EcoFusion: Energy-Aware Adaptive Sensor Fusion for Efficient Autonomous Vehicle Perception

Arnav Vaibhav Malawade*[1], Trier Mortlock*[2], Mohammad Abdullah Al Faruque[1,2]
Department of Electrical Engineering and Computer Science[1], Department of Mechanical and Aerospace Engineering[2]
University of California, Irvine, California, USA

ABSTRACT

Autonomous vehicles use multiple sensors, large deep-learning models, and powerful hardware platforms to perceive the environment and navigate safely. In many contexts, some sensing modalities negatively impact perception while increasing energy consumption. We propose **EcoFusion**: an energy-aware sensor fusion approach that uses context to adapt the fusion method and reduce energy consumption without affecting perception performance. EcoFusion performs up to **9.5%** better at object detection than existing fusion methods with approximately **60%** less energy and **58%** lower latency on the industry-standard Nvidia Drive PX2 hardware platform. We also propose several context-identification strategies, implement a joint optimization between energy and performance, and present scenario-specific results.

1 INTRODUCTION

Autonomous vehicles (AVs) are expected to improve mobility and road safety dramatically. However, these benefits come with rising energy costs [6]. AVs require large deep-learning (DL) models to perceive the environment and safely detect and avoid objects. The computational demands of these models significantly increase the hardware requirements of AVs, such that modern AV electrical/electronic (E/E) systems can require between several hundred watts (W) to over 1 kW of power. For example, the Nvidia Drive PX2, used for Tesla Autopilot from 2016-2018, has a Thermal Design Power (TDP) of 250 W [11], and modern successors have TDPs ranging from 500 W to 800 W [1]. These power demands can also increase the thermal demands on the vehicle's climate-control system. When combined, these demands can reduce vehicle range by over **11.5%** [14]. This impact is especially limiting for electric vehicles due to their limited battery range and long recharge times [26]. Furthermore, many other autonomous systems, including robotics, unmanned aerial vehicles, and sensor networks, operate in energy-constrained environments [5, 9, 21].

Recent works have attempted to address the energy demands of AV systems with application-specific hardware design, model pruning, and edge-cloud architectures [2, 3, 14, 16, 20]. These methods have specific downsides as they require expensive hardware modifications, extensive domain knowledge, and consistent network connectivity, respectively. Alternatively, efficient sensor-fusion approaches attempt to combine multiple sensing modalities to achieve good perception performance with less energy than conventional fusion [3, 9, 12]. However, these approaches are also limited because they use statically designed fusion algorithms (e.g., early or late fusion) that can lack robustness in difficult driving scenes [15]. Figure 1 illustrates the trade-off between performance and energy between different sensor fusion methods for two contexts: city and rain. *None* refers to using a single sensor with no fusion, *early fusion* combines raw sensor data before processing, and *late fusion* processes each sensor separately before fusing the final outputs. As shown, *no fusion* consumes the least energy but also performs the worst, *late fusion* performs much better but uses almost 3x more energy, and *early fusion* is energy efficient but performs poorly in difficult driving scenarios.

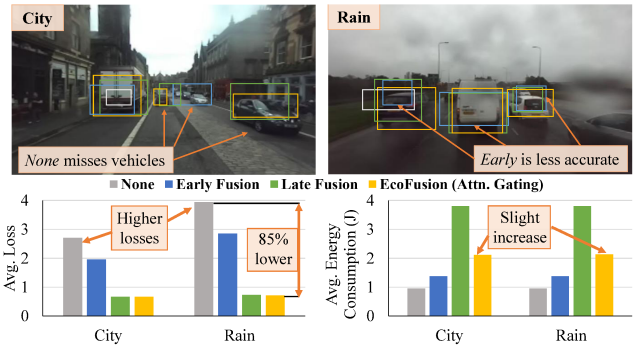

Figure 1: Performance and energy comparison for various AV perception sensor fusion methods in city and rainy driving.

In summary, our key research challenges include: (i) perceiving the environment accurately in difficult contexts, (ii) reducing the energy consumption of AV perception systems, and (iii) adapting the perception model to the current context to minimize energy consumption without compromising perception performance. We propose EcoFusion: an energy-aware sensor fusion approach that uses context to dynamically switch between different sensor combinations and fusion locations. Our approach can reduce energy consumption without degrading perception performance in comparison to both early and late fusion methods. As shown in Figure 1, our approach (shown in gold) achieves higher performance than other fusion methods while significantly reducing energy consumption. The key contributions of this paper are as follows:

(1) We propose an energy-aware sensor fusion approach that uses context to adapt the fusion method and reduce energy consumption without affecting perception performance.
(2) We propose novel gating strategies that can identify the context and use it to dynamically adjust the model architecture as part of a joint optimization between energy consumption and model performance.
(3) We benchmark the hardware performance of our approach on the industry-standard Nvidia Drive PX2 autonomous driving platform.

* Both authors contributed equally to this research.
This work was partially supported by the National Science Foundation (NSF) under awards CMMI-1739503 and CCF-2140154.

DAC '22, July 10–14, 2022, San Francisco, CA, USA

ACM ISBN 978-1-4503-9142-9/22/07.
https://doi.org/10.1145/3489517.3530489

(4) We present an in-depth analysis of the performance of each sensing modality in a range of difficult driving contexts.

2 RELATED WORK

In past years, research on energy-efficient AVs has focused mainly on reducing the energy needs for locomotion and actuation. However, due to the rise in DL perception algorithms and the computational requirements of modern AVs, minimizing the energy consumption of AV E/E systems is becoming a core problem [4, 6]. Authors in [3] focus on improving computational efficiency through algorithmic changes for a camera-lidar AV platform while using knowledge-based network pruning in their DL model. Selectively fusing sensors, as done in [7], also has potential benefits to save computational energy on AVs. Distinct from these methods, our approach utilizes the context of the environment to enable further energy optimization for AVs. Studies have demonstrated the value of context identification, such as in [12], where authors propose altering the power levels and operating state of an AV lidar sensor depending on the environmental factors, such as the vehicle's speed, to improve perception efficiency. Likewise, [9] proposes adjusting the sensing frequency for indoor robot localization according to environmental dynamics. However, these approaches are limited as they rely on statically designed context-based rules, whereas our approach employs a self-adaptive design to learn the context of the environment dynamically.

Trade-offs between the energy and performance of deep neural networks (DNNs), like those used in AV perception, have also been studied. [17] improves the computational efficiency of DNNs for classification by using component-specialization during training and component-selection during inference. [27] presents a structure simplification procedure that removes redundant neurons within DNNs. [25] performs incremental training with DNNs to consider energy-accuracy trade-offs at run-time. Unlike our approach, these works are only applied to classification using a single input modality and do not incorporate context. Additionally, we tackle the complex, cross-domain problem of AV energy optimization with our dynamic sensor fusion architecture, and present experiments involving real AV hardware.

3 PROBLEM FORMULATION

Here we detail the formulation for AV object detection and the joint energy-performance optimization implemented in our work.

3.1 Sensor Fusion for Object Detection

For each input sample, the goal of an object detector ϕ is to utilize the set of sensor measurements in the sample, $\mathbf{X}$, to accurately detect the objects in the scene, $\mathbf{Y}$:

$$\mathbf{Y} = \phi(\mathbf{X}), \text{ where } \mathbf{Y} = \{\mathbf{Y}^i_{class}, \mathbf{Y}^i_{reg}\}_{i=1...d} \quad (1)$$

where d is the number of objects in the sample. ϕ can be implemented via conventional sensor fusion techniques, an ML/DL model, or an ensemble of ML/DL models. The targets for object i in the sample are defined as follows:

$$\mathbf{Y}^i_{class} \in \{c_1, c_2, c_3, \dots\}, \ \mathbf{Y}^i_{reg} = [\mu_1, \nu_1, \mu_2, \nu_2] \in \mathbb{R}^2 \quad (2)$$

where $\mathbf{Y}^i_{class}$ represents the class of the object (e.g., c_1: car, c_2: truck, c_3: pedestrian) from a set of defined object classes, and $\mathbf{Y}^i_{reg}$ represents the 2D bounding box coordinates of the object in reference to the coordinate frame of the sample. We denote the model's estimate of $\mathbf{Y}$ as $\hat{\mathbf{Y}}$. Since $\mathbf{X}$ represents data from multiple heterogeneous sensing modalities, sensor fusion can be used to fuse the data to provide a better estimate of $\mathbf{Y}$. In early fusion, the raw sensor inputs are fused before being passed through the object detector as follows:

$$\hat{\mathbf{Y}} = \phi(\psi(\mathbf{X}_1, \mathbf{X}_2, \dots, \mathbf{X}_s)) \quad (3)$$

where ψ represents the function for fusing the different inputs. In contrast, *late fusion*, involves fusing the outputs of an ensemble of sensor-specific object detectors as follows:

$$\hat{\mathbf{Y}}_1, \hat{\mathbf{Y}}_2, \dots, \hat{\mathbf{Y}}_s = \phi_1(\mathbf{X}_1), \phi_2(\mathbf{X}_2), \dots, \phi_s(\mathbf{X}_s) \quad (4)$$

$$\hat{\mathbf{Y}} = \phi(\hat{\mathbf{Y}}_1, \hat{\mathbf{Y}}_2, \dots, \hat{\mathbf{Y}}_s) \quad (5)$$

3.2 Energy Modeling

In this work, we aim to jointly optimize the energy consumption and performance of the perception system of an AV. To enable this optimization, we use real-world measurements from three different sensors to model the energy consumption of various object detectors ϕ on the industry-standard Nvidia Drive PX2 autonomous driving hardware platform, depicted in Figure 2. For a given object detector implementation ϕ and fixed-size input $\mathbf{X}$, we model energy consumption $\mathbf{E}$ as follows:

$$\mathbf{E}(\phi, \mathbf{X}) = \mathbf{P}(\phi, \mathbf{X}) * t(\phi, \mathbf{X}) \quad (6)$$

where $t(\phi, \mathbf{X})$ represents the processing latency in seconds, and $\mathbf{P}(\phi, \mathbf{X})$ represents the hardware power consumption in Watts of running input $\mathbf{X}$ through ϕ as measured on the hardware. We measured the PX2's average power consumption under load as 45.4 Watts. Assuming X has a fixed size, we calculate $E(\phi)$ for all $\phi \in \Phi$ offline. Next, we use this energy calculation within a joint optimization framework.

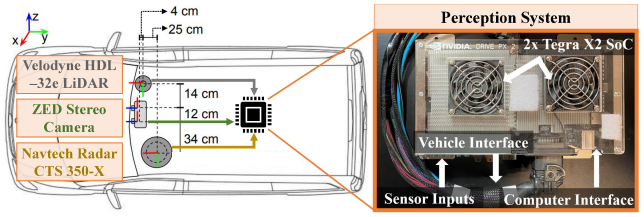

Figure 2: Sensor diagram [22] with our Nvidia Drive PX2.

3.3 Joint Energy-Performance Optimization

We formulate our optimization as a joint minimization problem between energy consumption and model loss. We denote the list of all object detector configurations as Φ. For each configuration ϕ in Φ, we use a model to predict the loss after the outputs of ϕ are fused via late fusion, denoted $L_f(\phi)$. The loss is defined as the combined regression and classification loss (using smooth L1 loss and cross-entropy loss, respectively) between the ground-truth $\mathbf{Y}$ and the $\hat{\mathbf{Y}}$ predicted by the model as defined in [19]. Then, the minimum fusion loss configuration ϕ' is identified. We also define the function ρ, which determines the set of ϕs that have a fusion loss within γ of ϕ'. This set Φ^* is defined as follows:

$$\Phi^* = \rho(L_f(\Phi), \gamma) = \{\phi \in \Phi \text{ s.t. } L_f(\phi) - L_f(\phi') \leq L_f(\phi') + \gamma\} \quad (7)$$

where γ is the maximum allowable difference in loss between any ϕ and ϕ' in order for ϕ to be included in Φ^*. γ can be defined based on the problem and represents the maximum deviation in performance from the best performing configuration ϕ' that is allowed to enable the exploration of more efficient configurations. In some cases, maximum performance may not be necessary, so energy can be saved by increasing γ. Otherwise, if maximum performance is desired, then γ can be set to 0, so only ϕ' is in Φ^*.

Given that $E(\phi)$ is known, we have the following joint loss function for each ϕ in Φ^*:

$$L_{joint}(\phi, \lambda_E) = (1 - \lambda_E) * L_f(\phi) + \lambda_E * E(\phi) \tag{8}$$

where $L(\phi)$ and $E(\phi)$ represent the predicted fusion loss and energy consumption, respectively, of ϕ; and $\lambda_E \in [0.0 - 1.0]$ is the weighting factor that weights the importance of energy consumption vs. performance in the joint optimization. Next, we select ϕ^*, a configuration in Φ^* which lies on the Pareto frontier of the following minimization:

$$\phi^* = \underset{\forall \phi \in \Phi^*}{\arg\min}(L_{joint}(\phi, \lambda_E)) \tag{9}$$

After ϕ^* is identified, it is executed to produce the final set of detections $\hat{\mathbf{Y}}$.

4 ECOFUSION METHODOLOGY

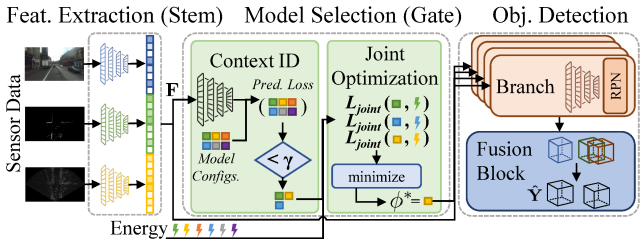

Figure 3: Our proposed EcoFusion framework.

We propose EcoFusion, a novel adaptive sensor fusion approach that jointly optimizes performance and energy consumption by identifying the context of an environment before subsequently adapting the model and fusion architecture. Our model can: (i) adapt between using no fusion, early fusion, and late fusion, (ii) select from one or more radar, lidar, or camera sensor inputs, and (iii) execute different types of fusion simultaneously depending on what it determines is the best execution path to minimize loss and energy consumption in the current context jointly.

The workflow for our approach is shown in Figure 3 and is detailed in Algorithm 1. First, sensor measurements are passed through modality-specific stem models, which produce an initial set of features $\mathbf{F}$ for each sensor. Next, the gate model uses $\mathbf{F}$ and the set of possible model configurations Φ to estimate the loss of each possible configuration for the given inputs. After selecting the candidates for optimization using γ, we pass these candidates Φ^*, their known energy consumption E, and their estimated losses L_f to produce L_{joint} for the optimization function. Then, the ϕ with the lowest L_{joint}, denoted ϕ^*, is selected to execute as is done in Equation 9. Since each ϕ represents an ensemble of one or more object detectors, denoted as branches, we run each branch in ϕ^* with its expected inputs and collect the results $\hat{\mathbf{Y}}^*$. These are then fused using our late fusion block, producing a final set of detections $\hat{\mathbf{Y}}$. The following subsections elaborate on the different components in our approach.

Algorithm 1: EcoFusion Algorithm

Input: $\mathbf{X}$, λ_E, Φ, γ, $E(\Phi)$
Output: Object Detections $\hat{\mathbf{Y}}$

Initialize feature vector $\mathbf{F}$ and branch output vector $\hat{\mathbf{Y}}^*$.
for *s in sensors* **do**
  $\mathbf{F}[s] \leftarrow stem(s)$ `// extract features by modality`
$L_f(\Phi) \leftarrow \text{gate}(\mathbf{F}, \Phi)$ `// estimate model losses`
$\Phi^* \leftarrow \rho(L_f(\Phi), \gamma)$ `// select candidates`
for ϕ *in* Φ^* **do**
  $L_{joint}(\phi, \lambda_E) \leftarrow (1 - \lambda_E) * L_f(\phi) + \lambda_E * E(\phi)$
$\phi^* \leftarrow \arg\min_{\forall \phi \in \Phi^*}(L_{joint}(\phi, \lambda_E))$ `// joint opt.`
for *branch in* ϕ^* **do**
  $\hat{\mathbf{Y}}^*[branch] \leftarrow branch(\mathbf{F}^*)$ `// pass subset of F`
$\hat{\mathbf{Y}} \leftarrow fusion\_block(\hat{\mathbf{Y}}^*)$ `// fuse branch detections`

4.1 Stem Model

The stem models are implemented as a small set of CNN layers that produce an initial set of features for each input modality. The stems are modality-specific, so there is one stem for each type of sensor used. The collection of features F output by the stems is collectively passed to the gate model to identify the context and select the set of branches to execute. Then, F is input to the selected branches.

4.2 Context-Aware Gating Model

We implement several gating strategies to estimate the fusion losses of each sensor configuration and facilitate the selection of ϕ^*. The goal of each gating model is to (i) identify the context based on the input features, (ii) estimate the performance of each model configuration in the context, and (iii) compute the optimization result and use it to select ϕ^*. Next, we detail the different methods we implemented for performing steps (i) and (ii).

4.2.1 Knowledge Gating. Our Knowledge Gating approach uses domain knowledge on the performance of each modality in different driving conditions to statically decide the best configuration for each rigidly-defined driving context (e.g., rain, snow, city, motorway). This gating approach assumes the context can be identified from external sources, such as weather information, GPS location, and time of day. Also, it assumes that the set of possible contexts is finite, which may limit scalability.

4.2.2 Deep Gating. This approach uses a deep-learning model with three CNN layers and one MLP layer to predict the loss for each model configuration for a given set of inputs. Then, the optimization function is run on these outputs.

4.2.3 Attention Gating. This approach is identical to the Deep Gating model, except for the addition of a self-attention layer to enable the gate to identify important areas of the input feature map.

4.2.4 Loss-Based Gating. In this strategy, the *a posteriori* ground-truth loss from each configuration for a given input is used to select ϕ^*. Thus, this implementation is not deployable in the real world but

represents the theoretical best-case performance for a gate model that can perfectly predict the fusion loss of every configuration for every input.

4.3 Branch Models

The branches in the model take the form of various object detectors. Each branch performs object detection by implementing a Faster R-CNN [19] object detector containing a ResNet-18 CNN model [10] to extract features from input images and a Region Proposal Network (RPN) to propose object locations across the feature map. The RPN proposals are then fed through a region-of-interest layer that predicts Y^i_{class}, Y^i_{reg} for each box i, as well as the confidence scores for the predicted boxes. We split each ResNet-18 model after the first convolution block, such that the first block becomes the stem, and the remaining three convolution blocks are used in each branch. Each branch can be configured to process either a single sensor or a set of sensors. In this work, we implement one branch for each input sensor and three early fusion branches that fuse both homogeneous and heterogeneous sets of sensors. Using the gate to select the branches, our model can dynamically choose between no fusion, early fusion, late fusion, and combinations of the three.

4.4 Fusion Block

The fusion block is implemented via a typical late-fusion algorithm. The detections from any number of branches are first converted to a uniform coordinate system before being statistically processed and fused using the weighted box fusion method from [23]. This process helps refine the accuracy of the bounding box predictions by reinforcing predictions with high confidence and overlap with other predictions.

5 EXPERIMENTS

In our experiments, we used the RADIATE [22] dataset, which provides annotated real-world object detection data from an AV with the following sensors: a Navtech CTS350-X radar, a Velodyne HDL-32e lidar, and a ZED stereo camera. The following classes of objects are annotated in the dataset: {*car, van, truck, bus, motorbike, bicycle, pedestrian, group of pedestrians*}. The dataset consists of various difficult driving contexts (*e.g., rain, fog, snow, city, motorway*) that are challenging for typical object detectors. In EcoFusion, we use a 70:30 train-test split across the dataset and train our model with all of the stems and branches enabled using supervised learning. Next, we take the trained stem and branch outputs and use them to separately train the gate model to select the branches that produce the lowest loss for a given stem output (**F**). We evaluate each model's performance at object detection using average loss and mean average precision (mAP), which is widely used for benchmarking object detection models [8, 19]. We compute the mAP for bounding boxes with an intersection-over-union (IoU) ≥ 0.5, aligning with the PASCAL Visual Object Classes (VOC) Challenge [8]. We calculated the energy consumption of each model configuration $\phi \in \Phi$ on the Nvidia Drive PX2 shown in Figure 2. We ignore the energy consumed by the gate models as we measured that they have negligible energy consumption (< 0.005 J) compared to the stems and branches of the model after TensorRT compilation. In all of our experiments, we set $\gamma = 0.5$ as we experimentally determined that it ensures performance at least as good as early and late fusion while enabling energy optimization. However, we note that γ can be tuned based on the requirements for a given application.

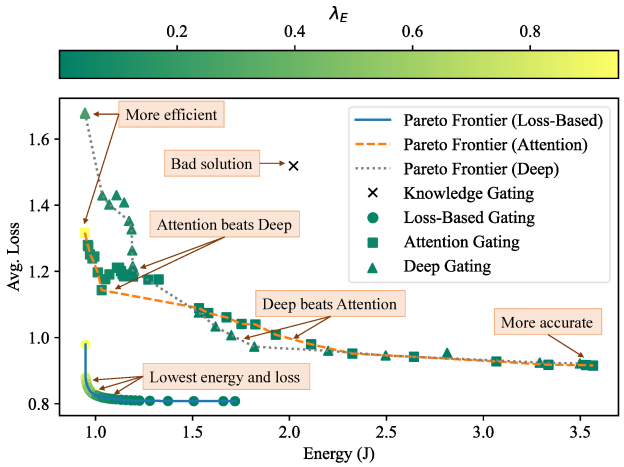

Figure 4: Analysis of the energy-loss trade-off of EcoFusion's optimization function with gating models and λ_E values.

5.1 Joint Optimization Analysis

We evaluated the trade-off between the performance (model loss) and energy consumption (in Joules) for each gating model in Figure 4. We varied λ_E between 0-1.0, where each point in the chart is color-coded according to its λ_E value. As shown, tuning λ_E higher or lower skews the model towards either increasing energy efficiency or increasing performance, respectively, so λ_E should be chosen depending on the requirements for a given application. The configuration for *Loss-Based* that best minimizes both objectives is $\lambda_E = 0.5$ with a loss of 0.966 and energy consumption of 0.844 J. *Attention* and *Deep* have similar Pareto frontiers, but *Attention* achieves better solutions for higher λ_E values while *Deep* achieves slightly lower loss with some low λ_E values. The gap between *Attention/Deep* and *Loss-Based* is likely due to modeling limitations and could potentially be closed using larger or more advanced gate models. For *Attention*, $\lambda_E = 1$ (most energy efficient) results in a loss of 1.317 and an energy consumption of 0.945 J, while $\lambda_E = 0$ (best performing) results in a loss of 0.9153 and an energy consumption of 3.566 J. As shown by the nearly flat trend on the right side of the plot, *Deep* and *Attention* can reduce energy significantly with little effect on loss by tuning λ_E. *Knowledge* is statically programmed such that, for each scenario type, we use domain knowledge to manually select the best sensor combination to use. Due to these constraints, *Knowledge* can be less efficient in some scenarios and is not tunable with our optimization.

5.2 Energy and Performance Evaluation

Our results for energy consumption and performance evaluation are shown in Table 1. In all of our experiments, early fusion takes in both cameras and lidar as input, while late fusion uses both cameras, lidar, and radar. The energy consumption and latency increase as the fusion method is varied from none to early to late, which is as expected as the latter methods require increasingly larger detection pipelines. The single-sensors are the most efficient, but their mAP

scores vary widely from 67% to 79%, likely due to inconsistent performance across scenarios. Early fusion is faster, more efficient, and achieves a higher mAP score and than late fusion; however, early fusion is insufficiently robust in poor driving conditions as will be discussed in Section 5.4. EcoFusion with $\lambda_E = 0.01$ achieves higher mAP than all other methods with less energy than late fusion. With $\lambda_E = 0.05$, EcoFusion still outperforms early fusion with less energy usage. As stated in [14], an AV must be able to process inputs at least once every 100 ms (10 frames per second) to ensure safety. In addition to meeting this latency requirement, EcoFusion also executes faster than both early and late fusion, which can improve safety and responsiveness by enabling the AV to process inputs more frequently. With $\lambda_E = 0.01$, EcoFusion achieves a mAP score **5.1%** and **9.5%** higher than early and late fusion, respectively, with **60%** less energy and **58%** lower latency than late fusion.

Fusion Type	Configuration	mAP (%)	Energy (J)	Latency (ms)
None	L. Camera (C_L)	74.48%	0.945	21.57
	R. Camera (C_R)	79.00%	0.945	21.57
	Radar (R)	67.74%	0.954	21.85
	Lidar (L)	70.45%	0.954	21.85
Early	$C_L + C_R + L$	80.26%	1.379	31.36
Late	$C_L + C_R + L + R$	77.98%	3.798	84.32
EcoFusion (Ours)	$\lambda_E = 0$	82.92%	3.566	81.49
	$\lambda_E = 0.01$	**84.32%**	**1.533**	**35.14**
	$\lambda_E = 0.05$	**82.16%**	**1.110**	**25.43**

Table 1: Energy Consumption and Performance Evaluation

5.3 Gating Method Evaluation

Table 2 shows mAP, loss, and energy results from evaluating our gating strategies at different λ_E values. With $\lambda_E = 0$, the models tend to pick better-performing branches regardless of their energy consumption. As λ_E increases, the joint optimization significantly reduces energy consumption while keeping loss within γ of the lowest-loss configuration. Although *Knowledge* achieves decent mAP scores, it lacks tunability and thus achieves the same loss and energy consumption for all λ_E; the encoded knowledge would need to be manually updated to adjust the trade-off. *Loss-Based* achieves the lowest loss and energy consumption but a lower mAP than *Deep* and *Attention*. This result is likely because loss is not perfectly correlated with mAP score; mAP primarily scores object classification over properly aligned bounding boxes, while loss is measured across both classification and box regression. Overall, *Attention* performs slightly better than *Deep* and offers the best trade-off of performance and energy.

5.4 Scenario-Specific Evaluation

Figure 5 shows loss and energy results for different driving scenarios in the dataset. We evaluated no fusion (radar-only), early fusion, late fusion, and EcoFusion with *Attention Gating*. As shown in the figure, EcoFusion performs similarly to late fusion in terms of loss across all scenarios. It is also clear that early fusion performs poorly in the difficult driving conditions present in the *Fog* and *Snow* scenarios. Late fusion is more robust and achieves relatively good performance across scenes; however, late fusion also consumes significantly more energy than all other methods. In contrast, EcoFusion's energy efficiency is on-par with early fusion and is significantly lower than that of late fusion. No fusion was the most energy-efficient but also had the highest overall loss.

λ_E	Gating Method	mAP (%)	Avg. Loss	Energy (J)
0	*Knowledge*	82.43%	1.519	**2.021**
0	*Deep*	82.68%	**0.915**	3.556
0	*Attention*	**82.92%**	**0.915**	3.566
0	*Loss-Based*	82.50%	0.808	1.719
0.01	*Knowledge*	82.43%	1.519	2.021
0.01	*Deep*	83.72%	1.124	**1.457**
0.01	*Attention*	**84.32%**	**1.089**	1.533
0.01	*Loss-Based*	81.65%	0.809	1.280
0.1	*Knowledge*	**82.43%**	1.519	2.021
0.1	*Deep*	81.98%	1.432	1.008
0.1	*Attention*	79.72%	**1.280**	**0.960**
0.1	*Loss-Based*	79.70%	0.818	1.044

Table 2: Gating method evaluation.

5.5 Discussion

5.5.1 Practicality. Since we evaluated our approach with the industry-standard Nvidia Drive PX2 autonomous driving platform, it is clear that our approach can save energy on real-world AV hardware while meeting real-time latency constraints. Furthermore, by achieving better object detection performance with lower latency, our approach improves safety and robustness over existing methods. Our evaluation on a diverse driving dataset proves that our approach is robust across scenarios and is thus more practical for real-world driving. To implement EcoFusion on a real driving system, the designer would first need to train the model on the appropriate dataset before selecting the best λ_E and γ for their design requirements. Then, the model can be compiled for hardware using TensorRT or a similar library and integrated into the AV stack.

5.5.2 Sensor Clock Gating. More energy could be saved by disabling unused sensors using clock gating. The Navtech CTS350-X radar uses 24 W [18], the Velodyne HDL-32E lidar uses 12 W [13] and the ZED camera uses 1.9 W [24], so reducing sensor energy usage can significantly improve AV efficiency. Temporal modeling can enable the context to be estimated across time instead of for a single input, allowing clock gating for specific periods. In Table 3, we analyze the benefits of sensor clock gating with our *Knowledge Gating* approach in each driving scenario since it uses external context to inform sensor selection. We also show baseline results with late fusion across the four sensors. Using the power consumption P and measurement frequency f of each sensor s, we estimate the energy that could be saved by stopping measurements without slowing the motor's rotation. We cannot completely power gate the rotating lidar and radar sensors because they have inertia and require several seconds to get back up to speed from a stand-still, which can compromise safety. We model the energy consumption E_s of each sensor and the total energy consumption E_{total} as follows:

$$E_s = (P_s^{meas.} + P_s^{motor}) * 1/f_s, \quad P_s^{meas.} = P_s - P_s^{motor} \tag{10}$$

$$E_{total} = E(\phi) + \sum_{s \in \phi} E_s \tag{11}$$

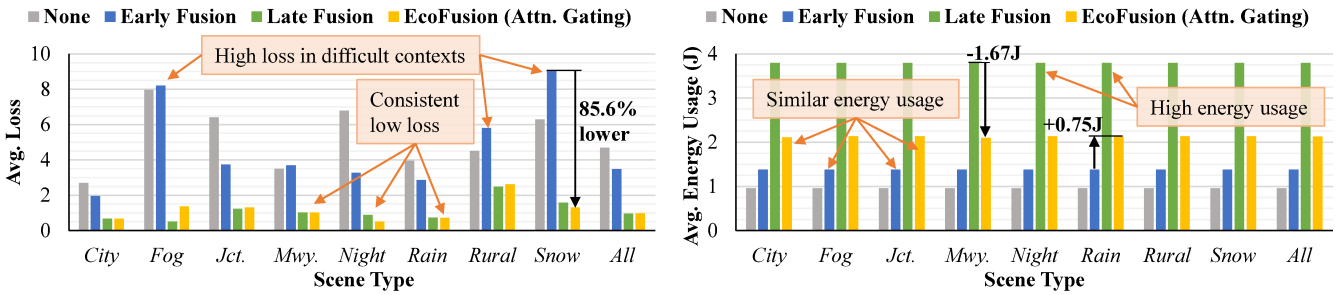

Figure 5: Average loss and energy consumption per scenario for each fusion method. Junction and Motorway are abbreviated as Jct. and Mwy., respectively. EcoFusion achieves low loss across scenes with 43.7% lower energy consumption than late fusion.

Fusion Method	Avg. Energy Consumption (J) by Scene Type								
	City	*Fog*	*Jct.*	*Mwy.*	*Night*	*Rain*	*Rural*	*Snow*	*Overall*
Late Fusion	13.27	13.27	13.27	13.27	13.27	13.27	13.27	13.27	13.27
EcoFusion (Ours)	**5.45**	13.96	**2.87**	**2.87**	**12.10**	13.29	**3.81**	13.96	**6.45**
EcoFusion Energy Savings	**58.91%**	-5.15%	**78.40%**	**78.40%**	**8.81%**	-0.09%	**71.28%**	-5.15%	**51.41%**

Table 3: Combined sensor and AV hardware platform energy consumption in each driving scenario.

where ϕ is the model configuration defined for the context. After our calculation, we set $P^{meas.} = 0$ to simulate clock gating of the sensor. The Navtech CTS350-X consumes 2.4 W to spin the motor, so its $P^{meas.} = 21.6$ W. Based on comparable lidar motor models, we estimate the Velodyne HDL-32E's $P^{meas.} = 9.6$ W. As shown in Table 3, EcoFusion would use up to **78.40%** less energy than late fusion in common driving scenarios. Our approach uses slightly more energy than late fusion in more difficult driving scenarios, but these scenarios are rare, so overall energy consumption is still lower. On average, clock gating unused sensors with EcoFusion uses **51.41%** less energy than running all sensors with late fusion and **43.90%** less energy than EcoFusion without sensor clock gating.

6 CONCLUSION

This paper introduces EcoFusion — a novel adaptive sensor fusion approach that uses contextual information to adapt its architecture and jointly optimize performance and energy consumption. We show that EcoFusion outperforms early and late fusion in terms of mAP (**84.32%** vs. 80.26% and 77.98%), with similar energy consumption and latency to early fusion. We also demonstrate that in difficult driving contexts, EcoFusion is more robust than early fusion (up to **85.6%** lower loss) and more efficient than late fusion (**60%** less energy). We additionally propose and evaluate multiple gating strategies and find that a learned strategy outperforms a knowledge-based strategy. Overall, we show that an energy-aware adaptive sensor fusion approach can significantly improve the energy efficiency and perception performance of AVs.

REFERENCES

[1] Sam Abuelsamid. 2020. Nvidia Cranks Up And Turns Down Its Drive AGX Orin Computers. *Forbes* (Jun 2020). https://www.forbes.com/sites/samabuelsamid/2020/05/14/nvidia-cranks-up-and-turns-down-its-drive-agx-orin-computers

[2] Sabur Baidya *et al.* 2020. Vehicular and edge computing for emerging connected and autonomous vehicle applications. In *DAC '20*. IEEE, 1–6.

[3] Dieter Balemans *et al.* 2020. Resource efficient sensor fusion by knowledge-based network pruning. *Internet of Things* 11 (2020), 100231.

[4] Jared A Baxter *et al.* 2018. Review of electrical architectures and power requirements for automated vehicles. In *ITEC '18*. IEEE, 944–949.

[5] Ivan Beretta *et al.* 2012. Design exploration of energy-performance trade-offs for wireless sensor networks. In *DAC '12*. 1043–1048.

[6] Justin M Bradley *et al.* 2015. Optimization and control of cyber-physical vehicle systems. *Sensors* 15, 9 (2015), 23020–23049.

[7] Changhao Chen *et al.* 2019. Selective sensor fusion for neural visual-inertial odometry. In *CVPR '19*. 10542–10551.

[8] Mark Everingham *et al.* 2010. The pascal visual object classes (VOC) challenge. *International Journal of Computer Vision* 88, 2 (2010), 303–338.

[9] Vineet Gokhale *et al.* 2021. FEEL: fast, energy efficient localization for autonomous indoor vehicles. In *ICC '21*. IEEE.

[10] Kaiming He *et al.* 2016. Deep residual learning for image recognition. In *CVPR '16*. 770–778.

[11] Fred Lambert. 2016. All new Teslas are equipped with NVIDIA's new Drive PX 2 AI platform for self-driving - Electrek. https://electrek.co/2016/10/21/all-new-teslas-are-equipped-with-nvidias-new-drive-px-2-ai-platform-for-self-driving.

[12] Sanghoon Lee *et al.* 2020. Accuracy–power controllable lidar sensor system with 3D object recognition for autonomous vehicle. *Sensors* 20, 19 (2020), 5706.

[13] Velodyne Lidar. 2021. Velodyne HDL-32e Datasheet. https://velodynelidar.com/products/hdl-32e/

[14] Shih-Chieh Lin *et al.* 2018. The architectural implications of autonomous driving: Constraints and acceleration. In *ASPLOS'18*.

[15] Arnav Vaibhav Malawade, Trier Mortlock, and Mohammad Abdullah Al Faruque. 2022. HydraFusion: Context-Aware Selective Sensor Fusion for Robust and Efficient Autonomous Vehicle Perception. In *ICCPS '22*.

[16] Arnav Malawade *et al.* 2021. SAGE: A Split-Architecture Methodology for Efficient End-to-End Autonomous Vehicle Control. *ACM TECS* 20, 5s (2021).

[17] Ravi Teja Mullapudi *et al.* 2018. Hydranets: Specialized dynamic architectures for efficient inference. In *CVPR '18*. 8080–8089.

[18] Navtech Radar. 2021. Navtech CTS Series. https://navtechradar.com/clearway-technical-specifications/compact-sensors

[19] Shaoqing Ren *et al.* 2015. Faster R-CNN: Towards real-time object detection with region proposal networks. *Advances in neural information processing systems* 28 (2015), 91–99.

[20] K. Samal *et al.* 2020. Attention-Based Activation Pruning to Reduce Data Movement in Real-Time AI: A Case-Study on Local Motion Planning in Autonomous Vehicles. *IEEE Journal on Emerging and Selected Topics in Circuits and Systems* 10, 3 (2020), 306–319.

[21] Shreyas Sen. 2016. Context-aware energy-efficient communication for IoT sensor nodes. In *DAC '16*. IEEE, 1–6.

[22] Marcel Sheeny *et al.* 2020. RADIATE: A radar dataset for automotive perception. *arXiv preprint arXiv:2010.09076* 3, 4 (2020).

[23] Roman Solovyev *et al.* 2021. Weighted boxes fusion: Ensembling boxes from different object detection models. *Image and Vision Computing* 107 (2021), 104117.

[24] Stereolabs. 2021. ZED Camera and SDK Overview. https://cdn.stereolabs.com/assets/datasheets/zed-camera-datasheet.pdf

[25] Hokchhay Tann *et al.* 2016. Runtime configurable deep neural networks for energy-accuracy trade-off. In *2016 CODES + ISSS*. IEEE, 1–10.

[26] Korosh Vatanparvar *et al.* 2015. Battery lifetime-aware automotive climate control for electric vehicles. In *DAC '15*. IEEE, 1–6.

[27] Boyu Zhang *et al.* 2018. Exploring energy and accuracy tradeoff in structure simplification of trained deep neural networks. *IEEE Journal on Emerging and Selected Topics in Circuits and Systems* 8, 4 (2018), 836–848.