Sensor Fusion Operators for Multimodal 2D Object Detection

Morteza Mousa Pasandi, Tianran Liu, Yahya Massoud(✉), and Robert Laganière

University of Ottawa, Ottawa, ON, Canada
ymass049@uottawa.ca

Abstract. Autonomous driving requires effective capabilities to detect road objects in different environmental conditions. One promising solution to improve perception is to leverage multi-sensor fusion. This approach aims to combine various sensor streams in order to best integrate the information coming from the different sensors. Fusion operators are used to combine features from different modalities inside convolutional neural network architectures. In this study, we provide a framework for evaluating early fusion operators using different 2D object detection architectures. This comparative study includes element-wise addition and multiplication, feature concatenation, multi-modal factorized bilinear pooling, and bilaterally-guided fusion. We report quantitative results of the performance as well as an analysis of computational costs of these operators on different architectures.

Keywords: Convolutional neural networks · Sensor fusion · Fusion operators · Object detection · Autonomous driving

1 Introduction

Nowadays, intelligent vehicles include several sensors of different types in order to produce robust perception systems. In that context, multi-modal sensor fusion represents a way to enhance the accuracy and reliability of perception algorithms and methods. Deep convolutional networks are particularly well adapted for sensor fusion as the network architecture can be designed to accommodate multiple branches coming from different sensor streams that eventually merge or fuse together to produce a single output.

Different strategies can be used to implement sensor fusion in a neural network. Early fusion aims at combining the input modalities before performing the feature extraction step. Late fusion is usually done at the very end of the network by fusing the high-level feature maps before making a final classification or detection decision. Mid-level fusion is performed on intermediate deep representations and applies further processing of the fused feature maps in order to come up with a final decision. However, no matter which strategy is used to implement a sensor fusion architecture, one has to make a decision on how fusion will be performed at the junction point where the branches from the different modalities merge.

The choice of an appropriate fusion mechanism is therefore a critical factor toward building a robust sensor fusion architecture. The fusion mechanism will enable proper interaction between the different modalities and their corresponding feature maps. Several fusion operators have been proposed in the recent literature. Some are based on simple element-wise arithmetic operators such as summation and multiplication, or on order statistics such as median or min/max operations. Although computationally efficient, these element-wise approaches generally do not provide good perceptual robustness as they are unable to capture all interactions between the different modalities and their associated feature maps. In order to design more advanced fusion operators, we need to capture the intrinsic interrelation between features. This idea seems to constitute an interesting alternative to simple fusion operators. But before adding computational complexity to a network, one must assess the efficacy of such complex fusion operators.

[1] presented an idea of employing learnable fusion mechanisms for the task of 3D object detection, and proposed extending such contribution to other tasks while exploring other variations of fusion modules. The objective of this paper is to experimentally compare the performance of different fusion operators in the context of multi-modal road user detection using deep neural networks. To do so, we selected an object detection architecture and use LiDAR and camera fusion as an illustrative application of sensor fusion. The performance of this architecture is then tested for different choices of fusion mechanisms. Our objective is to demonstrate the impact of a fusion operator on the precision of a detection network.

Section 2 presents some recent works in sensor fusion. Section 3 describes the detection network used in this experimental study. Section 4 introduces the different fusion operators. Section 5 present the experimental results. Section 6 is a conclusion.

2 Related Work

MV3D [2] presents a sensor fusion framework that leverages both camera and LiDAR sensors. The proposed architecture uses cylindrical frontal view projection of the LiDAR sensor to further enhance the performance of its frontal view branch. Each of the modalities is fed into a feature extraction network, then a 3D region-proposal generation step is applied. After that, region proposals are fed into a deep-fusion mechanism to enable interaction between all three modalities. Finally, the output of the deep-fusion mechanism is fed to classification and regression branches. It is worth noting that the deep-fusion mechanism comprises of a branching version of the element-wise mean fusion.

FrustumPointNets [3] uses RGB-D data for the tasks of object detection and localization. The framework applies a robust 2D detector to the RGB image to extract accurate 2D region-proposals, then aims to extract and segments 3D frustums from each region proposal by using the depth information. PointFusion [4] is a two-stage framework that uses inputs from both camera and LiDAR. PointFusion applies a SOTA 2D detector on the image, then fuses the extracted feature maps with the corresponding 3D points extracted from the raw LIDAR point cloud to mitigate any loss of information, not having to encode the whole raw point cloud.

AVOD [5] incorporates a two-stage architecture to operate on both camera and LiDAR sensors, while the intermediate fusion mechanism is an element-wise mean operation, followed by a series of fully-connected layers for each modality. ContFuse [6] is an end-to-end framework that uses the idea of continuous convolution to project frontal view feature maps extracted from a frontal view stream into BEV, then fuses these projected feature maps with original BEV feature maps extracted from a BEV feature extractor. The fusion mechanism is a simple element-wise summation. [1] proposes a two-stream multi-modal multi-view sensor fusion architecture that operates on LiDAR BEV representation, LiDAR frontal view features (height, depth, and intensity), and RGB images. Early learnable fusion for the frontal view stream is employed to boost performance. Moreover, learnable mid-level fusion is applied with multi-task learning to bypass the limited nature of fixed fusion operations (e.g. addition or multiplication). MMF [7] employed a strategy of using two sensors (LiDAR and camera) to learn four tasks and showed that a target task (3D object detection) can benefit from multi-task learning. MMF framework aimed at using RGB image as well as projection-based representations for LiDAR point clouds to simultaneously perform the following tasks: 2D and 3D object detection, online mapping, and depth completion.

3 Camera-LiDAR 2D Object Detector

The objective of this paper is to study the impact of various fusion operators on the performance of a sensor fusion framework. Therefore, we chose a multi-sensor fusion architecture that uses both LiDAR and camera inputs to perform 2D road object detection. We performed our experiments without introducing changes to the original network. Instead, we evaluated the performance by replacing the fusion operator that combines both input modalities. In Fig. 1, we show the architecture of the sensor fusion network. The objective behind our model selection is to combine both good performance and low complexity.

The camera stream processes a colored input image to detect 2D objects. The downside of using camera only is the impact of, among others, adverse lighting and weather conditions, noise, reflections, and limited depth cues. These challenging situations will negatively affect the performance of an image-based object detector. The LiDAR stream aims at complementing the visual information with 3D information provided in the form of 3D point clouds. In order to have compatible representations from both modalities, we opted for a frontal view representation of the LiDAR data by projecting intensity, depth, and height maps on the camera view.

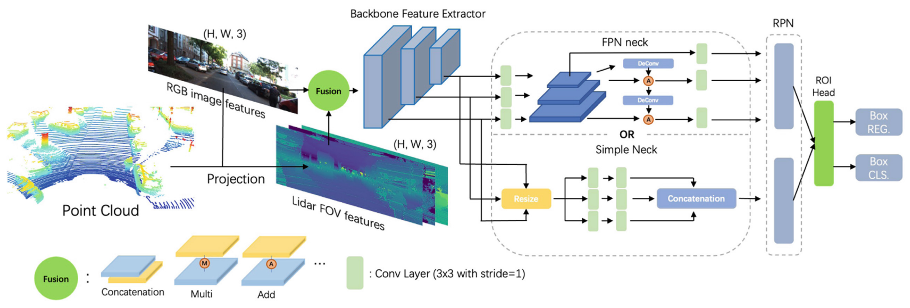

Fig. 1. Overall architecture of our sensor fusion framework. This network is used to assess and analyze the effectiveness of different fusion operators in a 2D detection task.

Both camera and LiDAR frontal views are then combined using a fusion operator forming the sensor fusion head of this early fusion network. The outcome of the fusion operation is then fed into the 2D object detection to perform inference. The architecture of the 2D detection network is inspired by popular detection frameworks such as Faster-RCNN [8] and Cascade-RCNN [9].

We use ResNet-30 [10] and Swin Transformer Tiny [11] as our feature extraction backbone networks. ResNet-30 has an initial layer containing 32 filters and a 7×7 kernel size, followed by three stages containing $\{4, 4, 6\}$ basic blocks and $\{32, 64, 128\}$ filters, respectively. Swin Transformer Tiny has a window size of 7 with four stages containing $\{96, 192, 384, 768\}$ embedding sizes, respectively. To further process the output of the backbone network, we employ one of two types of necks, namely Simple neck or a Feature Pyramid Network (FPN) neck [12]. Simple neck has three stages, each stage containing 96 Filters, resulting in 288 output channels after concatenation of all stages.

Lastly, Faster-RCNN [8] and Cascade-RCNN [9] are used as detection heads. Faster-RCNN employs a region proposal network (RPN) and a region-of-interest (ROI) detection head for generating object proposals. It extracts a fixed-length feature vector from each region proposal, based on which it assigns a classification score and a predicted bounding box. The outcome of both the RPN and the feature maps extracted from the neck are then fed into the ROI detection head to infer 2D objects. Cascade-RCNN is based on a sequential arrangement of three identical ROI heads, each feeding the next cascaded Faster RCNN detector. Figure 2 illustrates how both early- and mid-level fusion are applied in the proposed framework. Early fusion applies the fusion operation on the inputs, then feeds the fused inputs to the backbone network, while mid-level fusion feeds inputs to two parallel backbone networks first, then applies the fusion operation on the two output feature maps.

4 Sensor Fusion Operators

Fusion operators can be classified into two categories: (1) fixed and (2) learnable operators. The first category includes basic and computationally efficient

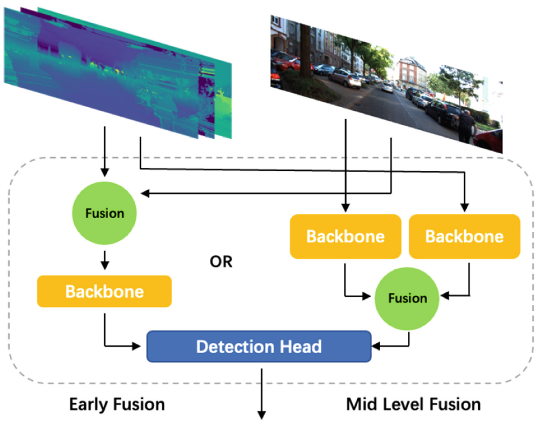

Fig. 2. Left: early fusion fuse inputs then feeds the outcome to the backbone network. Right: mid-level fusion feeds inputs to two backbone networks first, then fuses the output feature maps.

fusion methods such as element-wise summation, mean, max, or the concatenation operation. However, these methods are generally not robust enough to adequately capture the interactions between related features of different modalities. At the price of higher computational cost and training complexity, more sophisticated fusion schemes have been introduced. [13] proposed bilinear-pooling as a robust fusion mechanism. Despite the expressiveness of bilinear pooling, its use is impractical from a computational perspective. The authors present more efficient way to implement factorized versions of the robust bilinear-pooling fusion mechanism in deep learning architectures leading to multi-modal factorized bilinear pooling (MFB). MFB has been used in [14] for detecting activity in videos. MFB's implementation was further studied and optimized in [1] in order to be adapted for the task of 3D object detection. As a mid-level fusion mechanism, [1] showed a significant performance increase compared to element-wise fusion operators. Bilateral Guided Aggregation (BGA) is another fusion mechanism that has been applied in [15] to preserve Semantic features in semantic segmentation tasks. BGA aims at fusing multi-modal features with a cross-modal attention mechanism embedded into a neural sub-network. BGA has also been used in [16] to merge frontal view features with bird's eye view features in a camera-lidar fusion network. In our work, we employ and analyze both multi-modal factorized bilinear pooling (MFB) and bilateral guided fusion (BGF) as potential learnable fusion mechanisms. Both fusion mechanisms are illustrated in Fig. 3.

Multi-modal Factorized Bilinear Pooling. First, both colored image ($\mathcal{I}_{rgb}$) and LiDAR features ($\mathcal{I}_{lidar}$) are projected into a high-dimensional space using a convolutional layer that expands the number of channels from 3 to 6. The aforementioned step produces two expanded feature maps ($\mathcal{F}_a$) and ($\mathcal{F}_b$) as described

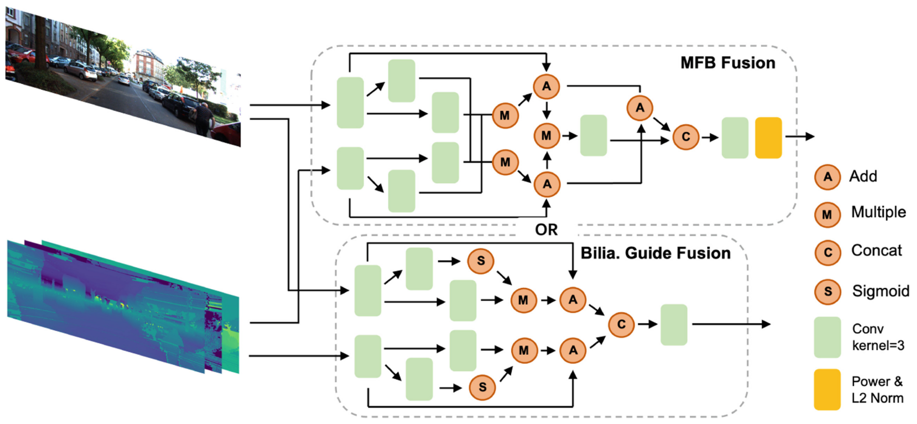

Fig. 3. The structure of two learnable fusion mechanisms: multi-modal factorized bilinear pooling (MFB) and bilateral guided fusion (BGF).

in Eq. 1. Then, each feature map is fed into two convolutional layers while preserving the size of the channel dimension (6 filters), producing four new feature maps: $\{\mathcal{F}_{a1}, \mathcal{F}_{a2}, \mathcal{F}_{b1}, \mathcal{F}_{b2}\}$. All four feature maps are then mixed together based on Eqs. 2 and 3. Then, two operations (addition and multiplication) are applied in parallel to $\mathcal{F}_{out1}$ and $\mathcal{F}_{out2}$. The multiplication outcome is then fed into a convolutional layer with 6 filters. Finally, the outcome of both convolution and addition operations is concatenated and fed into a final convolutional layer which reduces the channel dimension from 12 to 3 channels, followed by power and L_2 normalization. The final process is described in Eq. 4.

$$\begin{aligned}\mathcal{F}_a &= \text{Conv}_{3\longrightarrow 6}(\mathcal{I}_{rgb})\\ \mathcal{F}_b &= \text{Conv}_{3\longrightarrow 6}(\mathcal{I}_{lidar})\end{aligned} \tag{1}$$

$$\mathcal{F}_{out1} = \mathcal{F}_{b1} * \mathcal{F}_{a2} + \mathcal{F}_a \tag{2}$$

$$\mathcal{F}_{out2} = \mathcal{F}_{b2} * \mathcal{F}_{a1} + \mathcal{F}_b \tag{3}$$

$$\begin{aligned}\mathcal{F}_{mul} &= \mathcal{F}_{out1} * \mathcal{F}_{out2}\\ \mathcal{F}_{add} &= \mathcal{F}_{out1} + \mathcal{F}_{out2}\\ \mathcal{F}_{pre} &= \text{Concat}(\text{Conv}_{6\longrightarrow 6}(\mathcal{F}_{mul}), \mathcal{F}_{add})\\ \mathcal{F}_{final} &= \text{Norm}(\text{Conv}_{12\longrightarrow 3}(\mathcal{F}_{pre}))\end{aligned} \tag{4}$$

Bilateral Guided Fusion. Similar to steps 1 and 2 in MFB, both input modalities are first projected into a high-dimensional space using convolutions (Eq. 1), then fed into convolutions to produce four feature maps: $\{\mathcal{F}_{a1}, \mathcal{F}_{a2}, \mathcal{F}_{b1}, \mathcal{F}_{b2}\}$. Finally, all four feature maps are mixed based on Eq. 5 with a combination of sigmoid, addition, and multiplication to produce a final fused outcome with 3 output channels.

$$\begin{aligned}\mathcal{F}_{out1} &= (\sigma(\mathcal{F}_{a1}) * \mathcal{F}_{a2}) + \mathcal{F}_a \\ \mathcal{F}_{out2} &= (\sigma(\mathcal{F}_{b1}) * \mathcal{F}_{b2}) + \mathcal{F}_b \\ \mathcal{F}_{final} &= \mathrm{Conv}_{12\rightarrow 3}(\mathrm{Concat}(\mathcal{F}_{out1}, \mathcal{F}_{out2}))\end{aligned} \tag{5}$$

5 Experimental Results

In this section, we present the results of employing different operators in an early fusion mechanisms for various 2D detection architectures. Moreover, further experimentation is performed to discuss mid-level fusion and to assess the effectiveness and efficiency of each fusion operator.

5.1 Experimental Setting

To assess the effectiveness of early sensor fusion mechanisms, we train and evaluate four different models on the KITTI 2D object detection benchmark [17]. We rely on KITTI's validation set to showcase our experimental results and ablation study by following the same split suggested in [18]. Faster-RCNN variants are trained using stochastic gradient descent, while Cascade-RCNN with Swin Transformer is trained using the AdamW [19] optimizer. All architectures are trained on the 2D detection task for "Car" and "Pedestrian" classes. Models are trained for 30 epochs, while applying horizontal flipping as a data augmentation strategy. Detection accuracy is reported for three different categories: easy, moderate, and hard. We use average-precision (AP%) to report the detection accuracy for each class in all three difficulty levels.

5.2 Evaluation of Early Sensor Fusion

In Table 1, we compare and contrast the results of five early fusion operators and their impact on the performance of four 2D object detectors. The baseline detection accuracy for each architecture is called "No Fusion", where the detector operates on the colored image only without LiDAR information, hence, no early fusion mechanism is being applied. Other fusion operators are: (1) element-wise addition "Add", (2) feature concatenation "Concat", (3) element-wise multiplication "Multi", (4) bilaterally-guided fusion "BGF", and (5) multi-modal factorized bilinear pooling "MFB".

For the first set of experiments which use the R30 Simple Neck Faster-RCNN detector, MFB consistently outperforms all other early fusion operators in all difficulty levels for both classes of interest. Compared to the second-best fusion operator, MFB scores an increase of (+4.37%) and (+3.76%) in the moderate category for both car and pedestrian classes, respectively. In the second set of experiments, we use the R30 FPN Faster-RCNN detector. Feature concatenation yields better results for the pedestrian class, while BGF performs better for the

Table 1. Comaparing 2D detection accuracy of different early stage fusion operators on Faster-RCNN and Swin Transformer

Model components	Fusion operation	Car ($AP_{70}\%$)			Pedestrian ($AP_{50}\%$)		
		Easy	Moderate	Hard	Easy	Moderate	Hard
R30 Simple Neck Faster-RCNN	No fusion	78.04	56.03	47.25	29.04	24.55	20.93
	Add	77.45	59.18	48.34	30.73	23.25	19.52
	Concat	83.48	65.74	56.05	36.63	27.60	23.97
	Multi	77.09	56.39	47.51	28.98	22.18	18.71
	BGF	84.82	66.73	58.44	34.09	28.36	24.76
	MFB	**85.52**	**71.1**	**59.2**	**37.2**	**32.12**	**29.98**
R30 FPN Faster-RCNN	No fusion	89.92	81.26	72.56	63.18	53.12	46.18
	Add	92.57	81.88	71.7	54.27	45.32	38.28
	Concat	93.48	84.21	74.42	**65.59**	**55.41**	**47.94**
	Multi	87.39	70.25	62.6	58.18	48.91	42.2
	BGF	**93.79**	**84.89**	**76.15**	64.57	54.99	47.45
	MFB	92.93	82.37	72.23	64.12	53.15	47.11
R30 FPN Cascade-RCNN	No fusion	92.49	84.38	73.75	62.33	52.89	44.9
	Add	93.82	85.42	75.33	63.62	52.61	44.31
	Concat	94.49	87.5	77.74	70.21	58.98	50.41
	Multi	90.9	77.23	67.53	64.85	54.44	46.38
	BGF	**94.9**	**88.4**	**78.38**	**71.25**	**61.11**	**52.48**
	MFB	93.21	85.34	76.93	66.88	56.13	48.53
Swin FPN CascadeRCNN	No fusion	93.85	81.66	71.56	58.98	50.02	43.12
	Add	90.35	77.76	67.78	59.50	50.70	43.29
	Concat	94.87	86.36	77.76	**71.3**	**60.83**	**52.05**
	Multi	91.49	76.05	66.00	60.73	50.62	43.50
	BGF	95.38	85.81	77.68	67.66	56.78	47.9
	MFB	**96.57**	**88.12**	**78.13**	67.20	58.10	50.11

car class. That being said, both feature concatenation and BGF are on par. If we compare their performance based on the moderate difficulty, we find that BGF is superior to concatenation by (+0.68%) for the car class, but inferior to concatenation by (−0.42%) for the pedestrian class. When incorporated with R30-FPN Cascade-RCNN, BGF outperforms other fusion methods for both car and pedestrian classes in all difficulty levels.

Finally, we evaluate the Swin FPN Cascade-RCNN detector and observe MFB fusion outperforming other early fusion methods in the car class. As for the pedestrian class, it follows a similar pattern to R30-FPN Faster-RCNN, where feature concatenation outperforms all other fusion methods.

Based on the aforementioned results, we conclude the superiority of the learnable fusion mechanisms, both MFB and BGF, over other less-sophisticated fusion operators when detecting the car class. Using feature concatenation as the fusion operator yields overall good results, and even superior to learnable fusion in two cases only for the pedestrian class.

5.3 Evaluation of Mid-Level Sensor Fusion

In order to assess the importance of mid-level fusion, we train two ResNet30 in parallel, then fuse the extracted features from each network. Table 2 shows and contrasts the results between early- and mid-level fusion on the same model architecture, an R30 FPN Faster-RCNN 2D object detector. Sophisticated learnable fusion mechanisms prove ineffective in the case of mid-level fusion for both car and pedestrian class, while more simple element-wise addition or feature concatenation are scoring higher detection accuracy. Moreover, mid-level fusion with element-wise addition is superior to early fusion in the case of pedestrian detection, scoring an increase of (+2.25%), (+3.97%), and (+3.46%) in easy, moderate, and hard categories, respectively.

Table 2. Difference between early fusion and mid-level fusion. The reported results compare two different backbone feature extractors.

Model components	Fusion operation	Car (AP_{70}%)			Pedestrian (AP_{50}%)		
		Easy	Moderate	Hard	Easy	Moderate	Hard
R30 FPN Faster-RCNN Early Fusion	No Fusion	89.92	81.26	72.56	63.18	53.12	46.18
	Add	92.57	81.88	71.7	54.27	45.32	38.28
	Concat	93.48	84.21	74.42	**65.59**	**55.41**	**47.94**
	Multi	87.39	70.25	62.6	58.18	48.91	42.2
	BGF	**93.79**	**84.89**	**76.15**	64.57	54.99	47.45
	MFB	92.93	82.37	72.23	64.12	53.15	47.11
R30 FPN Faster-RCNN Mid-level Fusion	No Fusion	90.44	79.97	70.12	53.4	46.13	39.33
	Add	93.14	**83.80**	**73.88**	**67.84**	**59.38**	**51.40**
	Concat	**93.97**	83.77	73.65	67.12	58.18	50.53
	Multi	93.70	82.87	72.64	63.64	54.25	46.46
	BGF	93.41	82.50	72.44	60.75	51.91	44.72
	MFB	93.75	82.87	72.08	59.8	50.54	43.00

5.4 Complexity Analysis

Computation Overhead. In Table 3, we compare the computational complexity of five different object detectors without applying any fusion mechanisms. In terms of the number of learnable parameters, R30-FPN Faster-RCNN has the least capacity with 18.4M parameters, compared to Swin Tiny FPN Cascade-RCNN which has 72.5M parameters. R30 Simple Neck Faster-RCNN

Table 3. Comparison of the complexity of each model and their required floating point operations per second (FLOPS)

Model	Computation metrics	
	Parameters (Million)	FLOPS (Giga)
R30 Simple Neck Faster-RCNN	18.88	**79.44**
R30 FPN Faster-RCNN	**18.4**	191.71
2 Stream R30 FPN Faster-RCNN	20.87	217.78
R30 FPN Cascade-RCNN	46.2	219.51
Swin Tiny FPN Cascade-RCNN	72.5	336.13

has 79.44 GFLOPS, which is four times more efficient compared to Swin Tiny FPN Cascade-RCNN which has 336.13 giga-FLOPS.

An interesting aspect that should be emphasized is the relation between the number of parameters and GFLOPS for different fusion mechanisms. In Fig. 4, we compare concatenation to two learnable fusion mechanisms (MFB and BGF) by applying them on three different architectures: (1) R30 Simple Neck Faster-RCNN, (2) R30 FPN Faster-RCNN, and (3) Swin FPN Cascade-RCNN.

Normally, concatenation by itself does not add up more learnable parameters to the architecture, except for Swin FPN Cascade-RCNN. In contrast, both MFB and BGF add more learnable parameters to the model. That being said, the impact of concatenation on increasing GFLOPS is comparable, and in some cases even higher than both MFB and BGF.

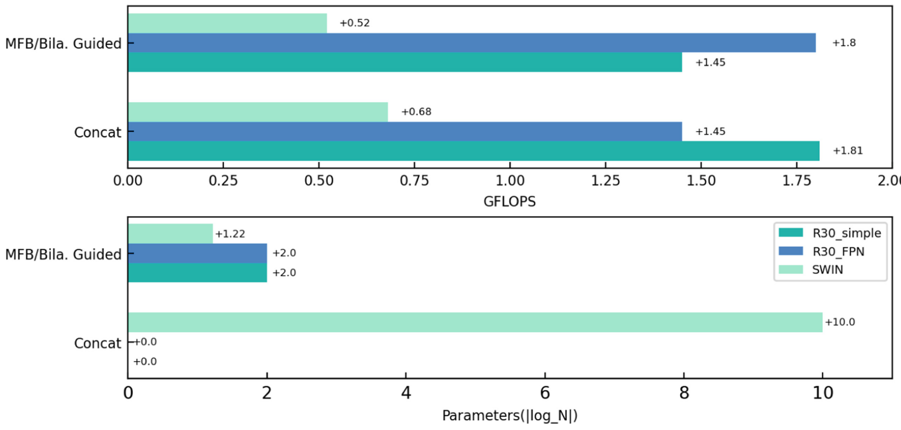

Fig. 4. Showing the relation between number of parameters (log-scaled) and the amount of GFLOPS when using different fusion operators. Even though learnable fusion mechanisms add up more learnable parameters to the architecture, their GFLOPS is still on par, and in some cases more efficient compared to feature concatenation.

Kernel Size of MFB. As denoted in [1], MFB was used with a kernel of size (1 × 1). In this work, we aimed to assess this design choice by comparing the usage of kernel size (1 × 1) and (3 × 3). We perform such experiments on R30 FPN Faster-RCNN architecture. In Table 4, we observe that (3×3) usually yields superior detection accuracy for both car and pedestrian classes, with only one exception in the moderate case for the pedestrian class.

Table 4. Comparing different kernel size options in MFB Fusion module. K_1: 1×1 kernel size, K_3: 3×3 kernel size. *Mid-level* flag denotes whether the setting was used in mid- or early-fusion. All four experiments were performed on Resnet 30 FPN Faster-RCNN architecture.

Exp	K_1	K_3	Mid-level	Car $AP_{70\%}$			Pedestrian $AP_{50\%}$		
				Easy	Moderate	Hard	Easy	Moderate	Hard
(a)	√		√	92.93	82.01	71.42	58.8	50.43	42.11
(b)		√	√	**93.75**	**82.87**	72.08	59.8	50.54	43.00
(c)	√			92.71	81.48	71.82	63.66	**54.12**	46.28
(d)		√		92.93	82.37	**72.23**	**64.12**	53.15	**47.11**

6 Conclusion

In this work, we provide a comprehensive analysis of both effectiveness and efficiency of various fusion operators when employed in different architectures to perform the task of multi-modal 2D object detection. In our experiments, we show that multi-modal representations leverage early fusion to provide more interaction between input features. Moreover, early fusion proves more effective compared to mid-level fusion of high-level feature maps. Multi-modal factorized bilinear pooling (MFB) showed performance improvements when used with Transformer-based 2D object detector. Also, we observe that element-wise multiplication negatively impacts the performance as it causes the gradients to vary dramatically in the upstream network. In contrary to early fusion experiments, mid-level fusion yielded higher performance when a simple element-wise addition fusion was applied.

References

1. Massoud, Y.: Sensor fusion for 3D object detection for autonomous vehicles. Master's thesis, Université d'Ottawa/University of Ottawa (2021)
2. Chen, X., Ma, H., Wan, J., Li, B., Xia, T.: Multi-view 3D object detection network for autonomous driving. In: Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, pp. 1907–1915 (2017)
3. Qi, C.R., Liu, W., Wu, C., Su, H., Guibas, L.J.: Frustum pointnets for 3D object detection from RGB-D data. In Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, pp. 918–927 (2018)

4. Xu, D., Anguelov, D., Jain, A.: Pointfusion: deep sensor fusion for 3D bounding box estimation. In: Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, pp. 244–253 (2018)
5. Ku, J., Mozifian, M., Lee, J., Harakeh, A., Waslander, S.L.: Joint 3D proposal generation and object detection from view aggregation. In: 2018 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS), pp. 1–8. IEEE (2018)
6. Liang, M., Yang, B., Wang, S., Urtasun, R.: Deep continuous fusion for multi-sensor 3D object detection. In: Ferrari, V., Hebert, M., Sminchisescu, C., Weiss, Y. (eds.) ECCV 2018. LNCS, vol. 11220, pp. 663–678. Springer, Cham (2018). https://doi.org/10.1007/978-3-030-01270-0_39
7. Liang, M., Yang, B., Chen, Y., Hu, R., Urtasun, R.: Multi-task multi-sensor fusion for 3D object detection. In: Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition, pp. 7345–7353 (2019)
8. Chen, X., Gupta, A.: An implementation of faster RCNN with study for region sampling. arXiv preprint arXiv:1702.02138 (2017)
9. Cai, Z., Vasconcelos, N.: Cascade R-CNN: delving into high quality object detection. In: Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, pp. 6154–6162 (2018)
10. 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, pp. 770–778 (2016)
11. Liu, Z., et al.: Swin transformer: hierarchical vision transformer using shifted windows. In: Proceedings of the IEEE/CVF International Conference on Computer Vision, pp. 10012–10022 (2021)
12. 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, pp. 2117–2125 (2017)
13. Yu, Z., Yu, J., Fan, J., Tao, D.: Multi-modal factorized bilinear pooling with co-attention learning for visual question answering. In: ICCV, pp. 1839–1848 (2017)
14. Rahman, Md.A., Laganière, R.: Mid-level fusion for end-to-end temporal activity detection in untrimmed video. In: BMVC (2020)
15. Changqian, Yu., Gao, C., Wang, J., Gang, Yu., Shen, C., Sang, N.: Bisenet v2: bilateral network with guided aggregation for real-time semantic segmentation. Int. J. Comput. Vision **129**(11), 3051–3068 (2021)
16. Deng, J., Zhou, W., Zhang, Y., Li, H.: From multi-view to hollow-3D: hallucinated hollow-3D R-CNN for 3D object detection. IEEE Trans. Circuits Syst. Video Technol. **31**(12), 4722–4734 (2021)
17. Geiger, A., Lenz, P., Urtasun, R.: Are we ready for autonomous driving? The KITTI vision benchmark suite. In: Conference on Computer Vision and Pattern Recognition (CVPR) (2012)
18. Chen, X., Kundu, K., Zhu, Y., Ma, H., Fidler, S., Urtasun, R.: 3D object proposals using stereo imagery for accurate object class detection. IEEE Trans. Pattern Anal. Mach. Intell. **40**(5), 1259–1272 (2017)
19. Loshchilov, I., Hutter, F.: Decoupled weight decay regularization. arXiv preprint arXiv:1711.05101 (2017)