Omni-CNN: A Modality-agnostic Neural Network for mmWave Beam Selection

Batool Salehi, Debashri Roy, Tong Jian, Chris Dick, Stratis Ioannidis, and Kaushik Chowdhury

Abstract*—Vehicles today are equipped with different sensors, such as GPS, camera, and LiDAR. We propose Omni-CNN, a machine-learning (ML) based framework that accepts one or more of such available sensor inputs to speed up beam selection in vehicular networks, instead of performing an exhaustive search among all possible beams. Omni-CNN proposes a modality-agnostic approach, where a single shared model can accommodate any combination of these sensor inputs but with appropriate weight-selection masks specific to each modality. In Omni-CNN, we first use the residual capacity of the shared model to learn the predictive task for the current modality. Second, we propose an adaptive algorithm to calculate the required capacity on a per-layer basis and release the excess capacity for the next modalities. In our design, we use the gradients to identify the sparsity constraints that result in minimum capacity usage, while maintaining the accuracy. Third, given the sparsity constraints, we solve an optimization problem to select modality-specific sub-models using Alternating Directions-Method of Multipliers (ADMM) algorithm. Finally, we include decision level aggregation to handle scenarios, where more than one modality is present. Results on a challenging real-world dataset reveal that Omni-CNN reduces the overall model size by** 91.4%**, while achieving** 80.89% **accuracy in the prediction of the optimal beam. Furthermore, it reduces the beam selection overhead by** 99.37% **while retaining** 93.34% **of the throughput, compared to the 802.11ad standard.

***Index Terms*—multimodal learning, beam selection, vehicular networks, model pruning.**

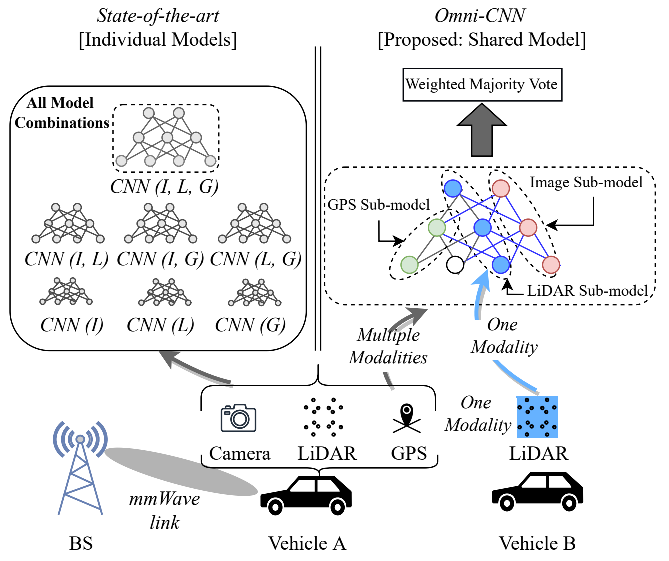

Fig. 1: State-of-the-art (left) versus proposed Omni-CNN framework (right). In the former, the models for all the modality combinations, seven models for GPS, LiDAR and image, must be on-board. In Omni-CNN, one shared modality-agnostic network is progressively learnt and the inference runs with any subset of present modalities.

I. Introduction

Vehicles today are equipped with sensors that continuously extract contextual knowledge of the environment. Such sensor data is used for autonomous driving [1], driver-assistance [2], road-safety [3], etc. Sensors typically relay large volumes of locally collected information to the cloud and may require data rates of tens of Gbps [4]. An option for enabling this high data-rate vehicle-to-infrastructure (V2I) connectivity is using the mmWave band [5], given the promise of wide (up to 2 GHz) channels.

A. Motivation for Sensor-guided mmWave Beam Selection

To combat the high attenuation in the mmWave band, a set of predefined directional *beams* are used to concentrate RF energy in narrow apertures [6]. Existing standards such as 802.11ad and 5G-NR require an exhaustive search through all beams. This involves bi-directional frame exchange with delays of up to tens of milliseconds [7], which is not feasible in dynamic scenarios where the optimum beam changes rapidly due to mobility. For example, the beam selection overhead in 5G-NR standard is ∼20ms for a set of 34 beams [8], [9]. On the other hand, the beam coherence time, the duration for which a beam is valid, is ∼100ms for a vehicle moving with the speed of 20mph. This considerable time overhead undermines the use of mmWave V2I links for two reasons. First, for higher vehicle speeds or narrower beams, the channel coherence time may be less than beam selection time, which can lead to a sub-optimal beam after an exhaustive search. Second, the more time vehicles spend on beam selection, the less time remains for data transmission. To speed up the

This work is supported by NSF AI Institute for Future Edge Networks and Distributed Intelligence (AI-EDGE) (grant CNS-2112471). The authors gratefully acknowledge Lopamudra Kundu from NVIDIA Inc for her constructive comments to shape this paper.

Batool Salehi, Tong Jian, Stratis Ioannidis, and Kaushik Chowdhury are with the Institute for the Wireless Internet of Things, Department of Electrical and Computer Engineering, Northeastern University, Boston, MA 02115 USA (e-mail: {bsalehihikouei, jian, ioannidis, krc}@ece.neu.edu) *(Corresponding author: Kaushik Chowdhury.)*.

Debashri Roy is with the Department of Computer Science and Engineering, University of Texas at Arlington, Arlington, TX 76019 USA (e-mail: debashri.roy@uta.edu).

Chris Dick is with the NVIDIA Inc, Santa Clara, CA 95051 (e-mail: cdick@nvidia.com).

beam selection, vehicle mounted sensors like GPS (Global Positioning System), cameras, and LiDAR (Light Detection and Ranging) can be used to identify the optimum mmWave beam [10]. Recent works use unimodal [11] or multimodal fusion-based [12], [13] deep learning (DL) architectures to infer the optimum beam in near real-time. These state-of-the-art methods demonstrate the beam selection accuracies with 56-88% for different scenarios and up to 95-96% improvement in beam selection speed over exhaustive search. While multimodal beam selection is a promising new approach, several key challenges below remain to be addressed.

B. Challenges in Multimodal Beam Selection

• **C1. Heterogeneity:** Lack of uniformity in vehicle-mounted sensors makes it difficult to train a common DL model for all vehicles. Fig. 1 shows our scenario of interest, where vehicles A (with image, LiDAR, and GPS) and B (with only a LiDAR) attempt to establish a mmWave link with a roadside base station (BS). A model that is tuned for vehicle A requires similar inputs from all three sensors GPS, image and LiDAR, and does not work for vehicle B as it only has a LiDAR sensor.
• **C2. Limited Storage:** The state-of-the-art DL methods for mmWave beam selection typically consist of over few million parameters, even for a single sensor modality [11], [14]. The parameter count further grows with fusion models that work with combinations of sensor inputs. Storing all possible models for all possible sensor combinations can quickly overflow limited memory, specially in resource constrained devices. A naive solution involving training seven distinct models for combinations of the camera, LiDAR, and GPS sensors is shown in Fig. 1, which is not extensible for general cases.
• **C3. Dynamic Deployment:** The state-of-the-art sensor fusion-based methods do not offer the flexibility in terms of extending a trained model, unless training is repeated from scratch with new sensor types as they become available. Thus, from Fig. 1, there is no simple way of taking the previously trained model for vehicle A and 'adding' inference capability for the previously unseen sensor in vehicle B. Moreover, fusion architectures designed for vehicle A cannot run the inference should an odd sensor fail unexpectedly. To address this, not only must all combinatorial models be stored, but also they must be switched, i.e., fetched and loaded to the memory to ensure the appropriate model is used for inference.
• **C4. Latency Requirements:** The permissible time-scale for collecting the data from multiple sensors and running inference is fairly small, in the order of few milliseconds. There may also be other local processes (emergency braking or lane change algorithms) that must share compute resources on vehicles with the beam selection models. Thus, we need to avoid performing excessive processing for communications, use simple inference models whenever possible and reserve computation power for other vehicle-centric tasks.

C. Omni-CNN: One Model for All Combinations of Modalities

The Omni-CNN framework directly addresses the above challenges by constructing a shared model that accommodates all combinations of sensors inputs. First, Omni-CNN selects the modality-specific disjoint sub-models within the larger shared model to run inference. It then performs decision level aggregation to account for combinations of sensors, thus ensuring the heterogeneity challenge (C1) is addressed. Second, since the model is shared among all modalities, only one model, instead of seven, must be stored on-board. This reduces the storage cost (C2). Third, since the discriminative quality of each sensor modality is acquired and contained independently in the shared model, Omni-CNN can be simply extended to new sensors without affecting the former modalities. This addresses the challenge of dynamic deployment (C3). This can also successfully handle sensor failures by triggering only the sub-models of functional sensors. Finally, in Omni-CNN, our innovative training approach ensures that each modality occupies the minimum model capacity, and thus, the computation cost, processing time and overhead are minimized (C4).

D. Summary of Contributions

Our main contributions are as follows:
(1) We propose Omni-CNN, the first of its kind modality-agnostic DL framework that predicts the optimum beam from any combination of GPS, camera, and LiDAR sensors. We use Alternating Directions-Method of Multipliers (ADMM) to select modality-specific sub-models from a shared model with minimal impact on prediction accuracy.
(2) We propose an algorithm to adaptively compute the required model capacity for each modality on a per-layer basis, regardless of the arrival sequence. Our design considers both loss reduction over training epochs computed using the model gradients and number of parameters to appropriately allocate sufficient capacity for each sensor modality. With Omni-CNN, the total number of used parameters decreases by 91.4% and 74.2% w.r.t. to using full model capacity and non-adaptive competing methods, respectively, with graceful degradation in accuracy. Moreover, we demonstrate that Omni-CNN is resilient to different arrival sequences of sensor modalities.
(3) We include decision level aggregation for handling multiple modalities and observe up to 4.08% improvement in accuracy compared to having a single modality.
(4) We compare Omni-CNN with the state-of-the-art methods involving deep learning for beam selection and demonstrate 50.39% and 89.24-95.65% improvement over prior works in accuracy and used model capacity, respectively. Moreover, We observe 99.37% improvement in beam selection speed over the 802.11ad standard while retaining 93.34% of the throughput.

II. Background and Related Works

A. Background on Conventional Exhaustive Search

The 802.11ad and 5G-NR standards use exhaustive search, wherein one of the nodes acts as the transmitter by sending out probe frames in all beams sequentially, while the receiver node listens to these frames with a quasi-omni-directional antenna setting. This process is then repeated with the transmitter-receiver roles reversed. For example, given a set of predefined transmitter beam codebooks of $C_{Tx} = \{t_1, \ldots, t_M\}$, the optimum beam is selected as:

$$t^* = \underset{1 \leq i \leq M}{\arg\max}\ y_{t_i}, \tag{1}$$

where y_{t_i} denotes the measured received signal strength at the receiver when the transmitter is configured at beam t_i. For example, in 5G-NR standard, the gNodeB sequentially transmits synchronization signals (SS) in each codebook element. The SS transmitted in a certain beam configuration is referred as the SS block, with multiple SS blocks from different beam configurations grouped into one SS burst [8]. The NR standard defines that the SS burst duration (T_{ssb}) is fixed to $5\,ms$, which is transmitted with a periodicity (T_p) of $20\,ms$ [9], [15]. In the mmWave band, a maximum of 32 SS blocks fit within a SS burst, which allows for 32 different beam pairs to be explored within one SS burst. As a result, the total time to explore M beams is expressed as:

$$T^{nr}(M) = T_p \times \left\lfloor \frac{M-1}{32} \right\rfloor + T_b\ (1 + (M-1) \bmod 32), \tag{2}$$

where $T_p = 20\,ms$ and $T_b = 5\,ms/32 = 156\,ns$ correspond to periodicity of SS bursts and required time to sweep one beam within a 5ms SS burst, respectively. We note that the beam selection time with exhaustive search (Eq. 1) for only 34 beams is $\sim$ 20ms. Moreover, the beam selection overhead scales with the number of beams in the codebook. As a result, it can not be timely completed in short contact times in vehicular networks, specially with a high number of beams.

B. Related Works

Model Pruning. Model pruning is a compression technique to reduce model size with acceptable accuracy degradation. The literature in model pruning can be categorized into *heuristic pruning algorithms* and *regularization-based pruning algorithms*. The former works on irregular, unstructured weight pruning, where arbitrary weights can be pruned. Han et al. [16] use an iterative algorithm to eliminate weights with small magnitude. Guo et al. [17] extended magnitude-based pruning to dynamically recover the pruned connections that are found to be important. Later, heuristic pruning algorithms have been generalized to the more hardware-friendly structured sparsity schemes. Dong et al. [18] present a Transformable Architecture Search (TAS), where the knowledge is transferred from the unpruned network to the pruned version. Yu et al. [19] define a "neuron importance score" and propagate this score to conduct the weight pruning process. Regularization-based pruning algorithms, on the other hand, have the unique advantage for dealing with structured pruning problems through group Lasso regularization [20]. Early works [21], [22] incorporate ℓ_1 or ℓ_2 regularization in loss function to solve filter/channel pruning problems. Recently, many articles are dedicated to pruning using regularization penalties [20], [23], [24] by incorporating the advanced optimization solution framework of ADMM. Leveraging the ability of producing sparse networks, weight pruning naturally fits in the need of modality-agnostic methods that wish to accumulate knowledge from a multitude of modalities [25], [26], [27]. We use the state-of-the-art ADMM pruning and propose a modality agnostic beam selection algorithm that addresses the challenges of heterogeneity, minimizing the storage, dynamic deployment, and minimizing the latency.

Paper	Heterogeneity	Minimize Storage	Dynamic Deployment	Minimize Latency	Validation on Real-world Data
Chen *et al.* [32]	✓	✗	✗	✗	✓
Liu *et al.* [33]	✓	✗	✗	✗	✓
Girdhar *et al.* [34]	✓	✗	✗	✗	✓
Singh *et al.* [35]	✓	✗	✗	✗	✓
Va *et al.* [11]	✗	✗	✗	✗	✗
Wang *et al.* [36]	✗	✗	✗	✗	✗
Alrabeiah *et al.* [37]	✗	✗	✗	✗	✗
Klautau *et al.* [14]	✗	✗	✗	✗	✗
Dias *et al.* [38]	✗	✗	✗	✗	✗
Zecchin *et al.* [13]	✗	✗	✗	✗	✗
Xu *et al.* [39]	✗	✗	✗	✗	✗
Reus-Muns *et al.* [40]	✗	✗	✗	✗	✗
Salehi *et al.* [15]	✗	✗	✗	✗	✓
Wang *et al.* [25]	✓	✗	✓	✗	✗
Omni-CNN	✓	✓	✓	✓	✓

TABLE I: Advantages of Omni-CNN over the state-of-the-art on multimodal learning methods and beam selection.

Multimodal Methods. Using sensor data such as GPS [28], camera [29], and LiDAR [30] are studied in IoT applications [31] in both unimodal and multimodal settings. Chen *et al.* [32] propose a framework to fuse multimodal data with different incompleteness patterns using the same architecture. While this method addresses the challenge of data heterogeneity, it fails in dynamic deployment scenarios. For example when a new modality becomes available, the proposed method must be trained from scratch to accommodate the new modality. Moreover, there are no constraints on minimizing the model size, which suggests that the proposed method might not be efficient from the storage and latency perspectives. Similarly, Liu *et al.* [33] propose to divide the incomplete multimodal data into several groups, where each of these groups contains complete data only. The authors then construct a hypergraph to model the relationships among instances in each group and learn the classification task. Girdhar *et al.* [34] and Singh *et al.* [35] propose using transformer-based architectures to train a single model on classification tasks from different modalities. However, they confront the same limitations as [32].

Beam Selection via Sensor Data. Several works perform beam selection on a single modality. Va *et al.* [11] and Wang *et al.* [36] exploit the location of vehicles while Alrabeiah *et al.* [37] use images and Klautau *et al.* [14] and Dias *et al.* [38] use LiDAR data to infer the optimum beam, exhibiting accuracies of up to 45% and inference time of $<$ 1ms. Multimodal methods design fusion architectures to combine information from multiple sensors [13], [39] with reported accuracies of up to 56-88% and slightly higher inference time due to complicated fusion designs. Muns *et al.* [40] use GPS and images to speed up the beam selection, while Salehi *et al.* [15] extend the sensor suite to GPS, image and LiDAR. However, these designs are based on strong assumptions, requiring all vehicles to have the same three sensors onboard and do not offer a general solution to allow inputs from different combinations of sensors.

Novelty over Prior Works. Nevertheless, the state-of-the-art methods fail to adaptively allocate the capacity to modalities and do not study the coexistence of multiple modalities. Moreover, unlike Omni-CNN, they do not address the challenges of heterogeneity, minimizing the storage, dynamic deployment, and minimizing the latency, simultaneously. We use as the starting point the Learn-Prune method by [25] and its supporting ADMM-based pruning strategy. In this paper, we integrate a novel, adaptive sparsity identifying scheme to allocate the necessary model weights for each sensor modality on a per-layer basis. Moreover, we study the coexistence of modalities and include an aggregation scheme to improve the accuracy. Finally, we demonstrate the first deployment of such modality-agnostic method for a challenging real-world wireless communication problem, i.e., multimodal beam selection with three different modalities. Table I highlights the unique features of Omni-CNN compared with the state-of-the-art for multimodal beam selection.

III. System Model and Problem Formulation

In this section, we first present our system model. We then formally state the problem we address in Omni-CNN.

A. System Model and Modality-agnostic Property

Instead of time-consuming exhaustive search prescribed by the standard, Omni-CNN exploits the inputs from any combinations of GPS, image and LiDAR sensors to infer the optimum beam and activate the mmWave link. In this paper, we use the above sensors as: (a) they provide geometrical information of the environment, which are relevant to the multimodal beamforming task and (b) they are typically present in modern vehicles. Nevertheless, our proposed method can be applied to any number or types of sensors.

Offline Training. In the training phase, we assume that the data from GPS, image, LiDAR and ground-truth are available for offline training. Omni-CNN progressively accumulates the knowledge from each sensor modality in a shared model. In Omni-CNN, the modalities can themselves arrive in any possible sequence. As an example, consider a scenario, where vehicle B in Fig. 1 is the first modality. In Omni-CNN framework, first, the data collected by LiDAR is used to train the shared model on the entire capacity. Second, Omni-CNN adaptively identifies the required model capacity for this modality. Third, the proposed ADMM pruning algorithm in Omni-CNN selects a modality-specific sub-model for LiDAR according to the target density and releases the remaining capacity. When the next modality arrives, say image, the same process repeats but on the residual capacity of the shared model, with no change to the sub-model of LiDAR. This process repeats for GPS as well. At the end of the training phase, a shared-model is obtained for GPS, image, and LiDAR sensors with modality-specific masks denoting the weights that must be used for each. In Omni-CNN, the residual capacity of the shared model is used to obtain a sub-model for the current sensor modality. Thus, the performance will be affected, if the shared model does not have enough residual capacity to accommodate the incoming sensor modalities.

Online Inference. At inference phase, since the overall shared model accepts all sensor modalities, the BS broadcasts only one model over the downlink channel that works for any future vehicle, regardless of there is only one or more of these previously encountered sensors mounted on them.

Modality-agnostic property. If a vehicle is equipped with only one of GPS, image and LiDAR sensors, the online inference happens only once using the sub-model of that specific modality. On the other hand, if a vehicle is mounted with more than one sensor, Omni-CNN runs independent inferences for each and exploits decision level aggregation to combine the predictions from all the present modalities. As a result, the shared model, along with modality-specific masks, can accommodate multiple sensor types independently and jointly, with minimum consumption of the model parameters and storage. When a new unseen modality arrives, the vehicles submit the sensor recordings and ground-truth data to the BS. If the shared model has enough capacity to accommodate the incoming modality, the shared model will be extended to the new sensor. Note that we do not need to re-learn the previous modalities since the sub-models are disjoint and learning the new modality will not affect previous training, subject to having enough residual capacity in the shared model.

B. Problem Formulation

Notation. We denote the dataset of sensor recordings from GPS, image and LiDAR obtained from the vehicles as $D_{\mathtt{C}} = \{X_{\mathtt{C}}, Y_{\mathtt{C}}\}$, $D_{\mathtt{I}} = \{X_{\mathtt{I}}, Y_{\mathtt{I}}\}$, $D_{\mathtt{L}} = \{X_{\mathtt{L}}, Y_{\mathtt{L}}\}$, with $X_{\mathtt{C}} \in \mathbb{R}^{N_t \times d_0^{\mathtt{C}} \times d_1^{\mathtt{C}}}$, $X_{\mathtt{I}} \in \mathbb{R}^{N_t \times d_0^{\mathtt{I}} \times d_1^{\mathtt{I}} \times d_2^{\mathtt{I}}}$, $X_{\mathtt{L}} \in \mathbb{R}^{N_t \times d_0^{\mathtt{L}} \times d_1^{\mathtt{L}} \times d_2^{\mathtt{L}}}$ being the data matrix of each, respectively. N_t is the number of training samples. Here, GPS has 2 elements: latitude and longitude, while the image has the dimensionality of $(d_0^{\mathtt{I}} \times d_1^{\mathtt{I}} \times d_2^{\mathtt{I}})$, with $d_2^{\mathtt{I}} = 3$ for RGB images. For LiDAR, we map the point clouds to a 3D quantized representation using [14] with dimensions $(d_0^{\mathtt{L}} \times d_1^{\mathtt{L}} \times d_2^{\mathtt{L}})$. On the other hand, $Y_{\mathtt{C}}, Y_{\mathtt{I}}, Y_{\mathtt{L}} \in \{0,1\}^{N_t \times M}$ denote the label matrix presenting the one-hot encoding of M beams for each modality, where the optimum beam is set to 1 and the rest are set to 0 as according to Eq. (1).

Problem Statement. Our goal is to learn a shared neural network f_W parameterized by weights $W \in R^c$ that accepts inputs from any (or all) of the these modalities. Here, R^c denotes the entire capacity of the network. Omni-CNN allows extracting disjoint sub-models for each modality from the overall shared model f_W. We refer to the weights associated with each sub-model as the modality-specific weights, denoted by $W^m,\ m = \{\mathtt{C}, \mathtt{I}, \mathtt{L}\}$. The sub-models are acquired by applying the modality-specific mask M^m on the shared model, i.e., $W^m = W \odot M^m$, with $\odot$ being the element-wise product.

We use the subscript $l \in \{1, \ldots, N\}$ to indicate layers of an N-layer CNN. We represent each layer via the matrix $W_l \in \mathbb{R}^{P_l \times Q_l}$ and the bias vector $b_l \in \mathbb{R}^{P_l}$. We also define $W := \{W_l\}_{l=1}^{N}$ and $b := \{b_l\}_{l=1}^{N}$ as the set of all weights and biases in the entire CNN. The loss of the each sub-model for dataset $\mathcal{D}^m,\ m = \{\mathtt{C}, \mathtt{I}, \mathtt{L}\}$ is calculated by $\mathcal{L}(W^m, b; \mathcal{D}^m)$, where

W^m denotes the modality-specific weights. In this case, the sub-model selection problem is formulated as:

$$\underset{W^m}{\text{Minimize:}} \quad \mathcal{L}(W^m, b; \mathcal{D}^m), \tag{3a}$$
$$\text{s.t} \quad W_l^m \in S_{l,m}(\alpha_l^m), \tag{3b}$$
$$\sum_m \alpha_l^m \leq 1, \tag{3c}$$
$$\text{supp}(W^m) \cap \text{supp}(W^{\bar{m}}) = \emptyset, \tag{3d}$$
$$W^m \in R^c, \tag{3e}$$

where $S_{l,m}$ denotes a fixed sparsity constraint parameterized by a sparsity constant $\alpha_l^m \in \mathbb{N}$, while W_l^m gives the specific weights for modality $m = \{\texttt{C}, \texttt{I}, \texttt{L}\}$ and layer $l = \{1, \cdots, N\}$. The constraint in Eq. (3b) enforces the support of W_l^m bounded by α_l^m. The constraint in Eq. (3c) allocates portion of a layer for each modality. Finally, with $\bar{m}$ denoting the set of previously seen modalities, constraint in Eq. (3d) ensures that the sub-models are disjoint.

C. Design Goals for the Modality-agnostic Neural Network

• **Adaptive Sparsity Constraints:** The sparsity constraints ($S_{l,m}$ in Eq. (3)) must be set adaptively on a per-layer basis and resilient to different modality arrival sequences (e.g., LiDAR could be first followed by images, and vice versa) to avoid unnecessary model occupation (see Sec. IV-B2).

• **Optimum Sub-model Selection:** Given the sparsity constraints $S_{l,m}$, we need a systematic approach for selecting a sub-model for each modality by solving Eq. (3). Omni-CNN addresses this challenge by employing the ADMM algorithm in Sec. IV-B3.

• **Coexistence of Modalities:** To handle all modality combinations, Omni-CNN uses decision level aggregation in Sec. IV-C; however, the optimum aggregation policy is not trivial.

IV. Detailed Description of Omni-CNN Design

We assume new sensor modalities become available in sequence and our goal is to identify disjoint sub-models for each of them. Ideally, these sub-models should (a) demonstrate prediction accuracy close to what is achieved if they were hypothetically using the full model capacity, and (b) utilize minimum model capacity. In Omni-CNN, we employ model pruning to remove less-important weights, while preserving accuracy. We then use the excess capacity, released by pruning, to learn the remaining sensor modalities.

A. Preprocessing: Homogenization with Zero Padding

The inputs to the shared model can have different dimensions. GPS data has two elements, latitude and longitude, which returns location. The RGB image data has three dimensions with their relative scales depending on the resolution of the camera. Raw LiDAR sensor data includes point clouds that measures the distance of objects with respect to the sensor location, but the dimensionality of the processed LiDAR data depends on the quantization level along each axis. For homogenization, we employ zero padding at the input layer to unify the shape of data before feeding that to the shared model. Given the sensor inputs having the shapes $(d_0^{\texttt{C}} \times d_1^{\texttt{C}})$, $(d_0^{\texttt{I}} \times d_1^{\texttt{I}} \times d_2^{\texttt{I}})$ and $(d_0^{\texttt{L}} \times d_1^{\texttt{L}} \times d_2^{\texttt{L}})$, we apply zero padding and generate homogenized data with shape $(\max(d_0^{\texttt{C}}, d_0^{\texttt{I}}, d_0^{\texttt{L}}) \times \max(d_1^{\texttt{C}}, d_1^{\texttt{I}}, d_1^{\texttt{L}}) \times \max(d_2^{\texttt{C}}, d_2^{\texttt{I}}, d_2^{\texttt{L}}))$. Note that for the GPS data, we add an entire channel with all zeros.

B. Omni-CNN at Training Phase

The Omni-CNN includes three steps at the training phase.

1) Train on Residual Capacity: Recall when a new modality becomes available, the previously encountered modalities already occupy a portion of the model, as shown by fixed weights in Fig. 2. We refer to the unoccupied portion of the shared model as the *residual capacity*. It follows that for the first modality, the entire CNN capacity is the residual capacity. Thus, in the first step of Omni-CNN, we use the entire residual capacity of the model to learn the predictive task for the current modality (see Fig. 2). We refer to the weights associated with modality m trained on residual capacity as $\mathbb{W}^m$. Moreover, since the biases are shared through all modalities, we only learn the biases for the first task.

2) Adaptive Sparsity Constraints: In Omni-CNN, different sensor modalities share the layers in the model. The allocated capacity to each sensor is identified by a sparsity constraint, which must be set on a per-layer basis. Coarsely setting sparsity constraints can impact Omni-CNN in two ways: (a) lowering the constraint may not leave sufficient model capacity to maintain the accuracy and (b) increasing the constraint may cause degradation in accuracy due to overfitting and increase computation and storage costs. In the absence of a systematic approach, identifying the optimum sparsity constraints $S_{l,m}$ requires offline trial-and-error and expert domain knowledge. Even worse, this hand-engineering must be repeated when a new sensor modality appears. To automatically identify the optimum sparsity constraints for each modality on a per-layer basis, we propose an algorithm that considers the gradients calculated over the residual model capacity.

Theorem 1. *The loss reduction over training epochs relates to the summation of the squared gradients over the residual model capacity i.e.,* $\mathcal{L}_{(\mathbb{W}^m)^e} - \mathcal{L}_{(\mathbb{W}^m)^{e-1}} \propto \sum_{r \in R} g^2_{(\mathbb{W}^m)^e}(r)$.

Proof. Over the training epochs, the residual model weights are computed by stochastic gradient descent (SGD) as:

$$(\mathbb{W}^m)^e = (\mathbb{W}^m)^{e-1} - \eta g_{(\mathbb{W}^m)^{e-1}} \odot \mathbb{M}^m, \tag{4}$$

where $(\mathbb{W}^m)^e$ denotes the residual weights in epoch e and η is the learning rate. Moreover, $g_{(\mathbb{W}^m)^{e-1}}$ is the gradient over residual capacity of $\mathbb{W}^m$ in epoch $e-1$, and $\mathbb{M}^m$ is a mask that enforces training only on the residual capacity, compared to the ultimate modality specific mask M^m, $||M^m||_0 \leq ||\mathbb{M}^m||_0$. We estimate the loss through the training epochs as:

$$\mathcal{L}_{(\mathbb{W}^m)^e} = \mathcal{L}_{(\mathbb{W}^m)^{e-1}} + \langle \nabla \mathcal{L}_{(\mathbb{W}^m)^{e-1}}, (\mathbb{W}^m)^e - (\mathbb{W}^m)^{e-1} \rangle \tag{5a}$$
$$= \mathcal{L}_{(\mathbb{W}^m)^{e-1}} - \eta \langle \nabla \mathcal{L}_{(\mathbb{W}^m)^{e-1}}, g_{(\mathbb{W}^m)^{e-1}} \odot \mathbb{M}^m \rangle \tag{5b}$$
$$= \mathcal{L}_{(\mathbb{W}^m)^{e-1}} - \eta || g_{(\mathbb{W}^m)^{e-1}} \odot \mathbb{M}^m ||^2. \tag{5c}$$

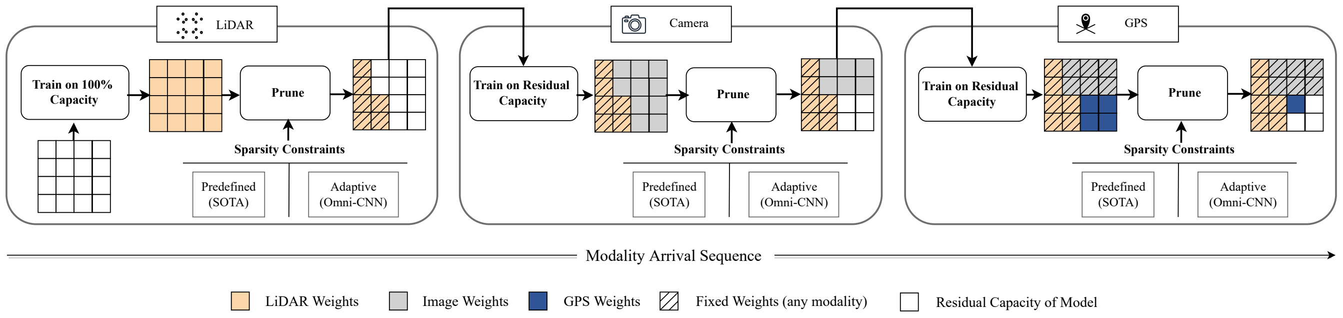

Fig. 2: Schematic of Omni-CNN framework at training phase for an example arrival sequence of LiDAR, image, and GPS sensors with three steps of *training on model capacity*, *adaptive sparsity constraints*, and *pruning*. Each modality is allowed to use only the residual capacity, while the modality-specific weights of prior modalities are fixed, shown with hatched blocks. Unlike state-of-the-art (SOTA) where the sparsity constraints are predefined, Omni-CNN discovers them adaptively using the gradients over residual capacity.

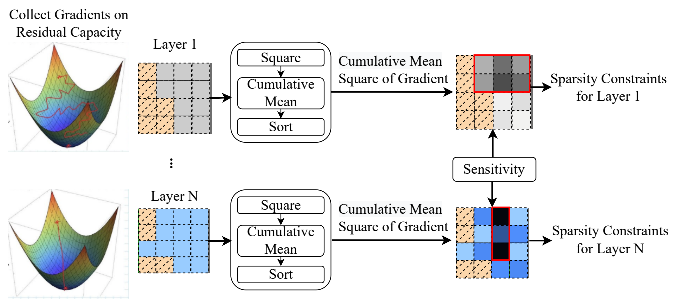
Collect Gradients on Residual Capacity
Layer 1
Layer N
Square
Cumulative Mean
Sort
Cumulative Mean Square of Gradient
Sensitivity
Sparsity Constraints for Layer 1
Sparsity Constraints for Layer N

Fig. 3: Adaptive sparsity constraints with Omni-CNN: we compute parameter Δ (Eq. (7)) to adaptively select the optimum sparsity constraint. Here, dark colored blocks depict higher values of Δ, which have lower probability of being pruned.

Here, Eq. (5a) is derived by estimating the loss with Taylor expansion, and Eq. (5b) is concluded from Eq. (4). Finally, Eq. (5c) is trivial obtained as $\nabla\mathcal{L}_{(\mathbb{W}^m)^{e-1}} = g_{(\mathbb{W}^m)^{e-1}}$. As a result, with R denoting the weights on the residual capacity, we conclude:

$$\mathcal{L}_{(\mathbb{W}^m)^e} - \mathcal{L}_{(\mathbb{W}^m)^{e-1}} \propto ||g_{(\mathbb{W}^m)^{e-1}} \odot M_{(\mathbb{W}^m)^{e-1}}||^2 \tag{6a}$$
$$= \sum_{r\in R} g^2_{(\mathbb{W}^m)^{e-1}}(r). \tag{6b}$$

□

On the other hand, increasing the number of parameters might result in overfitting and increase the computation cost. Thus, to account for both the loss and the number of model parameters, we define the parameter Δ that trades-off the loss with the model size. We calculate Δ as the cumulative mean square of the gradients. We compute this metric on the gradients derived in the last epoch, when the weights are finalized. Omitting the subscript e for simplicity:

$$\Delta(r) = \frac{\sum_{r\in R} g^2_{\mathbb{W}^m}(r)}{r}. \tag{7}$$

To compute Δ, we first calculate the gradients over the residual capacity $\mathbb{W}^m$ and sort them in decreasing order with respect to their absolute values. Next, we compute the cumulative mean square of the gradients as per Eq. (7). As a result, Δ

Algorithm 1: Omni-CNN Algorithm

```
Input: Any modality arrival sequence
Output: Sub-models per modality $W^m$
for $m \in Modalities$ do
    Learn: Train on residual capacity $\mathbb{W}^m$
    Adaptive sparsity constraint:
    Compute squared gradient on residual capacity $g^2_{\mathbb{W}^m}$
    for $l \in Layers$ do
        $r \leftarrow \arg\text{sort}\, g^2_{\mathbb{W}^m}$
        for $r \in R$ do
            Compute $\Delta(r)$ using Eq. (7)
        end
        $P \leftarrow \emptyset$
        $Range = max(\Delta) - min(\Delta)$
        for $j \in \Delta$ do
            while $\Delta(j) < min(\Delta) + \rho \times Range$ do
                $P = P + 1$
            end
        end
        $\alpha_l^m = \frac{P}{||\Delta||_0}$
    end
    Prune:
    Solve Eq. (3) using sparsity constraint $S_{l,m}(\alpha_l^m)$
end
```

is a decreasing function and the optimum sparsity constraint is satisfied upon convergence. We control the convergence via the parameter ρ that we refer to as *sensitivity*. Formally, convergence is achieved when Δ is within $\rho\times Range$ tolerance of the minimum value (see Alg. 1), where the $Range$ is the difference between maximum and minimum of Δ. We perform this operation on a per-layer basis and compute the optimum sparsity constraint $S_{l,m}$ as the fraction of the selected number of parameters over the entire parameter space of the layer (see Fig. 3).

3) Prune (Optimum Sub-model Selection): Given the weights trained on the residual capacity of the model (see Sec. IV-B1), i.e., $\mathbb{W}^m$, and the optimum sparsity constraints (see Sec. IV-B2), in the next step we select a sub-model W^m and release the excess capacity for the next modalities. This modality-specific sub-model is derived by solving Eq. (3). We observe that the optimization problem defined in Eq. (3) has combinatorial constraints, and cannot be solved via standard stochastic gradient descent. To deal with it, we exploit the ADMM algorithm suggested by Jian *et al.* [25] to identify the optimum sub-model for each modality.

The ADMM is a primal-dual algorithm designed for constrained optimization problems with decoupled objectives. With the help of an augmented Lagrangian, it alternates between (a) standard gradient descent with a quadratic proximal penalty, which forces the solution to be close to a value in the (non-convex) constraint space, and (b) an orthogonal projection operation to the constraint space. Hence starting from full weights $\mathbb{W}^m$ set to 1, we can progressively prune the weights, producing a feasible solution at convergence. From an implementation standpoint, to incorporate our constraints to ADMM, it suffices to produce polynomial-time functions that compute the orthogonal projection into constraint Eq. (3b), such as irregular, column, and filter pruning. For the Omni-CNN framework, following Jian *et al.* [25], we leverage irregular pruning, whereby sets $S_{l,m}$ are defined as:

$$S_{l,m} = \{\mathbb{W}_l^m \mid ||\mathbb{W}_l^m||_0 \leq \alpha_l^m\}, \quad (8)$$

where $||\cdot||_0$ denotes the number of non-zero elements, and $\alpha_l^m \in \mathbb{N}$ is a sparsity constant limiting the number of non-zero elements for modality m and layer l. Note that while the state-of-the-art method by Jian *et al.* solves Eq. (3) by imposing the same sparsity constraint to all layers, i.e., $S_{l,m} = S_{l',m} \ \forall\, l, l' \in \{1, \ldots, N\}$ and using the entire mode capacity $\sum_m \alpha_l^m = 1$, Omni-CNN relaxes these two assumptions by configuring the sparsity constraints on a per-layer basis and occupying as less as model capacity as possible; thus, granting an automated adaptive solution and minimum model occupation.

Omni-CNN Algorithmic Framework at Training Phase: Alg. 1 summarizes the working principle of Omni-CNN during training. We start with the training on residual capacity (see Sec. IV-B1) and obtain the weights $\mathbb{W}^m$. In the adaptive sparsity constraint configuration step, we compute the gradient on residual capacity and Δ in Eq. (7), and tune the convergence rate via sensitivity ρ. Finally, we plug in the derived sparsity constraint $S_{l,m}$ in Eq. (3) and use ADMM pruning (see Sec. IV-B3), to compute the sub-model weights W_l^m for each layer. The final sub-model is combination of individual sub-layers, $W^m = \{W_l^m\}_{l=1}^N$. Together, the three steps of training on residual capacity (Sec. IV-B1), adaptive sparsity constraints (Sec. IV-B2) and pruning (Sec. IV-B3) complete the Omni-CNN training phase. At the end of training phase, the modality specific sub-models $W^m \ \forall m \in \{\mathtt{C}, \mathtt{I}, \mathtt{L}\}$ are acquired for GPS, image and LiDAR modalities. The BS then transmits the final shared model to all vehicles in the downlink.

C. Omni-CNN at Inference Phase

At the end of training phase, the disjoint sub-models are obtained for each sensor modality. As a result, given the input samples of any of GPS, image, and LiDAR modalities,

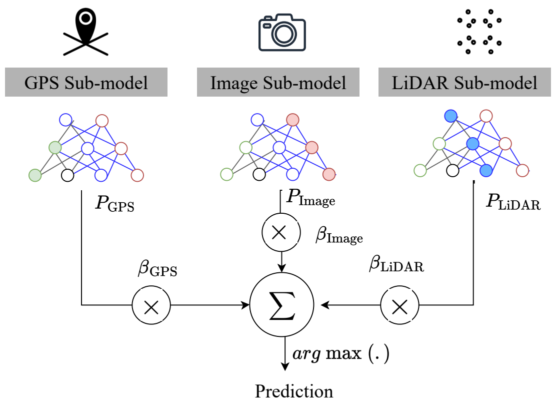

Fig. 4: At the inference phase, Omni-CNN uses the sub-models of available sensors, GPS, image and LiDAR in this example, to predict scores and then applies decision level aggregation for final prediction.

Omni-CNN generates individual prediction scores at the inference phase as:

$$P_m = f_{W^m}(X_m) \quad \forall m \in \{\mathtt{C}, \mathtt{I}, \mathtt{L}\}, \quad (9)$$

where f_{W^m} and X_m denote the sub-models and input data for modality m, respectively. Note that the cardinality of the prediction score P_m is equal to the number of beams in the codebook (i.e. $|P_m| = M$). We argue that the proposed Omni-CNN can be extended to account for a multitude of modalities for enhanced performance. Consider a scenario where a vehicle is equipped with more than one sensor modality, both GPS and LiDAR for instance. In this case, combining the knowledge from both modalities is expected to result in improved prediction accuracy as each sensor captures different information of the environment. For example, LiDAR data includes 3D mapping of the environment, while the precise location of the receiver is only achievable by GPS. Thus, taken together, different modalities provide a more comprehensive realization of the environment for the predictive task, compared to using each modality in isolation.

Coexistence of Modalities: We take advantage of decision level aggregation to reinforce the accuracy towards the final prediction, when more that one sensor modalities are present at the vehicles. We denote the combination of the available modalities by the set C. For example, if a car is equipped with both camera and LiDAR; then $C = \{\mathtt{I}, \mathtt{L}\}$. The decision aggregation policies of Omni-CNN include but are not limited to:

- **Majority vote:** Given the scores P_m for any modality combination $m \in C$, we select the class that is predicted by the majority of the modalities as:

$$t^* = mode\ \{i | \underset{i \in \{1,\ldots,M\}}{\arg\max}\ P_m(t_i)\ , \forall m \in C\} \quad (10)$$

  If the modality-specific outcomes do not agree on the wining class, i.e., each modality selects a distinct class for instance, we generate uniform probability distribution according to validation accuracy across the present modalities and select the winning class, accordingly. For example, in the case of $C = \{\mathtt{I}, \mathtt{L}\}$, we first compute the validation accuracies with the camera and LiDAR sensors as $v_{\mathtt{I}}$ and $v_{\mathtt{L}}$, respectively. We then generate uniform probability distribution for each sensor as:

$$\beta_{\mathtt{I}} = \frac{v_{\mathtt{I}}}{v_{\mathtt{I}} + v_{\mathtt{L}}} \quad , \quad \beta_{\mathtt{L}} = \frac{v_{\mathtt{L}}}{v_{\mathtt{I}} + v_{\mathtt{L}}} \tag{11}$$

Finally, we select the sensor modality to use for beam prediction considering $\beta_{\mathtt{I}}$ and $\beta_{\mathtt{L}}$ as the probability of camera and LiDAR sensors, respectively. We report the winning class as the one with the maximum prediction score from the selected sensor modality (see Eq. 10). Intuitively, by taking into account the validation accuracies, we select the sensor modality that is outperforming others, when the prediction scores results in selecting different classes.

- **Weighted majority vote:** We weight the prediction scores P_m by coefficient β_m for each sensor modality. We derive this coefficient by using validation accuracies as per Eq. 11. We select the winning class as the one that maximizes the weighted prediction score. Thus, more significant modalities have a higher impact on the final decision. Formally,

$$t^* = \{i | \operatorname*{arg\,max}_{i \in \{1,\ldots,M\}} \sum_{m \in C} \beta_m P_m(t_i)\}. \tag{12}$$

- **Multinomial logistic regression:** We use same equation as Eq. (12); however, the coefficients β_m are learnt through logistic regression [41]. In this regard, we train a logistic regression model with prediction probability of the present sensor modalities $P_m(t_i)$ ($m \in C$,$1 \leq i \leq M$) and ground-truth labels.

Omni-CNN Algorithmic Framework at Inference Phase: At the inference phase, if a vehicle has only one sensor mounted, it uses the associated sub-model and predicts the optimum beam. On the other hand, if a vehicle has more than one sensor modality installed, it performs independent inference runs for each and uses the aggregation polices in Sec. IV-C to combine the knowledge from all present modalities, see Fig. 4. In general, decision level aggregation i) improves the accuracy of predictions by combining the strengths of multiple modalities and reducing the impact of individual modality weaknesses ii) makes the predictions more robust by reducing the impact of outliers or errors in individual modality predictions [42]. On the other hand, unlike Omni-CNN, the state-of-the-art fusion methods [13], [15], [39] fail when being encountered with single or any other combination of the modalities than what they have been trained on. Taken together, the proposed decision level aggregation method i) ensures that the knowledge from all present modalities are combined towards the final prediction for boosted accuracy and robustness ii) without limiting the ability of the shared model to operate on any combination of the modalities. Note that in Omni-CNN disjoint sub-models are obtained from the shared model, which results in addressing the challenge of dynamic deployment (C3). As a result, there are no connections among different modalities. Nevertheless, Omni-CNN can be extended in the future works to account for modal interactivity [43], [44], modal redundancy [45], and data incompleteness [32], [33].

V. Evaluating Omni-CNN

A. Dataset, Experimental Setting and Evaluation Metrics

Dataset. We validate our proposed Omni-CNN framework on the publicly available FLASH dataset for multimodal beamforming [12], retrieved from the open-access repository [46]. The FLASH dataset studies a V2I scenario at 60GHz mmWave band and includes synchronized sensor data from on-board GPS, a GoPro Hero4 camera, and a Velodyne LiDAR, along with received signal strength indicator (RSSI) for all beams recorded by the Talon AD7200 mmWave radio [47]. The latitude and longitude of the vehicle and side view of the vehicle are recorded by the on-board GPS and RGB camera with shape (90, 160, 3), respectively, as the vehicle passes by a static BS. The LiDAR data is quantized into a 3D matrix with shape (20, 20, 20). The overall dataset has $\sim$32K samples, which we split into three available modalities, representing a distinct vehicle with one of GPS, image, and LiDAR sensors for cases with one sensor, and combinations of each for more.

Experiment Target. In Sec. V-B, we discard the adaptive sparsity constraint algorithm presented Sec. IV-B2. Instead, we study Omni-CNN, when the sparsity constraints are set by offline exploration. In Sec. V-C, we make a case on why we need to adaptively set the sparsity constraints for each modality on a per-layer basis. We then study the performance of Omni-CNN with the adaptive sparsity constraint algorithm presented in Sec. IV-B2, with respect to accuracy, model usage, and resilience to arrival sequences. In Sec. V-D, we provide results on coexistence of modalities based on the aggregation method presented in Sec. IV-C. In Sec. V-E, we benchmark Omni-CNN against having distinct models for each modality as well as the state-of-the-art on unified models and beam selection using sensor data. We study the overhead associated with sharing the models in the downlink with the vehicles in Sec. V-F. Finally, we compare Omni-CNN against exhaustive search with mmWave standard in Sec. V-G.

Experimental Setting and Data Preparation. We explore the effect of arrival sequences for the unimodal case and denote each experiment with the first letter of the sensors: e.g., ‘GIL’ represents the case where the arrival sequence is GPS, Image, and LiDAR, respectively. For our shared model, we use the ResNet [48] inspired LiDAR architecture previously released alongwith the FLASH dataset in [12]. This model has nine convolutional layers and three hidden layers. From the FLASH dataset, we use 90% and 10% of the data for training and testing, respectively. For a fair comparison, we fixed the randomness elements in our implementation to ensure the reproducibility of the experiments. Table II summarizes the dataset and setting in our experiments. These parameters are used throughout all experiments in this section. The source codes for our implementation are available in [49].

Term	Parameter	Information
Dataset	# Classes	34
	# Train samples per modality	28,636
	# Test samples per modality	3,287
Architecture	# Convolutional layers	9
	# Hidden layers	3
	# Params in Convolutional layers (M)	86,688
	# Params in Hidden layers (M)	1,589,248
Experiments	# Epochs	150
	Learning rate	0.001
	Optimizer	SGD

TABLE II: Dataset, architecture, and experiment setting summary.

Mod	GPS			Image			LiDAR		
Arr.	Omni-CNN (predefined sparsity constraint)								
	$S_{l,m}^1$	$S_{l,m}^2$	$S_{l,m}^3$	$S_{l,m}^1$	$S_{l,m}^2$	$S_{l,m}^3$	$S_{l,m}^1$	$S_{l,m}^2$	$S_{l,m}^3$
GIL	27.56	27.56	26.40	66.01	66.17	66.74	71.28	77.21	77.85
GLI	27.56	27.56	26.40	63.91	56.95	53.90	81.86	82.29	82.68
IGL	26.80	26.37	26.19	65.92	65.77	64.31	53.78	53.17	76.75
ILG	21.60	21.60	21.60	65.92	65.77	64.31	82.05	81.99	83.02
LGI	26.59	25.22	21.60	63.97	62.94	52.23	82.59	82.62	82.81
LIG	29.96	22.69	21.60	68.87	65.25	60.38	82.59	82.62	82.81
Range over. Arr.	21.60 - 29.96	21.60 - 27.56	21.60 - 26.40	63.91 - 68.87	56.95 - 66.17	52.23 - 66.74	53.78 - 82.59	53.17 - 82.62	76.75 - 82.81
Range over $S_{l,m}$	21.60-29.96			52.23-68.87			53.17-82.68		

TABLE III: Accuracy over six arrival sequences and three sparsity constrains with predefined sparsity constraints.

Evaluation Metrics. We use accuracy (Top-1) to evaluate the prediction performance. Moreover, we use model and layer usage metrics to assess the occupied capacity in each. With u and u_m, $m = \{\texttt{C}, \texttt{I}, \texttt{L}\}$ denoting the total number of parameters in the CNN and the number of parameters allocated to each modality, the modality-specific and total model usage are defined as $\frac{u_m}{u}$ and $\frac{\sum_m u_m}{u}$, respectively. Similar equations are used to compute the per-layer usage by considering the parameter space of each layer only.

B. Omni-CNN with Predefined Sparsity Constraints

In the first set of experiments, we consider a scenario where Eq. (3) is solved with a set of predefined sparsity constraints $S_{l,m}$. Here, the entire capacity of the shared model is used by three modalities, i.e., $\sum \alpha_l^m = 1$ in Eq. (3), and the same sparsity constraint is applied to all layers, as suggested by [25].

1) Effect of Sparsity Constraints: We evaluate the performance of Omni-CNN for three predefined sparsity constraints $S_{l,m}^1$={GPS:0.2, Image:0.5, LiDAR:0.3}, $S_{l,m}^2$={GPS:0.2, Image:0.4, LiDAR:0.4} and $S_{l,m}^3$={GPS:0.3, Image:0.2, LiDAR:0.5}. Thus, we gradually reduce the capacity of the image and allocate the excess capacity to either LiDAR or GPS. Results reveal that the prediction accuracy strongly depends on the sparsity constraints. For example, for sequence GIL, decreasing the capacity of the image barely affects the accuracy of GPS, while increasing the capacity of LiDAR improves its accuracy by 6.57% when changing the sparsity constraints from $S_{l,m}^1$ to $S_{l,m}^3$. On the other hand, for sequence GLI, the same modification in the sparsity constraint results in a drastic drop of 10.01% on the accuracy of the image, while the accuracy of the LiDAR improves by 0.821%.

Phenomenon 1. *With predefined sparsity constraints, beam selection accuracy varies between 21.60-29.96%, 52.23-68.87%, and 52.17-82.68% for GPS, image and LiDAR, respectively.*

2) Effect of Modality Arrival Sequence: Our proposed method is sequential, meaning that the modalities arrive successively and are learned progressively. From Table III, we observe that apart from the sparsity constraints, the accuracy is also affected by the arrival order, when the sparsity constraints are set non-adaptively.

Phenomenon 2. *For different arrival sequences under $S_{l,m}^1$, the accuracy of GPS, image and LiDAR varies in the range of 21.60-29.96%, 63.91-68.87%, 53.78-82.59%, respectively.*

Recommendation 1. *From this section, we observe that the predefined sparsity constraints scenarios are prone to performance degradation. This is because, in the absence of a systematic approach, the optimality of sparsity constraints is not guaranteed. This corroborates the importance of adaptive sparsity constraint algorithm proposed in Sec. IV-B2.*

C. Omni-CNN with Adaptive Sparsity Constraints

Using fixed sparsity constraints requires offline exploration to identify the optimum splits and suffers from degradation due to different arrival sequences. We show next how Omni-CNN addresses this limitation.

1) Need for Adaptive Sparsity Constraints: In Sec. IV-B2, we described that the sparsity constraints must be (a) adaptive to modalities and (b) defined on a per-layer basis. We validate this by studying the parameter Δ, i.e., cumulative mean square of the gradients, defined in Eq. (7). In Fig. 5a, we plot normalized Δ across three sensor modalities in the dataset and 12 layers of the model described in Table II. In this figure, the solid lines and error bars denote the average and range of Δ over all layers for each modalities. From Fig. 5a, we observe that the modalities have different convergence rates, with GPS being the fastest, which suggests that this modality requires less model capacity. Moreover, the range of error bars suggests that some layers are more significant than others.

Phenomenon 3. *The sparsity constraints $S_{l,m}$ must be allocated on a per-layer basis for each modality.*

2) Computing Adaptive Sparsity Constraints: In Fig. 5b, we show the layer usage percentage of all 12 model layers for three modalities with Omni-CNN. In this experiment, we consider the arrival sequence of 'GIL' where the sensitivity, ρ in Alg. 1 that controls the convergence condition, is set as 0.002. We observe that Omni-CNN adapts the layer usage of each modality. We also observe that GPS requires a small portion of the first convolutional layer. This is intuitive as this layer is responsible for capturing low-level features such as colour, edges, gradient orientation that are not included in GPS. Instead, image and LiDAR need much more capacity in the first layer. We note that using the same sparsity constraint

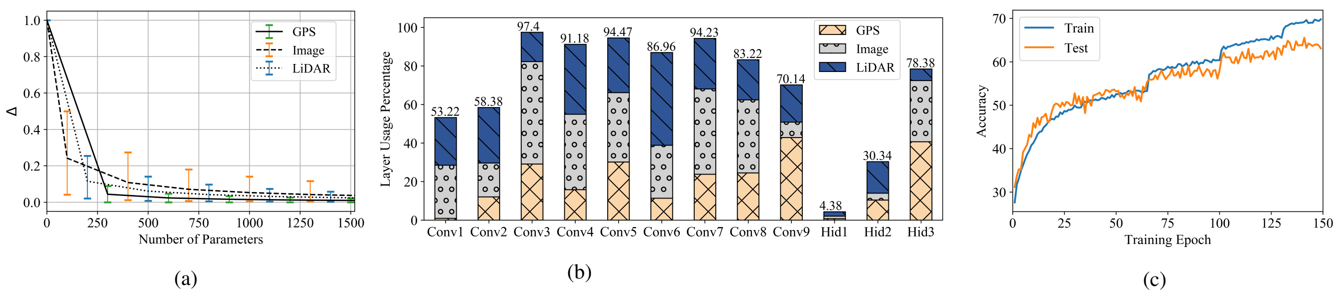

Fig. 5: (a) Average and range of normalized Δ over three modalities across 12 layers of the model. (b) Layer usage for arrival sequence of "GIL", where the optimum sparsity constraint is derived from Sec. IV-B2 on a per-layer basis. (c) Train and test accuracies for camera sensor, when the shared model is used at its full capacity. The shared model has higher number of parameters than customized model for camera. Thus, training the camera modality with this over-parameterized model might result in slight overfitting, which is corroborated from the results in Tab. VI. The same logic applies to the GPS sensor as well.

for all layers, as Sec. V-B, would result in unnecessary usage of this layer by GPS. Moreover, as we go deep in the network, the required capacity of the GPS increases for learning the task. As shown in this figure, some layers are more significant for each modality, `conv9` for GPS, `conv3` for image and `conv6` for LiDAR. Finally, we observe the minimum layer usage of 4.38% in the first hidden layer.

Phenomenon 4. *With Omni-CNN, we adaptively select the optimum sparsity constraints for each modalities across layers.*

3) Effect of Sensitivity on Accuracy and Model Usage: In Fig. 6, we evaluate the effect of sensitivity that controls the convergence condition in Omni-CNN (See Alg. 1). We consider three different arrival sequences of 'GIL', 'IGL', and 'LIG', and report the accuracy and model usage of GPS, image, and LiDAR sensor modalities, when the sparsity constraints are selected adaptively according to Sec. IV-B2.

Accuracy: As observed in the first column of Fig. 6, the higher values of sensitivity result in dropped accuracy due to extreme pruning. In contrast, decreasing the sensitivity improves the accuracy but increases the model usage percentage. We realize that the best performance across all three arrival sequences is achieved when the sensitivity is set to 0.002. In this case, the accuracy across three above arrival sequences ranges between 24.85-26.19%, 61.576-66.930%, and 76.42-77.79%. Compared to using predefined sparsity constraints as of Table III, where accuracy ranges are 21.60-29.96%, 52.23-68.87%, and 53.17-82.68%, we conclude that our proposed adaptive sparsity constraint algorithm selects the appropriate capacity for each sensor modality without offline exploration of different sparsity constraints.

Model Usage: In the second column of Fig. 6, we report the overall model usage, across all layers with Omni-CNN. We note that the first hidden layer has the highest (78% of total) number of parameters; thus weighting the total model usage over this layer. We observe that the maximum overall model usage of GPS, image, and LiDAR sensors for the lowest sensitivity is 4.84%, 10.05%, and 10.90%, respectively. Also, by focusing on the lower values of sensitivity, we note that the model usages are also adapted according to the modality arrival sequences, but stay within maximum deviation of 5.

Arr.	Accuracy (%)			Model Usage (%)		
	GPS	Image	LiDAR	GPS	Image	LiDAR
GIL	25.221	61.576	77.7	3.69	3.52	5.85
IGL	26.194	64.192	76.422	2.57	7.13	6.72
LIG	24.855	66.93	77.791	2.19	5.09	7.86

TABLE IV: Accuracy and model usage for three arrival sequences, where the sparsity constraints are set adaptively according to the algorithm proposed in Sec. IV-B2.

Resilience to Arrival Sequence: In Table. IV, we compare the accuracy and model usage for three arrival sequences of 'GIL', 'IGL', and 'LIG', where the sensitivity is set as 0.002 (which results in the best accuracy). From this table, we observe that accuracy for three sensor modalities stays within a close range, with the difference margin of 1.369-5.354% between the minimum and maximum over all modalities. Thus, our proposed method adaptively adjust the model usage for each modality to maintain the accuracy for different arrival sequences. Taken together, we conclude that our proposed adaptive sparsity constraint algorithm is resilient to different arrival sequences of the modalities.

Phenomenon 5. *Omni-CNN adapts the sparsity constraints for each layer with minimum and maximum layer usage of 4.38% and 97.4%, respectively. Moreover, it is resilient to different arrival sequences, by adaptively adjusting the sparsity constraints for each modality.*

Recommendation 2. *With Omni-CNN, we utilize minimum model capacity, while maintain the accuracy. This is because the sparsity constraints are set adaptively on a per-layer basis for each modality. Thus, using the adaptive sparsity constraint algorithm is recommended over predefined method.*

D. Omni-CNN and Coexistence of Modalities

We now explore the effect of decision level aggregation in Omni-CNN (see Sec. IV-C). We consider a single scenario, where the arrival sequence is GIL and the sensitivity is set as 0.002. In Table V, we compare the performance of Omni-CNN with presence of only one sensor modality versus

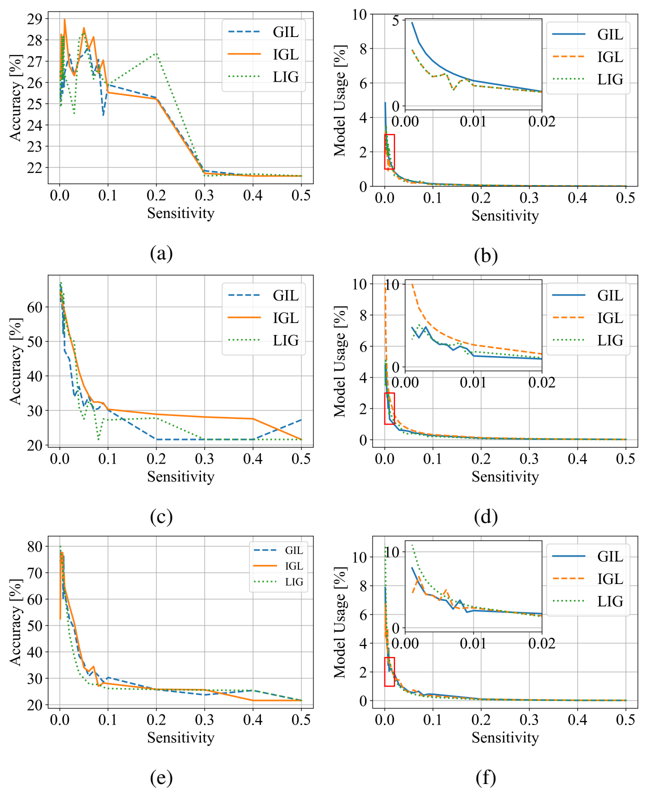

Fig. 6: Effect of sensitivity on the accuracy and model usage for three arrival sequences on GPS (a, b), Image (c, d), and LiDAR (e, f).

having more than one sensor and exploit the aggregation policies. For this experiment, we use half of our test set for validation and the rest for testing. We record the validation accuracy of 25.25%, 63.84%, and 77.17% for GPS, image, and LiDAR, resulting in coefficients of 0.15, 0.38, and 0.47 in Eq. (12) for the case of presence of all three modalities for instance, respectively. We conclude that the weighted majority vote is the most successful aggregation policy.

Phenomenon 6. *We observe that including GPS results in degradation in accuracy for the aggregation of GPS and image, while we observe 1.7%, 4.08%, and 3.2% improvement in accuracy for remaining multimodal combinations.*

Recommendation 3. *From Phenomena 6. it is evident that having multiple sensor modalities improves the prediction accuracy, since they provide a more comprehensive realization of the environment. Thus, when more than one modality is present, it is recommended to use the aggregation method presented in Sec. IV-C.*

E. Benchmarking Omni-CNN

1) Omni-CNN vs. Three Competing Methods: We compare the performance of Omni-CNN against three competing methods described below.

- *Customized models:* The FLASH dataset release [12] also includes customized model architectures for GPS, camera, and LiDAR sensors, with 695880, 827392, and 933280 trainable parameters, respectively. We train these individual models for each of the three modalities in this experiment. These models are designed based on specific features of each sensor. For example, LiDAR has the highest number of parameters due to complex nature of this modality.
- *Full capacity:* Among the three models released by FLASH dataset, we select the LiDAR model with 933280 trainable parameters as our shared model in Omni-CNN framework, since it has the highest capacity. In this experiment, we use the shared model to train the beam selection task for GPS, camera, and LiDAR. As a result, the same model is trained three times, separately, each time with a different modality. Note that since we use the same LiDAR model for both customized models and full capacity experiments, the accuracies for the LiDAR are the same in both cases. However, for GPS and image, we observe that the customized model outperform the full capacity setting. This is because the LiDAR model (i.e. shared model) has more parameters than customized models for GPS and camera. Thus, these two modalities may experience slight overfitting with this over-parameterized model. Fig. 5c shows the train and test accuracy for the camera in this experiment. We observe a small gap between train and test accuracy, specially in the last few epochs, that corroborates the slight overfitting.
- *Omni-CNN (Predefined):* We use the shared model and extract disjoint sub-models for each modality with predefined sparsity constraints. Here, we report the maximum accuracy across all of our experiments in Tab. III. Note that although the maximum accuracies are higher in predefined setting than the adaptive one, these accuracies are not achievable with the same sparsity constraint. For example, achieving the accuracy of 82.68% with LiDAR results in 26.40% and 53.90% accuracy for GPS and camera sensors, respectively, according to Tab. III.
- *Omni-CNN (Adaptive):* We use the shared model and extract disjoint sub-models for each modality with sparsity constraints obtained adaptively according to Sec. IV-B2. Here, we report the maximum accuracy for three modalities from Sec. V-C1 as well. However, as shown in

Omni-CNN (one sensor)	GPS	Image	LiDAR
	25.25	63.84	77.17

Omni-CNN (multiple sensors)	GPS Image	GPS LiDAR	Image LiDAR	GPS Image LiDAR
Majority vote	52.18	67.5	72.68	64.93
Weighted majority vote	59.25	**78.87**	**81.25**	**80.37**
Multinomial logistic regression	36.87	47.5	46.87	46.87

TABLE V: Evaluating the effect of decision level aggregation policies (while having multiple sensors) compared to having only one sensor modality in Omni-CNN.

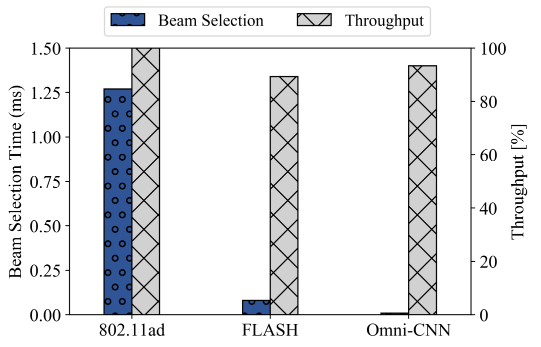

Fig. 7: Comparing Omni-CNN against standard exhaustive search and FLASH framework [12] in beam selection time and throughput.

Methodology	Test Accuracy (%)			Model Usage
	GPS	Image	LiDAR	
Customized models [12]	29.80	68.66	81.86	2.62×
Full Capacity	29.66	66.01	81.86	3×
Omni-CNN (Predefined)	29.96	68.87	82.68	×
Omni-CNN (Adaptive)	28.96	66.93	80.89	0.258×

TABLE VI: Benchmarking Omni-CNN against using full model capacity, customized models for each modality, and Omni-CNN with predefined sparsity constraints. The full capacity methodology refers to using 100% of the shared model for each modality, separately.

Sec. V-C1, using adaptive sparsity constraints is resilient to different arrival sequences of the modalities, unlike the predefined setting.

From Table VI, we observe that Omni-CNN performs similarly to three other competing methods in accuracy. Moreover, while the other approaches use $3\times$, $2.62\times$, and $1\times$ of the shared model capacity, Omni-CNN occupies only $0.258\times$ of the capacity. This compression also reflects in the downlink overhead, where the BS broadcasts the share model to all vehicles.

Phenomenon 7. *Omni-CNN reduces the model usage by 91.4%, 90%, and 74.2% w.r.t to using full capacity, customized models for each modality, and Omni-CNN with predefined sparsity constraints, respectively.*

2) Omni-CNN vs. the State-of-the-art Unified Models: We compare Omni-CNN with six state-of-the-art methods that generate unified models in computer vision domain in Table. VII. The competing method spans a wide range of methods with lifelong learning, pruning, weight consolidation, transformers, etc. Other works that consider different settings such as incomplete data ([32], [33]), modality interactivity [43], [44] or modal redundancy [45] have been kept out of the comparison. In this table, we report the average accuracy across all tasks, number of model parameters, as well as the experiment settings. From model parameter column, we observe that Omni-CNN exploits much less number of parameters than state-of-the-art methods. Moreover, we use the ratio of accuracy over number of model parameters as our criteria to benchmark different methods for a fair comparison. Intuitively, a higher ratio of accuracy over model size implies that the model demonstrates high prediction accuracy, while using low number of parameters. From the last column in Table. VII, we observe that Omni-CNN is superior to other competing methods in this regard.

Phenomenon 8. *The state-of-the-art on unified models exhibit competitive accuracy on their designated task (image classification). However, the models are large and the computation cost is disregarded. On the other hand, Omni-CNN results in the highest ratio of accuracy over number of parameters compared to state-of-the-art on unified models, which suggests that it requires minimum number of parameters while maintaining the accuracy.*

3) Omni-CNN vs. State-of-the-art Beam Selection: Here, we compare Omni-CNN against the state-of-the-art mmWave beam selection methods that use sensor data. As the majority of the literature are focused on a single LiDAR data, we report the accuracy on this modality only for a fair comparison.

Phenomenon 9. *From Table VII, we observe the Omni-CNN outperforms the Klautau* et al. *[55] framework by 50.39% in accuracy and Salehi* et al. *[12] by 89.24-95.65% in model usage.*

Recommendation 4. *Omni-CNN is superior to state-of-the-art methods due to design features that enables minimizing the model usage. Thus, it is a more suitable solution for wireless systems, where the communication overhead is a constraint.*

F. Accuracy and Overhead Trade-off in 5G-NR Deployment

Omni-CNN not only enables heterogeneity, efficient storage and less computation, but also incurs lower communication overhead for model initialization in the downlink. We compare the number of exchanged model parameters and communication overhead against a centralized learning scenario discussed in [12]. In this experiment, we consider the case of only one on-board sensor.

Overhead of centralized learning: Different vehicles with GPS, image, and LiDAR sensors use distinct model architectures each for beam selection. In the training phase, the BS gathers all the sensor data and trains individual models for each modality, with 0.69M, 0.82M, and 0.93M parameters (model architectures released with FLASH dataset [12]), for GPS, image and LiDAR, respectively. In the inference phase, all three distinct models with a total of 2.44M parameters must be transmitted in DL. This occurs because the BS is not aware of which specific sensor is available in a given vehicle and multiple vehicles may submit model requests simultaneously.

Overhead of Omni-CNN: In the training phase, the BS uses Omni-CNN framework to obtain modality-specific submodels for each modality (see Sec. IV-B). With Omni-CNN, the largest sub-model (LiDAR) has 0.1M parameters. In the inference phase, the BS broadcasts the shared model with 0.23M parameters, encapsulating the sub-models for three modalities in the DL, which is sent to all vehicles regardless

Paper	Task	Dataset	Classes	Acc (%)	Method	# of model params	Acc (%) over # params (M)
Unified Model Generation							
Jian *et al.* [25]	Image classification	Permuted MNIST	10	98.58	ADMM	5.5M	17.92
Mallya *et al.* [50]	Image classification	CIFAR	10	77.79	Pruning	0.88M	88.39
Zenke *et al.* [51]	Image classification	CIFAR	10	74.97	Elastic weight consolidation	0.88M	85.19
Shin *et al.* [52]	Image classification	CIFAR	10	63.61	Deep generative replay	0.88M	72.28
Girdhar *et al.* [34]	Image classification	ImageNet	1K	85.3	Swin Transformers [53]	145M	0.58
Singh *et al.* [35]	Image classification	ImageNet	1K	79.35	ViT Transformer [54]	86M	0.92
Modality Specific Beam Selection (LiDAR only)							
Klautau *et al.* [55]	Beam selection	Raymobtime	256	30.5	Centralized learning	0.1M	305
Salehi *et al.* [15]	Beam selection	Raymobtime	256	46.23	Centralized learning	2.3M	20.1
Salehi *et al.* [12]	Beam selection	FLASH	34	80.37	Centralized learning	0.93M	86.41
Omni-CNN	Beam selection	FLASH	34	80.89	Adaptive ADMM	0.1M	**808.9**

TABLE VII: Comparing Omni-CNN with state-of-the-art methods.

Learning Strategy	Test Accuracy (%)			Model Initialization Overhead
	GPS	Image	LiDAR	
Centralized learning [12]	29.81	68.75	80.37	2.44M, 0.1463
Omni-CNN	25.25	63.84	77.17	0.23M, 0.0137

TABLE VIII: Comparing the accuracy and model initialization overhead of centralized learning strategy [12] and the proposed Omni-CNN framework. The first value in model initialization overhead is the number of model parameters that must be exchanged and the second value is the communication overhead (in seconds) with the wireless link described in Sec. V-F.

of their available sensors. In Table VIII, we also report the communication overhead, while considering a 5G backchannel that supports a throughput of 63.59MBps.

Phenomenon 10. *Omni-CNN framework outperforms the centralized learning [12] by incurring 90.57% less overhead for model initialization at inference phase, with graceful degradation of up to 4.91% in accuracy.*

Recommendation 5. *Omni-CNN imposes the minimum overhead to the communication system by optimizing the number of model parameters. Thus, it is recommended as an alternative solution for resource constrained devices or when the communication overhead needs to be minimized.*

G. Comparison with Standard Exhaustive Search Approach

In Fig. 7, we compare the beam selection overhead of Omni-CNN against 802.11ad standard (see Sec. II-A), that is used for collection of FLASH dataset. Inferring the optimum beam using Omni-CNN requires three steps, (a) *Data acquisition and preprocessing*, which imposes negligible overhead given the high sampling rate of modern sensors and simplicity of LiDAR preprocessing step; (b) *Model inference* quantified as 0.008 ms by taking average over multiple inference runs over LiDAR sub-model; (c) *Sharing optimum beam*, which is an integer value with negligible communication overhead. Hence, the total beam selection time in Omni-CNN is $\sim$0.008ms, while sweeping all 34 beams with the 802.11ad standard in Talon routers takes 1.27ms [47]. We also report the throughput ratio that characterizes the ratio of degradation in throughput against the ideal exhaustive search method [12].

Phenomenon 11. *We observe 99.37% and 90% improvement in beam selection speed over 802.11ad and FLASH [12] while retaining 93.34% of the throughput with respect to 802.11ad and improving it by 4.02% compared to FLASH [12].*

Recommendation 6. *Omni-CNN results in much less overhead than exhaustive search based methods, since it is performed locally at the vehicles with a single inference run. Thus, it can address the beam initialization overhead and pave the way for widespread of mmWave systems.*

H. Discussions and Limitations

Despite demonstrates promising results, we discuss implications of some of our assumptions and limitations that may impact performance in practical scenarios. In this section, we discuss these limitations and identify possible future directions to address them.

- In Omni-CNN, the size of the shared model is fixed prior to considering available sensor modalities. As a result, Omni-CNN cannot accommodate unbounded number of modalities and the shared model will eventually exceed its representation capability. Thus, prior knowledge of sensors is required to select a shared model that can accommodate all the required modalities for the target application.
- Since each modality is trained on the residual capacity, the performance of the modalities (except for the first one) depends on the available residual capacity in the shared model. Thus, if the capacity is less that what required, we may observe degradation in the performance. A possible way to address this challenge is by using active dendrites ([56], [57]), which do mitigate such limitations.
- To manage sensor obsolescence, Omni-CNN can release the capacity from the previously learnt modalities through *targeted forgetting*. The Omni-CNN approach will need to be thoroughly revised to enable such forgetting through user-directives. In addition, the forgotten modalities must be learnt again upon being encountered in the future.
- The FLASH dataset has $\sim 32K$ samples for 34 classes (i.e. number of beams), while having the inputs data of maximum 43200 elements for the image with size (90, 160, 3). On the other hand, a benchmark dataset

such as MNIST [58] has $\sim 70K$ samples for 784-dimensional inputs, i.e. (28,28) images. Thus, the number of samples in the MNIST dataset have a better chance of representing the MNIST space (10×784 dimensional problem space) than the number of samples in the FLASH dataset representing the FLASH space ($\sim$ 34×43.2K dimensional problem space). As a result, the FLASH dataset is rather data-deficient, and conducting extensive data collection campaigns to obtain a richer dataset will ensure comparable results against more comprehensive benchmark datasets such as MNIST.

- We have not analytically argued why a certain model should be chosen as the starting point, other than by simply ordering them based on the number of trainable parameters (here, LiDAR). Thus, with models of similar sizes, a more principled way of choosing the shared model is needed. Finally, edge computing constraints for running model inference are not considered in the design. We envisage interesting trade-offs when a large model is chosen as the starting point (as we did) but available compute constraints affect latency of inference beyond acceptable levels. Joint GPU and model capacity provisioning is an exciting new area that can address this problem.

VI. Conclusions

Omni-CNN addresses a key problem in multimodal mmWave beam selection for vehicular networks: how to perform inference when different types of sensor modalities are present per vehicle. Disseminating a single shared model to any vehicle, compressed to 91.4% over the unpruned full capacity usage architecture, not only simplifies deployment but also demonstrates the baseline performance of $< 1\%$ accuracy degradation. As the first demonstration of model sharing for a communication-specific task, Omni-CNN framework can be extended in exciting ways: (i) factoring in the available computation resources into the sparsity constraint optimization problem, and (ii) adding new communication tasks (e.g., modulation-coding scheme selection instead of adding new sensor modalities) to the same shared model leading to thorough research on the impact of over-parameterization for communications-centric ML solutions.

References

[1] H. Bagheri, M. Noor-A-Rahim, Z. Liu, H. Lee, D. Pesch, K. Moessner, and P. Xiao, "5G NR-V2X: Toward Connected and Cooperative Autonomous Driving," *IEEE Communications Standards Magazine*, vol. 5, no. 1, pp. 48–54, 2021.

[2] D. Geronimo, A. M. Lopez, A. D. Sappa, and T. Graf, "Survey of Pedestrian Detection for Advanced Driver Assistance Systems," *IEEE transactions on pattern analysis and machine intelligence*, vol. 32, no. 7, pp. 1239–1258, 2009.

[3] R. S. Thoma, C. Andrich, G. Del Galdo, M. Dobereiner, M. A. Hein, M. Kaske, G. Schafer, S. Schieler, C. Schneider, A. Schwind *et al.*, "Cooperative Passive Coherent Location: A Promising 5G Service to Support Road Safety," *IEEE Communications Magazine*, vol. 57, no. 9, pp. 86–92, 2019.

[4] J. Choi, V. Va, N. Gonzalez-Prelcic, R. Daniels, C. R. Bhat, and R. W. Heath, "Millimeter-wave Vehicular Communication to Support Massive Automotive Sensing," *IEEE Communications Magazine*, vol. 54, no. 12, pp. 160–167, 2016.

[5] L. Kong, M. K. Khan, F. Wu, G. Chen, and P. Zeng, "Millimeter-wave Wireless Communications for IoT-Cloud Supported Autonomous Vehicles: Overview, Design, and Challenges," *IEEE Communications Magazine*, vol. 55, no. 1, pp. 62–68, 2017.

[6] W. Roh, J.-Y. Seol, J. Park, B. Lee, J. Lee, Y. Kim, J. Cho, K. Cheun, and F. Aryanfar, "Millimeter-Wave Beamforming as an Enabling Technology for 5G Cellular Communications: Theoretical Feasibility and Prototype Results," *IEEE Communications Magazine*, vol. 52, no. 2, pp. 106–113, 2014.

[7] T. Zugno, M. Drago, M. Giordani, M. Polese, and M. Zorzi, "Toward Standardization of Millimeter-wave Vehicle-to-Vehicle Networks: Open Challenges and Performance Evaluation," *IEEE Communications Magazine*, vol. 58, no. 9, pp. 79–85, 2020.

[8] C. N. Barati, S. Dutta, S. Rangan, and A. Sabharwal, "Energy and Latency of Beamforming Architectures for Initial Access in mmWave Wireless Networks," *Journal of the Indian Institute of Science*, pp. 1–22, 2020.

[9] M. Giordani, M. Polese, A. Roy, D. Castor, and M. Zorzi, "A Tutorial on Beam Management for 3GPP NR at mmWave Frequencies," *IEEE Communications Surveys & Tutorials*, vol. 21, no. 1, pp. 173–196, 2018.

[10] N. González-Prelcic, A. Ali, V. Va, and R. W. Heath, "Millimeter-Wave Communication with Out-of-Band Information," *IEEE Communications Magazine*, vol. 55, no. 12, pp. 140–146, 2017.

[11] V. Va, J. Choi, T. Shimizu, G. Bansal, and R. W. Heath, "Inverse Multipath Fingerprinting for Millimeter Wave V2I Beam Alignment," *IEEE Transactions on Vehicular Technology*, vol. 67, no. 5, pp. 4042–4058, 2017.

[12] B. Salehi, J. Gu, D. Roy, and K. Chowdhury, "FLASH: Federated Learning for Automated Selection of High-band mmWave Sectors," in *IEEE INFOCOM 2022-IEEE Conference on Computer Communications*. IEEE, 2022, pp. 1719–1728.

[13] M. Zecchin, M. B. Mashhadi, M. Jankowski, D. Gündüz, M. Kountouris, and D. Gesbert, "LIDAR and Position-Aided mmWave Beam Selection with Non-local CNNs and Curriculum Training," *IEEE Transactions on Vehicular Technology*, vol. 71, no. 3, pp. 2979–2990, 2022.

[14] A. Klautau, N. González-Prelcic, and R. W. Heath, "LIDAR Data for Deep Learning-Based mmWave Beam-Selection," *IEEE Wireless Communications Letters*, vol. 8, no. 3, pp. 909–912, 2019.

[15] B. Salehihikouei, G. Reus-Muns, D. Roy, Z. Wang, T. Jian, J. Dy, S. Ioannidis, and K. Chowdhury, "Deep Learning on Multimodal Sensor Data at the Wireless Edge for Vehicular Network," *IEEE Transactions on Vehicular Technology*, 2022.

[16] S. Han, H. Mao, and W. J. Dally, "Deep Compression: Compressing Deep Neural Networks with Pruning, Trained Quantization and Huffman Coding," in *ICLR*, 2016.

[17] Y. Guo, A. Yao, and Y. Chen, "Dynamic Network Surgery for Efficient DNNs," in *NeurIPS*, 2016, pp. 1379–1387.

[18] X. Dong and Y. Yang, "Network pruning via transformable architecture search," *Advances in Neural Information Processing Systems*, vol. 32, 2019.

[19] R. Yu, A. Li, C.-F. Chen, J.-H. Lai, V. I. Morariu, X. Han, M. Gao, C.-Y. Lin, and L. S. Davis, "Nisp: Pruning networks using neuron importance score propagation," in *Proceedings of the IEEE conference on computer vision and pattern recognition*, 2018, pp. 9194–9203.

[20] Z. Liu, M. Sun, T. Zhou, G. Huang, and T. Darrell, "Rethinking the Value of Network Pruning," in *ICLR*, 2019.

[21] W. Wen, C. Wu, Y. Wang, Y. Chen, and H. Li, "Learning structured sparsity in deep neural networks," in *NeurIPS*, 2016, pp. 2074–2082.

[22] Y. He, X. Zhang, and J. Sun, "Channel pruning for accelerating very deep neural networks," in *ICCV*, 2017, pp. 1389–1397.

[23] T. Zhang, S. Ye, K. Zhang, J. Tang, W. Wen, M. Fardad, and Y. Wang, "A Systematic DNN Weight Pruning Framework using Alternating Direction Method of Multipliers," in *ECCV*, 2018, pp. 184–199.

[24] T. Jian, Y. Gong, Z. Zhan, R. Shi, N. Soltani, Z. Wang, J. G. Dy, K. R. Chowdhury, Y. Wang, and S. Ioannidis, "Radio Frequency Fingerprinting on the Edge," *IEEE Transactions on Mobile Computing*, 2021.

[25] Z. Wang, T. Jian, K. Chowdhury, Y. Wang, J. Dy, and S. Ioannidis, "Learn-Prune-Share for Lifelong Learning," in *2020 IEEE International Conference on Data Mining (ICDM)*. IEEE, 2020, pp. 641–650.

[26] A. Mallya, D. Davis, and S. Lazebnik, "Piggyback: Adapting a Single Network to Multiple Tasks by Learning to Mask Weights," in *ECCV*, 2018, pp. 67–82.

[27] G. Sokar, D. C. Mocanu, and M. Pechenizkiy, "Spacenet: Make Free Space for Continual Learning," *Neurocomputing*, vol. 439, pp. 1–11, 2021.

[28] H. Gao, W. Huang, T. Liu, Y. Yin, and Y. Li, "Ppo2: Location privacy-oriented task offloading to edge computing using reinforcement learning for intelligent autonomous transport systems," *IEEE transactions on intelligent transportation systems*, 2022.

[29] H. Gao, J. Xiao, Y. Yin, T. Liu, and J. Shi, "A mutually supervised graph attention network for few-shot segmentation: the perspective of fully utilizing limited samples," *IEEE Transactions on neural networks and learning systems*, 2022.

[30] Y. Li, L. Ma, Z. Zhong, F. Liu, M. A. Chapman, D. Cao, and J. Li, "Deep learning for lidar point clouds in autonomous driving: A review," *IEEE Transactions on Neural Networks and Learning Systems*, vol. 32, no. 8, pp. 3412–3432, 2020.

[31] H. Gao, B. Qiu, R. J. D. Barroso, W. Hussain, Y. Xu, and X. Wang, "Tsmae: a novel anomaly detection approach for internet of things time series data using memory-augmented autoencoder," *IEEE Transactions on network science and engineering*, 2022.

[32] J. Chen and A. Zhang, "Hgmf: heterogeneous graph-based fusion for multimodal data with incompleteness," in *Proceedings of the 26th ACM SIGKDD international conference on knowledge discovery & data mining*, 2020, pp. 1295–1305.

[33] M. Liu, Y. Gao, P.-T. Yap, and D. Shen, "Multi-hypergraph learning for incomplete multimodality data," *IEEE journal of biomedical and health informatics*, vol. 22, no. 4, pp. 1197–1208, 2017.

[34] R. Girdhar, M. Singh, N. Ravi, L. van der Maaten, A. Joulin, and I. Misra, "Omnivore: A Single Model for Many Visual Modalities," in *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 2022, pp. 16 102–16 112.

[35] A. Singh, R. Hu, V. Goswami, G. Couairon, W. Galuba, M. Rohrbach, and D. Kiela, "Flava: A Foundational Language and Vision Alignment Model," in *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 2022, pp. 15 638–15 650.

[36] Y. Wang, A. Klautau, M. Ribero, M. Narasimha, and R. W. Heath, "MmWave Vehicular Beam Training with Situational Awareness by Machine Learning," in *2018 IEEE Globecom Workshops (GC Wkshps)*, 2018, pp. 1–6.

[37] M. Alrabeiah, A. Hredzak, and A. Alkhateeb, "Millimeter Wave Base Stations with Cameras: Vision-Aided Beam and Blockage Prediction," in *IEEE 91st Vehicular Technology Conference (VTC2020)*. IEEE, 2020, pp. 1–5.

[38] M. Dias, A. Klautau, N. González-Prelcic, and R. W. Heath, "Position and LIDAR-Aided mmWave Beam Selection using Deep Learning," in *2019 IEEE 20th International Workshop on Signal Processing Advances in Wireless Communications (SPAWC)*. IEEE, 2019, pp. 1–5.

[39] W. Xu, F. Gao, S. Jin, and A. Alkhateeb, "3D Scene-Based Beam Selection for mmWave Communications," *IEEE Wireless Communications Letters*, vol. 9, no. 11, pp. 1850–1854, 2020.

[40] G. Reus-Muns, B. Salehi, D. Roy, T. Jian, Z. Wang, J. Dy, S. Ioannidis, and K. Chowdhury, "Deep Learning on Visual and Location Data for V2I mmWave Beamforming," in *2021 17th International Conference on Mobility, Sensing and Networking (MSN)*. IEEE, 2021, pp. 559–566.

[41] D. Böhning, "Multinomial logistic regression algorithm," *Annals of the Institute of Statistical Mathematics*, vol. 44, no. 1, pp. 197–200, 1992.

[42] X. Dong, Z. Yu, W. Cao, Y. Shi, and Q. Ma, "A survey on ensemble learning," *Frontiers of Computer Science*, vol. 14, pp. 241–258, 2020.

[43] J.-B. Alayrac, A. Recasens, R. Schneider, R. Arandjelović, J. Ramapuram, J. De Fauw, L. Smaira, S. Dieleman, and A. Zisserman, "Self-supervised multimodal versatile networks," *Advances in Neural Information Processing Systems*, vol. 33, pp. 25–37, 2020.

[44] A. Nagrani, S. Yang, A. Arnab, A. Jansen, C. Schmid, and C. Sun, "Attention bottlenecks for multimodal fusion," *Advances in Neural Information Processing Systems*, vol. 34, pp. 14 200–14 213, 2021.

[45] P. Wang, X. Wang, B. Wang, Y. Zhang, L. Bai, and Y. Wang, "Countering modal redundancy and heterogeneity: A self-correcting multimodal fusion," in *2022 IEEE International Conference on Data Mining (ICDM)*. IEEE, 2022, pp. 518–527.

[46] https://www.rfdatafactory.com.

[47] D. Steinmetzer, D. Wegemer, M. Schulz, J. Widmer, and M. Hollick, "Compressive Millimeter-Wave Sector Selection in Off-the-Shelf IEEE 802.11ad Devices," *International Conference on emerging Networking EXperiments and Technologies (CoNEXT)*, 2017.

[48] K. He, X. Zhang, S. Ren, and J. Sun, "Deep Residual Learning for Image Recognition," in *CVPR*, 2016, pp. 770–778.

[49] "GitHub Repository of Omni-CNN is available at:," https://github.com/batoolsalehi/Omni-CNN, 2024.

[50] A. Mallya and S. Lazebnik, "Packnet: Adding multiple tasks to a single network by iterative pruning," in *Proceedings of the IEEE conference on Computer Vision and Pattern Recognition*, 2018, pp. 7765–7773.

[51] F. Zenke, B. Poole, and S. Ganguli, "Continual learning through synaptic intelligence," in *International conference on machine learning*. PMLR, 2017, pp. 3987–3995.

[52] H. Shin, J. K. Lee, J. Kim, and J. Kim, "Continual learning with deep generative replay," *Advances in neural information processing systems*, vol. 30, 2017.

[53] Z. Liu, Y. Lin, Y. Cao, H. Hu, Y. Wei, Z. Zhang, S. Lin, and B. Guo, "Swin Transformer: Hierarchical Vision Transformer using Shifted Windows," in *Proceedings of the IEEE/CVF International Conference on Computer Vision*, 2021, pp. 10 012–10 022.

[54] A. Dosovitskiy, L. Beyer, A. Kolesnikov, D. Weissenborn, X. Zhai, T. Unterthiner, M. Dehghani, M. Minderer, G. Heigold, S. Gelly *et al.*, "An Image is Worth 16x16 Words: Transformers for Image Recognition at Scale," *arXiv preprint arXiv:2010.11929*, 2020.

[55] A. Klautau, N. González-Prelcic, and R. W. Heath, "LIDAR Data for Deep Learning-based mmWave Beam-Selection," *IEEE Wireless Communications Letters*, vol. 8, no. 3, pp. 909–912, 2019.

[56] J. Guerguiev, T. P. Lillicrap, and B. A. Richards, "Towards deep learning with segregated dendrites," *Elife*, vol. 6, p. e22901, 2017.

[57] A. Iyer, K. Grewal, A. Velu, L. O. Souza, J. Forest, and S. Ahmad, "Avoiding catastrophe: Active dendrites enable multi-task learning in dynamic environments," *Frontiers in neurorobotics*, vol. 16, p. 846219, 2022.

[58] Y. LeCun, C. Cortes, and C. J. Burges, "The MNIST Database of Handwritten Digits, 1998," *URL http://yann. lecun. com/exdb/mnist*, vol. 10, p. 34, 1998.

Batool Salehi is currently pursuing a Ph.D. degree in computer engineering at Northeastern University under the supervision of Prof. K. Chowdhury. Her current research focuses on mmWave beamforming, Internet of Things, and the application of machine learning in the domain of wireless communication.

Debashri Roy received her MS (2018) and PhD (2020) degrees in Computer Science from University of Central Florida, USA. She is currently assistant professor at The University of Texas Arlington. Her research interests are in the areas of AI/ML enabled technologies in wireless communication, multimodal data fusion, nextG networks, and networked systems.

Tong Jian is currently pursuing a Ph.D. degree in the Department of Electrical and Computer Engineering, Northeastern University, Boston, Massachusetts. She received her M.Sc. (2016) in electrical engineering from Rensselaer Polytechnic Institute, New York. She works under the guidance of Prof. Stratis Ioannidis in the field of machine learning. Her current research efforts are focused on the application of machine learning in the domain of wireless communication.

Chris Dick is a wireless architect at NVIDIA and the technical lead for the application of AI and machine learning to 5G and 6G wireless. In his 24 years working in signal processing and communications he has delivered silicon and software products for 3G, 4G, and 5G baseband DSP and Docsis 3.1 cable access. He has performed research and delivered products for digital frontend (DFE) technology for cellular systems.

Stratis Ioannidis is an Associate Professor in the Electrical and Computer Engineering Department of Northeastern University, in Boston, MA, where he also holds a courtesy appointment with the Khoury College of Computer Sciences. Prior to joining Northeastern, he was a research scientist at the Technicolor research centers in Paris, France, and Palo Alto, CA, as well as at Yahoo Labs in Sunnyvale, CA. His research interests span machine learning, distributed systems, networking, optimization, and privacy.

Kaushik Chowdhury is a Professor at Northeastern University, Boston, MA. He is presently a co-director of the Platforms for Advanced Wireless Research (PAWR) project office. His current research interests involve systems aspects of networked robotics, machine learning for agile spectrum sensing/access, wireless energy transfer, and large-scale experimental deployment of emerging wireless technologies.