Dcha: Distributed-Centralized Heterogeneous Architecture Enables Efficient Multi-Task Processing for Smart Sensing

Erxiang Ren[1], Cheng Qu[2], Li Luo[1], Yonghua Li[2], Zheyu Liu[3], Xinghua Yang[4], Qi Wei[5], Fei Qiao[6]
[1] School of Electronic and Information Engineering, Beijing Jiaotong University, Beijing, China
[2] School of Information and Communication Engineering, Beijing University of Posts and Telecommunications, Beijing, China
[3] MakeSens AI, Beijing, China
[4] College of Science, Beijing Forestry University, Beijing, China
[5] Department of Precision Instrument, Tsinghua University, Beijing, China
[6] Department of Electronic Engineering, Tsinghua University, Beijing, China
Email:qiaofei@tsinghua.edu.cn

***Abstract*—The rapid development of artificial intelligence (AI) has accelerated the progression of IoT technology into the smart era. Integrating AI processing capabilities into IoT devices to create smart sensing systems holds significant promise. In this work, we propose a distributed-centralized heterogeneous architecture that enables efficient multi-task processing for smart sensing. This architecture improves the operational efficiency of sensing systems and enhances the deployment scalability through collaborative computing across end, edge, and center nodes. Specifically, we partition the network in traditional centralized sensing systems into several parts and perform algorithm-hardware co-design for each part on its respective deployment platform. We developed a sample design to validate the proposed architecture. By implementing a lightweight image encoder, we achieved an 88x reduction in encoder parameters and up to 9873x energy gain, facilitating deployment on resource-constrained devices. Experimental results demonstrate that the proposed architecture effectively reduces overall energy consumption by 0.0573x to 0.0889x, while maintaining robust multi-task inference capabilities. Moreover, energy consumption reductions of 2.88x to 3.22x on edge nodes and 6311.56x to 10037.23x on end nodes were observed.**

***Index Terms*—smart sensing, heterogeneous architecture, LLMs, multi-task**

I. Introduction

Smart sensing is on the rise, especially in some hot areas, such as embodied intelligence, autonomous driving, smart security, etc. In these fields, sensors play an important role in bridging the physical analog world with the electronic digital world. The physical environment will be sampled by the sensors in it, converted into electrical signals, and then transferred to the post-processing system. In traditional sensing systems, sensors are only sources of data, rather than information. Taking visual perception as an example, the sensor images the surrounding environment to generate an image array, and then transfers the array data to the post-processing system. In this process, the image sensor only truly records the surrounding environment and transfers the generated data, without the ability to perceive the information contained in the data. It is the above-mentioned processing paradigm that leads to data redundancy, and the consequent problems of data conversion, transmission bandwidth, power consumption, etc [1].

In response to these challenges, researchers are exploring ways to preprocess data by placing calculations closer to the front, such as near-sensor computing [2] [3] [4] and in-sensor computing [5] [6] [7]. By equipping the sensor with computing power, features are extracted and transmitted instead of raw data. This effectively alleviates the delay and power issues caused by the conversion and transmission of large amounts of raw data. On the other hand, in smart sensing systems, as the number of sensors increases, the burden on the backend processors gradually intensifies. It may even become unbearable, requiring an increase in the number of processors to alleviate this burden. Therefore, it is crucial to research the trade-off relationship between tasks and energy consumption in smart sensing systems. The emergence and development of large language models (LLMs) in recent years have provided an opportunity for addressing these issues, particularly with the advancement of multimodal LLMs, which have brought new developments to the processing of multiple tasks across various modalities.

As shown in Fig.1, the current mainstream Multimodal Large Language Models (MMLLMs) combine high-performance modality feature extraction networks (as backbones) with LLMs. Specifically, this approach uses different backbone networks to preprocess modality information separately and then aligns this information into a format that the LLMs can understand [28] [16] [14] [18]. Intuitively, this method naturally divides the model into three primary components: feature extraction networks, alignment networks, and the LLMs, as shown in Fig.1 (b). These components exhibit significant

The authors would like to acknowledge supports from National Natural Science Foundation of China under grant No. 92164203, 62334006 , and Key Research and Development Program of Xinjiang Uygur Autonomous RegionNo.2022B01008-3 , and Beijing National Research Center for Information Science and Technology.

978-3-9826741-0-0/DATE25/© 2025 EDAA

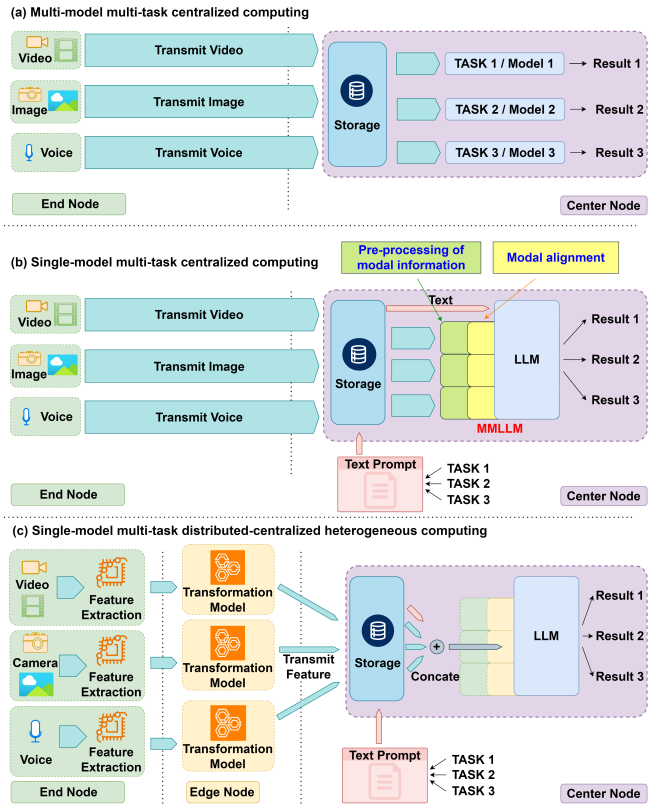


Fig. 1: Compare of there different multi-task solutions. (a) Multi-model multi-task centralized computing. (b) Single-model multi-task centralized computing. (c) Single-model multi-task distributed-centralized computing.

differences in network structure, parameter counts, and operator types. Therefore, separating them and deploying each on a suitable platform is a very promising approach. Performing algorithm-hardware co-design for the aforementioned networks on different platforms to achieve their respective optimal performance and energy efficiency, thereby realizing the best overall system performance and energy efficiency. This approach is more effective and feasible than performing optimization on a single centralized platform. Inspired by this, we propose a **distributed-centralized heterogeneous computing architecture (Dcha)** tailored for smart sensing scenarios. As shown in Fig.1 (c), this architecture uses an end-edge-center collaborative computing paradigm. Information is extracted by modal feature extraction models at distributed ends. This information is then transformed by models deployed at the edge. Finally, the transformed data is aggregated at the center, where powerful pre-trained LLMs are deployed. In this work, when we refer to "end," we mean sensors and near-sensor processors. Therefore, the near-sensor node is part of the end node. Although the aforementioned three-tier segmentation is coarse, it effectively demonstrates the advantages of this architecture. In this work, we focus on researching the impact of deploying the backbone network on the near-sensor node. Specifically, we will analyze the advantages of this architecture when performing multi-task processing. The main contributions of this work are as follows:

- We proposed a **distributed-centralized heterogeneous architecture for smart sensing**, which implements multi-task processing through feature sharing. The proposed architecture takes full advantage of each part to maximize the energy efficiency of the smart sensing system.
- We designed several neural network models suitable for **Dcha**, taking the visual modality as an example, with a focus on **lightweight encoder network designs** to accommodate deployment conditions at near-sensor nodes. We have also explored the impact of different parts of the model on various tasks, aiming to provide guidance for future system optimization.
- **Reusable end node algorithm hardware co-design** for different scenarios and tasks. When task-specific corrections need to be made, fine-tuning the transformation model deployed at the edge while the models in the near-sensor node and the center node are frozen. This feature enhances the flexibility of the proposed architecture and reduces deployment and maintenance costs.

II. Background and Related Works

LLMs have strong logical reasoning ability and a wide range of general knowledge reserves, so they have extremely strong text generation ability and perform well in various natural language processing (NLP) tasks. In recent years, with the explosion of the Generative Pre-trained Transformer (GPT) family, research on LLMs has also received widespread attention. Many excellent LLMs have emerged and begun to be applied, such as GPT-3 [8], InstructGPT [9], ChatGPT [10], and GPT-4 [11]. In addition, there are some excellent open-source LLM works, such as OPT [12], Flan-T5 [13], ChatGLM [14], LLaMA [15], which have significantly promoted the development of NLP and AGI fields.

Although LLMs can achieve good results in text tasks, how to perceive and process information from other modalities is still a problem worth researching. At the same time, high costs come with them. In BLIP-2 [16], the frozen pre-trained image encoders and frozen LLMs are bridged by the introduced Q-former. The Q-former includes a feature generator with a transformer decoder structure for generating image-text features, as well as a text encoder with a transformer encoder structure for assisting with cross-modal alignment training. The results of the Q-former are projected through a linear layer into image-text features suitable for the input format of pre-trained LLMs. These new features are then used by pre-trained LLMs to perform different types of image task inference. Meanwhile, training focuses on Q-former and costs less.

Based on BLIP-2, there are many studies under similar architectures. VisualGLM [17] extends the English model to support Chinese and English, while VisionLLM [18] can complete more visual tasks such as object detection and strength segmentation through instructions. ChatBridge [19] extends the original image text bimodal fusion to visual and audio. The architectures proposed by these works focus on the fusion of visual modality and LLMs. From another perspective, for smart sensing systems, these visual models are most suitable for deployment on the near-sensor node with computing power.

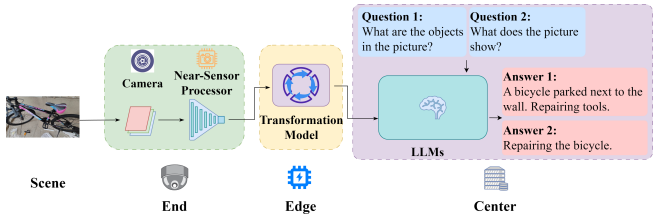


Fig. 2: An example design of the proposed distributed-centralized heterogeneous architecture.

However, these visual models are still too heavy for these ends. Therefore, the goal of this work is to explore the deployment of lightweight models on the near-sensor node, and then build a more bionic and efficient smart sensing system through end, edge, and center collaborative computing.

III. Distributed-Centralized Heterogeneous Architecture

A. Overview

In this work, we take a visual smart sensing system as an example to carry out the research. An example design for the proposed **Dcha** is shown in Figure 2. The subsequent discussion will also be based on this example design. We use a lightweight feature extraction network, specifically a modified ResNet [20], as the image encoder. Q-former [16] is adopted as the transformation model for modal feature alignment. Two LLMs, $OPT_{2.7B}$ [12] and $Flan_{T5xxl}$ [13], can be selectively deployed at the center node for multi-task processing. The processing data flow in **Dcha** is as follows:

❶ Near-sensor processor extract features from the raw data collected by sensors in the end node.
❷ At the edge node, the deployed transformation model generates feature sequences based on features collected from the end node, thereby completing the cross-modal alignment of vision and text.
❸ Concatenate presetting instructions for different tasks and feature sequences from different modalities at the center, as the input for LLMs.
❹ LLMs deployed at the center use the incoming data to complete corresponding tasks.

B. Analysis of Power Consumption

Unlike centralized architecture, distributed-centralized heterogeneous architecture focuses more on reducing system energy consumption by deploying models on high energy-efficiency platforms suitable for their network architecture. In terms of energy consumption, it is related to computation time and power. Device power often does not vary significantly when performing a single task. Therefore, for centralized systems, the total energy consumption E_{CEN} when a single model multi-tasking system performs inference tasks is as follows:

$$E_{CEN} = [\sum_{m=1}^{M}(T_{Encoder_m} + T_{Trans_m}) + T_{LLM}] * P_{Center} \quad (1)$$

For a distributed-centralized heterogeneous system, the total energy consumption E_{Dcha} during the execution of inference tasks can be expressed as follows:

$$E_{Dcha} = \sum_{m=1}^{M} T_{Encoder_m} * P_{End} + \sum_{m=1}^{M} T_{Trans_m} * P_{Edge} + T_{LLM} * P_{Center} \quad (2)$$

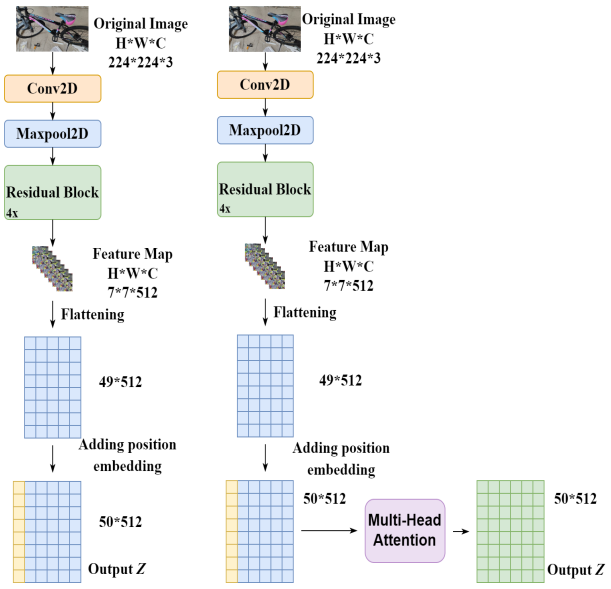
where *M* denotes the number of input modalities that need to be processed. $T_{Encoder_m}$ is the time required for the end to complete sensing and feature extraction on the *m-th* input modality. T_{Trans_m} is the time required for the edge node to perform the transformation model of the *m-th* modality. T_{LLM} is the time of the center to run the LLM. P_{End}, P_{Edge}, P_{Center} are the average power of the end node, edge node, and center node, respectively, when executing the current task. Clearly, a distributed-centralized architecture can leverage various heterogeneous platforms to optimize different nodes, resulting in lower average power and runtime at end and edge nodes, ultimately reducing overall energy consumption. Therefore, we encourage the use of low-power, high-performance front-end devices in such computing architectures to further reduce latency and power, thereby fully enhancing the architecture's performance.

$$N_{Dcha} = \sum_{m=1}^{M}(N_{EE_m} + N_{EC_m}) \quad (3)$$

After feature extraction of modal raw data by the model, the required transmission data volume for modal features is illustrated in Equation 3, where N_{EE} denotes the data volume required from the end to the edge, and N_{EC} denotes the data volume required from the edge to the center. The transmission data volume required for modal features is significantly smaller than that of the raw data. Consequently, under identical communication conditions, the communication energy consumption of the centralized distributed architecture based on feature sharing is bound to be far lower than that of the centralized architecture transmitting raw data. This assertion will be substantiated in the following subsection of this chapter. However, during this process, different communication methods and network designs result in varying reductions in communication energy consumption. Therefore, for the sake of facilitating energy consumption statistics, we disregard the impact of energy consumption during the communication process in this paper, focusing primarily on the impact during the computation process.

C. Near-Sensor Encoder Network Exploration

When deploying a feature extraction network as the encoder on the near-sensor node, it is essential to consider resource constraints and network lightweight. Additionally, the encoder's output must meet the input requirements of cross-attention computations in the transformation model. Therefore, we experimented with ResNet-50/ResNet-18/ResNet-18-A as encoders deployed at the near-sensor node.



(a) ResNet-18 (b) ResNet-18-A

Fig. 3: The scheme of using modified ResNet as the encoder.

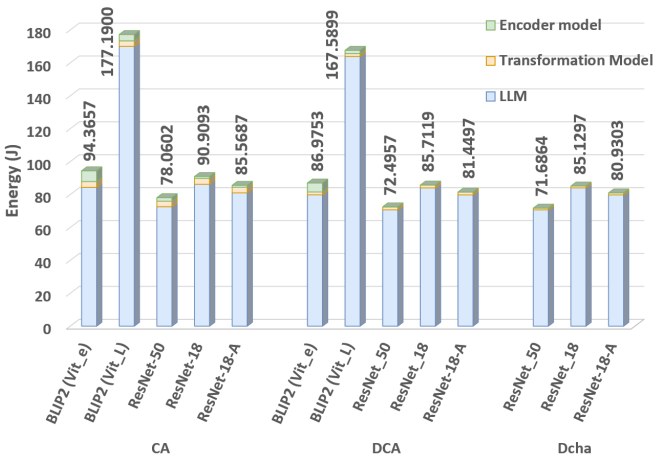


Fig. 4: Comparison of energy consumption during runtime under different architectures. CA: Running on Centralized Architecture. CDA: Running on Distributed-Centralized Architecture. Dcha: Running on Distributed-Centralized Heterogeneous Architecture.

1) ***ResNet-50, ResNet-18****:* Compared with the pure ResNet [20], our ResNet-50 and ResNet-18 replace the last pooling layer and linear layer with an embedding layer. Taking the ResNet-18 as an example, as shown in Fig.3a, the embedding layer first unfolds the three-dimensional (7x7x512) feature map into a two-dimensional (49x512) intermediate version, then adds a position embedding vector as the final outputs (50x512). In this example, the centralized architecture requires a total data volume of $224 * 224 * 3 = 150,528$ points for transmission, whereas N_{EE} is only $50 * 512 = 25,600$ points, and N_{EC} is $32 * 2560 = 81,920$ points. The total required transmission data volume is reduced by 43,008 points, a reduction of 28.57%. Additionally, if the edge node is deployed sufficiently close to the center end, N_{EC} can be neglected, and $N_{Dcha} = \sum_{m=1}^{M} N_{EE_m}$, resulting in a total required transmission data volume reduction of 82.99%. Additionally, in terms of model lightweighting, compared with the VIT-g encoder model, the ResNet-18 encoder model reduces the number of parameters by 88x, and its lighter design makes it easier to deploy on end nodes.

2) ***ResNet-18-A****:* As shown in Fig.3b, the ResNet-18-A adds a self-attention computing layer to the end of ResNet-18. It has been proven that the additional self-attention computing layer is beneficial for improving the accuracy of fine-tuning tasks. But it would reduce the generalization ability of the general model.

D. Model Training

1) ***Pre-training Data****:* We use datasets COCO [21], Visual Genome [22], SBU [23], and CC3M [24] for model pre-training. In our experiments, we totally select 150K image-text pairs from COCO, Visual Genome, SBU, and CC3M to train the model and verify their results on the image caption task and zero-shot Visual Question Answering (VQA) task.

2) ***Two-Stage Pre-training****:* Inspired by BLIP-2 [16], in the first stage of pre-training, we jointly train the image encoder and transformation model through the following three tasks: Image-Text Comparative Learning (ITC) for aligning image representation and text representation, Image-Text Matching (ITM) for learning fine-grained alignment between image and text representation, Image-grounded Text Generation (ITG) loss for training the transformation model to generate text based on input images. We believe that performing cross-modal feature alignment tasks will help the model to understand modal information. We define the loss function for the first-stage pre-training as the sum of losses for each task, as follows:

$$L_{stage1} = L_{ITC} + L_{ITM} + L_{ITG} \tag{4}$$

where L_{stage1} is the total loss for the first-stage pre-training, L_{ITC} represents the loss for the Image-Text Comparative Learning task, L_{ITM} is the loss for the Image-Text Matching task, and L_{ITG} denotes the loss for the Image-grounded Text Generation task. Unlike BLIP-2, the lightweight encoder allows us to train both the encoder model and the transformation model in the first stage, and the training cost does not increase significantly. In the second stage, we train the entire architecture model through Image grounded Text Generation task, where the encoder and transformation model use parameters obtained from the first stage. Parameters within the pre-trained LLM would not be updated during the second stage. In this stage, the training loss function will consist of L_{ITG}.

3) ***Fine-tuning****:* In practical deployment, we can use a pre-trained model in the second stage to infer various visual-language tasks. In addition, the pre-trained model can be fine-tuned on different tasks by inputting different instructions to LLM to improve the accuracy of the targeted task. During fine-tuned training, the training data can be input into the model in the form of image-text pairs: $\{Image\}\{Prompt : "Instruction", Answer : "Answer"\}$. Moreover, the image

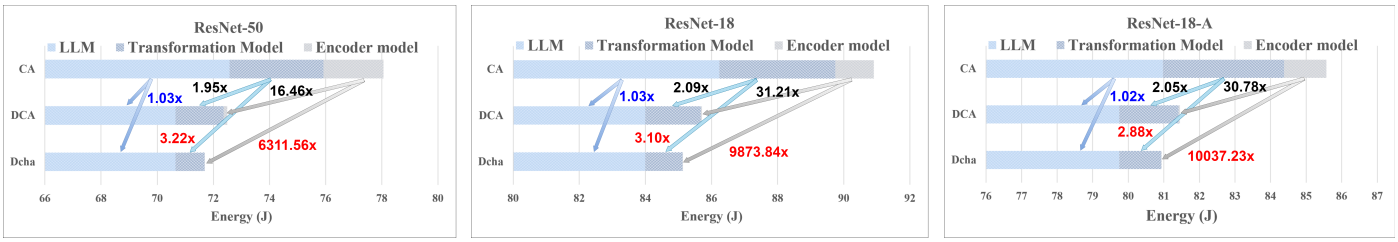


Fig. 5: Comparison of energy consumption during runtime under same networks with different architectures.

encoder can be optionally frozen or unfrozen during fine-tuning, greatly facilitating the deployment and maintenance of the model on the near-sensor node. This is because the image encoder has been well-pre-trained and can extract accurate image features. Therefore, there is no need to further train the image encoder during fine-tuning and training only the transformation model is sufficient to achieve good task performance. Once the model is deployed, the image encoder and related dedicated hardware on the near-sensor node does not need to be adjusted even when different scenarios or tasks change.

IV. Evaluation

To verify the proposed architecture and accompanying network model, we analysed the power consumption comparison between it and centralized systems, as well as the energy advantages brought by lightweight exploration of the near-sensor encoder network. To verify its multitasking ability, we also analysed its performance on zero-shot VQA tasks and fine-tuning image caption tasks. In addition, we assessed the impact of training methods.

A. Energy Consumption Comparison

In the energy consumption comparison, we randomly selected 5000 images from the Visual Genome [22] dataset and deployed different multi-task multi-modal LLMs in centralized architecture (CA), distributed-centralized architecture (CDA), and distributed-centralized heterogeneous architecture (Dcha) for multi-task processing. In terms of encoder models, we tested the VIT-g and VIT-L from the BLIP-2 solution, as well as our proposed network: ResNet-50, ResNet-18, and ResNet-18-A. All these schemes utilized the OPT-2.7B model as the LLM for task inference. It is noteworthy that due to the inapplicability of VIT-g and VIT-L models for near-sensor node (lack of related suitable hardware), we did not conduct Dcha verification for these two schemes.

During the evaluation, the centralized architecture employed a single NVIDIA RTX 3090 GPU for all computations. The centralized-distributed architecture, used an NVIDIA RTX 4080 Laptop GPU for the encoder, an NVIDIA RTX 3090 GPU for the transformation model, and another NVIDIA RTX 3090 GPU for the LLM. In the centralized-distributed heterogeneous architecture, several small-size advanced chips are chosen as candidates for near-sensor node. Specifically, the ResNet-50 was deployed on [31], ResNet-18 and ResNet-18-A were deployed on [32]. The transformation model was deployed on an NVIDIA RTX 4080 Laptop GPU, and the LLM was deployed on an NVIDIA RTX 3090 GPU.

Fig.4 illustrates the energy consumption of these networks in each part, as well as the total energy consumption of the entire system. It is evident that under the same architecture, different model designs exhibit different energy consumption. The BLIP-2 (VIT-g) and BLIP-2 (VIT-L) models [16] consume significantly more energy during image processing by the encoder, whereas our proposed ResNet-50, ResNet-18, and ResNet-18-A encoder schemes are capable of extracting image features with lower energy consumption.

In more detail, as shown in Figure 5, compared to the CA scheme, both the DCA and Dcha schemes effectively reduce the system power consumption during runtime. In our proposed ResNet-50, ResNet-18, and ResNet-18-A Encoder schemes, compared to CA, DCA can reduce the total energy consumption per task by a factor of **5.06%** to **7.68%**, and Dcha can further reduce it to a range of **5.73%** to **8.89%**.

Furthermore, it is observed that after distributing some computational load across different hardware, the power consumption of each hardware component has decreased. For instance, compared to the CA scheme, using DCA can reduce the energy consumption for running the transformation model by a factor of **1.95x** to **2.09x**, and the encoder model part can achieve a power consumption reduction of up to **31.21x**. Additionally, in the Dcha scheme, due to the application of the [31] chip and [32] chip, the energy consumption of ResNet-50, ResNet-18, and ResNet-18-A in the CA scheme can be reduced by **6311.56x** to **10037.23x**, and the transformation model can also benefit from a **2.88x** to **3.22x** reduction in energy consumption. Therefore, compared to the CA scheme, the DCA scheme can effectively reduce the energy consumption of each part of the system during the inference process, and Dcha can further reduce the power consumption through advanced heterogeneous hardware.

B. Multi-Task Processing Evaluation

In order to verify the multi-task processing capability of the proposed model and the influence of each part of the model on the task results, we used the relatively open Zero-Shot VQA task and the Image Captioning task that focuses more on image understanding ability for evaluation.

*1) **Zero-Shot Visual Question Answering Task**:* VQAv2 [27] dataset is used to perform VQA tasks. Fig.6(e) illustrates some representative results on the dataset. Quantitatively, the type of encoders has a slight influence, e.g., the combination of

TABLE I: Comparison of parameters and accuracy

Methods			*#Params*	*Zero-Shot VQA*	*Image Caption*	
	Image Encoder Backbone	Language Model	Image Encoder	VQAv2 val-dev	COCO (Finetuned) *CIDEr [29]*	*BLEU@4 [30]*
Frozen [25]	NF-ResNet-50	-	-	29.6	61.35	20.05
ClipCap [26]	Vit	Transformer	-	-	113.08	33.53
ClipCap [26]	Vit	MLP + GPT2 tuning	-	-	108.35	32.15
BLIP-2	ViT-L	$OPT_{2.7B}$	290.6M	50.1	-	-
BLIP-2	ViT-g	$OPT_{2.7B}$	985.9M	53.5	**145.8**	**43.7**
BLIP-2	ViT-g	Flan-$T5_{xxl}$	985.9M	**65.2**	144.5	42.4
VisionLLM [18]	ResNet-50	Alpaca	-	-	112.4	30.8
VisionLLM [18]	Intern-H	Alpaca	-	-	114.2	32.1
Dcha	ResNet-50	$OPT_{2.7B}$	23.6M	44.50	114.2	33.3
Dcha	ResNet-18	$OPT_{2.7B}$	11.2M	43.09	75.9	23.2
Dcha	ResNet-18-A	$OPT_{2.7B}$	11.2M	42.18	80.2	24.8
Dcha	ResNet-18-A	Flan-$T5_{xxl}$	11.2M	47.82	76.4	23.9

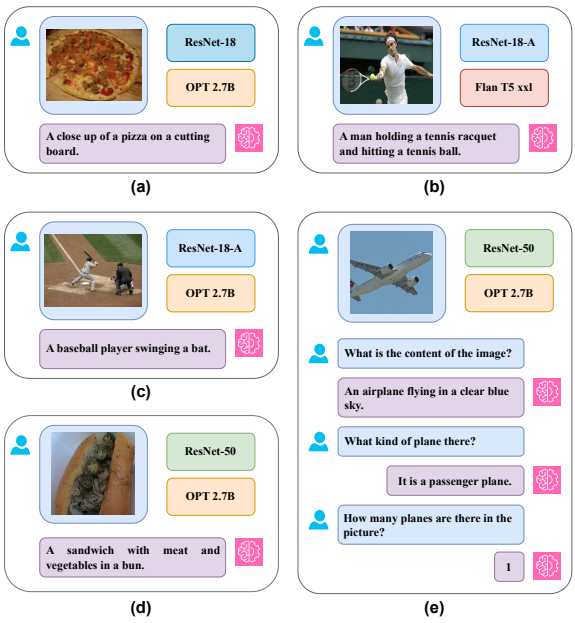


Fig. 6: Results of running image caption (a)∼(d) and VQA (e) tasks.

ResNet-18 and $OPT_{2.7B}$ has a 1.41% lower accuracy on VQA tasks than ResNet-50 and $OPT_{2.7B}$. While the choice of LLMs has a significant impact. For example, with the same ResNet-18-A encoder, $Flan_{T5xxl}$ is 5.64% accurate than $OPT_{2.7B}$.

Compared with the BLIP-2 scheme using ViT-L as encoder, under the same LLM ($OPT_{2.7B}$), the ResNet-50 scheme has a 5.6% decrease in accuracy, as well as the volume of parameters decreases by 12.3x.

*2) **Finetuned Image Caption Task**:* We fine-tune the model for the COCO image caption task. CIDEr [29] and BLEU@4 [30] are adopted as the measurement metrics. Fig.6(a)∼(d) shows some representative image caption results. As shown in Table I, under the same LLM ($OPT_{2.7B}$), different encoders have a significant impact on the results of image caption tasks. The ResNet-50 scheme has the highest accuracy, with CIDEr of 114.2 and BLEU@4 of 33.3. The accuracy of ResNet-18-A is slightly higher than that of the ResNet-18 scheme, which proves that adding a self-attention computing layer helps the model better focus on the task-specific features in the image.

Compared to using a universal model for image caption task inference, the accuracy after fine-tuning training has been improved to a certain extent. In the evaluation, the ResNet-50 scheme has an increase of 11.8 in CIDEr and 4 in BLEU@4 after fine-tuning training.

C. Frozen Encoder Fine-tuning Training

In the proposed Dcha architecture, we employ a strategy that combines freezing and fine-tuning, allowing us to change only the transformation model when tasks change while keeping the encoder and LLM unchanged. For example, experimental results with ResNet-18-A show that, compared to not freezing the encoder, freezing the encoder results in decreases of 0.4%@CIDEr and 0.1%@BLEU, respectively, on caption@COCO task.

V. Conclusion

In this paper, we propose a distributed-centralized heterogeneous architecture, which enables collaborative computing for smart sensing. Under this architecture, the end node equipped with computing power can be integrated into LLMs through a bridge. Distributed numerous "weak smart" sensor nodes converge information to the powerful center, which expands the boundaries of smart sensing systems. This work pragmatically focuses on network deployment at the near-sensor node. We explored several lightweight schemes for encoders, which achieved up to 88x parameter reduction and 9873x energy gains. Moreover, the advantages of multi-task processing in this architecture have also been proved. In the future, efforts should be made to explore the implementation of dedicated hardware that integrates encoders with sensors. In particular, in-sensor computing and near-sensor computing in the mixed-signal domain, which perform information pre-processing at the data source, warrant special attention as their level of intelligent processing determines the effectiveness of downstream tasks.

REFERENCES

[1] Shi, W., Cao, J., Zhang, Q., Li, Y., and Xu, L., "Edge computing: Vision and challenges," *IEEE Internet of Things Journal*, vol. 3, no. 5, pp. 637–646, Oct. 2016.

[2] Eki, R., Yamada, S., Ozawa, H., Kai, H., Okuike, K., Gowtham, H., Nakanishi, H., Almog, E., Livne, Y., Yuval, G., Zyss, E., and Izawa, T., "A 1/2.3inch 12.3Mpixel with On-Chip 4.97TOPS/W CNN Processor Back-Illuminated Stacked CMOS Image Sensor," in *2021 IEEE International Solid-State Circuits Conference (ISSCC)*, 2021, vol. 64, pp. 154-156.

[3] Liu, Z., Ren, E., Qiao, F., Wei, Q., Liu, X., Luo, L., Zhao, H., and Yang, H., "NS-CIM: A Current-Mode Computation-in-Memory Architecture Enabling Near-Sensor Processing for Intelligent IoT Vision Nodes," *IEEE Transactions on Circuits and Systems I: Regular Papers*, vol. 67, no. 9, pp. 2909-2922, 2020.

[4] Liang, F., Cai, C., Zhang, K., Zhang, L., Li, J., Bi, H., Wu, P., Zhu, H., Wang, C., Wang, H., Dong, Z., Luo, C., Luo, Z., Shan, C., Hu, W., and Wu, X., "Infrared Gesture Recognition System Based on Near-Sensor Computing," *IEEE Electron Device Letters*, vol. 42, no. 7, pp. 1053-1056, 2021.

[5] Hsu, T., Chen, Y., Chiu, M., Chen, G., Liu, R., Lo, C., Tang, K., Chang, M., and Hsieh, C., "A 0.8 V Multimode Vision Sensor for Motion and Saliency Detection With Ping-Pong PWM Pixel," *IEEE Journal of Solid-State Circuits*, vol. 56, no. 8, pp. 2516-2524, 2021.

[6] Xu, H., Lin, N., Luo, L., Wei, Q., Wang, R., Zhuo, C., Yin, X., Qiao, F., and Yang, H., "Senputing: An Ultra-Low-Power Always-On Vision Perception Chip Featuring the Deep Fusion of Sensing and Computing," *IEEE Transactions on Circuits and Systems I: Regular Papers*, vol. 69, no. 1, pp. 232-243, 2022.

[7] Ma, T., Boloor, A., Yang, X., Cao, W., Williams, P., Sun, N., Chakrabarti, A., and Zhang, X., "LeCA: In-Sensor Learned Compressive Acquisition for Efficient Machine Vision on the Edge," in *Proceedings of the 50th Annual International Symposium on Computer Architecture (ISCA '23)*, 2023, article no. 54, pp. 1-14.

[8] Brown, T. B., Mann, B., Ryder, N., Subbiah, M., Kaplan, J., Dhariwal, P., Neelakantan, A., Shyam, P., Sastry, G., Askell, A., Agarwal, S., Herbert-Voss, A., Krueger, G., Henighan, T., Child, R., Ramesh, A., Ziegler, D. M., Wu, J., Winter, C., Hesse, C., Chen, M., Sigler, E., Litwin, M., Gray, S., Chess, B., Clark, J., Berner, C., McCandlish, S., Radford, A., Sutskever, I., and Amodei, D., "Language Models are Few-Shot Learners," arXiv preprint arXiv:2005.14165, 2020.

[9] Ouyang, L., Wu, J., Jiang, X., Almeida, D., Wainwright, C., Mishkin, P., Zhang, C., Agarwal, S., Slama, K., Ray, A., Schulman, J., Hilton, J., Kelton, F., Miller, L., Simens, M., Askell, A., Welinder, P., Christiano, P. F., Leike, J., and Lowe, R., "Training language models to follow instructions with human feedback," in *Advances in Neural Information Processing Systems*, 2022, vol. 35, pp. 27730–27744.

[10] OpenAI, "ChatGPT: Optimizing language models for dialogue," 2022.

[11] OpenAI, "GPT-4 Technical Report," arXiv preprint arXiv:2303.08774, 2023.

[12] Zhang, S., Roller, S., Goyal, N., Artetxe, M., Chen, M., Chen, S., Dewan, C., Diab, M., Li, X., Lin, V. T., Mihaylov, T., Ott, M., Shleifer, S., Shuster, K., Simig, D., Koura, P. S., Sridhar, A., Wang, T., Zettlemoyer, L., and others, "OPT: Open Pre-trained Transformer Language Models," arXiv preprint arXiv:2205.01068, 2022.

[13] Chung, H. W., Hou, L., Longpre, S., Zoph, B., Tay, Y., Fedus, W., Li, Y. X., Wang, X., Dehghani, M., Brahma, S., Webson, A., Gu, S. S., Dai, Z., Suzgun, M., Chen, X., Chowdhery, A., Castro-Ros, A., Pellat, M., Robinson, K., Valter, D., Narang, S., Mishra, G., Yu, A., Zhao, V., Huang, Y., Dai, A., Yu, H. K., Petrov, S., Chi, E. H., Dean, J., Devlin, J., Roberts, A., Zhou, D., Le, Q. V., and Wei, J., "Scaling Instruction-Finetuned Language Models," arXiv preprint arXiv:2210.11416, 2022.

[14] Zeng, A., Liu, X., Du, Z., Wang, Z., Lai, H., Ding, M., Yang, Z., Xu, Y., Zheng, W., Xia, X., Tam, W. L., Ma, Z., Xue, Y., Zhai, J., Chen, W., Zhang, P., Dong, Y., and Tang, J., "GLM-130B: An Open Bilingual Pre-trained Model," arXiv preprint arXiv:2210.02414, 2023.

[15] Touvron, H., Lavril, T., Izacard, G., Martinet, X., Lachaux, M. A., Lacroix, T., Rozière, B., Goyal, N., Hambro, E., Azhar, F., Rodriguez, A., Joulin, A., Grave, E., and Lample, G., "LLaMA: Open and Efficient Foundation Language Models," arXiv preprint arXiv:2302.13971, 2023.

[16] Li, J., Li, D., Savarese, S., and Hoi, S., "BLIP-2: Bootstrapping Language-Image Pre-training with Frozen Image Encoders and Large Language Models," arXiv preprint arXiv:2301.12597, 2023.

[17] Du, Z., Qian, Y., Liu, X., Ding, M., Qiu, J., Yang, Z., and Tang, J., "GLM: General Language Model Pretraining with Autoregressive Blank Infilling," arXiv preprint arXiv:2103.10360, 2022.

[18] Wang, W., Chen, Z., Chen, X., Wu, J., Zhu, X., Zeng, G., Luo, P., Lu, T., Zhou, J., Qiao, Y., and Dai, J., "VisionLLM: Large Language Model is also an Open-Ended Decoder for Vision-Centric Tasks," arXiv preprint arXiv:2305.11175, 2023.

[19] Zhao, Z., Guo, L., Yue, T., Chen, S., Shao, S., Zhu, X., Yuan, Z., and Liu, J., "ChatBridge: Bridging Modalities with Large Language Model as a Language Catalyst," arXiv preprint arXiv:2305.16103, 2023.

[20] He, K., Zhang, X., Ren, S., and Sun, J., "Deep Residual Learning for Image Recognition," in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2016.

[21] Lin, T. Y., Maire, M., Belongie, S., Hays, J., Perona, P., Ramanan, D., Dollár, P., and Zitnick, C. L., "Microsoft COCO: Common Objects in Context," in *Computer Vision – ECCV 2014*, 2014, pp. 740–755.

[22] Krishna, R., Zhu, Y., Groth, O., Johnson, J., Hata, K., Kravitz, J., Chen, S., Kalantidis, Y., Li, L., Shamma, D. A., and others, "Visual genome: Connecting language and vision using crowdsourced dense image annotations," *International journal of computer vision*, vol. 123, pp. 32–73, 2017.

[23] Ordonez, V., Kulkarni, G., and Berg, T., "Im2Text: Describing Images Using 1 Million Captioned Photographs," in *Advances in Neural Information Processing Systems*, 2011, vol. 24.

[24] Sharma, P., Ding, N., Goodman, S., and Soricut, R., "Conceptual Captions: A Cleaned, Hypernymed, Image Alt-text Dataset For Automatic Image Captioning," in *Proceedings of the 56th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, 2018, pp. 2556–2565.

[25] Tsimpoukelli, M., Menick, J. L., Cabi, S., Eslami, S. M. A., Vinyals, O., and Hill, F., "Multimodal Few-Shot Learning with Frozen Language Models," in *Advances in Neural Information Processing Systems*, 2021, vol. 34, pp. 200–212.

[26] Mokady, R., Hertz, A., and Bermano, A. H., "ClipCap: CLIP Prefix for Image Captioning," arXiv preprint arXiv:2111.09734, 2021.

[27] Goyal, Y., Khot, T., Summers-Stay, D., Batra, D., and Parikh, D., "Making the v in VQA Matter: Elevating the Role of Image Understanding in Visual Question Answering," in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2017.

[28] Yin, S., Fu, C., Zhao, S., Li, K., Sun, X., Xu, T., and Chen, E., "A Survey on Multimodal Large Language Models," arXiv preprint arXiv:2306.13549, 2024.

[29] Vedantam, R., Zitnick, C. L., and Parikh, D., "CIDEr: Consensus-based image description evaluation," in *2015 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2015, pp. 4566–4575.

[30] Papineni, K., Roukos, S., Ward, T., and Zhu, W. J., "BLEU: a method for automatic evaluation of machine translation," in *Proceedings of the 40th Annual Meeting on Association for Computational Linguistics*, 2002, pp. 311–318.

[31] Kim, S., Li, Z., Um, S., Jo, W., Ha, S., Lee, J., Kim, S., Han, D., and Yoo, H., "DynaPlasia: An eDRAM In-Memory-Computing-Based Reconfigurable Spatial Accelerator with Triple-Mode Cell for Dynamic Resource Switching," in *2023 IEEE International Solid-State Circuits Conference (ISSCC)*, 2023, pp. 256–258.

[32] Wu, P. C., Su, J. W., Hong, L. Y., Ren, J. S., Chien, C. H., Chen, H. Y., Ke, C. E., Hsiao, H. M., Li, S. H., Sheu, S. S., Lo, W. C., Chang, S. C., Lo, C. C., Liu, R. S., Hsieh, C. C., Tang, K. T., and Chang, M. F., "A 22nm 832Kb Hybrid-Domain Floating-Point SRAM In-Memory-Compute Macro with 16.2-70.2TFLOPS/W for High-Accuracy AI-Edge Devices," in *2023 IEEE International Solid-State Circuits Conference (ISSCC)*, 2023, pp. 126–128.