# NAIR: An Efficient Distributed Deep Learning Architecture for Resource Constrained IoT System

Yucong Xiao, *Student Member, IEEE,* Daobing Zhang, *Member, IEEE,* Yunsheng Wang, *Member, IEEE,* Xuewu Dai, *Member, IEEE,* Zhipei Huang, *Member, IEEE,* Wuxiong Zhang, *Member, IEEE,* Yang Yang, *Fellow, IEEE,* Ashiq Anjum, *Senior Member, IEEE,* and Fei Qin, *Senior Member, IEEE*

***Abstract*—The distributed deep learning architecture can support the front-deployment of deep learning systems in resource constrained IoT devices and is attracting increasing interest. However, most ready-to-use deep models are designed for centralized deployment without considering the transmission loss of the intermediate representation inside the distributed architecture. This oversight significantly affects the inference performance of distributed deployed deep models. To alleviate this problem, a state-of-the-art work chooses to retrain the original model to form an intermediate representation with ordered importance and yields better inference accuracy under constrained transmission bandwidth. This paper first reveals that this solution is essentially a pruning-like solution, where unimportant information is adaptively pruned to fit within the limited bandwidth. With this understanding, a novel scheme named Naturally Aggregated Intermediate Representation (NAIR) has been proposed, which aims to naturally amplify the difference of importance embedded in the intermediate representation from a mature deep model and reassemble the intermediate representation into a hierarchy of importance from high-to-low to accommodate the transmission loss. As a result, this method shows further improved performance in various scenarios, avoids compromising the overall inference performance of the system, and saves astronomical retraining and storage costs. The effectiveness of NAIR has been validated through extensive experiments, achieving a 112% improvement in performance compared to the state-of-the-art work.**

This research was supported in part by the National Nature Science Foundation of China under Grant 62071450, in part by the National Key R&D Program of China under Grant 2021YFC3090204, in part by the Fundamental Research Funds for the Central Universities, in part by the Natural Science Foundation of Shanghai under Grant 19ZR1454100, in part by the NSF grants CNS 2128346, and in part by the EPSRC under Grant EP/Y00597X/1 and EP/Y018281/1. Part information of this paper appeared in IEEE SWC 2023. (Corresponding author: Daobing Zhang.)

Y.Xiao, Z.Huang, and F. Qin are with the School of Electronic and Electrical Communication Engineering, University of Chinese Academy of Sciences, Beijing 101408, China, email: (xiaoyucong20@mails.ucas.ac.cn, zhphuang@ucas.ac.cn, fqin1982@ucas.ac.cn).

D. Zhang is with the Aerospace Information Research Institute, Chinese Academy of Sciences, Beijing 100094, China, email: (zhangdb@aircas.ac.cn).

Y. Wang is with the Department of Computer Science, California State Polytechnic University, Pomona 91768, USA, email: (yunshengwang@cpp.edu).

X. Dai is with the Department of Mathematics, Physics and Electrical Engineering, Northumbria University, Newcastle-upon-Tyne, NE1 8ST, UK, email: (xuewu.dai@northumbria.ac.uk).

W. Zhang is with the Shanghai Institute of Microsystem and Information Technology (SIMIT), Chinese Academy of Sciences, Shanghai 200050, China, email: (wuxiong.zhang@mail.sim.ac.cn).

A. Anjum is with the School of Informatics, University of Leicester, Leicester LE1 7RH, UK (e-mail: aa1180@leicester.ac.uk).

Y. Yang is with the IoT Thrust and the Research Center for Digital World with Intelligent Things (DOIT) at HKUST (Guangzhou), also with Terminus Group, Beijing 100027, China, and also with Peng Cheng Laboratory, Shenzhen 518055, China (e-mail: yyiot@hkust-gz.edu.cn).

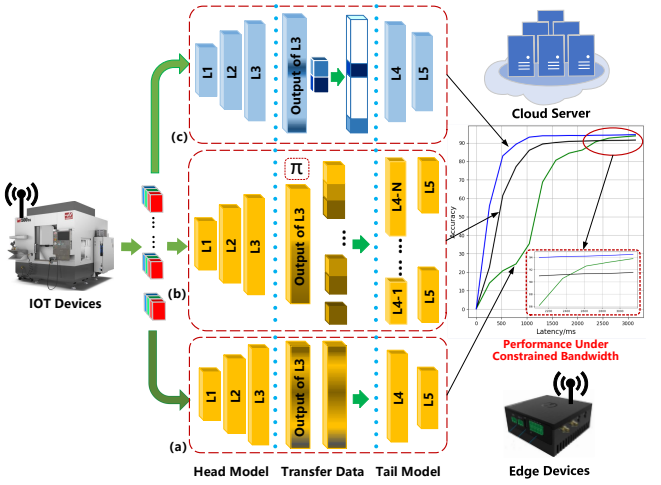


Fig. 1. The different deployment schemes of distributed deep learning systems.

***Index Terms*—Distributed Deep Learning, Convolutional Neural Network, Pruning, Transmission Loss**

## I. Introduction

In recent years, deep learning rapidly developed in various fields, including computer vision [1], [2], natural language processing [3], [4] and visual odometry [5], [6]. Due to the efficient feature extraction and analysis capability of Deep Neural Networks (DNN) [7], more advanced sensors with rich information can be deployed, which not only extend the boundary of sensing, but also enhance the capability of the Internet of Things (IoT) applications [8]. The high-dimension sensing data collected by these advanced sensors can be further extracted by a well-trained DNN model to generate the essential information, usually named as inference process. Without any doubt, this trend will significantly increase the performance of modern IoT applications.

These evolving changes are encouraging the front-deployment of the DNN models [9]. Although many Commercial off the Shelf (COTS) devices [10] and published works [11], [12] have already demonstrated this possibility, the computing and memory resources of these devices are usually several orders lower than the classical deep learning platforms. Both the model compression [13] or hand-craft mode to effectively compress and clip the DNN parameters may fail to fully exploit the capability of well-trained DNN models in

IoT devices. In response to this problem, the intuitive solution is to deploy the DNN model solely on the cloud system with sufficient memory resources and computing capabilities and let the raw observation be transmitted to the cloud through wireless transceivers. Nevertheless, the valuable observation will suffer from the transmission loss caused by multi-path fading effects, which means part of it will be lost during the transmission. With higher transmission loss, poor inference performance on the cloud side can be expected.

To alleviate this problem, researchers have proposed a new deployment pattern of deep learning systems in [14]–[16], which divides the DNN model into several parts to deploy in different devices and forms the distributed deep learning architectures. As shown in Fig. 1(a), the first several layers of a DNN model, termed the head model, will be deployed in the front-end IoT device, and the other layers, i.e., the corresponding tail model, will be deployed on the cloud server or edge devices with higher computing capabilities. The output of the head model, i.e., the intermediate representation, will be transmitted to the tail model through wireless transceivers. The partially processed intermediate representation data, instead of the raw observed data, will suffer from the transmission error. As generally believed, the essential useful information embedded in the input observations will be gradually concentrated through the convolution operations along with layers of the well-trained DNN model. In this context, if given the same transmitting error probabilities, the information loss in the transmission of intermediate representation will be lower than in the transmission of raw data, leading to higher inference performance [17]. However, as the location and the importance of concentrated information in the intermediate representation is a *posteriori* knowledge, the transmission efficiency of intermediate representation cannot be guaranteed through the utilization of mature Quality of Service (QoS) technologies in wireless communication systems, e.g., allocating higher transmission power for packets containing bits with higher importance.

A state-of-the-art work named CLIO has been proposed to tackle this limitation [17]. As shown in Fig. 1(b), a deep model will be retrained to force the importance of intermediate representation forming an ordered distribution among multiple slices. In detail, the loss function has been designed with a weighted linear combination consisting of and progressive number of slices, where is expected to indicate the probability that there is sufficient bandwidth available to transmit the given slice. With the continuous convergence of the loss function, it is expected that the distribution of importance will approach the *a priori* distribution of . Then, the system can decide how much intermediate information should be transmitted to the cloud in accordance with the instantaneous link quality of the time-varying fading channel. It should be noted that the performance gain of this method can be treated as a pruning process in essence. Different from the traditional pruning algorithms that directly prune the filters and feature maps corresponding to the less useful information in the intermediate representation, the CLIO is equivalent to dynamically pruning the less important intermediate information when there is no sufficient transmission bandwidth. However, the progressive manner will always result in multiple slices with increasing size, leading to a smaller probability of successfully transmitting the last slice under the limited bandwidth, i.e., a smaller $_i$. If the system is aware that the last slice is exactly the complete intermediate representation, the smaller $_i$ will then act as a negative coefficient, which results in a degradation of the overall inference performance, as illustrated in the area of Fig. 1. In other words, there is a manually involved trade-off between the effective aggregation of important information in intermediate representation and the overall inference performance of the model. Furthermore, the multiple slices with different dimensions require multiple tail models to be deployed in the second-tier device, which in practice consumes enormous resources.

To face this challenge, this paper presents a novel scheme named Naturally Aggregated Intermediate Representation (NAIR). Instead of forcing the importance of intermediate representation to form a high-to-low distribution through intuitive retraining, the proposed method aggregates the important information in the intermediate representation in a simple and efficient three steps manner, i.e., identifies the importance, enlarges the difference of importance, and reassembles the slices with different importance, which contributes to the keyword of its name: "naturally". As a result, the outputs of the head model will also form an ordered importance distribution, while the low importance data will also be adaptively pruned when the transmission bandwidth is not enough. As illustrated in Fig. 1(c), upon receiving, the non-received bits in the intermediate representation will be padded with zeros to restore dimensions before feeding into the tail model. Surprisingly, this simplified solution brings increased performance in all dimensions than the classical intuitive retraining solution. Firstly, as only a single tail model is involved, the astronomical retraining costs of multiple models with different tails can be avoided. Secondly, as no equalized penalty factor will be involved, the overall inference accuracy will not be degraded, i.e., a scenario without transmission loss. Lastly, and most importantly, the inference accuracy in any latency scenario has been increased as well. This is due to the enhanced efficiency of the proposed method to rearrange the importance, where the distribution of the parameters will be closer to the raw integrated model.

The contribution of this paper can be summarized as follows:

- We reveal that the ordered importance in the intermediate representation of the distributed deep model is equivalent to the pruning operation caused by transmission loss.
- We propose a simple and efficient algorithm to aggregate the important information in the intermediate representations without compromising the inference performance.
- We design a new distributed architecture that optimizes the resource consumption of the system and shows improved inference performance in various scenarios.

The rest of this paper is organized as follows: In section II, we review the related work, section III provides the problem formulation, and section IV introduces the model architecture and the naturally aggregation algorithm NAIR. The experimental results have been provided in Section VI to

validate the efficiency of the proposed method. Finally, section V concludes this paper and discusses potential future work.

## II. Related Work

Recent advances in deep learning technologies have attracted widespread attention to the implementation of DNN models in IoT environments. However, most existing DNN models are computationally and storage intensive, which is a significant concern for efficient inference on IoT devices with limited computational and energy capabilities [18], [19]. With respect to this important issue, there are two major fields under investigation.

DNN models are typically over-parameterized [20], and many studies have focused on model compression to reduce redundancy. Many of them can be classified into three categories: quantization, knowledge distillation, and pruning. Quantization-based methods [21]–[23] choose to convert weights and activations from high-precision floating points (32-bit) to lower n-bit precision with negligible loss in accuracy, thus reducing the memory and computational resources required by the DNN model and speeding up its inference. Knowledge Distillation (KD) was first proposed by Hinton *et al.* [24]. It is often characterized as a 'student-teacher' learning framework [25]–[27], in which the teacher exports knowledge and the student acquires it. The primary intention of KD is to mimic a pre-trained larger model or ensemble of models by training a smaller student model. DNN pruning [28]–[31] is a technique that effectively prunes unimportant 'side branches', i.e., redundant weights, and reduces both storage and computational costs. Its general flow typically follows three steps: (1) training the model to learn which connections are important; (2) removing the unimportant connections using a certain pruning approach; and (3) fine-tuning the pruned model to restore accuracy [32]. Although the above approaches can improve the application of DNN models in IoT environments, there remains a huge gap between available resources on the device and resources required for efficient inference [33].

Distributed deep learning is a feasible solution to this issue, which jointly utilizes the resources of IoT and edge devices to fully exploit the capabilities of DNN models. It can be broadly categorized into Model-Agnostic Distributed Deep Learning (MA-DDL) and Model-Dependent Distributed Deep Learning (MD-DDL). Federated Learning (FL) is a promising technique in MA-DDL that enables end devices to collaboratively train DNN models required by the FL server through transmitting the update weights [34]–[36]. As such, communication is a critical bottleneck in FL [37]. The authors in [38] propose an edge stochastic gradient descent algorithm that selects a fraction of important gradients for communication by comparing the loss value between iterations. A communication-mitigated FL algorithm is proposed in [39], which uploads only relevant local model updates to reduce communication costs while ensuring global convergence. In [40], Alham *et al.* accelerate the distributed gradient descent process by transmitting the portion of entries with the largest magnitude in each gradient. However, the majority of FL methods focus on improving the efficiency of distributed feedback training of DNN models. On the other hand, our work aims to optimize the forward inference performance of models while working under limited communication resources.

The collaborative training nature of the FL-based method enables each participating device to train well-performing models using only local data. However, the storage intensive nature of DNN models presents a challenge for their effective deployment in IoT devices with limited resources. The MD-DDL approach provides a reliable solution to this problem that divides the DNN model into several parts and offloads complex tasks to different devices. In [41], Xiao *et al.* utilize the multi-agent reinforcement learning algorithm to optimize the partition point of DNN models and edge servers to reduce the energy consumption of the system. The authors in [42] and [43] utilizes lossy compression techniques to reduce the volume of communication data by compressing intermediate representations. The work [44] trains DNN models by emulating packet loss through the dropout method, so that the models can cope with the packet loss problem. However, few efforts have responded to limited communication resources by effectively amplifying the importance differences of intermediate representations. Although the work in [17] has achieved this, the cost is to compromise the overall performance of the model. In this paper, we focus on optimizing the trade-off between communication costs and inference performance. The proposed method naturally aggregates the important information in intermediate representations, and the unimportant information is adaptively pruned to suit the limited communication bandwidth, resulting in improved performance in any latency scenario.

## III. Problem Formulation

A distributed DNN model $M$ containing $S$ layers can be modeled as a directed acyclic graph $G = (S; E)$ containing a set of directed edges $E$. The cut $C\ 2\ E$ will divide the model $M$ into head and tail models. To simplify the optimization problem, the cut $C$ is located at a specified layer $L_k$ of the model $M$. $L_1$ to $L_k$ are deployed as the head model for the IoT devices, and $L_{k+1}$ to $L_S$ are deployed as the tail model for the cloud or edge devices with ample computing capabilities. For simple discussion, let $O_k = f_{\ h}(x)$ denote the output of the $L_k$ layer, where $x$ is the input sensor data and $O_k$ is named the intermediate representation of the model. The output of the tail model can be defined as $\hat{y} = h_{\ t}(O_k)$, $_h$ and $_t$ are the head and tail model parameters, respectively. Then, the overall inference output of the distributed DNN model can be defined as:

$$\hat{y} = h_{\ t}(f_{\ h}(x)) = F_{\ t;h}(x): \tag{1}$$

Given a data set $D = f(x_n; y_n); n = 1 : Ng$, an optimization scheme to make the model output approximate the ground truth $y_n$ in the backward propagation stage can be formulated as:

$$_{t;h} = \arg\min_{t;h}[\sum_{n=1}^{N} L(y_n; \hat{y}_n)]; \tag{2}$$

where $L(\cdot)$ denotes a predefined loss function, which can be Cross-Entropy, Mean Square Error (MSE), KL Scatter, etc. Supposing enough training samples have been provided and suitable architecture of $M$ has been designed, a well-trained $M$ can be utilized to estimate the $\hat{y}$ by feeding high-dimension sensor data $x$ in the forward inference stage, where the intermediate representation $O_k$ will be transmitted from the head model to the tail model via wireless transceivers.

It is worth noting that if cut $C$ has been set before the first layer, the raw observation from the sensor will be transmitted to the tail model, which will then be the exact original model. However, the transmission errors will be applied to the observations and thus severely degrade the inference performance of the deep model, which advocates the development of distributed DNN model. The convolutional layers in the head model will efficiently extract features from the raw observation and generate an intermediate representation with important information concentrated. Assuming an identical wireless link, the performance of a distributed DNN model transmitting the intermediate representation will be better than the original model transmitting the raw observation. However, the intermediate representation with the random distribution of important information prevents the utilization of mature QoS techniques to guarantee its transmission efficiency. As a result, the system fails to demonstrate improved inference performance under limited bandwidth.

Instead of transmitting the raw intermediate representation $O_k$ to the tail model directly, a state-of-the-art work named CLIO chose to improve the organization of the intermediate representations for better inference performance under constrained wireless bandwidth [17]. The CLIO method divides $O_k$ into multiple slices with increasing sizes. We define $_{k;i} = slice(i; O_k)$ to indicate the progressive slicing algorithm that divides $O_k$ into $i$ slices and let $_i$ denote the probability that there is currently sufficient bandwidth to send $slice_i$. The output of the model that corresponds to $slice_i$ can be formulated as $\hat{y}_{slice_i} = _i(\ _{k;i})$, and the optimization scheme of this method can be reformulated as:

$$; \quad = \arg\min_{;} \mathbb{E}\ [\ \sum_{n=1}^{N} L(y_n;\ \ _i(slice(i; f\ (x))))]; \quad (3)$$

where $\mathbb{E}\ [\ ]$ denotes the expectation over distribution . and represent the parameters of the head and tail models, respectively. Note that has a separate set of parameters for each value of $i$ due to the different dimensions of $slice_i$. $f\ (x)$ denotes the computation function for intermediate representation $O_k$. With eq.(3), the well-trained head model is expected to learn an intermediate representation $O_k^0$ with ordered importance that minimizes the expected inference error under the progressive slicing operation for the given distribution . With the known and instantaneous wireless link quality estimation, the algorithm can now determine how much intermediate information should be transmitted to the cloud and the less useful intermediate information will be adaptively pruned for constrained bandwidth.

However, two interleaving facts still exist that limit the potential performance of this method. The first issue existed in the implementation domain, this method requires multiple tail models to suit various link qualities, i.e., different $slice_i$ from the intermediate representation $O_k$ must have a corresponding independent tail model to work with to guarantee the inference accuracy. In other words, tens or even hundreds more tail models must be trained and stored on the cloud side, which requires enormous memory and computational resources to afford this deployment pattern. This is why the authors in [17] have proposed several acceleration methods to increase the efficiency in training the tail models. The second issue is in the performance domain, where a reciprocal constraint existed in the model training strategy of this method. The smaller $_i$ corresponding to the complete intermediate representation will negatively impact the overall inference performance of the model under constrained bandwidth. Consequently, it is incapable of aggregating the important information in the intermediate representation without compromising the inference performance of the model.

To tackle these critical issues, we first analyze the essential factor of this method for generating an intermediate representation with ordered importance. The stochastic gradient descent (SGD) method is appointed to minimize the objective function in eq.(3), and the gradient of the model parameters ; can be derived as:

$$\frac{@L_{all}}{@(\ ;\ )} = \frac{@[\sum_{n=1}^{N} \sum_{i=1}^{I} \ _i L(y_n; \hat{y}_{slice_i})]}{@(\ ;\ )}; \quad (4)$$

where $I$ denotes the number of slices. Compared with the classical model parameters update process, now the parameters are obviously affected by an additional explicit weighting factor , which is intended to assign higher importance to $slice_i$ with higher $_i$. However, it is interesting to notice that in this form, the $\sum_{i=1}^{I}$ is not the only affecting weighting factor, i.e., both and $\sum_{i=1}^{I} L(y_n; \hat{y}_{slice_i})$ will affect the aggregation of important information in the intermediate representation.

First, let us assume an extreme scenario, where the effect of in the parameters updating process is eliminated. Then, it is easy to notice a latent weighting factor will be applied unintended due to the integration of $\sum_{i=1}^{I} L(y_n; \hat{y}_{slice_i})$. As shown in Fig. 1(b), the progressive slicing algorithm will divide the model $M$ into multiple slice based models with shared parameters of and $_{k+2;S}$, where $_{k+2;S}$ denotes the parameters of layer $L_{k+2}$ to $L_S$. The incremental dimension of $slice_1$ to $slice_I$ causes the number of parameters contained in its corresponding layer $L_{k+1}$ to increase. For modern neural networks, the larger models tend to have better performance [45]. Therefore, the loss corresponding to each slice based model $L(y_n; \hat{y}_{slice_i})$ holds the following rule:

$$L(y_n; \hat{y}_{slice_i}) \quad > \quad L(y_n; \hat{y}_{slice_{i+1}}); \quad (5)$$

where $1 \quad i \quad I \quad 1$. Then, if define $L^0(y_n; \hat{y}_{slice_i})$ as the contribution to the loss only by the current $i_{th}$ section without previous parameters, i.e. $L^0(y_n; \hat{y}_{slice_i}) = L(y_n; \hat{y}_{slice_i}) \ L^0(y_n; \hat{y}_{slice_{i\ 1}})$, with simple algebra we will have:

$$L(y_n, \hat{y}_{slice_i}) = \sum_{j=1}^{i} L'(y_n, \hat{y}_{slice_j}). \quad (6)$$

Note that $L'(y_n, \hat{y}_{slice_1})$ is equivalent to $L(y_n, \hat{y}_{slice_1})$. Then, eq.(4) can be rewrite as:

$$\frac{\partial L_{all}}{\partial (\ ,\ )} = \frac{\partial [\sum_{n=1}^{N} \sum_{i=1}^{I} \sum_{j=1}^{i} L'(y_n, \hat{y}_{slice_j})]}{\partial (\ ,\ )} \Big/ \frac{\partial [\sum_{n=1}^{N} \sum_{i=1}^{I} (I - (i-1)) L'(y_n, \hat{y}_{slice_i})]}{\partial (\ ,\ )}. \quad (7)$$

It is easy to notice that a latent weighing factor $I - (i-1)$ has already been involved even without . Obviously, this factor is linearly decreasing, and is also assumed as a monotonically decreasing distribution, i.e., the first slice will always be transmitted first while the other will have lower priority. Then, manually enlarged loss magnitude will be assigned to the front slices, which will further lead to a higher gradient magnitude. As a result, the information located at the head in the intermediate representation will dominate the performance of the model [46], i.e., be more important, which is the motivation of CLIO. Consequently, is not the essential factor in determining the ordered importance of intermediate representation. Even with the absence of , the monotonicity of the converged importance distribution can still be guaranteed with the progressive linear combination. In other words, is actually a penalization factor, which will further amplify the difference between the importance.

To demonstrate these facts, we employ the lightweight convolutional neural network MobileNetV2 [47] on the CIFAR-10 dataset [48] and replicate CLIO with three different optimization strategies, which are without pre-set , the probability of transmittable slices at the current bandwidth as , and the random . To characterize the importance distribution of intermediate representation concretely, we measure the relative importance of each channel in the intermediate representation by calculating the sum of absolute weights [31] for each filter in the intermediate layer, formulated as:

$$R_k = \sum_{i=1}^{n_i} |F_{i,j}| = \|F_{i,j}\|_1; \quad (8)$$

where $F_{i,j}$ denotes the 3D filters of intermediate layer and $n_i$ denotes the number of input channels for the filters.

Fig. 2(a) illustrates the relative importance of each channel in the intermediate representation corresponding to the three different optimization strategies. Obviously, the relative importance of each channel in the intermediate representation is showing a decrease regardless of . Nevertheless, if also obeys the decreasing trend, it will amplify the importance difference among channels in the intermediate representation, which can be observed in the area of Fig. 2(a) and Fig. 2(b).

Moreover, it is essential to note that the high-to-low importance distribution tended to induce the complete intermediate


Fig. 2. (a) The importance distribution of intermediate representation for CLIO in different scenarios. (b) The impact of each channel in intermediate representation on inference accuracy.

representation being assigned the smallest penalization factor, which degraded the overall inference performance compared to the original model before partitioning. To elaborate, the involvement of the penalization factor cannot aggregate important information in the intermediate representation and guarantee the overall inference accuracy of the model at the same time, i.e., a manually involved trade-off, which can be simply noticed in the area of Fig. 2(b).

Notwithstanding, the essential solution to support an efficient distributed deep learning model is to transmit the important information first, while the less important information will be adaptively pruned during transmission. This can improve the inference performance of the system under constrained bandwidth, which can be essentially recognized as a pruning solution but pruned by the limited bandwidth. To satisfy this target, the algorithm must know which bits contain more important information and should be transmitted with higher priority. The state-of-the-art work in CLIO chose to retrain the deep model with a specially designed loss function to force

Fig. 3. The intermediate representation distribution for different original model.

the important information relocated to the front bits of the intermediate representation, which, as discussed, suffers from both theoretical and implementation limitations. However, as the proposed method in CLIO is essentially a pruning-like operation, there should be a more efficient method hinted by the mature pruning algorithm, which is expected to keep the advantage of CLIO while avoiding the limitations.

## IV. NAIR: Naturally Aggregated Intermediate Representation

In this work, we propose a novel method named Naturally Aggregated Intermediate Representation (NAIR) following the well-investigated pruning principles, which fine-tunes the DNN model to identify and amplify the difference in importance of intermediate representation. This contributes to its name of naturally. As no retrain is involved, the massive number of tail models to optimize the performance of various slice$_i$ can be avoided. Similarly, as no negative is involved, the trade-off of the overall inference performance can also be avoided.

Given a well-trained deep model $M$, which is assumed to contain $S$ layers, and the partition point is located at the intermediate layer $L_k$ to divide it into head and tail parts. The output of $L_k$ is referred to as the intermediate representation $O_k$. To start, we slice the $O_k$ into multiple sub-vectors $O_k^i$ with the same $i$ channels. Suppose the number of sub-vectors is $J$, and the sequence of relative importance $R_k$ can be obtained through eq.(8):

$$R_k = \{R_k^1, R_k^2, \ldots, R_k^J\}. \tag{9}$$

It is well known that the importance of the intermediate information of $O_k$ is randomly distributed in nature, as demonstrated in Fig. 3. Instead of the complex retraining operation to force the importance to form a monotonicity distribution, we can simply sort the $R_k$ according to the relative importance to obtain an ordered sequence $R_k^{sort}$ with descending importance and record the channel-indexed sequence $I_k$ denoted as:

$$R_k \xrightarrow{sort} I_k = \{I_k^1, I_k^2, \ldots, I_k^J\}. \tag{10}$$

Fig. 4. Example architecture of NAIR when the number of $O_k^i$ is set to three.

With this index vector, the sub-vectors $O_k^i$ can be assembled into transmission slice $O_k^{I_1:I_j}$ with $i \times j$ channels, whose importance is following a high-to-low pattern. The dimension of the last slice $O_k^{I_1:I_J}$ is identical to $O_k$, except that it has ordered importance. Under constrained bandwidth, the slices with higher importance will be given priority for transmission to the tail model, which can also have a similar effect of adaptive pruning. Note that the above sorting process has been proposed in our previous work [49], which generates a controlled intermediate representation without compromising the overall accuracy. However, this method may still not suit the scenario with constrained bandwidth and the extensive resource consumption caused by multiple tail models still persists, which contributes to this work. Out of these considerations, a few bytes containing the index vector $I_k$ are additionally inserted into the packet for transmission. Then, these adaptively pruned slices can be selectively reassembled to the correct locations where they originally belonged in $O_k$. Subsequently, it will be padded with the zero vectors $O_{zero}$ to obtain the padded slice $O_k^{I_1:I_j;zero}$, which shares the same dimension with the original intermediate representation. As for the last transmission slice $O_k^{I_1:I_J}$, no padding operation is needed, but selectively reassemble is required to restore the original order. This procedure is named the selective assembly algorithm, which can effectively circumvent the

implementation limitation imposed by the multiple tail models. The execution flow is shown in Fig. 4 when $J$ is set to three, and the symbols mentioned above are listed in Table I with their meaning.

However, the intermediate representation from a well-trained DNN model commonly has filter importance without significant differences. Even though the scheme proposed in [17] can enlarge such difference, it manually involves a negative coefficient will compromise the overall inference performance of the model. To fill this gap, we employ a loss function which can naturally aggregate the important information in the intermediate representation. First, an importance weight is designed as:

$$= \begin{cases} (1 \quad R_{k;nor}^{v_j})^{w_{non}} & \text{if } 1 \quad j < J \\ 1 & \text{if } j = J \end{cases} \tag{11}$$

where the $w_{non}$ denotes the non-linear mapping weight, and the $R_{k;nor}^{v_j}$ is derived by using the relative importance of the complete intermediate representation as a scaling factor to normalize the relative importance of each transmission slice $O_k^{I_1:I_j}$. However, the increasing number of channels embedded in the transmission slices results in a low-to-high pattern of $R_{k;nor}^{v_j}$. Consequently, we compute the difference between $R_{k;nor}^{v_j}$ and $R_{k;nor}^{v_j}$ to obtain the weights following a decreasing distribution for aggregating the important information. Then, the weights are subjected to a non-linear mapping to serve as to further amplify the difference in importance among the slices. It is worth noting that the importance weight assigned to the complete intermediate representation is set as a constant value of 1 to guarantee the overall inference performance of the model.

In addition, a penalty has been formulated to suppress the unimportant information in the intermediate representation during the fine-tuning process, which is defined as:

$$= \sum_{i=2}^{J} R_k^{sort;i} : \tag{12}$$

Now, we define the selective assembly function as $O_k^{I_1:I_j;zero} = (O_k; O_{zero}; I_k)$ to map the intermediate representation $O_k$ to padded slices. The function $\hat{y}_j = g_t(O_k^{I_1:I_j;zero})$ is defined to represent the computational process of slices to be fed into the tail model that will generate the final prediction. Then the overall loss function that we are optimizing can be defined as:

$$_h; \quad _t = \arg\min_{h; t} \sum_{j=1}^{J} \quad _j [L_j(D; _h; _t)] + : \tag{13}$$

$$L_j(D; _h; _t) = \sum_{n=1}^{N} L(y_n; g_t( (f_h(x); O_{zero}; I_k))): \tag{14}$$

The model iteratively optimizes the loss function corresponding to each slice $O_k^{I_1:I_j;zero}$, thus generating an intermediate representation with aggregated important information. Deriving the gradients of the head and tail model parameters $_h, _t$ through the SGD algorithm will have:

$$G_{h; t} = \frac{\partial \left[ \sum_{n=1}^{N} \sum_{j=1}^{J} \quad _j L(y_n; \hat{y}_j) + \right]}{\partial( _h; _t)}: \tag{15}$$

In comparison with eq.(4), it can be seen that the negative coefficient which degrades the overall inference performance is converted into the importance weight As demonstrated in eq.(11), the importance weight will always ensure that the weight corresponding to the overall intermediate representation is the highest when amplifying the difference of importance. This design avoids the trade-off between effective aggregation of important information and overall inference performance. With the convergence of the model parameters, the penalty will continuously suppress the intermediate information with lower relative importance. In other words, the NAIR will strengthen the important information and weaken the unimportant information in the intermediate representation to improve the inference performance of the model under constrained transmission bandwidth.

Given any high-dimensional data from advanced sensors, the head model will now compute the intermediate representation with aggregated important information. With the known importance metrics, the intermediate representation can be simply reassembled into multiple slices that contain intermediate information with ordered importance. Under constrained

TABLE I
LIST OF SYMBOLS

| Symbol | Meaning |
|---|---|
| $O_k$ | Intermediate representation |
| $O_k^i$ | Sub-vectors of $O_k$ with $i$ channels |
| $R_k$ | Relative importance sequence related to $O_k$ |
| $I_k$ | Channel-indexed sequence obtained by sorting $R_k$ |
| $O_k^{I_1:I_j}$ | Transmission slice with $j$ channels |
| $O_{zero}$ | Zero vector |
| $O_k^{I_1:I_j;zero}$ | Padded slice with the same channels as $O_k$ |
| $F_{a;b}$ | Filter units of the intermediate layer |
| $M$ | Number of slices |
| $w_{non}$ | Non-linear mapping weight |
| $N$ | Number of images in data set |
| epoch | Number of training iterations |
| | Importance weight |
| | Penalty, initial value is 0 |
| $_h, _t$ | Model parameters |
| $cat(\ )$ | Concatenation operation |
| $\hat{I}_m$ | Channel-indexed sequence of received slice |
| $R_k^{sort}$ | Relative importance sequence with descending order |
| $R_k^v$ | Relative importance sequence related to $O_k^{I_1:I_j}$, $R_k^{v;1} = R_k^{sort;1}$ |

bandwidth, as much important information as possible will be transmitted and padded with zero vectors to feed into the single-tail model. On the other hand, the unimportant intermediate information is adaptively pruned. Consequently, the proposed method NAIR is essentially a pruning process, which naturally amplifies the difference of importance of the intermediate representation without compromising the overall inference accuracy of the mature model. The overall algorithm has been categorized into implementation and training, presented with Alg.1 and Alg.2, and the symbols are listed in Table I with their meaning.

**Algorithm 1** Distributed Inference Algorithm

$sort[I_1 : I_m] =) \quad \hat{I}_m \quad I_k$
for $I_j$ in $\hat{I}_m$ do
  if $I_j = 1$ then
    $\hat{O}_k = O_k^{I_j}$
  else
    $\hat{O}_k = cat(O_{zero}; O_k^{I_j})$
  end if
  if $I_{j+1} \quad I_j > 1$ then
    $O_k^{I_1:I_m;zero} = cat(\hat{O}_k; O_{zero})$
  else
    $O_k^{I_1:I_m;zero} = cat(\hat{O}_k; O_k^{I_{j+1}})$
  end if
end for

**Algorithm 2** Training Algorithm

$R_k^m = \sum_{a=1}^{n_i} |F_{a;b}| = \|F_{a;b}\|_1$
$R_k^{sort} \ (= sort[R_k^1 ::: R_k^M] =) \quad I_k$
for $m = 2$ to $M$ do
  $R_k^{v;m} = R_k^{v;m \ 1} + R_k^{sort;m}$
end for
if $1 \quad m < M$ then
  $= [1 \quad (R_k^{v;m} / max(R_k^v))]^{w_{non}}$
else if $m = M$ then
  $= 1$
end if
for $i = 1$ to epoch do
  for tensor $x$ in $[img_1 ::: img_N]$ do
    $O_k = f_h(x)$
    $O_k^{I_1:I_m;zero} = \quad (O_k; O_{zero}; I_k)$
    for $m = 2$ to $M$ do
      $= \quad + R_k^{sort;m}$
    end for
    $L_m(D; \ _h; \ _t) = \sum_{n=1}^{N} L(y_n; g_t(O_{I_1:I_m;zero}))$
    $_h; \ _t = \arg\min_{h; t} \sum_{m=1}^{M} \ _m [L_m(D; \ _h; \ _t)] +$
  end for
end for

## V. Experiment Results And Discussion

In this section, we validate the performance of the proposed method by applying it to a state-of-the-art lightweight neural network MobileNetV2 [47] with the CIFAR-10 dataset [48]. The CIFAR-10 dataset consists of 60,000 images, which are all scaled into 32×32 color images. All images have been classified into 10 classes, each with 6,000 images. There are 50,000 images used for training and 10,000 images used for evaluation. The MobileNetV2 is trained on an NVIDIA RTX 3080Ti GPU with Pytorch. The SGD optimizer is applied with a batch size of 80, and the non-linear mapping weight is set to 2. The learning rate is adjusted dynamically through the utilization of the cosine annealing algorithm [50], which avoids the loss function converging to the local optima, and the initial learning rate is set to 1e-2. The importance distribution of the intermediate representation can be acquired through the importance metric proposed in eq.(8) before transmission. Consequently, the transmission efficiency of the data can be guaranteed with the method proposed in [51], which utilizes the Multi-connectivity (MC) technique to elegantly increase the Mean Time To First Failure(MTTFF). The data rate of the wireless transceivers is set to the same 250kbps as in [17]. Our code is available at https://github.com/xyc916/NAIR.

The MobileNetV2 contains multiple bottleneck blocks and uses skip connections to connect the beginning and end of a bottleneck block. In this way, the model has the opportunity to access earlier activations that were not modified in the bottleneck block. Therefore, we set the partition point after any layer or bottleneck block in the MobileNetV2 and do not partition within each block. To reduce the training time when compiling NAIR to MobileNetV2, we choose every second channel of the intermediate representation as a unit in the selective assembly algorithm. Consequently, every two filters in the intermediate convolutional layer will be spliced together as a filter unit in accordance with the output channel dimension to calculate the relative importance.

Fig. 5 provides the accuracy results of MobileNetV2 across various bit error rates, where layer 1 refers to the raw sensor data. We emulate the transmission error of the wireless link with a uniform distribution of data errors ranging from 1% to 50%. Clearly, the model has almost no accuracy loss with only 1% bit error. However, with the increased error rate, different layers will have different performances. Take the scenario with 50% bit error rate as an example. If 50% of raw data was lost, then the inference loss will be more than 70%. If the intermediate representation after layer three is transmitted and suffers from 50% bit error, only 12% inference loss will

Fig. 5. The inference accuracy of the model with various bit error rates

be applied. This demonstrates the primary hypothesis of this work: the information is gradually concentrated along with the convolution layers.

In order to evaluate the performance of NAIR in generating an intermediate representation with aggregated important information, we characterize the importance distribution of the intermediate representation for the original model and NAIR respectively through eq.(8). As depicted in Fig. 6, the blue dashed line represents the importance distribution of the original model, while the blue solid line illustrates the importance distribution of the model that has been ne-tuned by NAIR. It is evident that the trend of the importance distribution for NAIR aligns with the original model. Nothing but the NAIR has magni ed the differences in relative importance, demonstrated by the red lines in Fig. 6. The red lines display the results in descending order of relative importance magnitude, with the dashed line corresponding to the original model and the solid line corresponding to the NAIR. As can be observed, the intermediate information for the eighth channel has the maximum relative importance, and its proportion of importance in the complete intermediate representation is improved from 9% to 21.63%. On the other hand, the less important information in the intermediate representation has been suppressed, such as the proportion of importance for the second channel is reduced from 7.21% to 4.62%. As a consequence, the proposed NAIR has the capability to amplify the difference of importance of the intermediate representation effectively.

Now, the inference performance of NAIR under constrained bandwidth will be validated. We compare the inference accuracy of the distributed deep learning system with three different transmission schemes, i.e., which data will be sent to the second-tier computing device. The rst type is the raw sensor data after data compression, the second is the intermediate representation of CLIO, and the third is the ne-tuned intermediate representation after NAIR. In terms of data compression algorithm, we use a state-of-the-art JPEG-based encoder, DeepN-JPEG [52], which is speci cally engineered for deep learning workloads and therefore performs better than the traditional JPEG for deep learning tasks.

In Fig. 7, we demonstrate the inference performances of the systems for the second-tier devices with limited resources and only deploy a single-tail model. In other words, the deployment scheme of the system in this given scenario is

Fig. 6. The importance distribution of intermediate representation for the original model and NAIR.

Fig. 7. The inference accuracy of distributed deep learning systems with single tail model over the latency domain.

similar to the conventional distributed deep learning system, but with a different solution for the arrangement of intermediate representations. The black solid line shows the inference accuracy under different latencies when transferring the intermediate representation after ne-tuning by NAIR. For CLIO, we acquired the intermediate representations with different aggregation effects by employing two different training strategies. The rst strategy aims to improve the performance of the system under severely constrained bandwidth, and a larger $_i$ is assigned to the head slice to achieve more important information to be aggregated at the head of the intermediate representation. Its accuracy results correspond to the green solid line in Fig. 7. The second strategy is to assign a normal $_i$ to the head slice. Correspondingly, the results of this strategy are depicted by the solid blue line. Both strategies will generate intermediate representations with important information aggregated at the head, although the rst strategy has a more powerful aggregation capability. The red solid line represents the accuracy under different latencies for transmitting the raw sensor data after the DeepN-JPEG compression algorithm.

TABLE II
COMPARISON OF THE INFERENCE ACCURACY FOR SYSTEMS OVER THE LATENCY DOMAIN

| Latency / Method | 262ms | 524ms | 786ms | 1.05s | 1.57s | 2.09s | Overall |
|---|---|---|---|---|---|---|---|
| Original | 16.0% | 20.0% | 28.96% | 62.0% | 77.0% | 83.0% | 93.79% |
| [1]$CLIO^{l}_{mt}$ | 83.54% | 87.93% | 89.12% | 89.67% | 89.77% | 90.1% | 90.29% |
| $CLIO^{n}_{mt}$ | 81.11% | 87.73% | 89.62% | 90.34% | 90.91% | 91.14% | 91.53% |
| [2]$CLIO^{l}_{st}$ | 43.18% | 80.51% | 84.98% | 88.3% | 89.66% | 90.06% | 90.29% |
| $CLIO^{n}_{st}$ | 22.75% | 61.48% | 77.12% | 85.99% | 90.2% | 90.97% | 91.53% |
| NAIR | 91.49% | 93.46% | 93.75% | 93.76% | 93.89% | 93.9% | 93.95% |

[1] $CLIO_{mt}$ represents the deployment of multiple tail models in the second-tier device, with the superscripts $l$ and $n$ denote larger and normal, respectively.
[2] $CLIO_{st}$ represents the deployment of single tail model in the second-tier device.

TABLE III
ABLATION STUDY FOR MAJOR COMPONENT OF NAIR

| Penalty | Importance Weight | Low Latency* | Overall |
|---|---|---|---|
| | | 77.16% | 93.77% |
| X | | 89.44% | 93.78% |
| X | X | 91.49% | 93.95% |

* This table only lists performance improvement in low latency scenario, as that is the focus of NAIR.

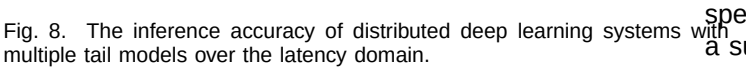

Fig. 8. The inference accuracy of distributed deep learning systems with multiple tail models over the latency domain.

It is evident that NAIR outperforms the CLIO and DeepN-JPEG algorithms, especially at lower latencies. At the latency of 262ms, NAIR has achieved an inference accuracy of 91.49%, representing an improvement of nearly 112% over the optimal accuracy of 43.18% for CLIO. Bene ting from the ef cient information aggregation capability of NAIR, its inference accuracy saturated rapidly. At the latency of 500ms, the accuracy of NAIR has reached 93.46%, approaching the optimal inference accuracy of 93.79% for a well-trained original model. In contrast, CLIO achieves an inference accuracy of 87.72% at the same latency. As presented in Table II, the overall inference accuracy of CLIO is decreased to 91.53% in comparison with the original model, which is caused by the negative coef cient . Simultaneously, the proposed NAIR achieves an overall inference accuracy of 93.95%. This is attributed to the fact that NAIR naturally aggregates the important information without compromising the overall inference performance of the model.

For the sake of fair comparison, we also display the inference accuracy over the latency domain in Fig. 8 for the second-tier device having the capability to deploy multiple tail models. As we can observe, the inference performance of CLIO has improved considerably compared to the prior deployment scheme with the single tail model. The reason is that each slice corresponds to a well-trained tail model that is speci cally designed for it. However, the NAIR still provides a superior performance. As shown in Table II, the NAIR has an improvement of 9% compared to the CLIO in the given low latency scenario. The proposed NAIR effectively ampli es the importance difference of the intermediate representation through the importance weight and penalty , utilizing zero vectors to concatenate the intermediate information with different importance to meet the input dimensionality requirement of the single tail model. This signi cantly reduces the enormous resource requirements caused by deploying multiple tail models and improves the inference performance of the distributed deep learning system under constrained bandwidth.

Table III evaluates the effectiveness of the two major components in the NAIR. Our baseline is a well-trained model that has been ne-tuned using the loss function shown in eq.(13), but removing the effects of penalty and importance weight. We can see that the baseline has an improved performance compared to the original model in the low latency scenario. As mentioned above, this is due to the latent weighting factor (j 1) always enhancing the important intermediate information. We then gradually add the impact of two components, both of which contribute to performance improvement. From Table III, the penalty and importance weight increase accuracy by 12.28% and 2.1% at low latency, respectively.

The overall accuracy shows no degradation compared to the original model. This benefits from the property of NAIR, which naturally amplifies the importance difference of the intermediate representation. Then, we evaluate the effectiveness of the zero-padding module by calculating the parameter counts of the tail models. Assuming that the partition point is located at layer three of the model and choosing every second channel number for slicing, CLIO will deploy twelve tail models to the second-tier device, with a parameter amount of approximately 36.2M. On the other hand, NAIR only requires a single tail model with a parameter count of about 3.1M, which is only 8.6% of the CLIO.

## VI. Conclusion

In this paper, we propose a novel approach named NAIR, which aggregates the important information in the intermediate representation in a simple and efficient manner. Compared to the state-of-the-art work, the proposed method can achieve increased inference accuracy in all latency scenarios, and successfully avoids the overall inference loss in the ideal communication scenarios, i.e., all the slices can be delivered to the tail model. The proposed method also avoids the involvement of multiple tail model to significantly enhance both deployment and retraining efficiency. The proposed work showcases a 112% improved inference performance over the state-of-the-art work in the single-tail model scenario and saves 91.4% parameters of the tail model.

In future work, we are aiming to expand the proposed NAIR method from the special head-tail model into a generalized multi-agent cooperative perception mode [53]–[56]. Most existing cooperative perception methods also assume the ideal communication among multiple agents, which is obviously not true and poses high risks in safety-critical applications. As a result, it was expected to extend this work to design a communication-efficient framework for multi-agent collaboration deep learning systems.

## References


[1] K. He, X. Zhang, S. Ren, and J. Sun, "Deep residual learning for image recognition," in *2016 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2016, pp. 770–778.

[2] A. Krizhevsky, I. Sutskever, and G. E. Hinton, "Imagenet classification with deep convolutional neural networks," *Commun. ACM*, vol. 60, no. 6, p. 84–90, may 2017. [Online]. Available: https://doi.org/10.1145/3065386

[3] A. Afkanpour, S. Adeel, H. Bassani, A. Epshteyn, H. Fan, I. Jones, M. Malihi, A. Nauth, R. Sinha, S. Woonna, S. Zamani, E. Kanal, M. Fomitchev, and D. Cheung, "Bert for long documents: A case study of automated icd coding," 2022. [Online]. Available: https://arxiv.org/abs/2211.02519

[4] D. Li, A. S. Rawat, M. Zaheer, X. Wang, M. Lukasik, A. Veit, F. Yu, and S. Kumar, "Large language models with controllable working memory," 2022. [Online]. Available: https://arxiv.org/abs/2211.05110

[5] S. Wang, R. Clark, H. Wen, and N. Trigoni, "Deepvo: Towards end-to-end visual odometry with deep recurrent convolutional neural networks," in *2017 IEEE International Conference on Robotics and Automation (ICRA)*, 2017, pp. 2043–2050.

[6] C. Chen, C. X. Lu, B. Wang, N. Trigoni, and A. Markham, "Dynanet: Neural kalman dynamical model for motion estimation and prediction," *IEEE Transactions on Neural Networks and Learning Systems*, vol. 32, no. 12, pp. 5479–5491, 2021.

[7] V. Sze, Y.-H. Chen, T.-J. Yang, and J. S. Emer, "Efficient processing of deep neural networks: A tutorial and survey," *Proceedings of the IEEE*, vol. 105, no. 12, pp. 2295–2329, 2017.

[8] L. Li, K. Ota, and M. Dong, "Deep Learning for Smart Industry: Efficient Manufacture Inspection System with Fog Computing," *IEEE Transactions on Industrial Informatics*, vol. 14, no. 10, pp. 4665–4673, 2018.

[9] E. Baccour, N. Mhaisen, A. A. Abdellatif, A. Erbad, A. Mohamed, M. Hamdi, and M. Guizani, "Pervasive ai for iot applications: A survey on resource-efficient distributed artificial intelligence," *IEEE Communications Surveys & Tutorials*, vol. 24, no. 4, pp. 2366–2418, 2022.

[10] GreenWave, "GAP8: Ultra-low power, always-on processor for embedded artificial intelligence," 2022. [Online]. Available: https://greenwaves-technologies.com/

[11] Y. He, J. Lin, Z. Liu, H. Wang, L.-j. Li, and S. Han, "AMC: AutoML for Model Compression and Acceleration on Mobile Devices," in *Proceedings of the European Conference on Computer Vision (ECCV) 2018*. [Online]. Available: http://arxiv.org/abs/1802.03494

[12] L. Lai and N. Suda, "Enabling deep learning at the IoT edge," *IEEE/ACM International Conference on Computer-Aided Design, Digest of Technical Papers, ICCAD*, 2018.

[13] C. Buciluundefined, R. Caruana, and A. Niculescu-Mizil, "Model compression," in *Proceedings of the 12th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, ser. KDD '06. New York, NY, USA: Association for Computing Machinery, 2006, p. 535–541. [Online]. Available: https://doi.org/10.1145/1150402.1150464

[14] B. A. Salau, A. Rawal, and D. B. Rawat, "Recent advances in artificial intelligence for wireless internet of things and cyber–physical systems: A comprehensive survey," *IEEE Internet of Things Journal*, vol. 9, no. 15, pp. 12 916–12 930, 2022.

[15] L. U. Khan, I. Yaqoob, N. H. Tran, S. M. A. Kazmi, T. N. Dang, and C. S. Hong, "Edge-computing-enabled smart cities: A comprehensive survey," *IEEE Internet of Things Journal*, vol. 7, no. 10, pp. 10 200–10 232, 2020.

[16] J. Huang, H. Guan, and D. Ganesan, "Re-thinking computation offload for efficient inference on iot devices with duty-cycled radios," in *Proceedings of the 29th Annual International Conference on Mobile Computing and Networking*, 2023. [Online]. Available: https://doi.org/10.1145/3570361.3592514

[17] H. Jin, S. Colin, G. Deepak, M. Benjamin, and K. Heesung, "CLIO : Enabling automatic compilation of deep learning pipelines across IoT and Cloud," in *The 26th Annual International Conference on Mobile Computing and Networking*, 2020, pp. 773–784.

[18] W. Shi, Y. Hou, S. Zhou, Z. Niu, Y. Zhang, and L. Geng, "Improving device-edge cooperative inference of deep learning via 2-step pruning," in *2019 IEEE Conference on Computer Communications (INFOCOM)*, 2019, pp. 1–6.

[19] H. Li, K. Ota, and M. Dong, "Learning iot in edge: Deep learning for the internet of things with edge computing," *IEEE Network*, vol. 32, no. 1, pp. 96–101, 2018.

[20] M. Denil, B. Shakibi, L. Dinh, M. Ranzato, and N. de Freitas, "Predicting parameters in deep learning," in *Proceedings of the 26th International Conference on Neural Information Processing Systems - Volume 2*, ser. NIPS'13. Red Hook, NY, USA: Curran Associates Inc., 2013, p. 2148–2156.

[21] Z. Dong, Z. Yao, A. Gholami, M. Mahoney, and K. Keutzer, "Hawq: Hessian aware quantization of neural networks with mixed-precision," in *2019 IEEE/CVF International Conference on Computer Vision (ICCV)*, 2019, pp. 293–302.

[22] Z. Cai, X. He, J. Sun, and N. Vasconcelos, "Deep learning with low precision by half-wave gaussian quantization," in *2017 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2017, pp. 5406–5414.

[23] B. Jacob, S. Kligys, B. Chen, M. Zhu, M. Tang, A. Howard, H. Adam, and D. Kalenichenko, "Quantization and training of neural networks for efficient integer-arithmetic-only inference," in *2018 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, 2018, pp. 2704–2713.

[24] G. Hinton, O. Vinyals, and J. Dean, "Distilling the knowledge in a neural network," in *Proceedings of the 28th International Conference on Neural Information Processing Systems*, ser. NIPS'15, 2015. [Online]. Available: https://arxiv.org/abs/1503.02531

[25] J. Yim, D. Joo, J. Bae, and J. Kim, "A gift from knowledge distillation: Fast optimization, network minimization and transfer learning," in *2017 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2017, pp. 7130–7138.

[26] L. Zhang, J. Song, A. Gao, J. Chen, C. Bao, and K. Ma, "Be your own teacher: Improve the performance of convolutional neural networks

via self distillation," in *2019 IEEE/CVF International Conference on Computer Vision (ICCV)*, 2019, pp. 3712–3721.

[27] Z. Li, P. Xu, X. Chang, L. Yang, Y. Zhang, L. Yao, and X. Chen, "When object detection meets knowledge distillation: A survey," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 45, no. 8, pp. 10 555–10 579, 2023.

[28] S. Han, J. Pool, J. Tran, and W. J. Dally, "Learning both weights and connections for efficient neural networks," in *Proceedings of the 28th International Conference on Neural Information Processing Systems - Volume 1*, ser. NIPS'15. Cambridge, MA, USA: MIT Press, 2015, p. 1135–1143.

[29] J. Zadnik, M. Makitalo, and P. Jaaskelainen, "Pruned lightweight encoders for computer vision," in *2022 IEEE 24th International Workshop on Multimedia Signal Processing (MMSP)*. IEEE, Sep. 2022. [Online]. Available: http://dx.doi.org/10.1109/mmsp55362.2022.9949477

[30] J. Guo and M. Potkonjak, "Pruning filters and classes: Towards on-device customization of convolutional neural networks," in *Proceedings of the 1st International Workshop on Deep Learning for Mobile Systems and Applications*, ser. EMDL '17. New York, NY, USA: Association for Computing Machinery, 2017, p. 13–17. [Online]. Available: https://doi.org/10.1145/3089801.3089806

[31] H. Li, A. Kadav, I. Durdanovic, H. Samet, and H. P. Graf, "Pruning filters for efficient convnets," in *International Conference on Learning Representations*, 2017. [Online]. Available: https://openreview.net/forum?id=rJqFGTslg

[32] N. Liu, G. Yuan, Z. Che, X. Shen, X. Ma, Q. Jin, J. Ren, J. Tang, S. Liu, and Y. Wang, "Lottery ticket preserves weight correlation: Is it desirable or not?" 2021. [Online]. Available: https://arxiv.org/abs/2102.11068

[33] S. Teerapittayanon, B. McDanel, and H. Kung, "Distributed deep neural networks over the cloud, the edge and end devices," in *2017 IEEE 37th International Conference on Distributed Computing Systems (ICDCS)*, 2017, pp. 328–339.

[34] T. Li, A. K. Sahu, A. Talwalkar, and V. Smith, "Federated learning: Challenges, methods, and future directions," *IEEE Signal Processing Magazine*, vol. 37, no. 3, pp. 50–60, 2020.

[35] E. Baccour, N. Mhaisen, A. A. Abdellatif, A. Erbad, A. Mohamed, M. Hamdi, and M. Guizani, "Pervasive ai for iot applications: A survey on resource-efficient distributed artificial intelligence," *IEEE Communications Surveys & Tutorials*, vol. 24, no. 4, pp. 2366–2418, 2022.

[36] S. Wang, T. Tuor, T. Salonidis, K. K. Leung, C. Makaya, T. He, and K. Chan, "Adaptive federated learning in resource constrained edge computing systems," *IEEE Journal on Selected Areas in Communications*, vol. 37, no. 6, pp. 1205–1221, 2019.

[37] K. Bonawitz, H. Eichner, W. Grieskamp, D. Huba, A. Ingerman, V. Ivanov, C. Kiddon, J. Konečný, S. Mazzocchi, H. B. McMahan, T. V. Overveldt, D. Petrou, D. Ramage, and J. Roselander, "Towards federated learning at scale: System design," 2019. [Online]. Available: https://arxiv.org/abs/1902.01046

[38] Z. Tao and Q. Li, "esgd: Communication efficient distributed deep learning on the edge," in *USENIX Workshop on Hot Topics in Edge Computing*, 2018. [Online]. Available: https://api.semanticscholar.org/CorpusID:51872328

[39] L. WANG, W. WANG, and B. LI, "Cmfl: Mitigating communication overhead for federated learning," in *2019 IEEE 39th International Conference on Distributed Computing Systems (ICDCS)*, 2019, pp. 954–964.

[40] A. F. Aji and K. Heafield, "Sparse communication for distributed gradient descent," in *Proceedings of the 2017 Conference on Empirical Methods in Natural Language Processing*. Association for Computational Linguistics, 2017. [Online]. Available: http://dx.doi.org/10.18653/v1/D17-1045

[41] Y. Xiao, L. Xiao, K. Wan, H. Yang, Y. Zhang, Y. Wu, and Y. Zhang, "Reinforcement learning based energy-efficient collaborative inference for mobile edge computing," *IEEE Transactions on Communications*, vol. 71, no. 2, pp. 864–876, 2023.

[42] H. Choi and I. V. Bajić, "Deep feature compression for collaborative object detection," in *2018 25th IEEE International Conference on Image Processing (ICIP)*, 2018, pp. 3743–3747.

[43] R. A. Cohen, H. Choi, and I. V. Bajić, "Lightweight compression of neural network feature tensors for collaborative intelligence," in *2020 IEEE International Conference on Multimedia and Expo (ICME)*, 2020, pp. 1–6.

[44] S. Itahara, T. Nishio, and K. Yamamoto, "Packet-loss-tolerant split inference for delay-sensitive deep learning in lossy wireless networks," in *2021 IEEE Global Communications Conference (GLOBECOM)*, 2021, pp. 1–6.

[45] T. Narayan, H. Jiang, S. Zhao, and S. Kumar, "Predicting on the edge: Identifying where a larger model does better," 2022. [Online]. Available: https://arxiv.org/abs/2202.07652

[46] S. Vandenhende, S. Georgoulis, W. Van Gansbeke, M. Proesmans, D. Dai, and L. Van Gool, "Multi-task learning for dense prediction tasks: A survey," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 44, no. 7, pp. 3614–3633, 2022.

[47] M. Sandler, A. Howard, M. Zhu, A. Zhmoginov, and L.-C. Chen, "Mobilenetv2: Inverted residuals and linear bottlenecks," in *2018 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, 2018, pp. 4510–4520.

[48] A. Krizhevsky and G. Hinton, "Learning multiple layers of features from tiny images," *Handbook of Systemic Autoimmune Diseases*, vol. 1, no. 4, 2009.

[49] Y.Xiao, Y.Wang, Z.Huang, F.Shen, and F.Qin, "A distributed deep learning system with controlled intermediate representation," in *2023 9th IEEE Smart World Congress (SWC2023)*, 2023, pp. 1–6.

[50] I. Loshchilov and F. Hutter, "Sgdr: Stochastic gradient descent with warm restarts," *arXiv: Learning*, 2016. [Online]. Available: https://api.semanticscholar.org/CorpusID:14337532

[51] T. Hößler, M. Simsek, and G. P. Fettweis, "Mission reliability for urllc in wireless networks," *IEEE Communications Letters*, vol. 22, no. 11, pp. 2350–2353, 2018.

[52] Z. Liu, T. Liu, W. Wen, L. Jiang, J. Xu, Y. Wang, and G. Quan, "Deepn-jpeg: A deep neural network favorable jpeg-based image compression framework," in *2018 55th ACM/ESDA/IEEE Design Automation Conference (DAC)*, 2018, pp. 1–6.

[53] Y. Hu, S. Fang, Z. Lei, Y. Zhong, and S. Chen, "Where2comm: Communication-efficient collaborative perception via spatial confidence maps," in *Advances in Neural Information Processing Systems (NeurIPS)*, vol. 35. Curran Associates, Inc., 2022, pp. 4874–4886. [Online]. Available: https://arxiv.org/abs/2209.12836

[54] R. Xu, H. Xiang, Z. Tu, X. Xia, M.-H. Yang, and J. Ma, "V2x-vit: Vehicle-to-everything cooperative perception with vision transformer," in *Proceedings of the European Conference on Computer Vision (ECCV)*, 2022. [Online]. Available: https://arxiv.org/abs/2203.10638

[55] Y. Liu, J. Tian, N. Glaser, and Z. Kira, "When2com: Multi-agent perception via communication graph grouping," in *2020 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, 2020, pp. 4105–4114.

[56] Y. Liu, J. Tian, C.-Y. Ma, N. Glaser, C.-W. Kuo, and Z. Kira, "Who2com: Collaborative perception via learnable handshake communication," in *2020 IEEE International Conference on Robotics and Automation (ICRA)*, 2020, pp. 6876–6883.

**Yucong Xiao** (Member, IEEE) received the B.S. degree in electronic information engineering from Zhengzhou University, Zhengzhou, China, in 2020. He is currently working toward the Ph.D. degree in signal and information processing with the School of Electronic and Electrical Communication Engineering, University of Chinese Academy of Sciences, Beijing, China.

His research interests include distributed deep learning and multi-agent reinforcement learning.

**Daobing Zhang** (Member, IEEE) received the M.S. and Ph.D. degrees in optics engineering from the Graduate University of the Chinese Academy Sciences, Beijing, China, in 2004 and 2007, respectively.

He is currently a Researcher with the Key Laboratory of Network Information System Technology (NIST), Aerospace Information Research Institute, Chinese Academy of Sciences, Beijing. His research interests include distributed deep learning and pruning.

**Yunsheng Wang** (Member, IEEE) is an assistant professor in the Department of Computer Science at California State Polytechnic University, Pomona. Before join Cal Poly Pomona in 2022, He was an associate professor in the Department of Computer Science, Kettering University. His current research interests include various topics in the application and protocols of wireless networks, Artificial Intelligence of Things (AIoT), Connected and Autonomous Driving, Edge Computing, and Cybersecurity. Dr. Wang regularly published in scholarly journals, conference proceedings, and books. He serves on several editorial boards, including the International Journal of Ad Hoc and Ubiquitous Computing and the IEEE Autonomous Driving Letters. He is the secretary of IEEE Special Technical Community on Autonomous Driving Technologies. Dr. Wang was general chair of The Fifth IEEE International Conference on Connected and Autonomous Driving (MetroCAD 2022). He was the track cochair of ICCCN 2020 and ICA3PP 2016. His research is supported by the US National Science Foundation, the US Department of Justice, and SAE international.

**Xuewu Dai** (Member, IEEE) received the B.Eng. in electronic engineering and the M.Sc. in computer science from Southwest University, China, in 1999 and 2003, respectively, and the Ph.D. from the University of Manchester, U.K., in 2008.

His research interests include robust state estimation, networked control systems and synchronization, and their applications to the time-sensitive Industrial Internet of Things. He is with the Department of Mathematics, Physics and Electrical Engineering, Northumbria University. Prior that, he was a postdoc at the Department of Engineering Science, University of Oxford and the Department of Electronic and Electrical Engineering, UCL. Dr. Dai was awarded the Early Career Research Prize by SWIG UK.

**Zhipei Huang** (Member, IEEE) received the B.Eng. degree in Communication Engineering and the M.Eng. degree in Communication and Information System from the University of Science and Technology of China. She obtained the Ph.D. degree in Communication and Information Systems from the Institute of Electronics, Chinese Academy of Sciences.

She is currently a professor in School of Electronic, Electrical and Communications Engineering, University of Chinese Academy of Sciences. Her recent research interests include physiological signal analysis, digital intelligent rehabilitation, dynamic human health status assessment and intelligent monitoring.

**Wuxiong Zhang** (Member, IEEE) is a professor at Shanghai Institute of Microsystem and Information Technology (SIMIT), Chinese Academy of Sciences. He received the bachelor degree in information security from Shanghai Jiao Tong University in 2008, and the Ph. D. degree from the University of Chinese Academy of Sciences in 2013. He was with a short-term academic exchange to Heriot-watt University in 2011. He has presided/participated in tens of important research project in China, including projects funded by the national natural science foundation of China, the national science and technology major projects, projects funded by the Shanghai Municipal Science and Technology Commission.

His research interest lies in 5G mobile communication technologies, vehicular networks and heterogeneous networks. He has published more than 70 academic papers, and was awarded by China Communications Society in 2019 for his contribution in Communication and Networking.

**Yang Yang** (Fellow, IEEE) received the B.S. and M.S. degrees in radio engineering from Southeast University, Nanjing, China, in 1996 and 1999, respectively, and the PhD degree in information engineering from The Chinese University of Hong Kong in 2002.

He is currently a Professor with the IoT Thrust, the Director of the Research Center for Digital World with Intelligent Things (DOIT), the Associate Vice-President for Teaching and Learning, and the Dean of the College of Education Sciences at the Hong Kong University of Science and Technology (Guangzhou), China. He is also the Chief Scientist of the IoT with Terminus Group, an Adjunct Professor with the Department of Broadband Communication, Peng Cheng Laboratory, and a Senior Consultant with Shenzhen Smart City Technology Development Group, China. Before joining HKUST (GZ), he has held faculty positions with The Chinese University of Hong Kong; Brunel University, U.K.; University College London (UCL), U.K.; CAS-SIMIT; and ShanghaiTech University, China. His research interests include multi-tier computing networks, 5G/6G systems, AIoT technologies, intelligent services and applications, and advanced wireless testbeds. He has published more than 300 articles and filed more than 120 technical patents in these research areas. He has been the Chair of the Steering Committee of Asia-Pacific Conference on Communications (APCC) from 2019 to 2021. He is also serving the IEEE Communications Society as the Chair for the 5G Industry Community and the Asia Region at Fog/Edge Industry Community.

**Ashiq Anjum** (Senior Member, IEEE) is a professor of Distributed Systems at the University of Leicester and the director of enterprise and impact for the School of Computing and Mathematical Sciences at Leicester. His areas of research include data intensive distributed systems, distributed machine learning models and self-adapting digital twins. Prof Anjum has been investigating digital twins to emulate the real time behaviour of mechanical and cyber-physical systems and his research work has been funded through a number of research grants including the current EPSRC projects in AI driven digital twins for net zero and clinical care.

**Fei Qin** (Senior Member, IEEE) received the B.Eng. degree in information engineering from the Huazhong University of Science and Technology in 2004, the M.Eng. degree in electronic engineering from the Beijing Institute of Technology in 2006, and the Ph.D. degree from University College London in 2012. He worked as the Product Manager of Crossbow Technology, Beijing, from 2006 to 2008. He is currently working as a Professor with the School of Electronic, Electrical and Communications Engineering, University of Chinese Academy of Sciences.

His current research interests include the joint optimization method of wireless networks and information system for the industrial applications.