HERACLES: Hierarchical Semantic Communications for Distributed Dynamic Sensor Fusion

HERACLES: Hierarchical Semantic Communications for Distributed Dynamic Sensor Fusion

***Abstract*—Distributed sensor fusion is a key component of a broad spectrum of applications, such as autonomous systems, where the ability of jointly process multi-sensor data at the edge boosts the range of operating conditions and overall task performance. However, existing distributed sensor fusion approaches encounter limitations in achieving efficient transmission and computation, primarily due to sensor data redundancy, unreliable sensor data transmission, and inflexible sensor fusion methods. In this paper, we propose HERACLES, a distributed sensor fusion framework that connects multi-branched dynamic neural network architectures, which we extend to include branches of different complexity, to (i) a computing methodology that distributes portions of the multi-branched neural network across mobile devices and edge servers, enabling flexible semantic feature extraction and sensor fusion; (ii) a hierarchical modulation-based transmission strategy, where multi-modal semantic features are allocated to different modulation layers to provide varying levels of error protection, and (iii) an infrastructure-level logic that controls the matching between semantic features and modulation layers, and the complexity of the neural model itself to meet an accuracy target while minimizing latency and energy consumption. As a result, HERACLES deeply connects computing, communications and resource allocations in a semantic and context-aware fashion. We evaluate HERACLES using real-world datasets and demonstrate that it can reduce the total delay and energy consumption by 20.39%–89.41% and 4.86%–88.17% (resp.), while maintaining near-optimal inference accuracy. The evaluation code is available at https://github.com/ICDCS-HERACLES/HERACLES.**

***Index Terms*—Distributed sensor fusion, Dynamic neural networks, Hierarchical modulation.**

I. Introduction

Multi-sensor fusion has become a widely adopted technique across various modern applications, such as environmental monitoring, robotics, and intelligent transportation systems [1], [2]. By combining information from multiple sensors, multi-sensor fusion compensates for individual sensor limitations and enhances perception, accuracy, and robustness through the integration of data from diverse sensors. To address the challenges of intensive data processing demands in real-time applications, edge-computing-based distributed sensor fusion has emerged as a key solution [3]. By offloading sensor data to edge servers for processing or fusion, distributed sensor fusion can result in substantially smaller delays, since it avoids slow processing at capability-constrained devices, and bandwidth consumption, as it avoids offloading tasks to cloud servers.

Although numerous studies have developed distributed multi-sensor fusion frameworks based on edge computing (see Sec. II for a detailed discussion) [3]–[6], some critical technical issues have not yet been fully addressed, specifically:

• **Sensor Data Redundancy:** With the technological advancement in sensor data acquisition, sensors are expected to generate large volumes of data. For example, a ZED stereo camera generates 672×376 image resolution at 15 fps, thus needing more than 10 Mb/s throughput for data transfer after quantization [7]. Transmitting all raw sensor data to the edge server would thus dramatically increase the bandwidth demand, or lead to unacceptable transmission delays. Therefore, efficient sensor data compression techniques are needed to reduce the amount of data to be transferred while preserving essential information.

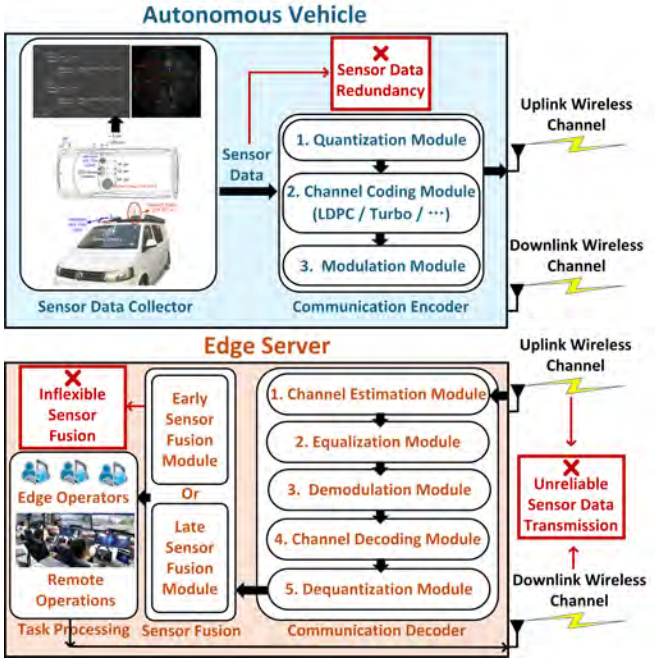

Fig. 1. Pictorial representation of existing distributed sensor fusion frameworks in the remote-assisted autonomous driving use case.

• **Unreliable Sensor Data Transmission:** Methodologies to ensure context-aware reliable transmission of multi-modal sensor data has been scarcely explored in existing work. As the contribution of each sensor modality to fusion quality varies depending on the task context [8], it is beneficial to allocate the limited communication resources to provide hierarchical error protection to multi-modal sensor data based on their importance, i.e., providing higher protection for critical sensor data to maintain sensor fusion accuracy, especially under poor channel conditions.

• **Lack of Flexibility in Sensor Fusion:** Transmitting all sensor data for fusion does not always guarantee optimal inference accuracy [9]. In certain contexts, the existing static distributed sensor fusion approaches lead to an increase in transmission as well as computational overhead without yielding proportional improvements in accuracy (e.g., camera data are not informative at night). Thus, the sensor data to be fused should be dynamically chosen according to the task context.

Moreover, it is worth noting that sensor data transmission

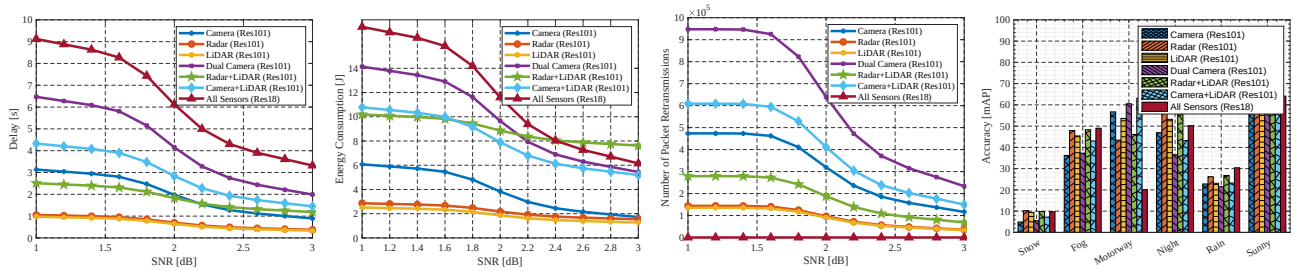

Fig. 2. Performance of existing distributed sensor fusion approaches under different system parameters.

and fusion are coupled processes, making it even more challenging to optimize a distributed sensor fusion system. Focusing solely on minimizing transmission delay or maximizing sensor fusion accuracy can negatively impact the other, as a sensor combination with higher fusion accuracy is usually more complex and requires dealing with a larger amount of data. Therefore, an infrastructure-level logic is needed to optimize the computing and communication processes across diverse channel conditions and task contexts.

To highlight the significance of the above challenges, we conduct a simulation based on existing distributed sensor fusion frameworks, using the relevant case study of remote-assisted autonomous driving where, to make an example, based on the results of sensor fusion, the autonomous vehicles may be controlled by a remote operator to enhance safety [10]. As depicted in Fig. 1, in these frameworks, all raw sensor data are directly transmitted from the autonomous vehicle to an edge server for object detection, without performing any pre-processing (other than image compression) at the vehicle. To evaluate the existing frameworks, we use the RADIATE dataset [7] – an autonomous vehicle perception dataset collected in a variety of task contexts. Accordingly, we consider the vehicle to be equipped with four sensors, namely, two ZED stereo cameras, a 32-channel Velodyne HDL-32e LiDAR, and a Navtech CTS350-X Radar. The task context includes various weather conditions and road types, such as 'Fog', and 'Night', 'Urban', and 'Motorway'. Quadrature Phase Shift Keying (QPSK) modulation and Low-Density Parity-Check (LDPC) coding are used to transmit the sensor data, and we adopt the sensor fusion model in [9], which is constructed based on the ResNet [11] with different complexity.

The average performance per inference task of the existing frameworks is shown in Fig. 2, in terms of: (i) total system delay, including transmission and processing (top left), (ii) total system energy consumption, including transmission and processing (top right), and (iii) number of packet retransmissions (bottom left), for varying values of signal-to-noise ratio (SNR) and sensor fusion accuracy, and (iv) achieved sensor fusion accuracy expressed as mean average precision (mAP) under different task contexts (bottom right). All results are plotted also under different levels of sensor fusion model complexity (indicated between brackets in the plots legend). It can be observed that directly transmitting sensor data leads to unacceptable delay and energy consumption, even for high SNR values. Notably, this approach fails to meet the stringent delay requirements for time-sensitive applications such as autonomous driving, which typically demands response times within 50–100 ms. Furthermore, simply transmitting all sensor data ("All Sensors") does not guarantee the highest possible accuracy, highlighting the inefficiency of a blanket approach to data transmission. Finally, varying contexts significantly influences the accuracy of sensor fusion, underscoring the need for adaptive sensor fusion strategies that account for context-specific requirements.

To overcome the above challenges, we propose HERACLES, an edge-end collaborative dynamic sensor fusion framework based on hierarchical communication. HERACLES has the following features: *(i)* To reduce data redundancy, it deploys dynamic gated neural networks (NN) called "stems" at the mobile device to enable flexible semantic feature extraction of multi-modal sensor data; *(ii)* To ensure reliable transmission of sensor data features, it exploits hierarchical modulation to provide hierarchy error protection for different data modalities, while increasing the transmission rate; *(iii)* To enable flexible sensor fusion, it deploys dynamic gated NNs, referred to as "branches," at the edge, offering a hybrid sensor fusion approach that supports varying model complexities and sensor combinations; *(iv)* To minimize energy consumption and delay, it supports infrastructure-level computing and transmission strategy orchestration. The semantic feature extraction, hierarchical modulation layer allocation, and dynamic sensor fusion strategies are therefore jointly optimized, based on channel conditions and task contexts.

Our main contributions can thus be summarized as follows:

- We propose HERACLES, an edge-end collaborative dynamic sensor fusion framework based on hierarchical communication. HERACLES supports context-aware sensor data compression and flexible sensor fusion via dynamic gated NNs. Further, HERACLES can enable hierarchical error protection for sensor data with different modalities via hierarchical communication. We use remote-assisted autonomous driving as a case study to illustrate HERACLES's workflow.
- Taking into account the dynamic nature of sensor fusion task contexts and channel conditions, we formulate the DeFuSe (Distributed Fusion of Sensor data) problem to jointly minimize the average energy consumption of the sensor fusion system and the delay under inference accuracy constraints. The DeFuSe problem leads to the *joint* and *context-aware*

optimization of the computing and the transmission process, including the pre-processing stem selection, branch selection for fusion, and the allocation of the sensor feature to modulation layers. We remark that by connecting the dynamic shape of the model to the communication and resource allocation logics, we make the latter responsive to the semantics of the data and context. In light of its NP-hardness, we develop an optimization engine based on Rainbow deep Q-Network (Rainbow DQN), enabling more efficient exploration of optimal communication and computation strategies compared to traditional deep Q-Network (DQN). By leveraging the DRL-based optimization engine, HERACLES can support context-aware dynamic strategy orchestration.

• We conduct an extensive experimental evaluation using real-world autonomous driving and wireless datasets to demonstrate the effectiveness of HERACLES. Our results show that channel conditions and task contexts play a critical role in influencing distributed sensor fusion system performance. Also, the DRL-based optimization engine in HERACLES exhibits a notable degree of generalization capability across varying contexts and SNRs, even if it is trained in an environment with static reward weights, SNRs, and mixed task contexts. Compared to state-of-the-art distributed sensor fusion schemes, HERACLES reduces total delay and energy consumption by 20.39%–89.41% and 4.86%–88.17%, respectively, while maintaining near-optimal inference accuracy.

II. Related Work

Our work mainly relates to the areas of distributed sensor fusion at the network edge and semantic communications.

Edge Computing-Enabled Distributed Sensor Fusion. These frameworks fully offload data analysis to edge servers, enhancing efficiency for applications with strict latency and accuracy targets, such as autonomous driving and industrial automation. [3] proposes a fusion perception algorithm for cameras and mm-wave radar, which leverages DBSCAN clustering and the Munkres algorithm to enhance edge-based data acquisition, processing, and fusion. [4] addresses low-latency street information collection with a fusion method for anti-multipath camera-radar sensors, using object-level data for higher accuracy. EdgeServe [5], instead, manages machine learning inference and data movement in an edge network, coordinating multi-sensor fusion tasks. A seamless edge and fog platform is introduced in [6], integrating complex sensors through a symbiotic HW/SW design for efficient data fusion.

However, the above works mainly focus on the design of sensor fusion methods: they either execute sensor fusion entirely on local devices or offload all processing to edge servers. As a result, users may suffer either high local processing delays due to limited on-device resources or substantial data transmission delays associated with offloading all sensor data to the edge. In contrast, HERACLES introduces an edge-end collaborative approach to sensor fusion, where an adaptive portion of sensor data processing is performed locally on the device, greatly reducing the amount of data that needs to be transmitted. The computationally intensive sensor fusion tasks are then handled by the more powerful edge server, trading-off transmission efficiency with processing capability.

Multi-Modal Semantic Sommunications. These techniques focus on transmitting "meaningful" and "task-oriented" information in a semantic fashion rather than raw data as plain traffic. Recent studies have developed end-to-end frameworks for multi-modal semantic feature extraction and transmission. In [12], a two-layer knowledge graph-based model (KG-MSF) extracts shallow-level semantics directly and derives deep-level semantics through logical reasoning, both represented as triplets for efficient transmission. [13] introduces a rate-adaptive mechanism that adjusts coding rates based on semantic importance and channel conditions, while [14] proposes a vector-wise dynamic scheme with a lightweight feature selection module and unified codebook to minimize transmission overhead. Additionally, CA_DeepSC [15] utilizes cross-modal alignment to enhance resilience, while [16] presents a channel-level fusion method where the channel itself acts as the fusion medium, using semantic precoding to counteract channel randomness.

It is worth noting that these approaches adopt static sensor fusion methods that use all sensors without considering the task contexts [12], [13], [15], [16]. Further, they fail to ensure reliable transmission for multi-modal data of different importance [12], [14]–[16]. HERACLES, instead, (i) uses context-aware dynamic sensor fusion via dynamic NNs, and (ii) supports error protection under different modal sensor data, via hierarchical communication, according to a context-specific data relevance.

III. System Model

In this section, we describe the workflow of HERACLES referring to the remote-assisted autonomous driving scenario as a use case, where[1] autonomous vehicles transmit multi-modal sensor data to the edge server for remote sensor fusion and object detection. As illustrated in Fig. 3, the information flow in HERACLES begins with the collection of multi-modal sensor data (e.g., camera, LiDAR, Radar) on the vehicle side. A dynamic-gated NN-based semantic encoder extracts semantic features, which are then processed by a channel encoder for coding and hierarchical modulation. These semantic features are transmitted to the edge via a wireless channel. At the edge side, they undergo demodulation, decoding, and semantic decoding, followed by sensor fusion. Below, we detail and model each of these components.

Dynamic Semantic Encoder. Consider an autonomous vehicle equipped with a set $\mathcal{N}=\{1,\ldots,n,\ldots,N\}$ of sensors, characterized by the modality set $\mathcal{M}=\{1,\ldots,m,\ldots,M\}$. The system operates in time slots indexed by $\mathcal{T}=\{0,\ldots,T\}$, where, at each slot t, a remote-assisted computer vision task (e.g., object detection) is generated and offloaded to the edge server for processing. The task context, denoted by $\psi(t)$,

[1] HERACLES can also be directly applied to other multi-modal sensor fusion tasks, like visual question answering (VQA), based on computer vision and text information [17], by changing the sensor inputs and tasks after fusion.

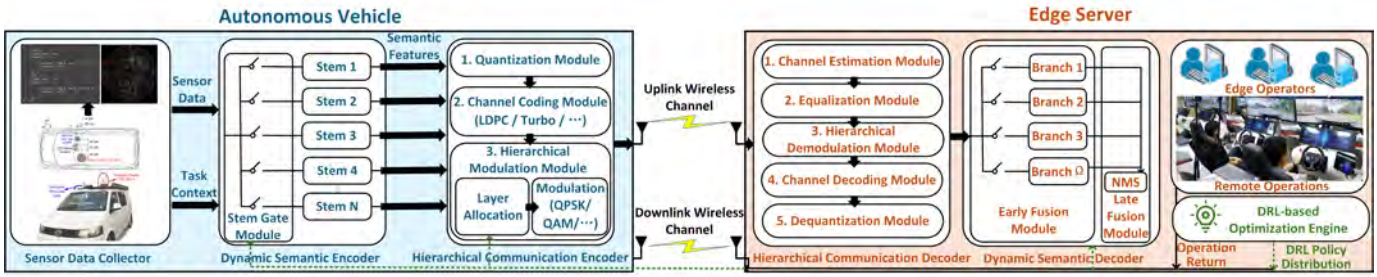

Fig. 3. The framework of HERACLES using the remote-assisted autonomous driving as use case.

captures dynamic conditions such as weather (sunny, rainy, etc.), time (day, night), and road type (highway, urban). A dynamic gated NN-based semantic encoder extracts semantic features from multi-modal sensor data, using multiple distinct convolutional NN (CNN) blocks (corresponding to the first block of the ResNet architecture) called "stems" to process each modality [18]. The task context $\psi(t)$ greatly influences sensor and stem selection for semantic feature extraction. Sensors like LiDAR and radar are prioritized in low-visibility or adverse conditions due to their robustness, while urban environments may require more diverse sensor data compared to highway scenarios. Let $\mathbf{S}^{m,n}(t)$ be the extracted semantic features from sensor n of modality m, and $\widetilde{\mathcal{N}}\subseteq\mathcal{N}$ the set of sensors selected for feature extraction (with $\widetilde{N}=|\widetilde{\mathcal{N}}|$). We define a binary vector $\boldsymbol{\pi}^s(t)=[\pi_n^s(t)]_{n\in\mathcal{N}}$ to represent the stem selection strategy, where $\pi_n^s(t)=1$ if semantic features $\boldsymbol{S}^{m,n}(t)$ will be extracted at time slot t, and $\pi_n^s(t)=0$ otherwise. Since each different stem will process sensor data for a corresponding modality, $\widetilde{N}=\sum_{n\in\mathcal{N}}\pi_n^s(t)$. The stem selection strategy $\boldsymbol{\pi}^s(t)$ dynamically adjusts active stems based on $\psi(t)$, optimizing computational efficiency and reducing energy consumption by processing only those sensor data that are relevant for the context at hand.

Hierarchical Channel Encoder. To ensure reliable wireless transmission, we propose a hierarchical channel encoder comprising three modules: quantization, channel coding, and hierarchical modulation. The first one converts real-number semantic features extracted by stems into binary data suitable for channel encoding. Specifically, each extracted real-number semantic feature $\boldsymbol{S}^{m,n}(t)$ is normalized to $[0,1]$ using its maximum absolute value and quantized into an L-bit fixed-point representation [19]. The quantized feature is then encoded by the channel coding module. The channel coding module applies error correction codes to enhance robustness with redundancy via methods like Turbo, LDPC, and Polar codes, enabling error detection and correction at the edge to reduce bit error rate (BER). The hierarchical modulation module allocates semantic features from different sensors to various transmission layers based on feature importance, BER tolerance, channel conditions, and context, using different modulation schemes per layer to achieve hierarchy transmission rates and BER protection level.

The structure of the hierarchical modulation module is depicted in Fig. 4(left side). We consider a K-layer ($K=\widetilde{N}$) hierarchical modulation scheme for semantic transmission, where semantic features from the $\widetilde{N}$ selected sensors are allocated to different modulation layers based on channel conditions and task contexts. Let $\mathcal{K}=\{1,\ldots,K\}$ be the set of layers; when $K=1$, the traditional single layer modulation will be used. Let $\boldsymbol{o}(t)=[o_k(t)]_{1\le k\le K}$ be the modulation order vector for each layer in time slot t. The first (K-th) layer, carrying the most (least) important feature, uses the $o_1(t)$ ($o_K(t)$)-order modulation. Constellations in hierarchical modulation follow Gray mapping, ensuring symbols at minimum Euclidean distance differ by only one bit. To allocate sensor semantic features to different layers, we define a K-dimensional integer variable $\boldsymbol{\pi}^h(t)$ representing the hierarchical allocation strategy at time t, with $\pi_k^h(t)\in\widetilde{\mathcal{N}}$. This indicates that a feature can only be assigned to a layer if selected for transmission. Again, the task context $\psi(t)$ plays a crucial role in the layer allocation for hierarchical transmission. Different contexts affect the relevance of sensor features, leading to dynamic sensor selection and feature prioritization across transmission layers. For instance, in adverse weather or nighttime conditions, features from LiDAR or radar sensors, which are more robust to environmental changes, may be assigned higher priority layers, ensuring reliable delivery. Conversely, in highway scenarios, fewer sensors with smaller feature outputs may suffice, which can reduce the transmission delay.

Assume the transmitted symbols of different layers $\boldsymbol{x}_k(t)$, $k=1,\ldots,K$, where row vector $\boldsymbol{x}_k(t)$ is the sub-packet of the k-th layer after serializing the corresponding semantic features selected by $\boldsymbol{\pi}^s(t)$. Without loss of generality, we assume that both the autonomous vehicle and edge server are equipped with a single antenna [19]; the transmission model and problem formulation can be easily extended to the multi-antenna case. Let $p_k(t)$ be the transmit power of the k-th layer signal in slot t, and the power vector (in W) is $\boldsymbol{p}(t)=[p_k(t)]_{k\in K}$. The power is allocated based on sub-packet importance, with higher importance bits receiving more power [20]. Assuming the vehicles maximum transmit power is p^m, we have $p_1(t)>\ldots>p_K(t)$, where $\sum_{k=1}^K p_k(t)=p^m$. Then the signals from different modulation layers will be superposed and transmitted in series. $x(t)$ denotes the superposed symbols, which satisfy $\boldsymbol{x}(t)=\sum_{k=1}^K p_k(t)\boldsymbol{x}_k(t)$.

Wireless Transmission. The superposed signals are sent via the wireless channel. For the received symbol, the first $\log_2(o_1(t))$ bits are the first layer data, the next $\log_2(o_2(t))$ bits are the second layer data, till the last $\log_2(o_K(t))$ bits coming from the last layer. The signal received at the edge

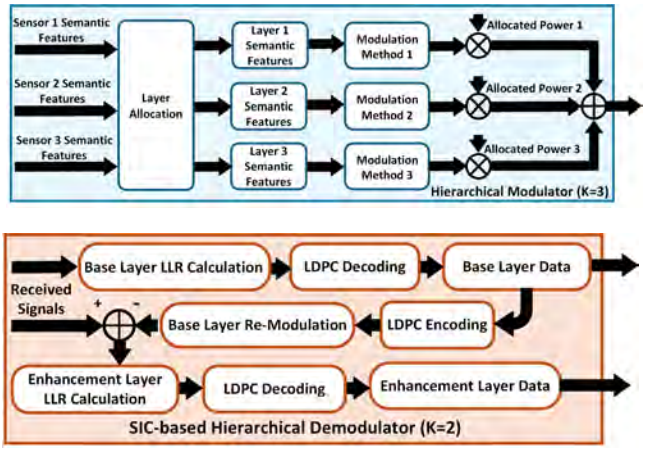

Fig. 4. Architecture of the Hierarchical Modulation module (top) and of the Demodulation module (bottom).

at slot t is $\boldsymbol{y}(t)=h(t)\boldsymbol{x}(t)+\boldsymbol{z}(t)$ where $h(t)$ is the channel between the vehicle and edge server and $\boldsymbol{z}(t)$ is the white Gaussian noise (AWGN) with elements i.i.d. with 0 mean and variance $(\sigma^n)^2$. We adopt a block fading model with a constant channel gain within a slot but i.i.d. across different slots. The task context, $\psi(t)$, will affect transmission performance since the data size is directly determined by the selected stems. Also, the data transmitted at each layer varies in volume, leading to different BER and packet error rates (PER), which in turn yield different numbers of retransmissions.

Hierarchical Channel Decoder. The hierarchical channel decoder comprises 5 modules: the channel estimation module, channel equalization module, hierarchical demodulation module, channel decoding module, and dequantization module. In the first module, considering imperfect channel state information (CSI), we write the estimated channel as $\hat{h}(t)=h(t)+\Delta h(t)$, where $\Delta h(t)\sim\mathcal{CN}(0,(\sigma^e)^2)$ is the estimation error, with σ^e indicating channel estimation accuracy [21]. In the channel equalization module, least squares is used for channel equalization. After channel estimation, the estimated sub-packet of the first layer, $\hat{\boldsymbol{x}}_1(t)$, can be obtained as $\hat{\boldsymbol{x}}_1(t)=\frac{\boldsymbol{y}(t)}{\hat{h}(t)}$. We adopt soft demodulation based on the maximum likelihood rule; the log-likelihood ratio (LLR) of the first layers data (with the highest power level) is calculated based on the distance from each constellation point. To mitigate inter-layer interference, successive interference cancellation (SIC) is applied [22]. Once the demodulated sub-packet of the first layer is obtained, it is re-coded, re-modulated, multiplied by the estimated channel $\hat{h}(t)$, and subtracted to cancel the first layers interference. The sub-packet of the second layer is then calculated as: $\hat{\boldsymbol{x}}_2(t)=\frac{\boldsymbol{y}(t)-\hat{h}(t)\bar{\boldsymbol{x}}_1(t)}{\hat{h}(t)}$, where $\bar{\boldsymbol{x}}_1(t)$ is the re-modulated estimated sub-packet of the first layer. The LLR of the second layer (with the next highest power level) is then calculated, and this sub-packet is also removed from $\boldsymbol{y}(t)$ for third-layer demodulation. This process continues until all sub-packets across layers are estimated. The computed LLRs from soft demodulation are then fed into the channel decoding module and dequantization module to reconstruct the semantic features. The structure of the hierarchical demodulation module is illustrated in Fig. 4 (right side).

For simplicity, we assume HERACLES is an error-free system, i.e., the edge server discards erroneous packets and requests retransmission. To reduce the PER, the superposed packet $\boldsymbol{x}(t)$ is divided into smaller packets of equal size. A Hybrid Automatic Repeat Request (HARQ) mechanism is adopted, combining cyclic redundancy check (CRC) for error detection with forward error correction. Corrupted packets failing the CRC are stored, retransmitted, and combined with the stored one to improve the likelihood of successful decoding.

Dynamic Sensor Fusion. The estimated semantic features are sent to the sensor fusion module for further processing to produce outputs for specific tasks such as object detection. Similarly to the dynamic semantic encoder at the vehicle side, the dynamic sensor fusion module is also built based on dynamic gated NNs, allowing flexible sensor fusion depending on task contexts and channel conditions. We adopt a hybrid sensor fusion approach [9], which synergistically leverages both methods: early fusion integrates raw data for detailed information, while late fusion combines high-level decisions, providing computational efficiency and robustness against sensor failures. The sensor fusion module includes a early fusion module and a late fusion module, connected by multiple gates. For the estimated semantic features, early fusion controls gates to select a subset of appropriate branches. These branches, implemented using a 2D convolutional layer, integrate the concatenated features, which then serve as inputs to subsequent ResNet layers. The complexity of the branches varies, influencing the sensor fusion's accuracy. The dynamic gated mechanism ensures that branch complexity aligns with task demands and channel conditions. In scenarios requiring high accuracy, such as dense urban environments or at nighttime, more complex branches may be selected to handle diverse sensor inputs and provide detailed feature integration. Meanwhile, during simpler tasks or under constrained channel conditions, lower-complexity branches can be utilized to maintain efficiency without significantly compromising accuracy.

Let Ω be the total number of branches at the edge, and $\Omega=\{1,\ldots,\Omega\}$. We define $\boldsymbol{f}_\omega=f_\omega^n,\forall\omega\in\Omega,\forall n\in\mathcal{N}\}$ as an N-dimensional binary variable where $f_\omega^n=1$ if the branch ω can process feature $\boldsymbol{S}^{m,n}(t)$ from sensor n. The Ω-dimensional binary vector $\boldsymbol{\pi}^b(t)=[\pi_\omega^b(t)]_{\omega\in\Omega}$ denotes the branch selection strategy, where $\pi_\omega^b(t)=1$ if branch ω is selected at time slot t. Else, $\pi_\omega^b(t)=0$ if all sensor data required by branch ω are not transmitted, i.e., $f_\omega^n=1$, $\pi_n^s(t)=0$, $\forall n\in\mathcal{N}$, $\forall\omega\in\Omega$. Notably, a sensors feature can be processed by multiple branches. Then the outputs from different branches are further processed by the late fusion module to produce the final task inference result. For late fusion, we apply non-maximum suppression [23], which retains the highest confidence bounding box for each target and removes overlapping boxes based on the intersection over union ratio between predicted bounding boxes.

IV. Optimal Semantic Distributed Sensor Fusion

We now present the key performance metrics of a distributed sensor fusion system (Sec. IV-A). Based on these metrics, we

formulate the DeFuSe (Distributed Fusion of Sensor data) optimization problem that aims at maximizing efficient sensor data transmission and sensor fusion in HERACLES (Sec. IV-B).

A. Relevant Metrics

Total Delay. The total system delay in slot t, $T^a(t)$, consists of the stem processing delay $T^s(t)$, wireless transmission delay $T^c(t)$, and sensor fusion delay at the edge $T^b(t)$. Given its negligible impact, we ignore the channel encoding/decoding delay. Let $\eta_k(t)$ (given below) be the achievable rate of layer k in slot t; the transmission delay can be written as $T^c(t)=\sum_{k\in K}\frac{D_k(t)}{\eta_k(t)}$. The stem and branch processing delay can be calculated based on the NN floating point operations and HW capacity; to do so, we adopt the delay model in [24].

Total Energy Consumption. The total energy consumption in slot t, $E^a(t)$, includes the stem processing $E^s(t)$, the wireless communication $E^c(t)$, and the branch processing $E^b(t)$ components. We have: $E^c(t)=\sum_{k\in K}p_k(t)\frac{D_k(t)}{\eta_k(t)}$, while the stem and branch processing energy consumption can be obtained according to the model in [25].

Communication Performance. The relevant metrics for data transfer over the wireless channel are achievable rate, BER, and PER. Let $\mathrm{SNR}_k(t)$ be the SNR of layer k and B the bandwidth; using Shannon's formula, the achievable rate $\eta_k(t)$ of layer k is given by $B\log_2(1+\mathrm{SNR}_k(t))$. Considering the SIC technique, we have: $\mathrm{SNR}_k(t)=\frac{p_k(t)}{(\sigma^n)^2+\sum_{i=k+1}^{K}p_i(t)}$. Denoting the BER of layer k with $\zeta_k^b(t)$, the general BER expressions for hierarchical QAM/PSK are given in [26], [27], but they ignore the channel coding gain. As the channel gain is hard to be captured analytically, following the existing literature, in Sec. VI we measure the BER of hierarchical modulation under different SNR values via numerical simulation.

Next, assume the decoding processes of different layers to be independent and that the bit error process within a packet is Bernoulli distributed. Then, we can obtain the PER when decoding the sub-packets at a specific set of layers, with the set of layers being determined by the stem, branch selection, and layer allocation strategy. Let $\boldsymbol{\pi}(t)=<K,\boldsymbol{\pi}^s(t),\boldsymbol{\pi}^h(t),\boldsymbol{\pi}^b(t)>$ be the joint communication and processing strategy in the slot t, and $\zeta^p(t)$ the PER for a given $\boldsymbol{\pi}(t)$. Then, the PER for the sub-packets at the layers specified by $\boldsymbol{\pi}(t)$ is $\zeta^p(t)=1-\prod_{k\in K}(1-\zeta_k^b(t))^{D_k(t)}$.

Inference Accuracy. In object detection for remote-assisted autonomous driving, accuracy, denoted by $\xi(t)$, refers to the mAP in detecting and recognizing targets in the surrounding environment based on sensor fusion. As mentioned in Sec. III, we assume an error-free system. For a given stem/branch selection strategy, the inference accuracy is always the same if all the packets are received without errors. Thus, $\xi(t)$ can be obtained in advance by measuring the inference accuracy of each possible stem/branch selection strategy.

B. The DeFuSe Optimization Problem

We define an optimization problem, called DeFuSe (Distributed Fusion of Sensor data), that aims at efficient sensor data transmission and sensor fusion by minimizing the weighted sum of delay and energy consumption, under the system and accuracy constraints. Pivotal to achieving this goal is the joint optimization of (i) the sensors to use for data transmission and processing, which also determines the number K of hierarchical transmission layers; (ii) the stem selection strategy $\boldsymbol{\pi}^s(t)$, (iii) the hierarchical layer allocation strategy $\boldsymbol{\pi}^h(t)$, and (iv) the branch selection strategy $\boldsymbol{\pi}^b(t)$. Such decisions in the slot t are represented by the joint strategy $\boldsymbol{\pi}(t)=<K,\boldsymbol{\pi}^s(t),\boldsymbol{\pi}^h(t),\boldsymbol{\pi}^b(t)>$. Let $\boldsymbol{\pi}=\{\boldsymbol{\pi}(t),\forall t\in\mathcal{T}\}$ be the joint strategies in T slots; then we formulate the problem as:

The DeFuSe problem

$$\mathrm{P1}: \quad \min_{\boldsymbol{\pi}} \frac{1}{T}\sum_{t=1}^{T}\left[\kappa_1 T^a(t)+\kappa_2 E^a(t)\right] \tag{1a}$$

$$s.t \quad \xi(t)\geq\xi^{th}(t),\forall t\in\mathcal{T}, \tag{1b}$$

$$\pi_n^s(t)\in\{0,1\},\forall n\in\mathcal{N},\forall t\in\mathcal{T}, \tag{1c}$$

$$\sum_{n\in\mathcal{N}}\pi_n^s(t)=K,\forall t\in\mathcal{T}, \tag{1d}$$

$$\pi_k^h(t)\in\widetilde{\mathcal{N}},\forall k\in\mathcal{K},\forall t\in\mathcal{T}, \tag{1e}$$

$$\pi_k^h(t)\neq\pi_{k'}^h(t),k\neq k',\forall k,k'\in\mathcal{K},\forall t\in\mathcal{T}, \tag{1f}$$

$$\pi_\omega^b(t)\in\{0,1\},\forall\omega\in\Omega,\forall t\in\mathcal{T}. \tag{1g}$$

$$\pi_\omega^b(t)=0,\ \text{if } f_\omega^n=0 \text{ or } \pi_n^s(t)=0, \tag{1h}$$

$$\forall n\in\mathcal{N},\forall\omega\in\Omega,\forall t\in\mathcal{T}.$$

(1b) imposes that the accuracy in each slot does not drop below the target value, while (1c) to (1h) represent the feasible domain of the three decision variables. Theorem 1 below proves that the problem is NP-hard.

Theorem 1. *The DeFuSe problem is NP-hard.*

Proof. DeFuSe is a non-linear integer problem with coupled variables. The number of selected sensors K, the stem selection $\boldsymbol{\pi}^s(t)$, hierarchical layer allocation $\boldsymbol{\pi}^h(t)$, and branch selection $\boldsymbol{\pi}^b(t)$ are interdependent and influential. The selected stem also determines the set of the branches to be considered. Since features with different modalities have different data size, the layer allocation will directly affect the BER performance and, hence, the number of packet retransmissions. Given the chosen branch and layer allocation, the remaining optimization (i.e., the stem selection) can be regarded as a generalized assignment problem (GAP), which is known to be NP-hard. Thus, since a GAP instance can be reduced to a DeFuSe instance in polynomial time, the thesis is proven. □

V. Optimization Methodology

To efficiently solve the DeFuSe problem, HERACLES adopts a deep reinforcement learning (DRL) approach that offers several advantages over traditional iteration-based online optimization methods. *First*, DRL provides adaptive decision-making, allowing HERACLES to adjust the strategies $\boldsymbol{\pi}(t)$ to contexts and channel conditions. *Second*, DRL can use pre-trained models for rapid real-time decision-making, which is particularly suitable for delay-sensitive applications. Below, we detail the proposed DRL-based optimization scheme, be-

ginning with the Markov decision process (MDP) formulation of the problem, followed by the DRL optimization.

A. Markov Decision Process Formulation

In HERACLES, we assume that the DRL-based optimization engine is deployed at the edge, where a DRL agent will be trained and is responsible for selecting the transmission and computing strategies $\boldsymbol{\pi}(t)$ at each time t. The decision-making process for the joint strategy $\boldsymbol{\pi}(t)$ is formulated as an MDP with elements $\{\mathcal{S}, \mathcal{A}, \mathcal{R}, \mathcal{P}, \gamma\}$. $\mathcal{S}=\{\boldsymbol{s}(t), \forall t \in \mathcal{T}\}$ is the state set of HERACLES, $\mathcal{A}=\{\boldsymbol{a}(t), \forall t \in \mathcal{T}\}$ is the action set to determine the joint strategies, $\mathcal{P}=\{P(s(t+1)|s(t), a(t)), \forall t \in \mathcal{T}\}$ denotes the set of state transition probabilities capturing the system dynamics, $\mathcal{R}=\{r(t+1)|s(t), a(t), \forall t \in \mathcal{T}\}$ is the set of the actions' reward, and $\gamma \in [0, 1]$ is the discount factor for future rewards (the larger the γ the more long-term rewards are prioritized). At each slot t, the agent observes state $\boldsymbol{s}(t)$, performs action $a(t)$, transitions to state $s(t+1)$, and receives a reward $R(t)$. This process repeats until the reward stabilizes, indicating an optimal strategy. Next, we define the state, action, and reward spaces for the agent.

State Space. The system state, $\boldsymbol{s}(t)$, includes the estimated channel $\hat{h}(t)$, task context $\psi(t)$, and accuracy threshold $\xi^{th}(t)$, i.e., $\boldsymbol{s}(t) \triangleq \{\hat{h}(t), \psi(t), \xi^{th}(t)\}$. The estimated channel can be obtained via channel estimation feedback, while the task context and accuracy threshold can be obtained respectively from the sensor features and the application requirements.

Action Space. The agents action in time slot t is the joint discrete strategy $\boldsymbol{\pi}(t)$, which includes the number of sensors used for fusion (i.e., the number of layers for hierarchical communication) and the stem selection, layer allocation, and branch selection strategies. Thus, the action set in slot t is defined as $\boldsymbol{a}(t) \triangleq \{K, \boldsymbol{\pi}^s(t), \boldsymbol{\pi}^h(t), \boldsymbol{\pi}^b(t)\}$.

Reward Function. After executing the joint strategy, the agent at the edge calculates the reward based on the task performance. Consistently with the DeFuSe objective, at each slot t the reward function is given by the weighted sum of: 1) the delay penalty $T^a(t)$, 2) the energy consumption penalty $E^r(t)$, and 3) the accuracy constraint penalty $\mathrm{erf}(\xi(t) - \xi^{th}(t))$, where $\mathrm{erf}(\cdot)$ is the error function, i.e., $r(t) = -\kappa_1 T^a(t) - \kappa_2 E^r(t) + \kappa_3 \,\mathrm{erf}(\xi(t) - \xi^{th}(t))$, where κ_1, κ_2, and κ_3 are adjustable weights.

B. DRL-based Optimization Scheme Design

Next, we introduce the DRL optimization scheme based on the MDP formulated above. We adopt Rainbow DQN [28], a state-of-the-art DRL scheme for solving the MDP with discrete action space. Compared to traditional DQN, Rainbow DQN integrates the following six variants of DQN. Specifically, (1) Double DQN, which reduces overestimation bias by decomposing the target network into the action selection and action evaluation steps; (2) Prioritized Experience Replay, enhancing learning efficiency by allowing the agent to replay more important experiences more frequently; (3) Dueling DQN, which improves estimation of the state values and action advantages separately, leading to better policy evaluation; (4) Multi-step Learning utilizing temporal difference errors over multiple steps, thus providing a more stable and consistent target signal for the Q-value updates; (5) Distributional Learning accounting for the full distribution of possible reward outcomes, rather than just the expected value, enabling more nuanced decision-making under uncertainty; (6) Noisy Net, which increases the exploration ability of the model by adding noise to the NNs. Thus, Rainbow DQN can improve the training stability and robustness of the DRL optimization scheme, which is well-suited for solving the high-dimensional problem DeFuSe with discrete action spaces.

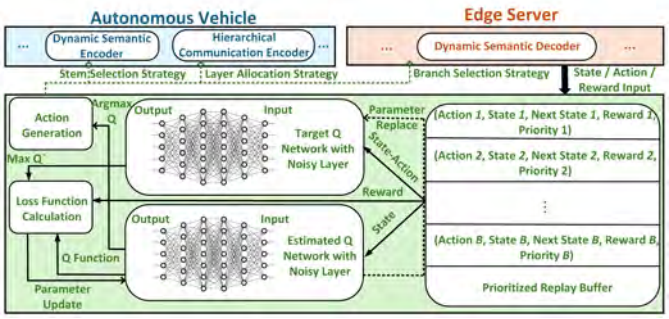

Fig. 5. DRL-based Optimization Engine of HERACLES.

We use Rainbow DQN to address the DeFuSe problem through a two-phase procedure: **Offline Training** and **Online Deployment**. The Rainbow DQN-based optimization engine we develop for solving the DeFuSe problem in HERACLES is illustrated in Fig. 5 and detailed below.

Offline Training. In this phase, the agent at the edge interacts with the vehicle through three steps to learn optimal decision-making strategies. *(i) Negotiation:* In each time slot, the edge server first negotiates with the vehicle to determine the computing and transmission strategy $\boldsymbol{\pi}(t)$. The vehicle sends the task context $\psi(t)$ and accuracy threshold $\xi^{th}(t)$ to the edge, then the agent estimate the wireless channel $\hat{h}(t)$ and obtain the state vector $\boldsymbol{s}(t) \triangleq \{\hat{h}(t), \psi(t), \xi^{th}(t)\}$. Next, the agent inputs the state to the Q-network and obtains the action $\boldsymbol{a}(t) \triangleq \{K, \boldsymbol{\pi}^s(t), \boldsymbol{\pi}^h(t), \boldsymbol{\pi}^b(t)\}$. Then the agent returns to the vehicle the stem selection strategy $K, \boldsymbol{\pi}^s(t)$ and layer allocation strategy $\boldsymbol{\pi}^h(t)$. Since the transmitted data size is relatively small, we ignore the negotiation overhead. *(ii) Execution:* The vehicle executes the semantic extraction and transmits semantic features according to the received strategies. The agent at the edge then performs sensor fusion and computes the reward $r(t)$, and it transitions to the next state $\boldsymbol{s}(t+1)$. These interactions, including state, action, reward, and next state, are stored in a prioritized replay buffer. *(iii) Training.* Once the buffer reaches its capacity, the agent samples mini-batches of experiences and updates its Q-networks using the J-step target Q-distribution. The Q-network is updated via gradient descent to minimize the temporal difference error, allowing the agent to accurately approximate optimal Q-values.

Online Deployment. The trained Rainbow DQN agent is now deployed in a real-world environment to make real-time decisions. This phase consists of two steps: *(i) Negotiation* and *(ii) Execution*, which are the same as in the Offline Training.

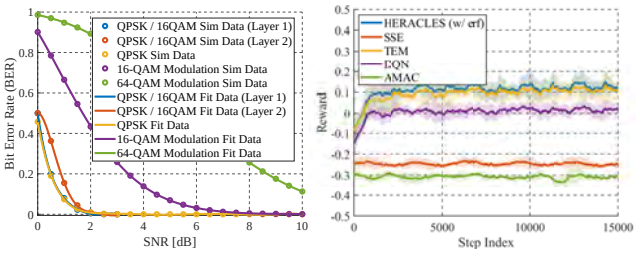

Fig. 6. BER performance of different modulations (left). Convergence performance of different schemes (right).

However, in this phase no further parameter update is required. Also, the agent can periodically adjust network parameters through online learning, ensuring robust performance in dynamic channel conditions or task contexts.

VI. Performance Evaluation

In this section, we evaluate HERACLES with real-world datasets. The experimental settings, our benchmarks, the evaluation results and analysis are introduced below.

A. Experimental Settings

Datasets. We use the RADIATE dataset [7], which has been introduced in Sec. I. For the wireless channel, we adopt a real-world 5G wireless dataset [29] collected from a major Irish mobile operator, which includes channel quality indicator (CQI) and SNR traces of a moving vehicle.

Dynamic NN Models. They are based on ResNet-18, ResNet-50, and ResNet-101 backbones [11]. We consider 3 types of stems per sensor and 21 branches for various fusion combinations and computing complexities. Also, the vehicle is equipped with an NVIDIA Jetson Nano platform, while the edge server computing power is 12× that of the vehicle. The RADIATE dataset is split with 70% for training and 30% for evaluation, as in [9]. For single-modal branches, data is processed directly with the ResNet backbone, while for multi-modal branches, multi-sensor features are first concatenated and fused via a 2D convolutional layer for early fusion.

Hierarchical Communications. To handle unreliable wireless channels in extreme weather, as in [20], the maximum number of layers for hierarchical modulation[2] is set to 2. It should be noted that HERACLES is also compatible with typical single-layer modulation, i.e., the DRL-based optimization engine may also choose one sensor only in some cases. Specifically, HERACLES adopts QPSK for $K{=}1$, and QPSK/16-QAM hierarchical modulations for $\tilde{N}{=}2$. The maximum transmit power p^m is set to 1 W, with a power ratio of $\frac{p_1(t)}{p_2(t)}{=}9$. LDPC coding with a fixed rate of $\frac{1}{2}$ is used for channel coding, and we set the system bandwidth $B{=}20$ MHz.

Rainbow DQN Agent. The agents target and estimated Q-networks include a 2D convolutional hidden layer. Advantage and value streams are processed by noisy-enhanced linear layers with hidden size 512, noise standard deviation of 0.1, and ReLU activation. The discount factor is 0.99. The prioritized replay buffer has a size 32, with proportional prioritization and an exponent of 0.5. Multi-step learning uses $J{=}3$, and distributional learning has 51 atoms with $V_{min}{=}-10$ and $V_{max}{=}10$. The Adam optimizer has a learning rate of $6.25{\times}10^{-5}$ and a hyperparameter of $1.5{\times}10^{-4}$. Batch size is 128, memory size 10,000, and target network updates occur every 100 steps. We train the model at SNR=2 dB, with κ_1, κ_2, and κ_3 set to 2, 1, and 1 (resp.).

[2]Increasing modulation layers offers diminishing transmission gain while significantly raising system complexity. Evaluation results also show that 2-layer hierarchical modulation is enough to ensure robust performance.

B. Benchmarks

We compare our approach to the following four baselines:

- **Adaptive Modulation and Coding (AMAC)** [19]: it uses static NNs to extract semantic features, and all features are sent to the edge for fusion via single-layer modulation. AMAC adopts exhaustive search to obtain optimal modulation type, order, and coding rate. The feasible transmission schemes are listed in [19, Table 1].
- **DQN** [30]: Conventional DQN is used to verify the effectiveness of the Rainbow DQN-based optimization algorithm in providing the joint strategies.
- **Static Semantic Encoding (SSE)**: To verify the effectiveness of the stem and branch selection, we consider a static semantic encoder where all the stems are activated and all the semantic features are transmitted based on the four-layer QPSK hierarchical modulation with a power ratio of 4:3:2:1.
- **Dynamic Sensor Fusion with Typical Modulation (TEM)**: To verify the effectiveness of the hierarchical modulation, we consider this baseline, in which semantic features are sent using one-layer 16-QAM.

Also, to assess the impact of the reward design, we implement an alternative version of HERACLES with linear reward: $r(t){=}\kappa_1(T^m{-}T^a(t)){+}\kappa_2 E^r(t){+}\kappa_3(\xi(t){-}\xi^{th}(t))$. We distinguish between them by **"(w/ erf)"** and **"(w/o erf)"**.

C. Results and Analysis

BER Performance. Since deriving analytically the BER of the hierarchical modulation is challenging when considering the coding gain, we first show in Fig. 6(left) the BER performance of the various modulation schemes under different SNR values, obtained via simulation and fitting the obtained scatter points. The LDPC coding rate is 0.5 (for 64-QAM, the coding rate is $\frac{2}{3}$), and the channel estimation error parameter is 0.5. As observed, QPSK/16-QAM hierarchical modulation can achieve similar BER performance to conventional QPSK. However, hierarchical modulation can substantially increase the transmission rate since it superimposes multi-layer modulation symbols, thus reducing the size of the transferred data.

DRL Convergence. Fig. 6(right) shows the schemes' convergence. All DRL agents are trained in a "mixed" environment with contexts that change every 100 slots within each a 3,000-slot episode. The reward curves represent the average of 5 runs with different seeds, with the shaded area indicating the standard deviation. Once can observe that Rainbow DQN scheme converges rapidly, stabilizing within 10,000 steps.

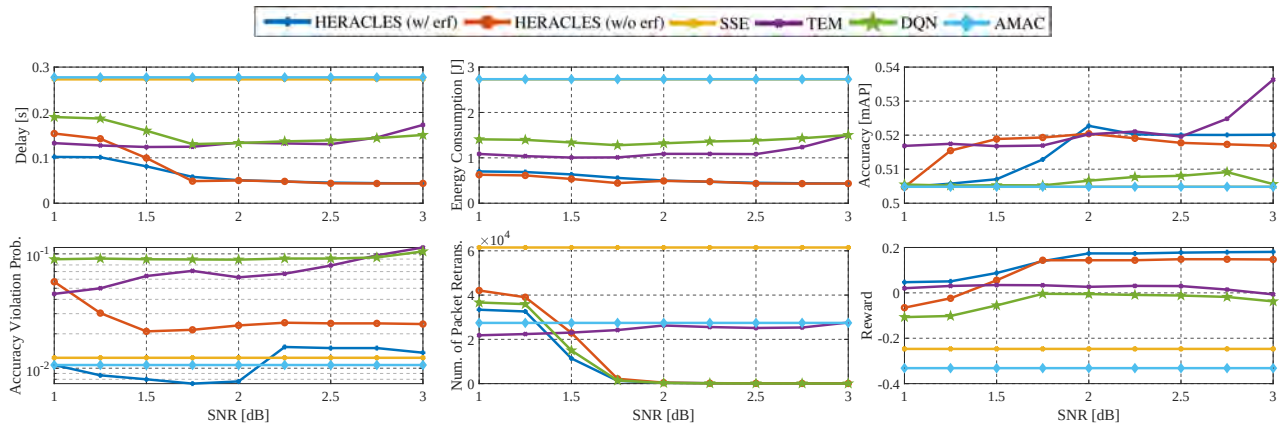

Fig. 7. Performance of different schemes under varying SNRs: delay (top-left), energy consumption (top-middle), accuracy (top-right), accuracy violation probability (bottom-left), number of packet retransmissions (bottom-middle), and reward (bottom-right).

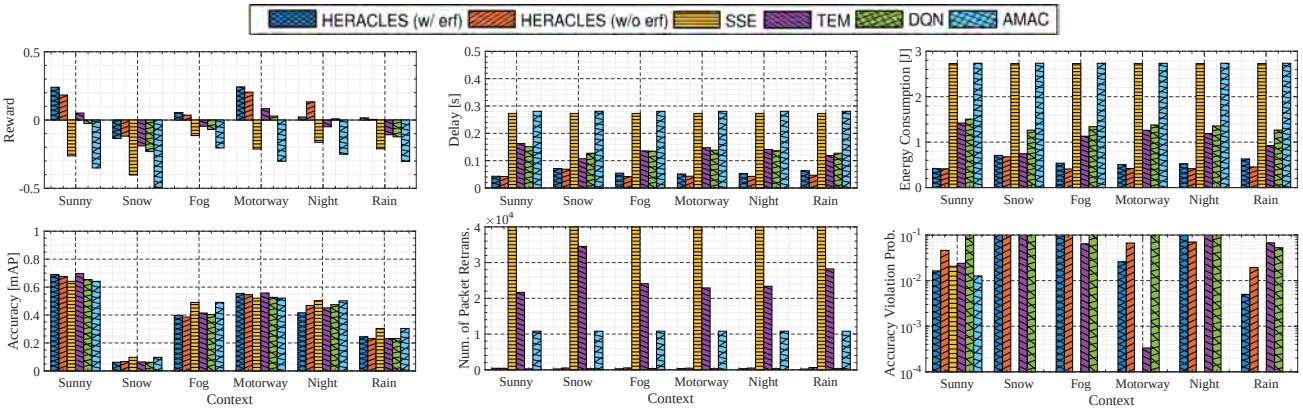

Fig. 8. Performance of different schemes under varying contexts: delay (top-left), energy consumption (top-middle), accuracy (top-right), accuracy violation probability (bottom-left), number of packet retransmissions (bottom-middle), and reward (bottom-right).

Also, compared to DQN, it achieves higher rewards, demonstrating that the Rainbow DQN technique effectively manage context switches and integrate experiences across diverse environments. Importantly, the proposed scheme with the erf function yields the highest average reward, indicating that the HERACLES framework, coupled with the erf-based reward design, reduces significantly delay and energy consumption while preserving accuracy.

Performance Under Different SNRs. We further analyze the impact of SNR values on key metrics, including average reward, delay, energy consumption, accuracy, accuracy violation probability, and number of retransmissions. Fig. VI-C highlights that, as the SNR increases from 1 dB to 2 dB, HERACLES yields a notable reward, hence overall performance, increase. As expected, higher SNR values result in reduced delay and energy consumption, as the algorithm leverages better channel conditions for reliable sensor feature transfer with fewer retransmissions. For TEM and HERACLES (w/o erf), an increased SNR also boosts inference accuracy, reducing accuracy violation probability as agents will be more inclined to choose actions with higher delay and energy consumption, but achieving higher accuracy. Thus, the reward first increases with the SNR and then tends to flatten or even slightly decrease. Also, although TEM can achieve optimal average accuracy performance in most of the cases, its accuracy violation probability is much higher than for HERACLES, indicating that our scheme can achieve much more stable accuracy performance. Unlike HERACLES, AMAC and SSE do not show significant gains with increasing SNR, indicating their inability to flexibly adapt the strategy to channel dynamics. These results underscore the robustness of HERACLES in balancing key metrics across different SNRs: it can reduce the average total delay and energy consumption by 20.39%–75.67% and 33.75%–72.34% (resp.), with at most 3.04% accuracy loss.

Performance Under Different Contexts. Next, we assess how the different schemes cope with dynamic, real-world applications where task contexts frequently change. Fig. 8 underlines HERACLES's robust adaptability, as it achieves optimal reward, delay, energy consumption, and retransmission metrics consistently across various contexts, while maintaining an accuracy level that matches that of other schemes. Specifically, HERACLES reduces the average total delay and energy consumption by 33.23%–73.5% and 4.86%–70.38% (resp.), with at most 7.85% accuracy loss. Notably, the DRL agents are trained in a mixed-context environment, which allows them to generalize across diverse contexts with no need for individual model adjustments. Although training separate mod-

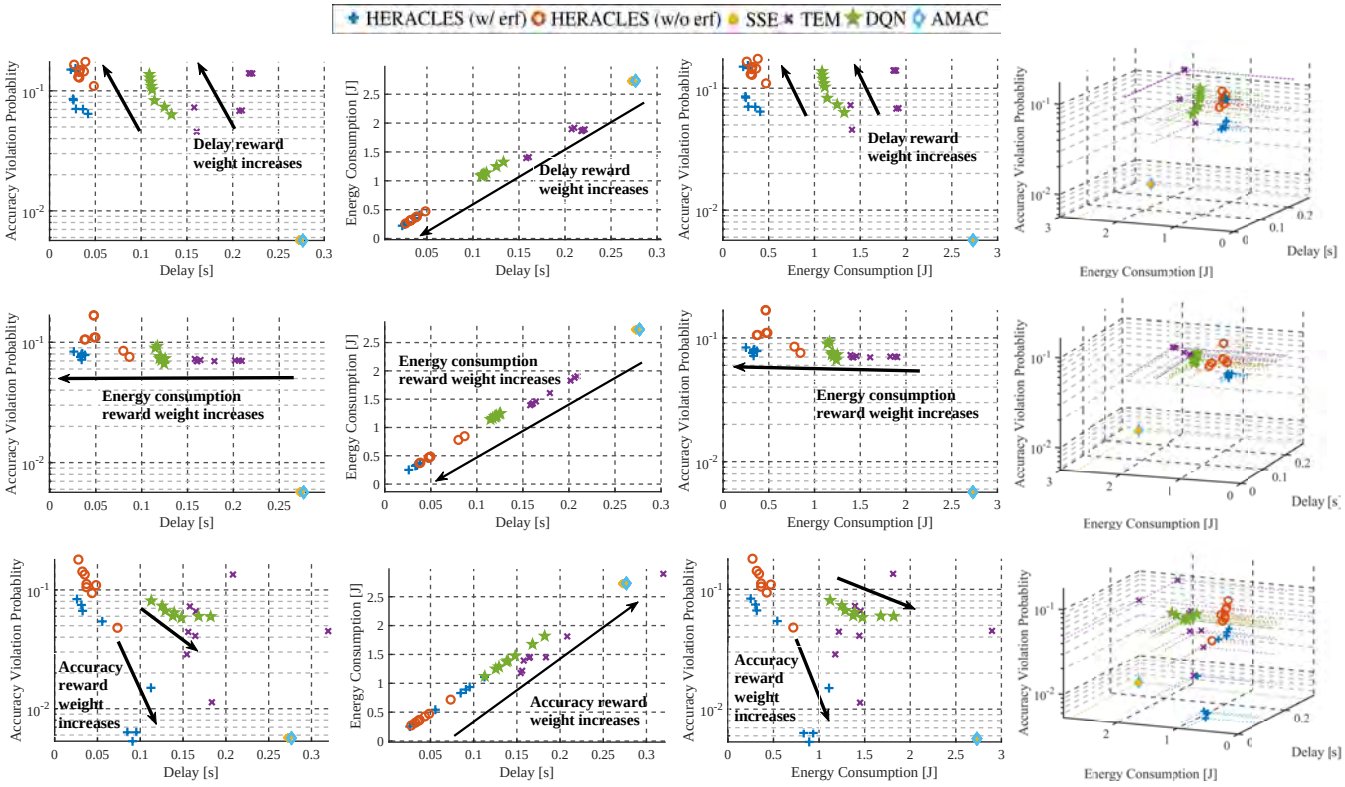

Fig. 9. Trade-offs offered by different schemes for varying reward weights: delay (top), energy consumption (middle) and accuracy (bottom). From left to right: delay vs. accuracy violation probability, delay vs. energy consumption, energy consumption vs. accuracy violation probability, and a 3D visualization of delay, energy consumption, and accuracy violation probability.

els for each context could potentially enhance performance, this approach would substantially increase training costs and model-switching overhead during online deployment. Thus, the mixed-context training strategy offers an efficient trade-off, providing high adaptability and stability across contexts while minimizing operational complexity.

Performance Comparison Under Different Reward Weights. Here, we explore how varying reward weights κ_1, κ_2, and κ_3 affect the behavior of DRL agents. Fig. 9 depicts the delay, energy consumption, and accuracy violation probabilities by different schemes trained under various reward weights ranging from 0.5 to 4 (with others fixed at 1 when one is varied). In each row, the first three subplots illustrate pairwise metric relationships, while the final subplot provides a 3D performance view. We can see that increasing delay or energy weight biases the agent toward sensor combinations with lower delay and energy consumption, which generally leads to higher accuracy violation probability. Compared to its benchamarks, HERACLES consistently achieves optimal performance, all metrics are closest to the origin in both 2D and 3D plots. Fig. 9 confirms the intuition that, as the accuracy weight increases, the agent favors sensor combinations with higher inference accuracy, requiring more delay and energy. For a given value of accuracy reward weight, HERACLES outperforms the other schemes in most cases in terms of all metrics, with the erf function scheme often being the best. Specifically, HERACLES reduces average total delay, energy consumption, and accuracy violation probability by 27.17%–89.41%, 5.79%–88.17%, and −3.07%–13.74% (resp.).

HERACLES Adaptive Strategy Orchestration. To demonstrate HERACLESs adaptive computing and transmission capability, we examine the actions selected by "HERACLES (w/ erf)" scheme under different channel conditions and contexts. Fig. 10 illustrates the probability of selecting various actions, along with their corresponding labels. The numbers between parentheses denote sensor indices: "1" and "2" for left and right cameras, "3" for radar, and "4" for LiDAR. Actions involving multiple sensors correspond to hierarchical modulation, with the order reflecting layer allocation (e.g., preceding numbers indicate the first layer). "Res" followed by a number signifies the ResNet backbone, where higher values indicate more complex models. The results show that HERACLES dynamically adjusts strategies very effectively in response to SNR and environmental context. As the SNR increases, the agent tends to select more complex sensor configurations, utilizing better channel conditions to improve data transmission and target accuracy. Additionally, cameras are more frequently selected in "Sunny" contexts compared to "Night." These results confirm the frameworks adaptability to real-time conditions, balancing accuracy, resource utilization, and responsiveness across various operational contexts. Since the agent is not specifically trained for every context, selected actions still exhibit some level of similarity across conditions.

Summary: We can summarize the key takeaways from the above experimental analysis as follows:

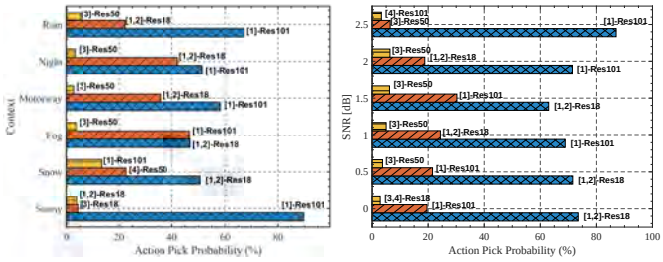

Fig. 10. HERACLES's action pick probability for different SNR and contexts.

• For distributed sensor fusion frameworks, channel conditions and task context play a critical role in influencing system latency, energy consumption, and inference accuracy. By adapting the orchestration strategy to channel conditions and task contexts, when compared to its benchmarks, HERACLES can reduce delay and energy consumption by $20.39\%-75.67\%$ and $33.75\% - 72.34\%$ (resp.) across different SNRs, and $33.23\%-73.5\%$ and $4.86\%-70.38\%$ across different contexts (resp.), while maintain near-optimal accuracy.

• For HERACLES's DRL-based optimization engine, the reward weights, SNRs, and contexts for training significantly influence the agents decision-making preferences. When DRL agents are trained under mixed contexts or a fixed SNR value, the results demonstrate a notable degree of generalization capability of DRL agents across varying contexts and SNRs.

• The optimal transmission and computation strategy varies noticeably with changes in channel conditions and task context (e.g., cameras are typically favored during the day, while radar is more reliable at night). Also, as channel conditions get better, HERACLES selects higher complexity models to enhance inference accuracy. Related to the dynamic NN, fusing data from both the camera and radar using ResNet-18 ensures superior performance in most of the cases.

VII. Conclusions and Future Work

We proposed HERACLES, a distributed sensor fusion framework that ensures dynamic sensor fusion and reliable transmission via semantic hierarchical communication. HERACLES deploys dynamic gated neural networks between mobile devices and edge servers: "stems" at the mobile device extract semantic features from sensor data, while edge-side "branches" process these features for hybrid sensor fusion. Also, HERACLES adopts hierarchical communication for reliable wireless transmission, where multi-modal semantic features are transmitted with different modulation methods and power levels (hence, offering variable error protection and improved transmission rates) according to their relevance. To enable optimal strategy orchestration in HERACLES, we first formulated an (NP-hard) optimization problem called DeFuSe, aiming at minimizing delay and energy consumption under accuracy constraints. We then proposed a Rainbow DQN-based scheme for efficient computing and communication. Our results, obtained using real-world datasets, demonstrate that HERACLES reduces delay and energy consumption up to 89.4% and 88.2% (resp.), while yielding near-optimal inference accuracy. Future work will focus on addressing the accuracy-constrained problem using advanced techniques such as constrained DRL and evaluating HERACLES through hardware testbed experiments.

References

[1] L. Liu *et al.*, "Computing systems for autonomous driving: State of the art and challenges," *IEEE Internet of Things Journal*, vol. 8, no. 8, 2021.

[2] Z. Wang *et al.*, "Multi-sensor fusion in automated driving: A survey," *IEEE Access*, vol. 8, 2019.

[3] Y. Fu *et al.*, "A cameraradar fusion method based on edge computing," in *IEEE EDGE*, 2020.

[4] Y. Luo *et al.*, "Towards high accuracy low latency real-time road information collection: An edge-assisted sensor fusion approach," in *MSN*, 2021.

[5] T. Shaowang *et al.*, "Sensor fusion on the edge: initial experiments in the edgeserve system," in *ACM BiDEDE*, 2022.

[6] G. Mujica *et al.*, "Edge and fog computing platform for data fusion of complex heterogeneous sensors," *Sensors*, vol. 18, no. 11, 2018.

[7] M. Sheeny *et al.*, "RADIATE: a radar dataset for automotive perception in bad weather," in *IEEE ICRA*, 2021.

[8] N. Rashid *et al.*, "SELF-CARE: selective fusion with context-aware low-power edge computing for stress detection," in *DCOSS*, 2022.

[9] A. V. Malawade *et al.*, "Hydrafusion: Context-aware selective sensor fusion for robust and efficient autonomous vehicle perception," in *ACM/IEEE ICCPS*, 2022.

[10] S. Lu, R. Zhong, and W. Shi, "Teleoperation technologies for enhancing connected and autonomous vehicles," in *IEEE MASS*, 2022.

[11] K. He *et al.*, "Deep residual learning for image recognition," in *IEEE CVPR*, 2016.

[12] C. Xing *et al.*, "Representation and fusion based on knowledge graph in multi-modal semantic communication," *IEEE Wir. Comm. Letters*, 2024.

[13] Y. He *et al.*, "Rate-adaptive coding mechanism for semantic communications with multi-modal data," *IEEE ToC*, 2024.

[14] G. Zhang *et al.*, "A unified multi-task semantic communication system for multimodal data," *IEEE ToC*, 2024.

[15] W. Wang *et al.*, "CA_DeepSC: Cross-modal alignment for multi-modal semantic communications," in *IEEE GLOBECOM*, 2023.

[16] X. Luo *et al.*, "Multimodal and multiuser semantic communications for channel-level information fusion," *IEEE Wir. Comm.*, 2024.

[17] H. Xie *et al.*, "Task-oriented multi-user semantic communications," *IEEE JSAC*, 2022.

[18] Y. Han *et al.*, "Dynamic neural networks: A survey," *IEEE TPAMI*, 2022.

[19] H. Gao *et al.*, "Adaptive modulation and retransmission scheme for semantic communication systems," *IEEE TCCN*, 2024.

[20] S. Wang and B. K. Yi, "Optimizing enhanced hierarchical modulations," in *IEEE GLOBECOM*, 2008.

[21] L. Qin *et al.*, "Joint transmission and resource optimization in NOMA-Assisted IoVT with mobile edge computing," *IEEE TVT*, 2024.

[22] H. S. Ghazi *et al.*, "Improved detection in successive interference cancellation NOMA OFDM receiver," *IEEE Access*, vol. 7, 2019.

[23] A. Neubeck and L. Van Gool, "Efficient non-maximum suppression," in *ICPR*, 2006.

[24] H. Qi *et al.*, "Paleo: A performance model for deep neural networks," in *ICLR*, 2017.

[25] S. Han *et al.*, "Deep compression: Compressing deep neural networks with pruning, trained quantization and huffman coding," arXiv:1510.00149 [cs], 2016.

[26] P. Vitthaladevuni and M.-S. Alouini, "BER computation of 4/M-QAM hierarchical constellations," *IEEE Trans. on Broad.*, 2001.

[27] P. Vitthaladevuni and M. Alouini, "A recursive algorithm for the exact BER computation of generalized hierarchical QAM constellations," *IEEE TIT*, 2003.

[28] M. Hessel *et al.*, "Rainbow: Combining improvements in deep reinforcement learning," arXiv:1710.02298 [cs], 2017.

[29] D. Raca *et al.*, "Beyond throughput, the next generation: a 5g dataset with channel and context metrics," in *ACM MMSys*, 2020.

[30] J. Lan *et al.*, "Improved Q-Learning-based motion control for basketball intelligent robots under multi-sensor data fusion," *IEEE Access*, 2024.