applied sciences

MDPI

Article

SSDLiteX: Enhancing SSDLite for Small Object Detection

Hyeong-Ju Kang

School of Computer Science and Engineering, Korea University of Technology and Education, Cheonan 31253, Republic of Korea; hjkang@koreatech.ac.kr; Tel.: +82-41-560-1420

Abstract: Object detection in many real applications requires the capability of detecting small objects in a system with limited resources. Convolutional neural networks (CNNs) show high performance in object detection, but they are not adequate to resource-limited environments. The combination of MobileNet V2 and SSDLite is one of the common choices in such environments, but it has a problem in detecting small objects. This paper analyzes the structure of SSDLite and proposes variations leading to small object detection improvement. The feature maps with the higher resolution are utilized more, and the base CNN is modified to have more layers in the high resolution. Experiments have been performed for the various configurations and the results show the proposed CNN, SSDLiteX, improves the detection accuracy AP of small objects by 1.5 percent points in the MS COCO data set.

Keywords: convolutional neural networks; object detection; single shot multibox detector

Citation: Kang, H.-J. SSDLiteX: Enhancing SSDLite for Small Object Detection. *Appl. Sci.* **2023**, *13*, 12001. https://doi.org/10.3390/app132112001

Academic Editor: Hui Yuan

Received: 10 August 2023
Revised: 31 October 2023
Accepted: 31 October 2023
Published: 3 November 2023

1. Introduction

Recently, convolutional neural networks (CNNs) show high performance in computer vision tasks including image classification [1–6], object detection [7–12], and image segmentation [13]. However, the enormous requirement of the storage and computation prohibits CNNs from being adopted in embedded systems that are power- or resource-limited. CNNs in such systems should have high performance with the small number of operations and coefficients.

Many computer vision applications including autonomous vehicles and surveillance systems are related to object detection. A CNN for object detection is usually constructed by adding some auxiliary layers to a base CNN for image classification. There are various kinds of object detection CNNs, and widely used for resource-limited systems is the combination of MobileNet V2 and SSDLite [12,14–16]. The network shows an appropriate detection capability with low complexity, but its performance is sometimes not so satisfactory, especially in detecting small objects.

In this paper, some optimizations will be applied to SSDLite. It has been known that SSDLite maintains the performance but reduces the complexity compared to the previous version, SSD (Single Shot Multibox Detector) [11]. However, the detailed changes have not been analyzed. This paper will compare SSDLite and SSD, discussing the pros and cons of the difference. The analysis shows that SSDLite does not utilize high resolution feature maps sufficiently. By utilizing such feature maps, SSDLite can be optimized to have better performance, especially in detecting small objects, with slightly increased complexity.

This paper is structured as follows. After discussing object detection neural networks in Section 2, the structures of SSD and SSDLite are compared in Section 3. Experimental results are presented in Section 4, and conclusions are described in Section 5.

2. Object Detection CNN

A CNN consists of various types of layers including a convolutional layer, a batch normalization and scaling layer, a pooling layer, an activation layer, and a fully connected layer, but most of the operations are concentrated in the convolutional layer.

2.1. Convolutional Layer

The main component of a CNN is the convolutional layer, which is based on a two-dimensional convolution operation. The layer assumes N input feature maps fi of height H and width W and produces M output features maps fo of height R and width C with the kernels of the size K by

$$fo(m,y,x) = \sum_{n=0}^{N-1} \sum_{i=0}^{K-1} \sum_{j=0}^{K-1} w(m,n,i,j) \times fi(n, S \times y + i, S \times x + j) + bias(m), \quad (1)$$

where S is the stride, $w()$ is the weights, and $bias()$ is the bias offset for each output feature map. The number of coefficients in a convolutional layer is $K \times K \times N \times M$, and that of multiplications is $R \times C \times K \times K \times N \times M$.

MobileNet replaces a normal convolutional layer with a combination of depthwise and pointwise convolutional layers. The depthwise convolutional layer calculates the output data by individually applying a $K \times K$ size kernel to each input channel as

$$fo(n,y,x) = \sum_{i=0}^{K-1} \sum_{j=0}^{K-1} w(n,i,j) \times fi(n, S \times y + i, S \times x + j) + bias(n). \quad (2)$$

The pointwise convolutional layer is a 1×1 convolution layer.

In Equations (1) and (2), S is the stride, and if S is larger than one, the height and width of the output feature map is less than those of the input feature map. A pooling layer performs the maximum or average operation instead of the convolution operation with a larger-than-one S value, reducing the spatial size too. An image classification CNN usually reduces the feature map spatial size five times with $S = 2$, making $1/32$ scale feature map data, and then processes the feature maps with a fully connected layer.

2.2. Object Detection

An object detection CNN can be constructed from a classification CNN in a two-stage architecture or a one-stage architecture. In the two-stage architecture, the regions of interest (RoIs) are proposed first and the detection is performed on the proposed regions. The representative networks are R-CNN [8], Fast R-CNN [9], and Faster R-CNN [10]. The one-stage networks do not have the region proposal stage. They have low detection capability than the two-stage networks, but can be processed faster. In the category are SSD series [12,17] and YOLO series [18–20]. To overcome the low performance of the one-stage networks, feature pyramid structures were proposed like Feature Pyramid Networks [21], RetinaNet [22], and EfficientDet [23].

Among the various object detection CNNs, the MobileNet and SSD combination is still widely used in the resource-limited environments because of its low complexity and fair detection capability. SSD is constructed by adding additional layers to a base image classification CNN like MobileNet. When the size of the input is 300, the SSD adds four stages consisting of a 1×1 convolutional layer and a 3×3 convolutional layer with the stride of two, reducing the horizontal and vertical size of the data to 1. The object detection is performed with extracted outputs of the stages, as shown in Figure 1. The size of the output of the base CNN is modified to be $1/16$ scale and $1/8$ scale data are extracted in the middle of the base CNN. The $1/8$ and $1/16$ scale data are also used for the object detection. SSDLite is basically constructed by decomposing the 3×3 convolutional layers in the auxiliary stages of SSD into a depthwise convolutional layer and a pointwise convolutional layer.

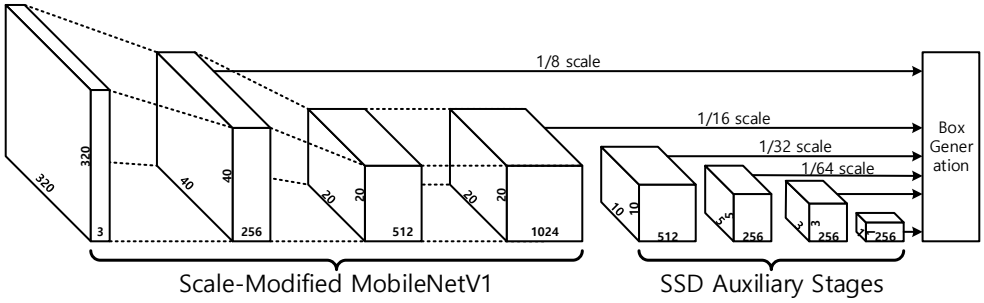

Figure 1. MobileNet V1 and SSD with input image 320 × 320.

3. Comparison of SSD and SSDLite

SSDLite reduced the amount of calculations and coefficients, but the effectiveness of the applied techniques has not been discussed in detail. This paper will try to find a better structure by analyzing the difference between SSD and SSDLite.

3.1. Prior Box

In SSD and SSDLite, the location and size of an object are predicted based on the given prior boxes for each location. SSD extracts the feature maps of six kinds of scales for object detection, from 1/8 to 1/256. The prior boxes for the first, fifth, and sixth scale feature maps have the sizes and aspect ratios described as Type 1 in Table 1. For the other scale feature maps, the sizes and aspect ratios are described as Type 2.

Table 1. Prior Box Types

Type	(Size, Aspect Ratio)
Type 1	(1, 1) ($\sqrt{2}$, 1) (1, 2) (1, 1/2)
Type 2	(1, 1) ($\sqrt{2}$, 1) (1, 2) (1, 1/2) (1, 3) (1, 1/3)
Type 3	(1/2, 1) (1, 2) (1, 1/2)

In the table, the size of 1 is the reference size, meaning 0.1, 0.2, 0.37, 0.54, 0.71, and 0.88 times the input image size for the VOC data set. For the MS COCO data set, the reference sizes are 0.07, 0.15, 0.33, 0.51, 0.69, and 0.87 [11]. The size $\sqrt{2}$ is the geometric mean of the reference size of the current scale and that of the next scale.

In contrast, SSDLite uses *Type 3* prior boxes for the first scale feature maps and *Type 2* prior boxes for the other feature maps [24]. The reference sizes are also changed to 0.2, 0.35, 0.5, 0.65, 0.8, 0.95 for the MS COCO data set. From the prior box configurations, it can be known that SSD uses more various prior boxes than those of SSDLite for the first extracted feature maps, whose scale is small and resolution is high. The prior boxes of SSD has the size 0.7 with the aspect ratios of 1, 2, 1/2 and the size 0.10 with the aspect ratios of 1. SSDLite, however, uses the prior boxes of size = 0.1 and the aspect ratio = 1 and also size = 0.2 and aspect ratio = 2 and 1/2. Instead, SSDLite uses more prior boxes in large-scale and low-resolution feature maps.

3.2. Scale of Extracted Feature Maps

In SSD, the base CNN is modified to have the scale of 1/16 at the last convolutional layer, and then the 1/8 scale and 1/16 scale feature maps are extracted to be used for the objected detection. In addition, the feature maps of four scales, 1/32, 1/64, 1/128, and 1/256, are generated by the auxiliary stages and they are also used for the object detection, as shown in Figure 1. In SSDLite, on the contrary, the base CNN scale is not modified and the 1/16 scale and 1/32 scale feature maps are extracted from the base CNN. The auxiliary stages generate 1/64, 1/128, 1/256, and 1/512 scale feature maps, as shown in Figure 2. Compared to SSD, SSDLite uses large-scale data rather than small-scale data, so the performance of extracting small-sized objects may deteriorate. In addition, the number

of output channels in the last layer is 256 for SSD and 128 for SSDLite, of which the effect will be examined via an experiment.

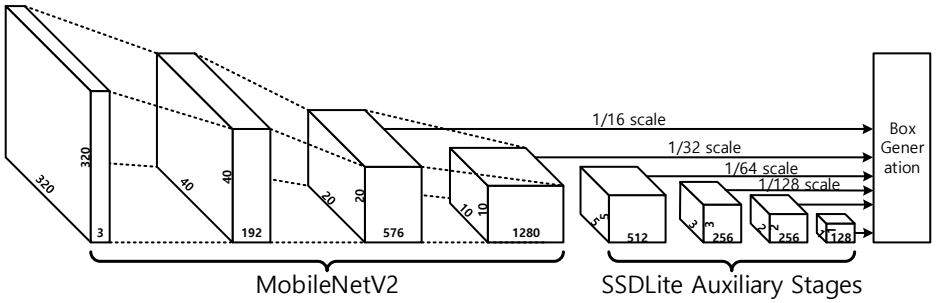

Figure 2. MobileNet V2 and SSDLite with input image 320 × 320.

3.3. Reduction Method of Spatial Size

In CNN, the spatial size of feature maps can be reduced using a strided convolutional layer or a pooling layer. Both SSD and SSDLite use a convolutional layer with the stride of two for this purpose. However, SSD uses a 3 × 3 convolutional layer with zero pad without the stride if reducing the spatial size in half is equivalent to reducing it by two (for example, from 5-by-5 to 3-by-3). SSDLite always uses a 3 × 3 convolutional layer with the stride of two.

In addition to the difference of SSD and SSDLite, it is also considerable to use a pooling layer. In the early CNNs, a pooling layer is commonly used, but a convolutional layer with a stride is prevalent in the modern CNNs. Some studies, however, told that a pooling layer has a advantage, especially in hardware implementation [25,26]. It would be needed to analyze which one is better in the view of performance and complexity.

3.4. Fully Connected Layer Substitution

An image classification CNN usually has a few fully connected layers at the end to collect the information and make a decision. SSD replaces the fully connected layers with convolutional layers. For example, the fc6 and fc7 fully connected layers of VGG16 are replaced with a 3 × 3 convolutional layer and a 1 × 1 convolutional layer. The fc6-replacement layer uses a dilation of six to have a larger receptive field. However, SSDLite does not include such layers. Using such layers can increase the complexity, but may also improve the detecting capability.

3.5. Bottleneck in Auxiliary Stage

An auxiliary stage in SSD consists of a 1 × 1 convolutional layer and a 3 × 3 convolutional layer. To decrease the complexity, the 1 × 1 convolutional layer reduces the number of channels, which is restored in the succeeding 3 × 3 convolutional layer as shown in Figure 3a. In the figure, each box represents a convolutional layer or a depthwise convolutional layer, and p0 and s2 mean the pad of zero and the stride of two, respectively. SSDLite transforms the 3 × 3 convolutional layer in the auxiliary stage into a 3 × 3 depthwise convolutional layer and a 1 × 1 pointwise convolutional layer. An auxiliary stage in SSDLite, therefore, consists of a 1 × 1 convolutional layer reducing the number of channels, a 3 × 3 depthwise convolutional layer with the reduced number of channels, and a 1 × 1 convolutional layer restoring the number of channels, as shown in Figure 3b. This structure, however, does not conform to the inverse bottleneck concept of MobileNet V2, where the channels are expanded before a 3 × 3 depthwise convolutional layer. This paper will examine an auxiliary stage consisting of a 3 × 3 depthwise convolutional layer and a 1 × 1 convolutional layer in Figure 3c, which maintain the number of channels with the slight increase in complexity.

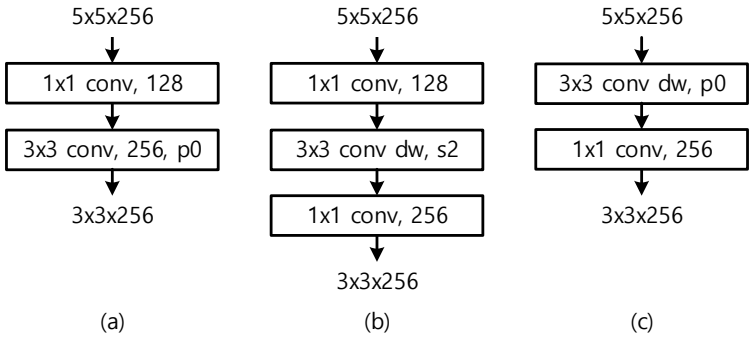

Figure 3. Auxiliary stage of (**a**) SSD, (**b**) SSDLite, and (**c**) SSDLiteX.

3.6. Layers in 1/8 Scale

MobileNet V2 consists of seven bottleneck sequences, each of which has one to four convolutional blocks. The third to sixth bottleneck sequences are described in Table 2. In the table, t, c, n, and s are the channel number expansion factor, the number of output channels, the number of convolutional block repetitions, and the stride.

The third bottleneck sequence receives 1/4 scale data, the feature map size is reduced at the first convolutional block of the sequence, and then the data are processed via two more convolutional blocks, as described in the left side of the first row in the table. Therefore, the 1/8 scale data are processed via two and a half convolutional blocks in the sequence. The 1/8 scale data are reduced to 1/16 scale data at the first convolutional block of the forth sequence and they are processed via three more convolutional blocks of the forth sequence and three convolutional blocks of the fifth sequence. The 1/16 scale data are processed via six and a half convolutional blocks in the forth and fifth sequences, with much more layers than those for the 1/8 scale data. The small number of layers in 1/8 scale can degrade the capability of detecting small objects. This paper will try the modification where the scale is changed from 1/8 to 1/16 at the first convolutional block of the fifth sequence rather than the forth sequence. In the modified version, 1/8 scale feature maps are processed via six and a half convolutional blocks, as shown in the right side of Table 2.

Table 2. MobileNet V2 Structure.

Input	Operator	t, c, n, s	Input	Operator	t, c, n, s
...			...		
$56^2 \times 24$	bottleneck	6, 32, 3, 2	$56^2 \times 24$	bottleneck	6, 32, 3, 2
$28^2 \times 32$	bottleneck	6, 64, 4, 2	$28^2 \times 32$	bottleneck	6, 64, 4, 1
$14^2 \times 64$	bottleneck	6, 96, 3, 1	$28^2 \times 64$	bottleneck	6, 96, 3, 2
$14^2 \times 96$	bottleneck	6, 160, 3, 2	$14^2 \times 96$	bottleneck	6, 160, 3, 2
...			...		

4. Experimental Results

In order to measure the effects of various changes mentioned in the previous section, experiments were performed applying various settings. The configurations are named as SSDLiteX and described in Table 3 with the related subsections. The configuration C0 has the same setting as SSDLite does. In C1, the prior boxes of the SSD type are applied as described in Section 3.1, and object detection is also performed from the 1/8 scale data as described in Section 3.2. C2 applies no-padding instead of strides to reduce the horizontal and vertical sizes in some additional layers. C3 uses a pooling instead of a convolutional layer with stride and C4 uses the fc6 layer. SSD uses the fc6 layer, a 3×3 convolutional layer with a dilation of six and the fc7 layer, a 1×1 convolutional layer. In this paper, the fc7 layer is not used, and the fc6 layer is separated into a depthwise convolutional layer with the dilation of one and a pointwise convolutional layer with 512 output channels. In C5, the auxiliary stage structure of Section 3.5 is used. Two kinds of base CNNs are used: the original MobileNet V2 and the modified version with more 1/8 layers described in

Section 3.6. Figure 4 shows the CNN structure where the configurations C1–C5 are used with the modified MobileNet V2.

Table 3. SSDLiteX Configurations.

Configuration	Explanation	Subsection
C0	SSDLite	-
C1	C0 + Prior box of SSD and using 1/8 scale data	Sections 3.1 and 3.2
C2	C1 + Reduction by no-padding and 256 channels on the last	Sections 3.2 and 3.3
C3	C2 + Reduction by pooling	Section 3.3
C4	C3 + Adding fc6 layer	Section 3.4
C5	C4 + Less 1 × 1 convolution	Section 3.5

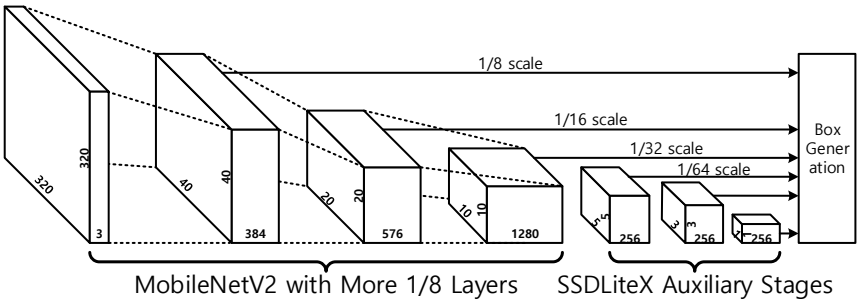

Figure 4. MobileNet V2 and SSDLiteX with input image 320 × 320.

4.1. Ablations Study of Detection Capability

Table 4 shows the ablation study results for the configurations described in Table 3 with the original MobileNet V2 base CNN and the modified version of more 1/8 scale layers. As the properties of Table 3 are added, the accuracy increases. The accuracy increases much when the configuration changes from C0 to C1 and from C3 to C4. Using the modified version of MobileNet V2 also leads to accuracy enhancement. The configuration C1 applies more prior boxes and more layers for the small-scale feature maps. The modification of MobileNet V2 is also related to the placement of more layers for small-scale feature maps. This implies the importance of utilizing small-scale feature maps.

Table 4. Ablation study of MS COCO AP (%).

Configuration	Original	More 1/8 Layers
C0	21.06	21.73
C1	21.11	22.39
C2	21.12	22.51
C3	21.18	22.36
C4	21.62	22.80
C5	21.52	22.86

Table 5 analyzes the detection capability in detail. The configuration C5 with the original base MobileNet V2 improves the detection capability of medium objects and the base MobileNet V2 with more 1/8 layers has similar effects. The C5 configuration and the structure improvement have a synergy effect, leading to much higher detection capability of small and medium objects, as shown in the last row of the table.

Table 5. Detection accuracy analysis.

Configuration	AP_{small}	AP_{medium}	AP_{large}
Original + C0	0.024	0.156	0.384
Original + C5	0.021	0.166	0.382
More 1/8 + C0	0.024	0.162	0.386
More 1/8 + C5	0.039	0.187	0.390

4.2. Trade-Off with Complexity

Table 6 compares the complexity of the configurations. The weight amount of C1 is much smaller than that of C0. This seems to be due to the decrease in the number of additional layers by utilizing more data from the base CNN. The configuration C2 has slightly more weights than C1 because of the channel number increase at the last auxiliary layer. The weight amount increase at C4 is due to the addition of the fc6 layer. The configurations C2 to C5 have more weights than C1 but still less weights than C0. All of the proposed configurations show a lesser weight amount and higher accuracy than SSDLite. If we compare two cases, SSDLite with the original MobileNet V2 and SSDLiteX C5 with the modified MobileNet V2, the accuracy is increased by 8.5%, with a weight amount decrease by 3%. Figure 5 illustrates the two ablation studies of the accuracy and the weight amount. It also compares the combination of MobileNet V3 and SSDLite [27], which has more weights and similar accuracy.

Table 6. Ablation study of complexity.

Configuration	Original		More 1/8 Layers	
	Weight	Computation	Weight	Computation
C0	4.28 M	0.79 G	4.28 M	1.02 G
C1	3.80 M	1.17 G	3.87 M	1.28 G
C2	3.85 M	1.17 G	3.92 M	1.28 G
C3	3.85 M	1.17 G	3.92 M	1.28 G
C4	4.01 M	1.19 G	4.08 M	1.30 G
C5	4.09 M	1.18 G	4.15 M	1.29 G

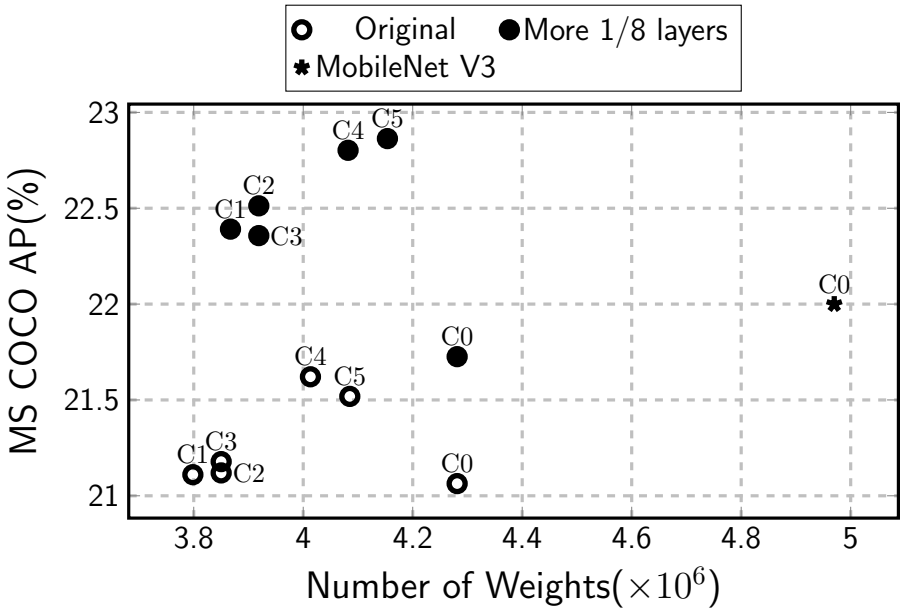

Figure 5. Number of weights and MS COCO AP for the configurations.

The amount of computation and the number of MAC operations show a slightly different aspect as shown at the third and fifth column. Basically, 1/8 scale layers have a

larger horizontal and vertical size than 1/16 scale layers, so they require more calculations. Therefore, C1, which extracts and uses 1/8 scale data from the base CNN, increases the amount of computations compared to C0. With the same reason, the modified MobileNet V2 with more 1/8 scale layers requires more computations. C4, in which the fc6 layer was additionally placed, also increases the amount of computations compared to the others. As the amount of computation increases, however, the performance of object detection also improves as shown in Table 4.

The proposed SSDLiteX configurations improve the detection capability with a lesser number of weights and an increased amount of computation. In an environment with a CNN accelerator, the convolutional layer processing is bound by the external DRAM communication rather than the computation capacity [28]. In such environments, the small weight amount of the proposed SSDLiteX will suggest a good trade-off between the complexity and the object detection performance.

5. Conclusions

This paper proposed a variation of SSDLite, SSDLiteX, to resolve its weaknesses. SSDLite comes from SSD, but the effects of the modifications have not been analyzed. This paper compares SSD and SSDLite and reveals SSDLite has a weakness in detecting small objects. CNN structures utilizing small-scale and high-resolution feature maps are proposed to improve the detecting ability of small objects with a reduced number of weights and a slightly increased number of operations. Experimental results show that the MS COCO AP on small objects increases by a 1.5 percent point. Small object detection is one of the weak points of compact CNNs where SSDLiteX can be a practical solution.

Funding: This work was supported by Basic Science Research Program, through the National Research Foundation of Korea (NRF), funded by the Ministry of Education (2021R1I1A3059617).

Institutional Review Board Statement: Not applicable.

Informed Consent Statement: Not applicable.

Data Availability Statement: The trained CNN models in the paper are available in https://github.com/HyeongjuKang/SSDLiteX accessed on 1 September 2023.

Conflicts of Interest: The authors declare no conflict of interest. The funders had no role in the design of the study; in the collection, analyses, or interpretation of the data; in the writing of the manuscript; or in the decision to publish the results.

Abbreviations

The following abbreviations are used in this manuscript:

AP	Average Precision
DRAM	Dynamic Random Access Memory
MAC	Multiplication and Accumulation

References

1. Krizhevsky, A.; Sutskever, I.; Hinton, G.E. ImageNet classification with deep convolutional neural networks. *Adv. Neural Inf. Process. Syst.* **2012**, *25*, 1097–1105.
2. Simonyan, K.; Zisserman, A. Very deep convolutional networks for large-scale image recognition. *arXiv* **2015**, arXiv:1409.1556.
3. Szegedy, C.; Liu, W.; Jia, Y.; Sermanet, P.; Reed, S.; Anguelov, D.; Erhan, D.; Vanhoucke, V.; Rabinovich, A. Going deeper with convolutions. In Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, Boston, MA, USA, 7–12 June 2015; pp. 1–9.
4. He, K.; Zhang, X.; Ren, S.; Sun, J. Deep residual learning for image recognition. In Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, Las Vegas, NV, USA, 27–30 June 2016; pp. 770–778.
5. Iandola, F.N.; Han, S.; Moskewicz, M.W.; Ashraf, K.; Dally, W.J.; Keutzer, K. SqueezeNet: AlexNet-level accuracy with 50× fewer parameters and <0.5 MB model size. *arXiv* **2016**, arXiv:1602.07360. [CrossRef]
6. Howard, A.G.; Zhu, M.; Chen, B.; Kalenichenki, D.; Wang, W.; Weyand, T.; Andreetto, M.; Adam, H. MobileNets: Efficient Convolutional Neural Networks for Mobile Vision Applications. *arXiv* **2017**, arXiv:1704.04861. [CrossRef]

7. Sermanet, P.; Eigen, D.; Zhang, X.; Mathieu, M.; Fergus, R.; LeCun, Y. OverFeat: Integrated recognition, localization and detection using convolutional networks. In Proceedings of the International Conference on Learning Representations, Banff, AB, Canada, 14–16 April 2014.
8. Girshick, R.; Donahue, J.; Darrell, T.; Malik, J. Rich feature hierarchies for accurate object detection and semantic segmentation. In Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, Columbus, OH, USA, 23–28 June 2014; pp. 580–587.
9. Girshick, R. Fast R-CNN. In Proceedings of the 2015 IEEE International Conference on Computer Vision, Santiago, Chile, 7–13 December 2015; pp. 1440–1448.
10. Ren, S.; He, K.; Girshick, R.; Sun, J. Faster R-CNN: Towards real-time object detection with region proposal networks. In Proceedings of the Advances in Neural Information Processing Systems, Montreal, Canada, 7–12 December 2015, pp. 91–99.
11. Liu, W.; Anguelov, D.; Erhan, D.; Szegedy, C.; Reed, S.; Fu, C.Y.; Berg, A.C. SSD: Single shot multibox detector. In Proceedings of the European Conference on Computer Vision (ECCV), Amsterdam, The Netherlands, 11–14 October 2016.
12. Sandler, M.; Howard, A.; Zhu, M.; Zhmoginov, A.; Chen, L.C. MobileNetV2: Inverted residuals and linear bottlenecks. In Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition, Salt Lake City, UT, USA, 18–23 June 2018; pp. 4510–4520.
13. He, K.; Gkioxair, G.; Dollar, P.; Girshick, R. Mask R-CNN. In Proceedings of IEEE International Conference on Computer Vision, Venice, Italy, 22–29 October 2017.
14. Zhang, J.; Jing, J.; Lu, P.; Song, S. Improved MobileNetV2-SSDLite for automatic fabric defect detection system based on cloud-edge computing. *Measurement* **2022**, *201*, 111665. [CrossRef]
15. Liu, T.; Zhu, Y.; Wu, K.; Yuan, F. Underwater Accompanying Robot Based on SSDLite Gesture Recognition. *Appl. Sci.* **2022**, *12*, 9131. [CrossRef]
16. Anggraini, N.; Ramadhani, S.H.; Wardhani, L.K.; Hakiem, N.; Shofi, I.M.; Rosyadi, M.T. Development of Face Mask Detection using SSDLite MobilenetV3 Small on Raspberry Pi 4. In Proceedings of the 2022 5th International Conference of Computer and Informatics Engineering (IC2IE), Jakarta, Indonesia, 13–14 September 2022; pp. 209–214. [CrossRef]
17. Liu, S.; Du, Z.; Tao, J.; Han, D.; Luo, T.; Zie, Y.; Chen, Y.; Chen, T. Cambricon: An instruction set architecture for neural networks. *ACM Sigarch Comput. Archit. News* **2016**, *44*, 393–405. [CrossRef]
18. 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, Las Vegas, NV, USA, 27–30 June 2016.
19. Redmon, J.; Farhadi, A. YOLO9000: Better, faster, stronger. In Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, Honolulu, HI, USA, 21–26 July 2017; pp. 6517–6525.
20. Redmon, J.; Farhadi, A. YOLOv3: An incremental improvement. *arXiv* **2018**, arXiv:1804.02767. [CrossRef]
21. Lin, T.Y.; Dollár, P.; Girshick, R.; He, K.; Hariharan, B.; Belongie, S. Feature pyramid networks for object detection. In Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, Honolulu, HI, USA, 21–26 July 2017; pp. 936–944.
22. Lin, T.Y.; Goyal, P.; Girshick, R.; He, K.; Dollar, P. Focal loss for dense object detection. In Proceedings of the IEEE International Conference on Computer Vision, Venice, Italy, 22–29 October 2017; pp. 2999–3007.
23. Tan, M.; Pang, R.; Le, Q.V. EfficientDet: Scalable and efficient object detection. In Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition, Seattle, WA, USA, 13–19 June 2020; pp. 10778–10787.
24. Huang, J.; Rathod, V.; Birodkar, V.; Myers, A.; Lu, Z.; Votel, R.; Chen, Y.; Chow, D. TensorFlow Object Detection API. Available online: https://github.com/tensorflow/models/blob/master/research/object_detection (accessed on 1 September 2023).
25. Ma, Y.; Zheng, T.; Cao, Y.; Vrudhula, S.; Seo, J.s. Algorithm-hardware co-design of single shot detector for fast object detection on FPGAs. In Proceedings of the 2018 IEEE/ACM International Conference on Computer-Aided Design (ICCAD), San Diego, CA, USA, 5–8 November 2018; pp. 1–8.
26. Kang, H.J. Real-Time Object Detection on 640 × 480 Image With VGG16+SSD. In Proceedings of the 2019 International Conference on Field-Programmable Technology (ICFPT), Tianjin, China, 9–13 December 2019; pp. 419–422.
27. Howard, A.; Sandler, M.; Chen, B.; Wang, W.; Chen, L.C.; Tan, M.; Chu, G.; Vasudevan, V.; Zhu, Y.; Pang, R.; et al. Searching for MobileNetV3. In Proceedings of the IEEE/CVF International Conference on Computer Vision, Seoul, Republic of Korea, 27 October–2 November 2019; pp. 1314–1324. [CrossRef]
28. Zhang, C.; Li, P.; Sun, G.; Guan, Y.; Xiao, B.; Cong, J. Optimizing FPGA-based accelerator design for deep convolutional neural networks. In Proceedings of the 2015 ACM/SIGDA International Symposium on Field-Programmable Gate Arrays, Monterey, CA, USA, 22–24 February 2015; pp. 161–170.