applied sciences

MDPI

Article

SAFP-YOLO: Enhanced Object Detection Speed Using Spatial Attention-Based Filter Pruning

Hanse Ahn [1], Seungwook Son [2], Jaehyeon Roh [1], Hwapyeong Baek [1], Sungju Lee [3,*], Yongwha Chung [1,*] and Daihee Park [1]

[1] Department of Computer Convergence Software, Korea University, Sejong 30019, Republic of Korea; hansahn@korea.ac.kr (H.A.); tngkrshsh@korea.ac.kr (J.R.); qorrns156@korea.ac.kr (H.B.); dhpark@korea.ac.kr (D.P.)
[2] Info Valley Korea Co., Ltd., Anyang 14067, Republic of Korea; sso7199@invako.kr
[3] Department of Software, Sangmyung University, Cheonan 31066, Republic of Korea
* Correspondence: peacfeel@smu.ac.kr (S.L.); ychungy@korea.ac.kr (Y.C.)

Abstract: Because object detection accuracy has significantly improved advancements in deep learning techniques, many real-time applications have applied one-stage detectors, such as You Only Look Once (YOLO), owing to their fast execution speed and accuracy. However, for a practical deployment, the deployment cost should be considered. In this paper, a method for pruning the unimportant filters of YOLO is proposed to satisfy the real-time requirements of a low-cost embedded board. Attention mechanisms have been widely used to improve the accuracy of deep learning models. However, the proposed method uses spatial attention to improve the execution speed of YOLO by evaluating the importance of each YOLO filter. The feature maps before and after spatial attention are compared, and then the unimportant filters of YOLO can be pruned based on this comparison. To the best of our knowledge, this is the first report considering both accuracy and speed with Spatial Attention-based Filter Pruning (SAFP) for lightweight object detectors. To demonstrate the effectiveness of the proposed method, it was applied to the YOLOv4 and YOLOv7 baseline models. With the pig (baseline YOLOv4 84.4%@3.9FPS vs. proposed SAFP-YOLO 78.6%@20.9FPS) and vehicle (baseline YOLOv7 81.8%@3.8FPS vs. proposed SAFP-YOLO 75.7%@20.0FPS) datasets, the proposed method significantly improved the execution speed of YOLOv4 and YOLOv7 (i.e., by a factor of five) on a low-cost embedded board, TX-2, with acceptable accuracy.

Keywords: object detection; deep learning; attention mechanism; filter pruning

Citation: Ahn, H.; Son, S.; Roh, J.; Baek, H.; Lee, S.; Chung, Y.; Park, D. SAFP-YOLO: Enhanced Object Detection Speed Using Spatial Attention-Based Filter Pruning. *Appl. Sci.* **2023**, *13*, 11237. https://doi.org/10.3390/app132011237

Academic Editor: Andrea Prati

Received: 19 August 2023
Revised: 9 October 2023
Accepted: 10 October 2023
Published: 12 October 2023

1. Introduction

Object detection accuracy has significantly improved with advancements in deep learning techniques [1]. As a result, many real-time applications have adopted one-stage detectors like YOLO [2–8] due to their fast execution speed and accuracy. YOLOv4 has incorporated numerous recent techniques, and the COCO accuracies of later YOLO versions, including YOLOv4 [5], YOLOv5 [6], YOLOx [7], and YOLOv7 [8], far surpass earlier iterations, such as YOLOv1 [2], YOLOv2 [3], and YOLOv3 [4]. Therefore, many studies have applied later versions of YOLO to many real-time applications, such as 24 h pig monitoring [9–29] and autonomous driving [30–42].

However, for a practical deployment, deployment costs should be considered, as there is a general trade-off between accuracy and execution speed in any object detection system. Although embedded board implementations of YOLO have been reported, there has been an even greater demand for "integrated" performance (=accuracy × speed) on a low-cost embedded board.

In this paper, a method is proposed to reduce the number of unimportant filters of YOLO to provide improved integrated performance on a low-cost embedded board. Generally, attention mechanisms have been widely used to improve the accuracy of deep

learning models [43–46], while pruning techniques have been used to improve the execution speed [47–52]. In the proposed SAFP method, spatial attention mechanisms [45,46] are used to improve the execution speed of YOLO by evaluating the importance of each YOLO filter and pruning unimportant filters (i.e., filter pruning [49,50]). That is, the feature maps before and after spatial attention are compared, and then the unimportant filters of YOLO are pruned based on this comparison. Our research distinguishes itself from recent research [53–57] that utilizes attention for filter pruning in image classification tasks. The technique for pruning unimportant filters of YOLO is proposed by leveraging information from spatial attention, in contrast to the previous methods for image classification. By comparing the feature map difference after spatial attention and the foreground–background information derived from ground-truth (GT) box locations, unimportant filters of YOLO can be determined.

Figure 1 illustrates the application of spatial attention to feature maps detecting features, highlighting the differences between them: (a) represents a feature map extracted from the input image, capturing the characteristics of the objects effectively. In contrast, (b) represents the outcome of applying spatial attention to the feature map in (a), emphasizing the object's characteristics. The difference between (a) and (b), in addition to the GT box locations of the training image, is depicted in (c), illustrating that applying spatial attention to a feature map that well-represents the object's characteristics can enhance the emphasis on those features (i.e., important filters should not be pruned).

On the other hand, (d) represents a feature map obtained from the input image that does not adequately represent the object's characteristics. (e) shows the result of applying spatial attention to the feature map in (d), while (f) represents the difference between (d) and (e), in addition to the GT box locations of the training image. As evidenced by the results in (f), even the application of spatial attention does not emphasize the features in a map that lacks a proper representation of the object's characteristics. In this study, convolution filters that produce feature maps similar to (f), which fail to accurately represent the object's characteristics, even after applying spatial attention, are identified as unimportant filters for object detection. These filters recognize them as non-essential to enhance the overall efficacy of the detection process (i.e., unimportant filters can be pruned).

To demonstrate its effectiveness, the proposed method was applied to YOLOv4 [5] and YOLOv7 [8]. With the pig and vehicle datasets, the proposed method significantly improved the execution speed of YOLOv4 and YOLOv7 on a low-cost embedded board, TX-2, with acceptable accuracy. Compared to the baseline YOLO model, the proposed method can improve integrated performance (accuracy x speed) up to five times. The contributions of this work are summarized as follows:

- Attention mechanisms have been used to improve the accuracy of deep learning models, while pruning techniques have been used to improve the execution speed. A step is taken forward from previous research by combining attention and pruning to achieve real-time YOLO on a low-cost embedded board with acceptable accuracy. To the best of our knowledge, this represents the first instance of concurrently considering both the accuracy and speed of YOLO by comparing the foreground–background information derived from GT box locations and the feature map difference before and after spatial attention.
- To validate the feasibility of our proposed method, the experiments were conducted comparing the proposed method to baseline models in terms of accuracy and speed across various settings. The experiments with two datasets (pig and vehicle), two attention mechanisms (convolutional block attention network, CBAM and coordinate attention, CA), two detection models (YOLOv4 and YOLOv7), and two model sizes (medium and tiny) were conducted. Additionally, comparative experiments were carried out with conventional filter-pruning techniques based on magnitude and randomness.

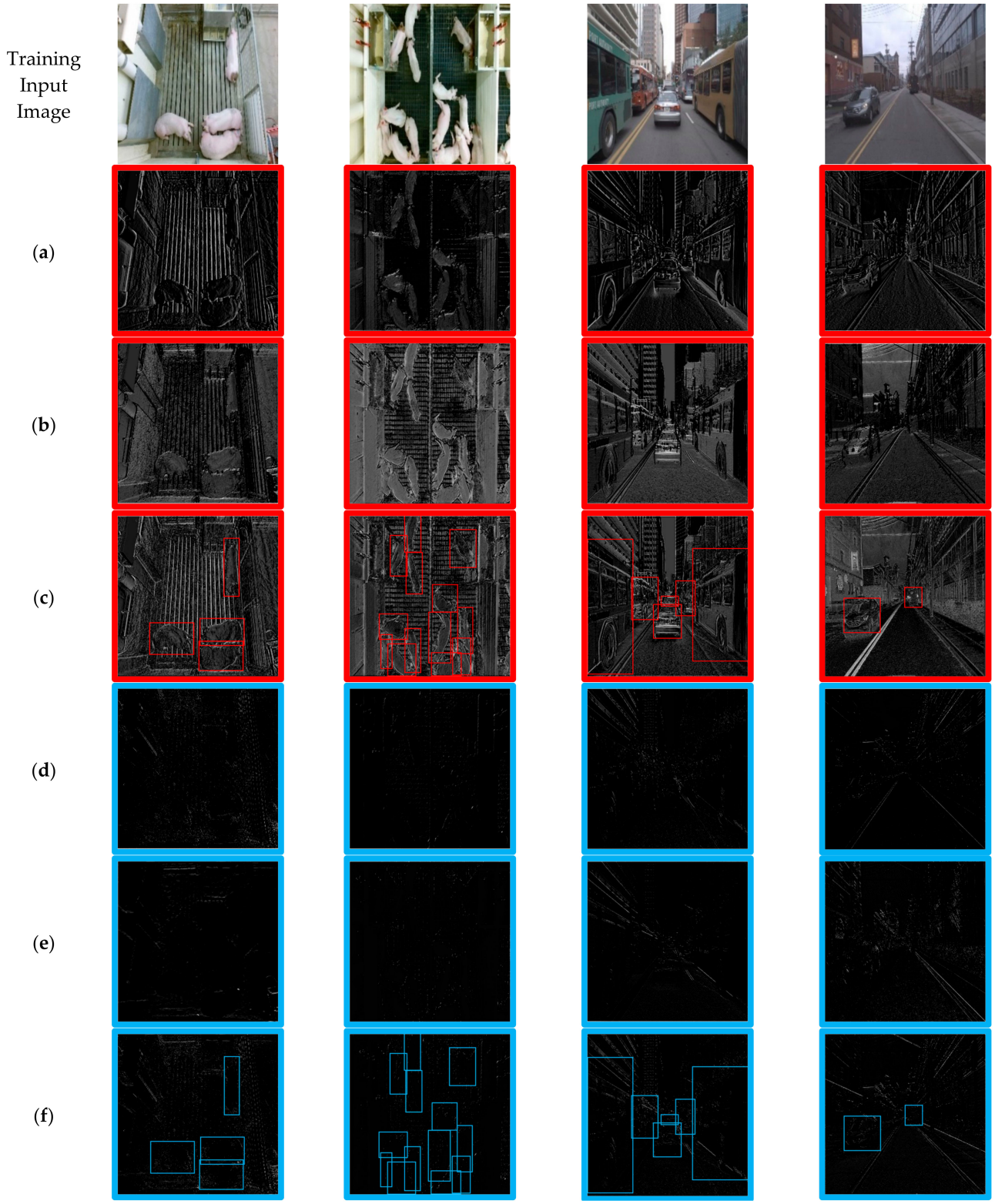

Figure 1. For important filters (shown as red rows), the comparison result (**c**) of the foreground–background information derived from GT box locations and the feature map difference before (**a**) and after (**b**) spatial attention can be used not to prune the filters. For unimportant filters (shown as blue rows), the comparison result (**f**) of the foreground–background information derived from GT box locations and the feature map difference before (**d**) and after (**e**) spatial attention can be used to prune the filters.

2. Background

This study aims to improve the execution speed of object detection. Although YOLO is relatively fast and has been widely used in many real-time applications, our objective is to further enhance its execution speed, particularly for practical deployment on low-cost embedded boards. In particular, there is a general trade-off between accuracy and execution speed in any object detection system, including YOLO. Although our main goal is to improve the execution speed of YOLO, the 'integrated' performance (=accuracy × speed) should also be improved by considering the accuracy of the proposed method. Given that attention mechanisms have proven effective in enhancing the accuracy of deep learning models, our approach leverages spatial attention mechanisms to not only improve YOLO's execution speed but also enhance its overall integrated performance.

Attention. The methods for diverting attention to the most important locations in an image and disregarding unimportant locations are called attention mechanisms [44]. Attention mechanisms have provided benefits in many computer vision tasks, such as image classification and object detection. For image classification tasks, researchers have developed channel attention (what to pay attention to) [43,44] to improve the accuracy of deep learning-based classification models. In contrast, for object detection tasks, researchers have developed spatial attention (where to pay attention), such as CBAM [45] and CA [46], to improve the accuracy of deep learning-based detection models. In fact, CBAM and CA provide both channel and spatial attention mechanisms. Since they provide a spatial attention (where to pay attention) mechanism, they were selected as the attention mechanisms of the proposed method for object detection tasks. Figure 2 shows the structure of CBAM, and Figure 3 shows the structure of CA.

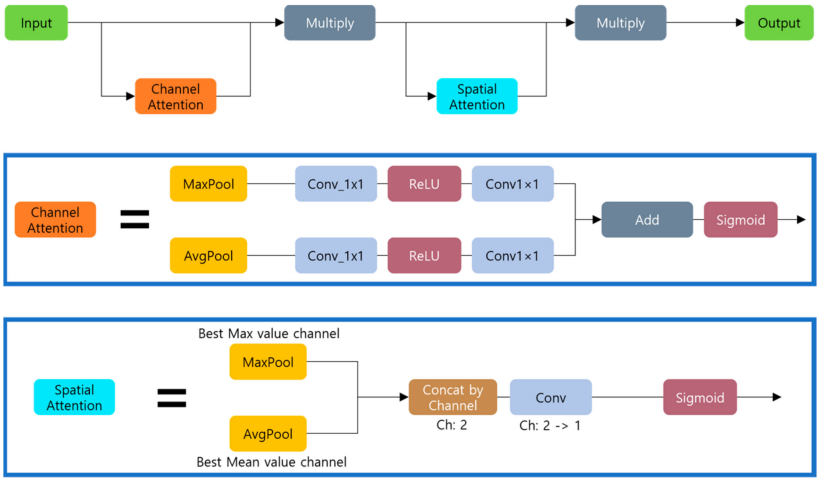

Figure 2. Structure of CBAM [45].

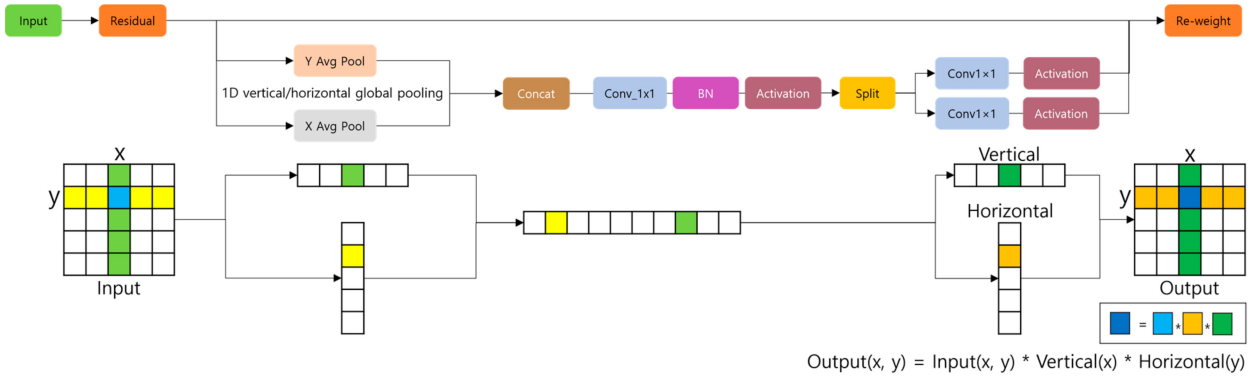

Figure 3. Structure of CA [46].

Pruning. The methods for removing the components of a model to produce sparse models for acceleration and compression are called pruning techniques [50]. Pruning aims to minimize the number of parameters in a model without significantly degrading its accuracy. In contrast to weight pruning (which results in unstructured models), structured pruning provides realistic acceleration by producing GPU-friendly models. Therefore, this study pruned unimportant filters (i.e., filter pruning) as an example of structured pruning. Furthermore, like attention, most research on pruning has been conducted on convolutional neural networks (CNNs) for image classification tasks, which are the foundation for other tasks, such as object detection. Conventionally, filter pruning has been performed using magnitude pruning [51] and random pruning [52]. In the case of magnitude pruning, filters are considered more important when their values are larger, and the importance of filters is determined using metrics like L2-norm. On the other hand, random pruning involves randomly selecting filters for pruning.

In general, attention mechanisms have been widely used to improve the accuracy of deep learning models, whereas pruning techniques have been used to improve their execution speed. Table 1 shows a summary of the attention-based pruning results published between 2019 and 2022. As shown in Table 1, few classification results have been reported for attention-based pruning, while the detection results of attention for accuracy and pruning for speed have been reported. To the best of our knowledge, no prior results on the use of spatial attention-based pruning considering YOLO's detection accuracy and speed have been reported. In general, there exists an accuracy/speed trade-off, and therefore, introducing attention mechanisms to improve accuracy increases the resource requirements (time and memory), while pruning for speed sacrifices accuracy. For example, in [26], applying attention improved accuracy from 90.4% to 90.8%; however, the speed decreased slightly from 83FPS to 82FPS. Similarly, in [39], the application of attention improved accuracy from 95.2% to 97.2%; however, it was reported that the model size increased slightly from 13.7MB to 14.4MB. However, to the best of our knowledge, no prior results on the use of spatial attention-based pruning for simultaneously considering YOLO detection accuracy and speed have been reported.

Table 1. Summary of attention-based pruning results (2019–2022) for classification speed and detection accuracy/speed.

Computer Vision Task	Attention or Pruning	Reference
Image Classification	Channel Attention for Speed	[53]
	Channel Attention for Speed	[54]
	Channel Attention for Speed	[55]
	Channel Attention for Speed	[56]
	Channel Attention for Speed	[57]
Object Detection	Attention for Accuracy (Not Report Speed/Model Size)	[13]
	Attention for Accuracy (Not Report Speed/Model Size)	[25]
	Attention for Accuracy (Speed Degradation)	[26]
	Attention for Accuracy (Model Size Increase)	[39]
	Pruning for Speed (Not Report Accuracy)	[18]
	Spatial Attention-based Pruning for Accuracy and Speed	Proposed Method

3. Proposed Method

In this study, a lightweight spatial attention model is proposed. First, spatial attention is applied to the target model. Next, the filters within the convolution layer are evaluated using an attention map. Finally, filter pruning is performed using the evaluated filters. The overall structure of the proposed method is illustrated in Figure 4.

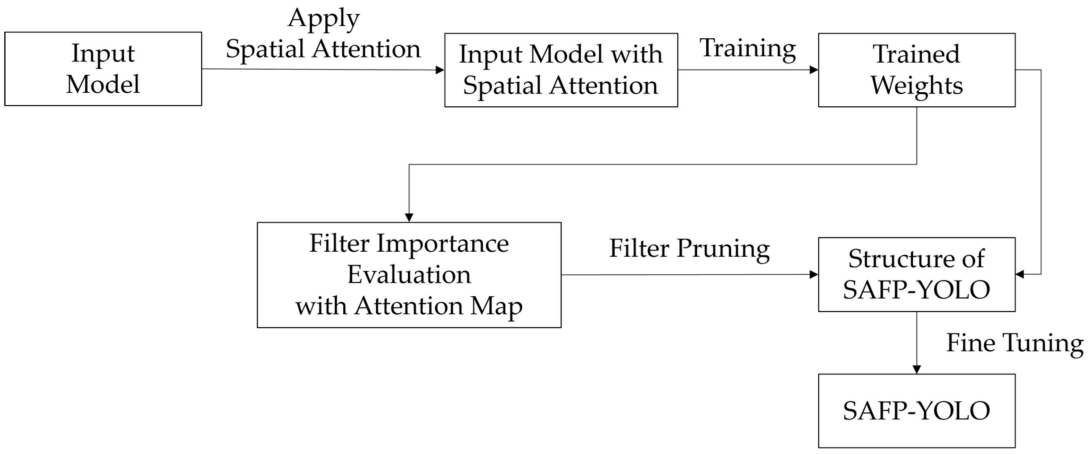

Figure 4. Overview of the proposed method SAFP-YOLO.

3.1. Spatial Attention

In this section, spatial attention modules CBAM [45] and CA [46] are introduced, and we discuss how they are applied to the YOLOv4 and YOLOv7 frameworks with the intention of enhancing the accuracy and efficiency of object detection. Similar to the principles of SENet, CBAM examines the relationships among filters to ascertain which ones need encoding. This is not the full extent of CBAM's functionality; it also encodes pixel-wise attention across the entire filter, helping to determine the regions that require more focused processing. This targeted approach ensures that the most relevant information is captured. Ultimately, these encoded attention maps are amalgamated with the original input feature map, thus refining the overall feature representation. In contrast, CA serves a slightly different purpose. CA is a specific spatial attention mechanism designed to guide the learning model towards filters and positions that demand focus. CA achieves this by applying attention based on the values gleaned from the global average pooling along the horizontal and vertical axes of each filter. This method is not arbitrary; it simultaneously considers the importance and spatial information among filters. The outcome is an improved object detection accuracy, as the model is better informed about where to concentrate its resources. While conventional filter-pruning techniques employing attention modules have predominantly utilized channel attention modules, this study aims to implement pruning by combining spatial attention modules with GT Box. This preference stems from our belief that, in detection tasks as opposed to classification, spatial attention modules are more effective than channel attention modules.

To effectively integrate these attention mechanisms into the YOLOv4 framework, they are strategically applied at the end of each CBM (Conv-Bn-Mish) and CBL (Conv-Bn-Leaky) module in the backbone and neck of the architecture since it enables to obtain attention maps from the feature maps of the convolution layer. Such integration is not merely an add-on; it is an integral part of enhancing the model's responsiveness to the spatial features that are used for accurate object detection. Figure 5 illustrates the structure of YOLOv4 with the Spatial Attention (SA) modules, showcasing how these SA modules blend seamlessly into the existing architecture. For the YOLOv7 framework, SA modules are applied to each end of the MP (Max Pooling), ELAN (Efficient Layer Aggregation Networks), and ELAN-H (ELAN with HorNet) modules in the backbone and neck to obtain attention maps from the feature map of the convolution layer, and Figure 6 illustrates the structure of YOLOv7 with

SA. By incorporating the SA module, attention maps can be derived from the convolutional layers of both YOLOv4 and YOLOv7.

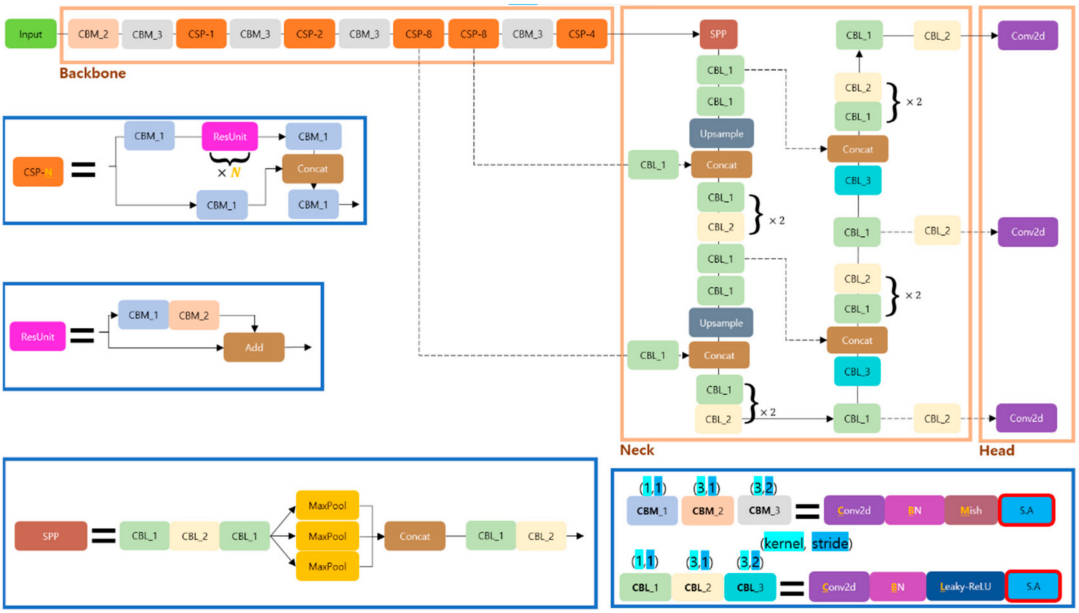

Figure 5. Structure of YOLOv4 with SA (incorporated into backbone and neck).

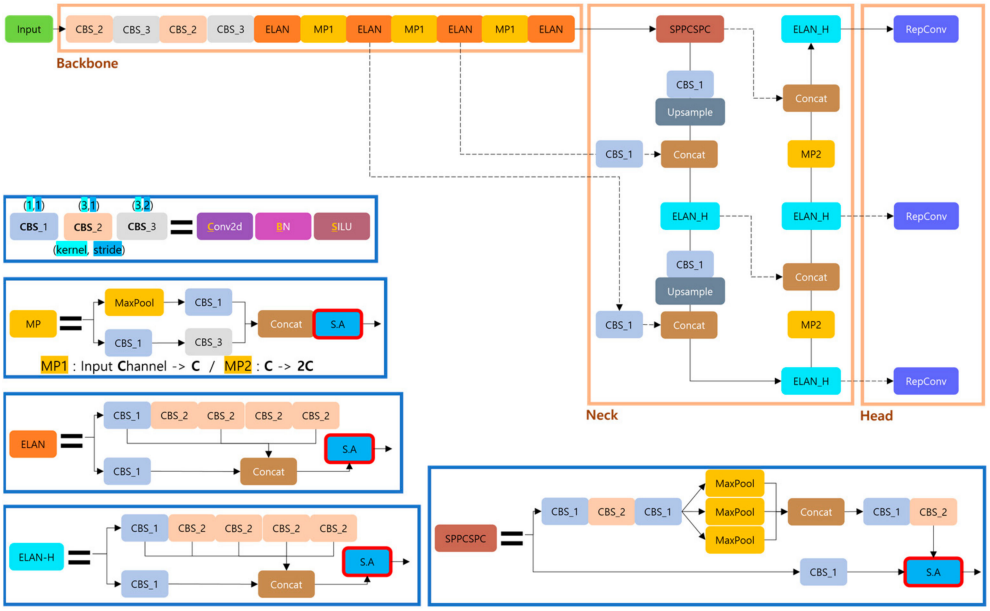

Figure 6. Structure of YOLOv7 with SA (incorporated into backbone and neck).

3.2. Filter Importance Evaluation

In this section, a comprehensive method is introduced to assess the significance of each filter within the convolutional layers of an object detection model. The primary aim of this evaluation is to understand the roles and contributions of different filters within the convolutional architecture, an essential aspect that drives the overall performance of the model. An evaluation is commenced by using a trained model and inputting an image containing the object for detection. As the input image contains the GT detection box's information, the precise location of the object can be located within the image and can serve as the basis for further analysis. Following the input process, feature maps are obtained from each convolution layer. Feature maps contain filter responses within a particular

layer, which provide intricate operations of the convolutional layers, allowing us to discern the importance of individual filters. To assess filter importance systematically, the feature maps are divided into two distinct regions, foreground and background, leveraging the GT detection box information for a clear differentiation between the object of interest and its surroundings.

With the regions defined, L2-norm values are calculated for both the foreground and background areas. Then, the absolute difference is computed in the L2-norm values between these regions, as it provides a quantitative measure of the filter's significance. A larger difference in the L2-norm values signifies the filter's pronounced role in distinguishing features. A significantly higher L2-norm value for the foreground region indicates that the filter excels at extracting features related to either the object or its surroundings. Such filters are regarded as highly relevant for the object detection task. By doing so, they can be ranked in terms of importance. This ranking not only provides insights into filter dynamics but also facilitates informed decisions regarding filter pruning and other optimization techniques, all designed to enhance the model's performance through fine-tuning without unnecessary complexity.

The entire process leads to a more refined understanding of the model, which is based on an empirical analysis of the filters' roles. In this study, Algorithm 1 is proposed to outline the specific filter-pruning algorithm used, offering a clear pathway to achieve the described objectives.

Algorithm 1. Filter evaluation using attention map

Input: model, featureMaps, groundTruthBox, inputImage
Output: optimizedModel

```
foreach (layer in convolutionLayers(model)) {
  featureMaps = extractFeatureMaps(layer, inputImage)

   foregroundRegion = getRegion(featureMaps, groundTruthBox)
  backgroundRegion = getRegion(featureMaps, complementOf(groundTruthBox))

  L2NormForeground = calculateL2Norm(foregroundRegion)
  L2NormBackground = calculateL2Norm(backgroundRegion)

  difference = abs(L2NormForeground − L2NormBackground)

   if (difference is significantly large) {
    markFilterAsImportant(layer)
   }
  }

 rankedFilters = rankFiltersBasedOnImportance(convolutionLayers(model))

 optimizedModel = applyOptimizations(model, rankedFilters)

 return optimizedModel
```

3.3. Filter Pruning

In this section, a method is discussed for creating a filter-pruned model after evaluating the filters within the convolutional layers. Our proposed filter-pruning method focuses on removing filters at a fixed ratio, such as 50%, a critical step for achieving lightweight models and efficient deployment in deep learning applications.

Initially, the reduction ratio 'r' is determined for filter pruning through a series of systematic experiments aimed at discovering the optimal value that balances efficiency and effectiveness. Once this ratio is established, the filters are sorted in each convolution

layer in descending order based on their previously evaluated importance, ensuring that we preserve the filters considered most vital for the model's operation.

Next, filters are pruned in each convolution layer, starting with the least important ones. This dual-purpose approach allows us to reduce the model's size and enhance the inference speed without sacrificing core functionalities. This step is crucial for tailoring the model to various resource constraints and computational environments. Following the pruning process, the values of the remaining filters are applied in each convolutional layer to a modified version of the model, where spatial attention is not employed. At this stage, the model undergoes filter pruning with the fixed ratio 'r'. This adaptation streamlines the model's structure and sets the stage for subsequent fine-tuning.

Then, fine-tuning techniques are employed on the model devoid of spatial attention, inheriting the values of the filters from the previous convolution layer. This approach allows us to preserve the model's accuracy while achieving a significant reduction in the number of parameters. This strategic decision plays a crucial role in the overall design, as it fulfills the objectives of enabling the model to function well in resource-constrained environments like embedded systems and mobile devices.

Furthermore, excluding spatial attention from the final model is a deliberate step taken to simplify the deployment of deep learning models in scenarios with limited computational resources. This consideration aligns with broader trends in the field, addressing the increasing demand for adaptable and efficient models. The integration of these techniques may result in performance and efficiency that meet the demands of real-world conditions, especially in settings where deep learning models must operate under various resource constraints. Algorithm 2 presents the filter-pruning algorithm proposed in this study.

Algorithm 2. Filter pruning using attention map

```
Input: model
Output: finalModel
Initialize: reduction ratio r

  newModel = createModelWithoutSpatialAttention()

  foreach (layer in convolutionLayers(model)) {
    sortedFilters = sortFiltersByImportance(layer)
    int numFiltersToPrune = numberOfFilters(layer) * r

    for int i = 0 to numFiltersToPrune do
      removeFilter(sortedFilters, sortedFilters[numberOfFilters(sortedFilters) – i − 1])


    applyPrunedFiltersToNewModel(sortedFilters, newModel)
  }

 finalModel = fineTuning(newModel)

  return finalModel
```

4. Experimental Results

4.1. Experimental Setup and Resources for the Experiment

In this study, pig and vehicle datasets were used to evaluate the proposed method. The pig dataset was released by Riekert et al. [58], while the vehicle dataset was released by Argoverse [59]. As shown in Figure 7, the pig dataset was captured by either top-view or tilted-view cameras across various pig pen structures, and the vehicle dataset using a side-view camera showed various scale vehicles.

Figure 7. Test sample images of pig [58] and vehicle [59] datasets.

The deep learning model was trained on a PC with an AMD Ryzen 5950 × 16-core processor, a GeForce RTX 3090 GPU, and 32 GB of RAM. The model was trained on 10 epochs with a learning rate of 0.00261. In addition, the anchor-box configuration comprised nine anchors that were determined using the k-means algorithm. For the embedded board implementation, the Nvidia Jetson TX-2 embedded board was used (dual-core Denver2 64-bit CPU and quad-core ARM A57 complex, NVIDIA Pascal™ architecture with 256 NVIDIA CUDA cores, 8 GB 128-bit LPDDR4) [60].

4.2. Evaluation of Detection Performance

In this section, we evaluate the proposed method. Tables 2 and 3 present the results of applying the proposed method to the YOLOv4 and YOLOv7 baselines for the pig dataset and vehicle dataset. The inference speeds (FPS) of the models were measured on a TX-2 embedded board [60], and the integrated performance was defined as the product of accuracy and speed. Although there was a decrease in accuracy, 84.4 to 78.6 in YOLOv4 and 87.9 to 81.2 in YOLOv7, the actual gain in inference speed was substantial: 3.9 to 20.9 in YOLOv4 and 3.8 to 20.0 in YOLOv7, which led to an increase in the integrated performance of the lightweight models. Furthermore, both YOLOv4 and YOLOv7 showed performance improvements in terms of the number of model parameters as well. This confirms the applicability of the proposed method to various models. Additionally, although CBAM and CA exhibited differences in accuracy, they both contributed to an improved integrated performance. Thus, the proposed method can be applied to various spatial attention mechanisms. Also, this confirms that the proposed method can be applicable to various dataset types.

Figure 8 illustrates the detection results of the proposed SAFP-YOLOv7 with CA (87.5% pruning). As shown in Tables 2 and 3, the accuracy of the proposed SAFP-YOLOv7 with CA (87.5% pruning) has decreased compared to the baseline YOLOv7, yet it is confirmed that most objects were successfully detected. However, false detections were observed when objects were clustered or distant, leading to smaller sizes within the image or when obscured by infrastructure. Among the cases of false detection, it is anticipated that the detection accuracy could be improved while maintaining detection speed for those cases where objects were distant, leading to smaller sizes within the image or obscured by infrastructure, by employing the methods described in [20,38].

Figure 8. Detection results of the proposed SAFP-YOLOv7 with CA (87.5% pruning). Even with a pruning rate of 87.5% from YOLOv7, most objects were successfully detected except occluded pigs and small vehicles.

Table 2. Comparison of the proposed method with YOLOv4 and YOLOv7 for the pig dataset on a TX-2.

	Method		Accuracy ↑ ($AP_{0.5}$, %)	Speed ↑ (FPS)	No. of Model Parameters ↓ (M)
YOLOv4 [5]	Baseline YOLOv4		84.4	3.9	52.5
	Proposed SAFP with CBAM [45]	50% Pruning	84.7	6.0	36.7
		75% Pruning	80.5	12.0	25.7
		87.5% Pruning	78.1	20.9	15.7
	Proposed SAFP with CA [46]	50% Pruning	85.0	6.0	36.7
		75% Pruning	81.6	12.0	25.7
		87.5% Pruning	78.6	20.9	15.7
YOLOv7 [8]	Baseline YOLOv7		87.9	3.8	36.7
	Proposed SAFP with CBAM [45]	50% Pruning	86.7	5.8	25.8
		75% Pruning	83.7	11.5	16.3
		87.5% Pruning	80.5	20.0	10.2
	Proposed SAFP with CA [46]	50% Pruning	86.9	5.8	25.8
		75% Pruning	84.2	11.5	16.3
		87.5% Pruning	81.2	20.0	10.2

Table 3. Comparison of the proposed method with YOLOv4 and YOLOv7 for the vehicle dataset on a TX-2.

	Method		Accuracy ↑ ($AP_{0.5}$, %)	Speed ↑ (FPS)	No. of Model Parameters ↓ (M)
YOLOv4 [5]	Baseline YOLOv4		78.1	3.9	52.5
	Proposed SAFP with CBAM [45]	50% Pruning	77.4	6.0	36.7
		75% Pruning	74.3	12.0	25.7
		87.5% Pruning	72.1	20.9	15.7
	Proposed SAFP with CA [46]	50% Pruning	77.5	6.0	36.7
		75% Pruning	74.4	12.0	25.7
		87.5% Pruning	72.6	20.9	15.7
YOLOv7 [8]	Baseline YOLOv7		81.8	3.8	36.7
	Proposed SAFP with CBAM [45]	50% Pruning	80.7	5.8	25.8
		75% Pruning	78.2	11.5	16.3
		87.5% Pruning	75.4	20.0	10.2
	Proposed SAFP with CA [46]	50% Pruning	80.8	5.8	25.8
		75% Pruning	78.7	11.5	16.3
		87.5% Pruning	75.7	20.0	10.2

4.3. Discussion

In this section, the validation of the proposed method is presented. Figures 9 and 10 show the results of applying the conventional filter-pruning techniques (i.e., magnitude pruning [51] and random pruning [52]) to the pig dataset and vehicle dataset. Magnitude pruning involved determining the importance of filters based on their L2-norm values and then performing filter pruning. On the other hand, random pruning simply involves randomly selecting filters for pruning.

As shown in Figures 9 and 10, the application of filter pruning increased the inference speed and reduced the model size of the baseline model. However, the conventional filter-pruning technique, by using L2-norm [51] or randomly removing filters [52], led to substantial accuracy degradation, regardless of the dataset. On the other hand, the proposed SAFP could provide an accuracy of 81.2% (vs. 87.9% baseline YOLOv7) for the pig dataset and 75.7% (vs. 81.8% baseline YOLOv7) for the vehicle dataset, even with 87.5% pruning. That is, with acceptable accuracy, SAFP could increase the inference speed (by a factor of five) and reduce the model size (by a factor of three) of the baseline model, regardless of the dataset.

Figure 11 illustrates the feature maps generated by the convolution filters pruned through both the method proposed in this study and the conventional filter-pruning techniques [51,52]. As depicted in the figure, the feature maps produced by the convolution filters pruned using the proposed method were observed to either not represent or minimally represent the features of the objects that should be detected, indicating that filters likely to be unimportant in object detection were pruned. Conversely, the feature maps representing the convolution filters pruned through the conventional filter-pruning techniques were found to have also eliminated filters that accurately represent the features of the objects that need to be detected. Thus, even filters that were likely to be important in object detection were pruned. Consequently, it was confirmed that by selectively pruning filters that do not adequately represent the features of the objects through the proposed method, a reduction in accuracy can be minimized.

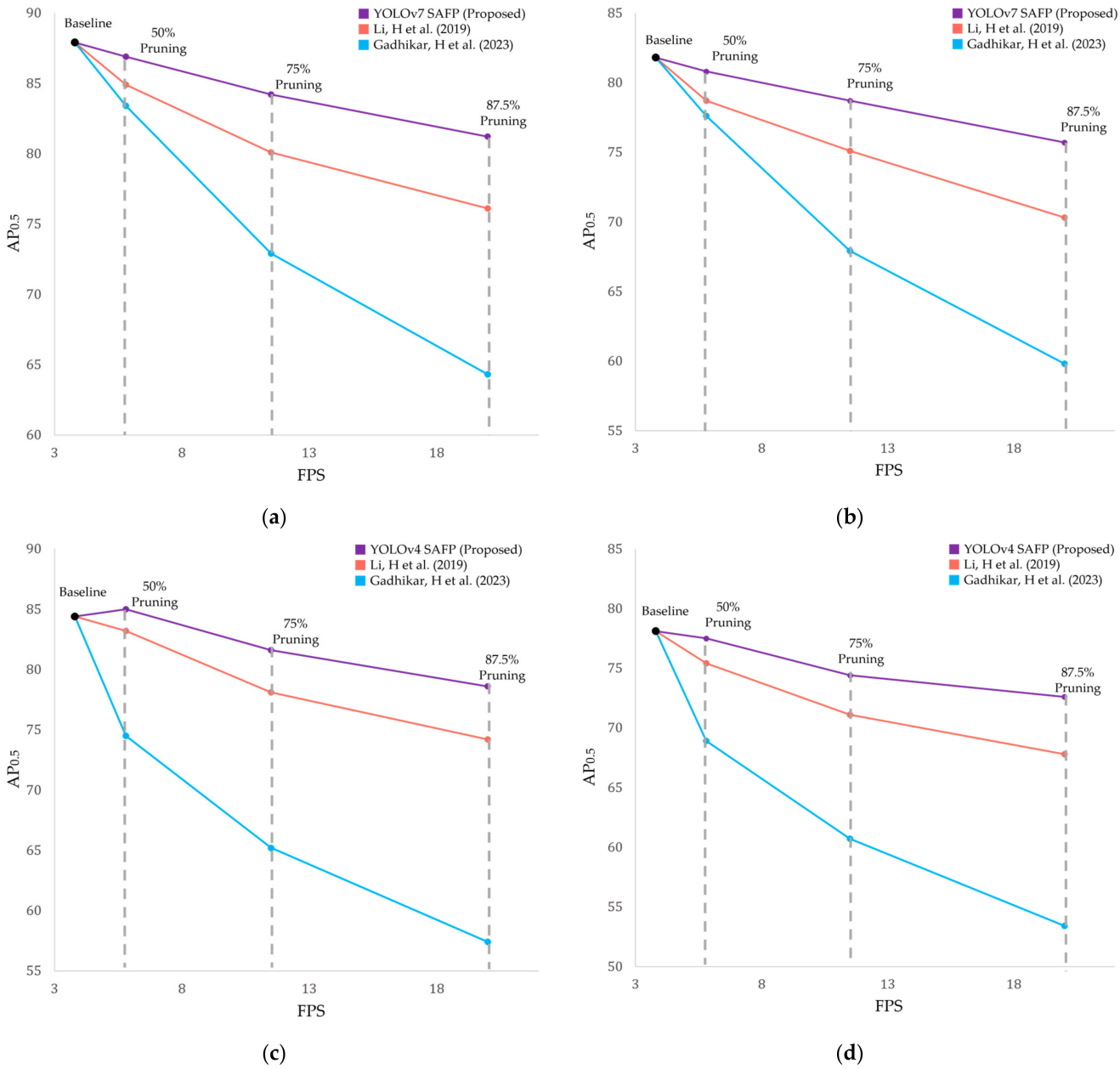

Figure 9. Accuracy–speed comparison of the filter-pruning techniques with YOLOv4 and YOLOv7 (**a**,**c**) shows the results on the pig dataset and (**b**,**d**) shows the results on the vehicle dataset. Compared to the conventional filter-pruning techniques [51,52], the proposed SAFP could increase the execution speed of the baseline model with acceptable accuracy degradation.

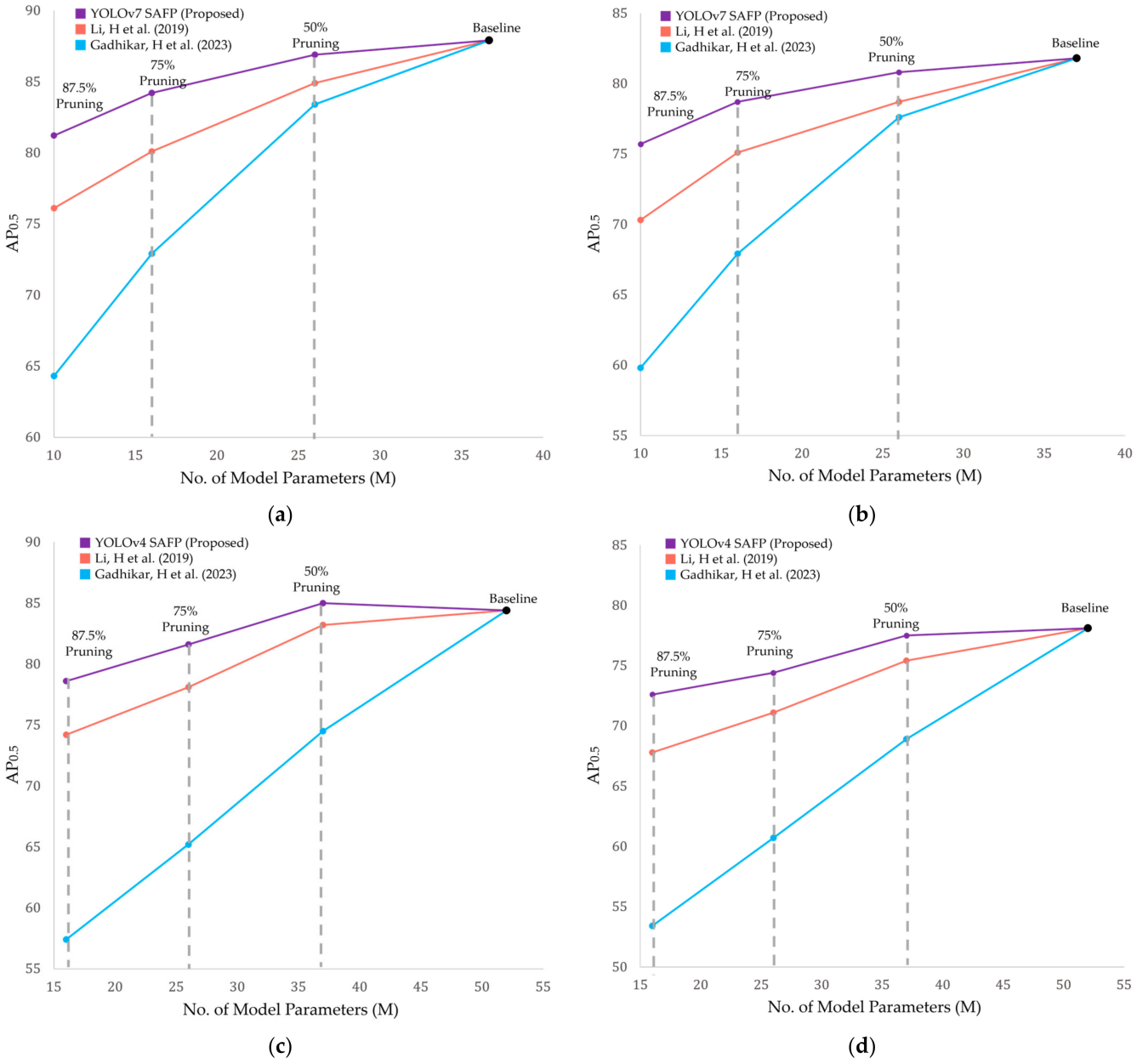

Figure 10. Accuracy–model size comparison of the filter-pruning techniques with YOLOv4 and YOLOv7 (**a**,**c**) shows the results on the pig dataset and (**b**,**d**) shows the results on the vehicle dataset. Compared to the conventional filter-pruning techniques [51,52], the proposed SAFP could reduce the model size of the baseline model with acceptable accuracy degradation.

Figure 12 presents the attention maps obtained by applying Grad-Cam [61] to the models using the proposed method and the conventional filter-pruning techniques [51,52]. Similar to Figure 11, it can be observed that in the case of the proposed method, attention is generally focused on the objects of interest to be detected. On the other hand, in the models applying the conventional filter-pruning techniques, some attention is directed towards objects; however, it is also noticeable that more attention is concentrated on areas such as the background that should not be detected. In other words, when applying the proposed method presented in this paper, it can be confirmed that more important filters for detection are retained compared to the conventional filter-pruning techniques, thereby aiding in the detection process.

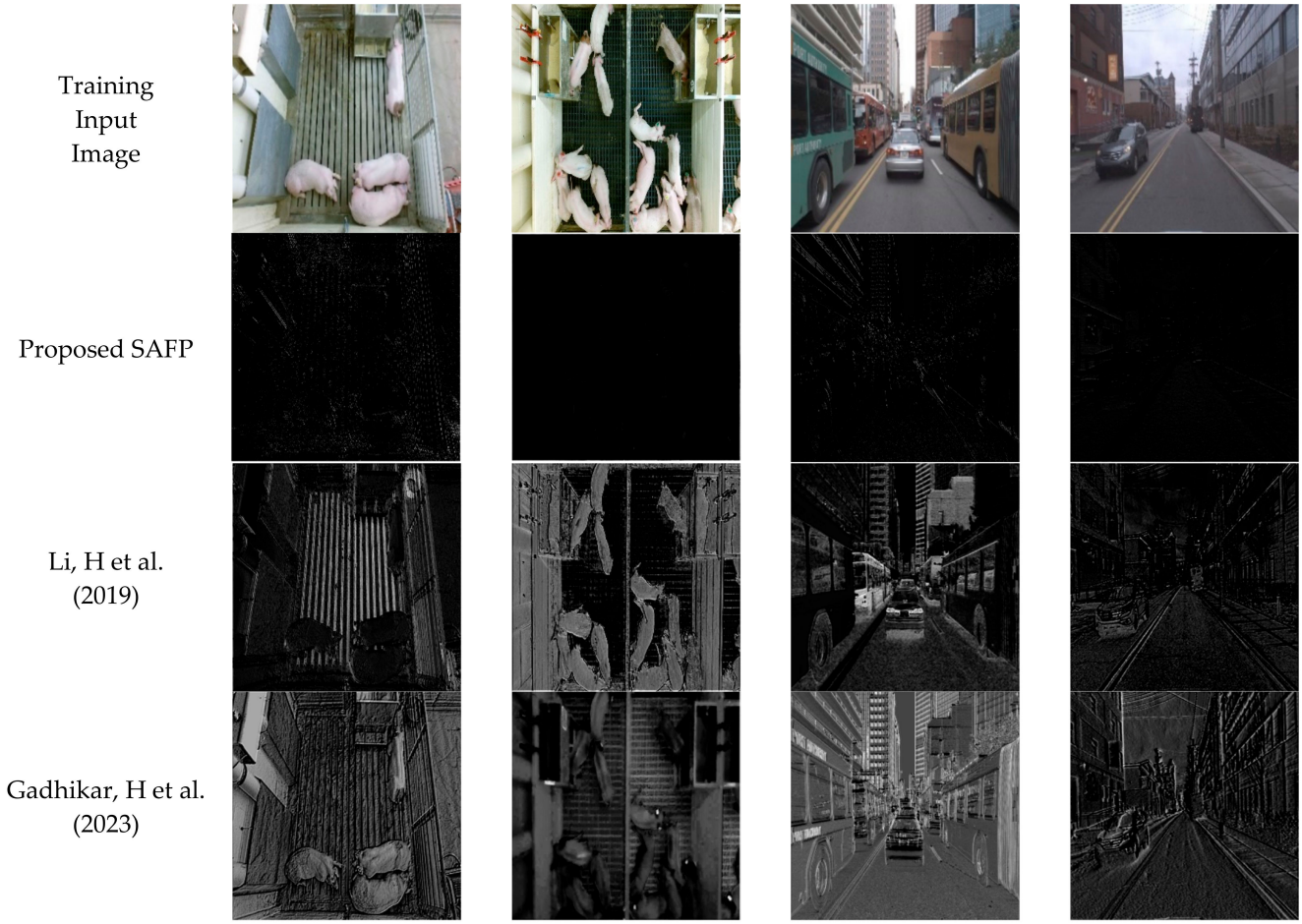

Figure 11. Feature maps from pruned filters from the proposed method and the conventional filter-pruning techniques [51,52]. Unimportant filters were pruned in the proposed method, while important filters were pruned in the conventional filter-pruning techniques.

Tables 4 and 5 present the results of applying the proposed method to TinyYOLOv4 and TinyYOLOv7, which are lightweight models of YOLOv4 and YOLOv7, respectively, on the pig and vehicle datasets. As shown in Tables 4 and 5, there was a decrease in accuracy; however, the relative increase in inference speed compared to the accuracy loss was substantial. Furthermore, the proposed method could significantly reduce the model size. This confirmed that the proposed method can be applicable to tiny models. Also, this confirms that the proposed method with YOLOv4 and YOLOv7 could provide better performance than TinyYOLOv4 and TinyYOLOv7 baseline models, regardless of the dataset type. With the pig dataset, the proposed method (applied to YOLOv4 with 78.6% accuracy at 20.9FPS and YOLOv7 with 81.2% accuracy at 20.0FPS) outperformed the tiny baseline (TinyYOLOv4 with 78.2%@19.2FPS and TinyYOLOv7 with 81.9%@18.3FPS). Additionally, it has been confirmed that the proposed method (applied to YOLOv4 with 72.6%@20.9FPS and YOLOv7 with 75.7%@20.0FPS) was superior for detecting vehicles when compared to the tiny baseline (TinyYOLOv4 with 68.5%@19.2FPS and TinyYOLOv7 with 71.3%@18.3FPS).

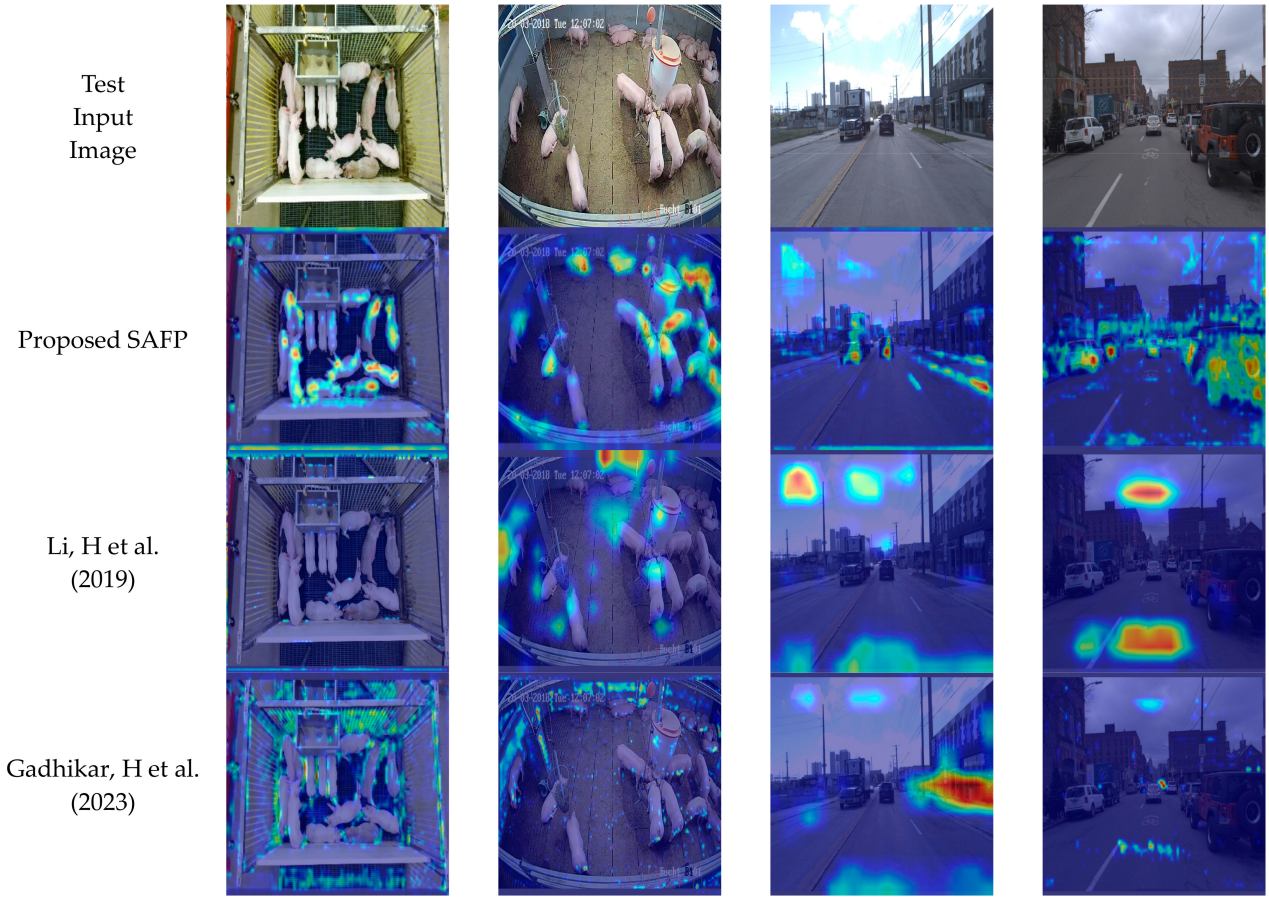

Figure 12. Attention maps from the proposed method and the conventional filter-pruning techniques [51,52]. The attention maps from the proposed method predominantly focus on the target objects. In contrast, using the conventional filter-pruning techniques was observed as lacking this focused attention on the objects.

Table 4. Comparison of the proposed method with TinyYOLOv4 and TinyYOLOv7 for pig dataset on a TX-2.

	Method		Accuracy ↑ ($AP_{0.5}$, %)	Speed ↑ (FPS)	No. of Model Parameters ↓ (M)
Tiny YOLOv4 [5]	Baseline TinyYOLOv4		78.2	19.2	6.1
	Proposed SAFP with CBAM [45]	50% Pruning	76.8	28.4	2.4
		75% Pruning	74.1	40.2	1.0
		87.5% Pruning	71.4	52.9	0.4
	Proposed SAFP with CA [46]	50% Pruning	77.4	28.4	2.4
		75% Pruning	74.9	40.2	1.0
		87.5% Pruning	72.3	52.9	0.4
Tiny YOLOv7 [8]	Baseline TinyYOLOv7		81.9	18.3	6.0
	Proposed SAFP with CBAM [45]	50% Pruning	79.3	27.2	2.3
		75% Pruning	77.6	38.4	1.0
		87.5% Pruning	74.7	50.7	0.4
	Proposed SAFP with CA [46]	50% Pruning	81.0	27.2	2.3
		75% Pruning	78.5	38.4	1.0
		87.5% Pruning	75.8	50.7	0.4

Table 5. Comparison of the proposed method with TinyYOLOv4 and TinyYOLOv7 for vehicle dataset on a TX-2.

	Method		Accuracy ↑ ($AP_{0.5}$, %)	Speed ↑ (FPS)	No. of Model Parameters ↓ (M)
Tiny YOLOv4 [5]	Baseline TinyYOLOv4		68.5	19.2	6.1
	Proposed SAFP with CBAM [45]	50% Pruning	68.1	28.5	2.4
		75% Pruning	65.7	40.2	1.0
		87.5% Pruning	63.3	52.9	0.4
	Proposed SAFP with CA [46]	50% Pruning	67.7	28.5	2.4
		75% Pruning	65.7	40.2	1.0
		87.5% Pruning	63.4	52.9	0.4
Tiny YOLOv7 [8]	Baseline TinyYOLOv7		71.3	18.3	6.0
	Proposed SAFP with CBAM [45]	50% Pruning	70.9	27.2	2.3
		75% Pruning	68.4	38.4	1.0
		87.5% Pruning	65.9	50.7	0.4
	Proposed SAFP with CA [46]	50% Pruning	71.0	27.2	2.3
		75% Pruning	68.4	38.4	1.0
		87.5% Pruning	66.0	50.7	0.4

Figure 13 illustrates the detection results of the proposed SAFP-TinyYOLOv7 with CA (87.5% pruning). As shown in Tables 4 and 5, the accuracy of the proposed SAFP-TinyYOLOv7 with CA (87.5% pruning) has decreased compared to the baseline TinyYOLOv7. However, it is confirmed that most objects were successfully detected except occluded pigs and small vehicles.

Figure 13. Detection results of the proposed SAFP-TinyYOLOv7 with CA (87.5% pruning). Even with a pruning rate of 87.5% from TinyYOLOv7, most objects were successfully detected except occluded pigs and small vehicles.

5. Conclusions

Although accurate object detection is crucial for general applications, fast object detection is also important for many real-time applications, such as 24-hour surveillance and autonomous driving. In this study, a method was proposed to improve the execution speed of a one-stage YOLO detector using SAFP. In general, attention mechanisms have been widely used to improve the accuracy of deep learning models, whereas pruning techniques have been used to improve their execution speed. In the proposed SAFP method, a spatial attention mechanism was used to improve the execution speed of YOLO by evaluating the importance of each filter and pruning the unimportant filters. That is, the feature maps were first compared with before and after spatial attention, and then the unimportant filters of YOLO were pruned based on this comparison of feature maps. Because spatial attention emphasized the target locations in a training image, the importance of each YOLO filter was evaluated by checking the changes in the feature map values at the target locations.

The proposed method was applied to YOLOv4 and YOLOv7 to demonstrate its effectiveness. Using the vehicle dataset, the proposed method with both CBAM and CA improved the execution speed of YOLOv4 from 3.9FPS to 20.9FPS on a low-cost embedded board, TX-2, with acceptable accuracy and thus improved the integrated performance (= accuracy $\times$ speed) from 304.6 to 1517.3. Using the pig dataset, the proposed method with CBAM and CA improved the execution speed of YOLOv7 from 3.8FPS to 20.0FPS on a low-cost embedded board, TX-2, with acceptable accuracy and thus improved the integrated performance from 334.0 to 1624.0. For the pig dataset, there were performance improvements of 78.2%@19.2FPS (TinyYOLOv4) to 72.3%@52.9FPS and 81.9%@18.3FPS (TinyYOLOv7) to 75.8%@50.7FPS, which is confirmed to satisfy real-time object detection using tiny detection models. In other words, the proposed method demonstrated superior overall performance compared to the baseline in all aspects, including two datasets (pig and vehicle), two attention mechanisms (CBAM and CA), two detection models (YOLOv4 and YOLOv7), and two model sizes (medium and tiny). Furthermore, it significantly outperformed the conventional pruning techniques (magnitude- and randomness-based) in terms of accuracy. Also, note that the proposed method can also be applied to other model compression techniques [47], such as quantization.

Author Contributions: S.L. and Y.C. conceptualized and designed the experiments; H.A., S.S., H.B., J.R., and H.B. designed and implemented the detection system; Y.C. and D.P. validated the proposed method; H.A. and S.S. wrote the paper. All authors have read and agreed to the published version of the manuscript.

Funding: This work was the result of a local government–university cooperation-based regional innovation project (2021RIS-004), carried out with the support of the Korea Research Foundation with the funding of the Ministry of Education in 2021 and the National Research Foundation of Korea (NRF) grant with funding from the Korea government (MSIT) (2022R1F1A1062775).

Institutional Review Board Statement: Not applicable.

Informed Consent Statement: Not applicable.

Data Availability Statement: Not applicable.

Conflicts of Interest: The authors declare no conflict of interest.

References

1. Zhao, Z.; Zheng, P.; Xu, S.; Wu, X. Object Detection with Deep Learning: A Review. *IEEE Trans. Neural Netw. Learn. Syst.* **2019**, *11*, 3212–3232. [CrossRef] [PubMed]
2. Redmon, J.; Divvala, S.; Girshick, R.; Farhadi, A. You Only Look Once: Unified, Real-Time Object Detection. In Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR), Las Vegas, NV, USA, 27–30 June 2016.
3. Redmon, J.; Farhadi, A. YOLO9000: Better, Faster, Stronger. In Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR), Honolulu, HI, USA, 21–26 July 2017.
4. Redmon, J.; Farhadi, A. YOLOv3: An Incremental Improvement. *arXiv* **2018**, arXiv:1804.02767.

5. Bochkovskiy, A.; Wang, C.; Liao, H. YOLOv4: Optimal Speed and Accuracy of Object Detection. *arXiv* **2020**, arXiv:2004.10934.
6. Ultralytics/yolov5. Available online: https://github.com/ultralytics/yolov5 (accessed on 25 June 2020).
7. Ge, Z.; Liu, S.; Wang, F.; Li, Z.; Sun, J. YOLOx: Exceeding YOLO Series in 2021. *arXiv* **2021**, arXiv:2107.08430.
8. Wang, C.; Bochkovskiy, A.; Liao, H. YOLOv7: Trainable Bag-of-Freebies Sets New State-of-the-Art for Real-Time Object Detectors. *arXiv* **2022**, arXiv:2207.02696.
9. Shirke, A.; Saifuddin, A.; Luthra, A.; Li, J.; Williams, T.; Hu, X.; Kotnana, A.; Kocabalkanli, O.; Ahuja, N.; Green-Miller, A.; et al. Tracking Grow-Finish Pigs across Large Pens using Multiple Cameras. *arXiv* **2021**, arXiv:2111.10971.
10. Ahn, H.; Son, S.; Kim, H.; Lee, S.; Chung, Y.; Park, D. EnsemblePigDet: Ensemble Deep Learning for Accurate Pig Detection. *Appl. Sci.* **2021**, *11*, 5577. [CrossRef]
11. Shao, H.; Pu, J.; Mu, J. Pig-Posture Recognition based on Computer Vision: Dataset and Exploration. *Animals* **2021**, *11*, 1295. [CrossRef]
12. Bhujel, A.; Arulmozhi, E.; Moon, B.; Kim, H. Deep-Learning-based Automatix Monitoring of Pigs' Physico-Temporal Activities at Different Greenhouse Gas Concentrations. *Animals* **2021**, *11*, 3089. [CrossRef]
13. Luo, Y.; Zeng, Z.; Lu, H.; Lv, E. Posture Detection of Individual Pigs based on Lightweight Convolutional Neural Networks and Efficient Channel-Wise Attention. *Sensors* **2021**, *21*, 8369. [CrossRef]
14. Li, S.; Kang, X.; Feng, Y.; Liu, G. Detection Method for Individual Pig based on Improved YOLOv4 Convolutional Neural Network. In Proceedings of the 4th International Conference on Data Science and Information Technology, Shanghai, China, 23–25 July 2021.
15. Witte, J.; Gomez, J. Introducing a New Workflow for Pig Posture Classification based on a Combination of YOLO and EfficientNet. In Proceedings of the 55th Hawaii International Conference on System Sciences, Maui, HI, USA, 4–7 January 2022.
16. Ocepek, M.; Žnidar, A.; Lavrič, M.; Škorjanc, D.; Andersen, I. DigiPig: First Developments of an Automated Monitoring System for Body, Head, and Tail Detection in Intensive Pig Farming. *Agriculture* **2022**, *12*, 2. [CrossRef]
17. Ji, H.; Yu, J.; Lao, F.; Zhuang, Y.; Wen, Y.; Teng, G. Automatic Position Detection and Posture Recognition of Grouped Pigs based on Deep Learning. *Agriculture* **2022**, *12*, 1314. [CrossRef]
18. Kim, J.; Suh, Y.; Lee, J.; Chae, H.; Ahn, H.; Chung, Y.; Park, D. EmbeddedPigCount: Pig Counting with Video Object Detection and Tracking on an Embedded Board. *Sensors* **2022**, *22*, 2689. [CrossRef] [PubMed]
19. Bo, Z.; Atif, O.; Lee, J.; Park, D.; Chung, Y. GAN-based Video Denoising with Attention Mechanism for Field-Applicable Pig Detection System. *Sensors* **2022**, *22*, 3917. [CrossRef] [PubMed]
20. Son, S.; Ahn, H.; Baek, H.; Yu, S.; Suh, Y.; Lee, S.; Chung, Y.; Park, D. StaticPigDet: Accuracy Improvement of Static Camera-based Pig Monitoring using Background and Facility Information. *Sensors* **2022**, *22*, 8315. [CrossRef] [PubMed]
21. Ding, Q.; Chen, J.; Shen, M.; Liu, L. Activity Detection of Suckling Piglets based on Motion Area Analysis using Frame Differences in Combination with Convolution Neural Network. *Comput. Electron. Agric.* **2022**, *194*, 106741. [CrossRef]
22. Ding, Q.; Liu, L.; Lu, M.; Liu, K.; Chen, J.; Shen, M. Social Density Detection for Sucking Piglets based on Convolutional Neural Network Combined with Local Outlier Factor Algorithm. *Comput. Electron. Agric.* **2022**, *202*, 107423. [CrossRef]
23. Kim, T.; Kim, Y.; Kim, S.; Ko, J. Estimation of Number of Pigs Taking in Feed using Posture Filtration. *Sensors* **2023**, *23*, 238. [CrossRef]
24. Chen, J.; Zhou, J.; Liu, L.; Shu, C.; Shen, M.; Yao, W. Sow Farrowing Early Warning and Supervision for Embedded Board Implementations. *Sensors* **2023**, *23*, 727. [CrossRef]
25. Li, G.; Shi, G.; Jiao, J. YOLOv5-KCB: A New Method for Individual Pig Detection using Optimized K-Means, CA Attention Mechanism, and a Bi-Directional Feature Pyramid Network. *Sensors* **2023**, *23*, 5242. [CrossRef]
26. Lai, J.; Liang, Y.; Kuang, Y.; Xie, Z.; He, H.; Zhuo, Y.; Huang, Z.; Zhu, S.; Huang, Z. IO-YOLOv5: Improved Pig Detection under Various Illuminations and Heavy Occlusion. *Agriculture* **2023**, *13*, 1349. [CrossRef]
27. Lee, S.; Lee, W.; Park, J. A Study on Tracking Moving Objects: Pig Counting with YOLOv5 and StrongSORT. In Proceedings of the 29th International Workshop on Frontiers of Computer Vision (IW-FCV 2023), Yeosu, Republic of Korea, 20–22 February 2023.
28. Huang, E.; He, Z.; Mao, A.; Ceballos, M.; Parsons, T.; Liu, K. A Semi-Supervised Generative Adversarial Network for Amodal Instance Segmentation of Piglets in Farrowing Pens. *Comput. Electron. Agric.* **2023**, *209*, 107839. [CrossRef]
29. Odo, A.; Muns, R.; Boyle, L.; Kyriazakis, I. Video Analysis using Deep Learning for Automated Quantification of Ear Biting in Pigs. *IEEE Access* **2023**, *11*, 59744–59757. [CrossRef]
30. Zhang, Y.; Song, X.; Bai, B.; Xing, T.; Liu, C.; Gao, X.; Wang, Z.; Wen, Y.; Liao, H.; Zhang, G.; et al. 2nd Place Solution for Waymo Open Dataset Challenge—Real-Time 2D Object Detection. In Proceedings of the CVPRW, Virtual, 19–25 June 2021.
31. Nikolay, S. 3rd Place Waymo Real-Time 2D Object Detection: YOLOv5 Self-Ensemble. In Proceedings of the CVPRW, Virtual, 19–25 June 2021.
32. Jeon, H.; Tran, D.; Pham, L.; Nguyen, H.; Tran, T.; Jeon, J. Object Detection with Camera-Wise Training. In Proceedings of the CVPRW, Virtual, 19–25 June 2021.
33. Zhang, S.; Song, L.; Liu, S.; Ge, Z.; Li, Z.; Sun, J. Workshop on Autonomous Driving at CVPR 2021: Technical Report for Streaming Perception Challenge. In Proceedings of the CVPRW, Virtual, 19–25 June 2021.
34. Chen, Z.; Yang, D.; Xu, G.; Zhu, Q.; Wang, S.; Zhao, F. Solution to Streaming Perception Challenge for Detection-Only and Full-Stack Tracks. In Proceedings of the CVPRW, Virtual, 19–25 June 2021.

35. Gu, Y.; Wang, Q. Team CASIT_CV: Solution to Streaming Perception Challenge for Detection-Only Track. In Proceedings of the CVPRW, Virtual, 19–25 June 2021.
36. Yang, J.; Liu, S.; Li, Z.; Li, X.; Sun, J. Real-Time Object Detection for Streaming Perception. In Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR), New Orleans, LA, USA, 18–24 June 2022.
37. Farid, A.; Hussain, F.; Khan, K.; Shahzad, M.; Khan, U.; Mahmood, Z. A Fast and Accurate Real-Time Vehicle Detection Method using Deep Learning for Unconstrained Environments. *Appl. Sci.* **2023**, *13*, 3059. [CrossRef]
38. Yu, S.; Son, S.; Ahn, H.; Baek, H.; Nam, K.; Chung, Y.; Park, D. EnsembleVehicleDet: Detection of Faraway Vehicles with Real-Time Consideration. *Appl. Sci.* **2023**, *13*, 3939. [CrossRef]
39. Zhao, Q.; Ma, W.; Zheng, C.; Li, L. Exploration of Vehicle Target Detection Method based on Lightweight YOLOv5 Fusion Background Modeling. *Appl. Sci.* **2023**, *13*, 4088. [CrossRef]
40. Zhang, Y.; Sun, Y.; Wang, Z.; Jiang, Y. YOLOv7-RAR for Urban Vehicle Detection. *Sensors* **2023**, *23*, 1801. [CrossRef] [PubMed]
41. Ammar, A.; Koubaa, A.; Boulila, W.; Benjdira, B.; Alhabashi, Y. A Multi-Stage Deep-Learning-based Vehicle and License Plate Recognition System with Real-Time Edge Inference. *Sensors* **2023**, *23*, 2120. [CrossRef] [PubMed]
42. Lin, J.; Guo, J.; Shivanna, V.; Chang, S. Deep Learning Derived Object Detection and Tracking Technology based on Sensor Fusion of Millimeter-Wave Radar/Video and Its Application on Embedded Systems. *Sensors* **2023**, *23*, 2746. [CrossRef] [PubMed]
43. Sun, J.; Jiang, J.; Liu, Y. An Introductory Survey on Attention Mechanisms in Computer Vision Problems. In Proceedings of the 6th International Conference on Big Data and Information Analytics (BigDIA), Shenzhen, China, 4–6 December 2020.
44. Guo, M.; Xu, T.; Liu, J.; Liu, Z.; Jiang, P.; Mu, T.; Zhang, S.; Martin, R.; Cheng, M.; Hu, S. Attention Mechanisms in Computer Vision: A Survey. *arXiv* **2021**, arXiv:2111.07624. [CrossRef]
45. Woo, S.; Park, J.; Lee, J.; Kweon, I. CBAM: Convolutional Block Attention Module. In Proceedings of the European Conference on Computer Vision (ECCV), Munich, Germany, 8–14 September 2018.
46. Hou, Q.; Zhou, D.; Feng, J. Coordinate Attention for Efficient Mobile Network Design. In Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR), Nashville, TN, USA, 20–25 June 2021.
47. Lebedev, V.; Lempitsky, V. Speeding-up Convolutional Neural Networks: A Survey. *Bull. Pol. Acad. Sci. Tech. Sci.* **2018**, *66*, 799–810.
48. Blalock, D.; Ortiz, J.; Frankle, J.; Guttag, J. What is the State of Neural Network Pruning? *arXiv* **2020**, arXiv:2003.03033.
49. Vadera, S.; Ameen, S. Methods for Pruning Deep Neural Networks. *IEEE Access* **2022**, *10*, 63280–63300. [CrossRef]
50. He, Y.; Xiao, L. Structured Pruning for Deep Convolutional Neural Networks: A Survey. *arXiv* **2023**, arXiv:2303.00566.
51. Li, H.; Kadav, A.; Durdanovic, I.; Samet, H.; Graf, P. Pruning Filters for Efficient Convnets. *arXiv* **2016**, arXiv:1608.08710.
52. Gadhikar, H.; Mukherjee, S.; Burkholz, R. Why Random Pruning Is All We Need to Start Sparse. In Proceedings of the International Conference on Machine Learning, Honolulu, HI, USA, 23–29 July 2023.
53. Wang, X.; Yao, W.; Fu, H. A Convolutional Neural Network Pruning Method based on Attention Mechanism. In Proceedings of the 31st International Conference on Software Engineering and Knowledge Engineering, Lisbon, Portugal, 10–12 July 2019.
54. Yamamoto, K.; Maeno, K. PCAS: Pruning Channels with Attention Statistics for Deep Network Compression. *arXiv* **2019**, arXiv:1806.05382.
55. Zhang, S.; Wu, G.; Gu, J.; Han, J. Pruning Convolutional Neural Networks with an Attention Mechanism for Remote Sensing Image Classification. *Electronics* **2020**, *9*, 1209. [CrossRef]
56. Chen, R.; Qi, H.; Liang, Y.; Yang, M. Identification of Plant Leaf Diseases by Deep Learning based on Channel Attention and Channel Pruning. *Front. Plant Sci.* **2022**, *13*, 1023515. [CrossRef]
57. Chen, Y.; Shuai, M.; Lou, S.; An, Z.; Zhang, Y. FPAR: Filter Pruning via Attention and Rank Enhancement. In Proceedings of the 2022 IEEE International Conference on Multimedia and Expo (ICME), Taipei, Taiwan, 18–22 July 2022.
58. Riekert, M.; Klein, A.; Adrion, F.; Hoffmann, C.; Gallmann, E. Automatically Detecting Pig Position and Posture by 2D Camera Imaging and Deep Learning. *Comput. Electron. Agric.* **2020**, *174*, 105391. [CrossRef]
59. Argoverse-HD. Available online: https://www.kaggle.com/datasets/mtlics/argoversehd (accessed on 19 August 2023).
60. NVIDIA. NVIDIA Jetson TX2. Available online: http://www.nvidia.com/object/embedded-systems-dev-kitsmodules.html (accessed on 19 August 2023).
61. Selvaraju, R.; Cogswell, M.; Das, A.; Vedantam, R.; Parikh, D.; Batra, D. Grad-Cam: Visual Explanations from Deep Networks via Gradient-Based Localization. In Proceedings of the IEEE International Conference on Computer Vision (ICCV), Venice, Italy, 22–29 October 2017.