed, which was 2.87 times faster than YOLOv3 and 2.18 times faster

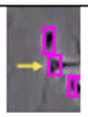
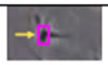
Case	Original frame	Detection result
1. Failure example from the validation dataset (artifacts)		
2. Failure example from the test dataset (occlusion)		
3. Failure example from the test dataset (occlusion)		
4. Failure example from the test dataset (blurry sperm)		

Fig. 12. Some detection failures.

than YOLOv4. YOLOv4′s architecture includes max-pooling layers, which reduces computation and makes it faster than YOLOv3, despite the fact that the two have a similar amount of parameters. With only 30 convolutional layers, the proposed architecture was trained 1.6 times faster compared to YOLOv3 (78 layers) and 2.9 times compared to YOLOv4 (109 layers). Figs. 10 and 11 show the speed comparison.

3.6. Failure case analysis

In general, our proposed method has been able to detect sperm cells better than the other methods, summarized in Figs. 8 and 9. However, we still encountered some detection errors. Fig. 12 presents some of the detection failures which were highlighted by the arrows.

To summarize, these were the reasons for the failures: the sperm were almost fully occluded (second and third cases), the artifacts were more or less similar to a sperm cell (first case), and the sperm was very blurry (fourth case).

3.7. Overall comparison

For clarity, we compared the performances of all the methods in a single table (Table 10). The results in bold were the best results achieved. YOLOv3 delivered better accuracy in detecting small objects, compared to other methods in [21]. It has almost the same accuracy as Faster R-CNN, but it is 10 times faster [26]. YOLOv4 is the successor to YOLOv3, which includes numerous improvements and achieves state-of-the-art performance for generic object detection in terms of accuracy and speed. There is a common thread of between accuracy and speed. Our proposed method, on the other hand, can achieve substantially faster performance while maintaining accuracy and using fewer resources. Table 10 presents the overall results comparison. We average several parameters that are the same or comparable across different splits.

4. Conclusions

This study proposed a deep neural network architecture, with its hyper-parameters and configurations detailed in the material and methods section, for robust detection of bull sperm cells. It was robust to partial occlusion, artifacts, vast number of moving objects, object's tiny size, low contrast, low video resolution, blurred objects, and different lighting conditions.

To summarize, the proposed method surpassed all the methods in terms of accuracy, speed, and resource need. 0.93 F1-score and 94.11 mAP on the test dataset, which is 0.05 points higher and 0.01 mAP lower than the state-of-the-art method (YOLOv4). In terms of speed, the proposed method achieved real-time performance with 50.3 average fps, which was 2.18 times faster than the state-of-the-art method. Our training time was also faster, up to 2.9 times that of the state-of-the-art method. With that performance, it eventually needed a small dataset containing only 120 video frames.

The memory usage of the proposed architecture was also significantly smaller than both YOLOv3 and YOLOv4. Only 30 layers were used in the suggested architecture, which is 2.6 times smaller than YOLOv3 layers and 3.6 times smaller than YOLOv4. Our proposed architecture has 3.5 million trainable parameters, which is one-twentieth of YOLOv3 or YOLOv4 parameters. There were some advantages to such architectures. The training and testing times were fast, less GPU memory was needed (1.18 times less than YOLOv3 and 1.07 times less than YOLOv4), and less amount of file storage was needed (weights file's size was one-twentieth of the size of YOLOv3 and YOLOv4 weights file).

In the future, we shall apply the proposed method for sperm cell tracking. We also want to test it for different cases such as detecting blood cells, bacteria, or any other biomedical case. We believe that this architecture shall perform well in these cases too.

5. Ethical approval

The authors took the authorization for the use of the dataset for the submission of this paper, and ethical approval was not required for this study.

Declaration of Competing Interest

The authors declare no conflict of interest regarding the publication of this paper.

Acknowledgments

This research was funded by Lembaga Pengelola Dana Pendidikan (LPDP) RI [grant number PRJ-6897 /LPDP.3/2016]. The authors would like to thank ITB World Class University (WCU) program, Prof. Ryosuke Furuta, Mr. Ridwan Ilyas, Mr. Kurniawan Nur Ramdhani, and Mrs. Tri Harsi.

References

[1] Pusat Data dan Sistem Informasi PertanianPusat Data dan Sistem Informasi Pertainian, Buku Outlook Komoditas Peternakan Daging Sapi, Sekretaris Jenderal - Kementrian Pertanian Indonesia, Sekretariat Jenderal - Kementrian Pertanian Indonesia, Jakarta, 2020.

[2] P. Hidayatullah, Mengko, L.E.R. Tati, R. Munir, A. Barlian, Bull sperm tracking and machine learning-based motility classification, IEEE Access 9 (2021) 12, doi:10.1109/ACCESS.2021.3074127.
[3] J. Auger, Intra- and inter-individual variability in human sperm concentration, motility and vitality assessment during a workshop involving ten laboratories, Hum. Reproduct. 15 (11) (Nov. 2000) Art. no. 11, doi:10.1093/humrep/15.11.2360.
[4] M.K. Hoogewijs, S.P. De Vliegher, J.L. Govaere, C. De Schauwer, A. De Kruif, A. Van Soom, Influence of counting chamber type on CASA outcomes of equine semen analysis: counting chamber type influences equine semen CASA outcomes, Equine Vet. J. 44 (5) (Sep. 2012) Art. no. 5, doi:10.1111/j.2042-3306.2011.00523.x.
[5] H.M. Dott, G.C.A. Foster, The estimation of sperm motility in semen, on a membrane slide, by measuring the area change frequency with an image analysing computer, Reproduction 55 (1) (Jan. 1979) 161–166, doi:10.1530/jrf.0.0550161.
[6] G.M. Centola, Comparison of manual microscopic and computer-assisted methods for analysis of sperm count and motility, Arch. Androl. 36 (1) (Jan. 1996) 1–7, doi:10.3109/01485019608987878.
[7] M.L.W.J. Broekhuijse, E. Šoštarić, H. Feitsma, B.M. Gadella, Additional value of computer assisted semen analysis (CASA) compared to conventional motility assessments in pig artificial insemination, Theriogenology 76 (8) (Nov. 2011) 1473–1486 .e1, doi:10.1016/j.theriogenology.2011.05.040.
[8] B.A. Keel, B. Webster, CRC Handbook of the Laboratory Diagnosis and Treatment of Infertility, CRC Press, Boston, 1990.
[9] M.M. Awad, Effect of some permeating cryoprotectants on CASA motility results in cryopreserved bull spermatozoa, Anim. Reproduct. Sci. 6 (2011).
[10] A. Januskauskas, et al., Effect of cooling rates on post-thaw sperm motility, membrane integrity, capacitation status and fertility of dairy bull semen used for artificial insemination in sweden, Theriogenology 52 (4) (Sep. 1999) 641–658, doi:10.1016/S0093-691X(99)00159-4.
[11] J. Verstegen, M. Iguer-Ouada, K. Onclin, Computer assisted semen analyzers in andrology research and veterinary practice, Theriogenology 57 (1) (2002) Art. no. 1.
[12] P. Hidayatullah, T.L.E.R. Mengko, R. Munir, A survey on multisperm tracking for sperm motility measurement, IJMLC 7 (5) (Oct. 2017) 144–151, doi:10.18178/ijmlc.2017.7.5.637.
[13] M. Iguer-ouada, J.P. Verstegen, Evaluation of the 'Hamilton Thorn computer-based automated system' for dog semen analysis, Theriogenology 55 (3) (Feb. 2001) 733–749, doi:10.1016/S0093-691X(01)00440-X.
[14] P. Hidayatullah, M. Zuhdi, Automatic sperms counting using adaptive local threshold and ellipse detection, in: 2014 International Conference on Information Technology Systems and Innovation (ICITSI), Nov. 2014, pp. 56–61, doi:10.1109/ICITSI.2014.7048238.
[15] A. Akbar, E. Sukmawati, D. Utami, M. Nuriyadi, I. Awaludin, P. Hidayatullah, Bull sperm motility measurement improvement using sperm head direction angle, TELKOMNIKA (Telecommunication Computing Electronics and Control), 16, Aug. 2018 Art. no. 4, doi:10.12928/telkomnika.v16i4.8685.
[16] F. Shaker, S.A. Monadjemi, A.R. Naghsh-Nilchi, Automatic detection and segmentation of sperm head, acrosome and nucleus in microscopic images of human semen smears, Comput. Methods Programs Biomed. 132 (Aug. 2016) 11–20, doi:10.1016/j.cmpb.2016.04.026.
[17] H. S. Mahdavi, S. A. Monadjemi, and A. Vafaei, "Sperm detection in video frames of semen sample using morphology and effective ellipse detection method," *J Med Signals Sens*, vol. 1, no. 3, Art. no. 3, 2011, doi:10.4103/2228-7477.95392.
[18] F.N. Rahatabad, M.H. Moradi, V.R. Nafisi, A Multi steps algorithm for sperm segmentation in microscopic image, Int. J. Bioeng. Life Sci. 1 (12) (2007) 3, doi:10.5281/zenodo.1061515.
[19] V.S. Abbiramy, V. Shanthi, C. Allidurai, Spermatozoa detection, counting and tracking in video streams to detect asthenozoospermia, in: 2010 International Conference on Signal and Image Processing, Dec. 2010, pp. 265–270, doi:10.1109/ICSIP.2010.5697481.
[20] M. S. Nissen, O. Krause, K. Almstrup, S. Kjærulff, T. T. Nielsen, and M. Nielsen, "Convolutional neural networks for segmentation and object detection of human semen," in *Image Anal.*, vol. 10269, P. Sharma and F. M. Bianchi, Eds. Cham: Springer International Publishing, 2017, pp. 397–406. doi: 10.1007/978-3-319-59126-1_33.
[21] J. Redmon and A. Farhadi, "YOLOv3: An Incremental Improvement," *arXiv:1804.02767 [cs.CV]*, p. 6, 2018.
[22] A. Bochkovskiy, C.-Y. Wang, and H.-Y. M. Liao, "YOLOv4: optimal speed and accuracy of object detection," p. 17, 2020, doi: https://arxiv.org/abs/2004.10934v1.
[23] A. Rosebrock, Deep Learning for Computer Vision with Python: Starter Bundle, 1st ed., PyImageSearch, 2017 [Online]Available: https://books.google.co.id/books?id=9UI-tgEACAAJ .
[24] A. Zheng, Evaluating Machine Learning Models: A Beginner's Guide to Key Concepts and Pitfalls, First. O'Reilly Media, Inc., 2015.
[25] A. Bochkovskiy, "Yolo_mark: Windows & Linux GUI for marking bounded boxes of objects in images for training Yolo v3 and v2," 2019. https://github.com/AlexeyAB/Yolo_mark (accessed Aug. 24, 2019).
[26] Y. Liu, P. Sun, N. Wergeles, Y. Shang, A survey and performance evaluation of deep learning methods for small object detection, Expert Syst. Appl. 172 (Jun. 2021) 114602, doi:10.1016/j.eswa.2021.114602.
[27] N.-D. Nguyen, T. Do, T.D. Ngo, D.-D. Le, An evaluation of deep learning methods for small object detection, J. Electr. Comput. Eng. 2020 (Apr. 2020) 1–18, doi:10.1155/2020/3189691.
[28] J. Redmon, S. Divvala, R. Girshick, A. Farhadi, You only look once: unified, real-time object detection, in: *2016 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, Las Vegas, NV, USA, Jun. 2016, pp. 779–788, doi:10.1109/CVPR.2016.91.
[29] K. He, X. Zhang, S. Ren, J. Sun, Deep residual learning for image recognition, in: *2016 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, Las Vegas, NV, USA, Jun. 2016, pp. 770–778, doi:10.1109/CVPR.2016.90.
[30] N. Tajbakhsh, et al., Convolutional neural networks for medical image analysis: full training or fine tuning? IEEE Trans. Med. Imag. 35 (5) (May 2016) 1299–1312, doi:10.1109/TMI.2016.2535302.
[31] J. Redmon, "YOLO: real-time object detection," 2019. https://pjreddie.com/darknet/yolo/ (accessed Aug. 24, 2019).
[32] A. Bochkovskiy, "Windows and Linux version of Darknet Yolo v3 & v2 neural networks for object detection (Tensor Cores are used): AlexeyAB/darknet," Aug. 24, 2019. https://github.com/AlexeyAB/darknet (accessed Aug. 24, 2019).
[33] A. Bochkovskiy, *AlexeyAB/darknet*. 2021, Jun. 25, 2021. https://github.com/AlexeyAB/darknet.
[34] J. Redmon and A. Farhadi, "YOLO9000: better, faster, stronger," presented at the 2017 IEEE Conference on Computer Vision and Pattern Recognition (CVPR), Jul. 2017. doi:10.1109/CVPR.2017.690.
[35] M. Bernico, Deep Learning Quick Reference: Useful Hacks For Training And Optimizing Deep Neural Networks with TensorFlow and Keras, 1st ed., Packt Publishing Ltd., Birmingham, 2018.