Architecting Efficient Multi-modal AIoT Systems

Xiaofeng Hou
xfhelen@gmail.com
[1]ACCESS – AI Chip Center for Emerging Smart Systems, InnoHK Centers, Hong Kong Science Park
Hong Kong, China
[2]Shanghai Jiao Tong University
Shanghai, China

Jiacheng Liu
jcliu@cse.cuhk.edu.hk
[3]Department of Computer Science and Engineering, The Chinese University of Hong Kong
Hong Kong, China
[2]Shanghai Jiao Tong University
Shanghai, China

Xuehan Tang
xuehantang00@gmail.com
[2]Department of Computer Science and Engineering, Shanghai Jiao Tong University, 800 Donchuan Road
Shanghai, China

Chao Li*
lichao@cs.sjtu.edu.cn
[2]Department of Computer Science and Engineering, Shanghai Jiao Tong University, 800 Donchuan Road
Shanghai, China

Jia Chen
jjiachen@ust.hk
[1]ACCESS – AI Chip Center for Emerging Smart Systems, InnoHK Centers, Hong Kong Science Park
Hong Kong, China

Luhong Liang
luhong@ust.hk
[1]ACCESS – AI Chip Center for Emerging Smart Systems, InnoHK Centers, Hong Kong Science Park
Hong Kong, China

Kwang-Ting Cheng*
timcheng@ust.hk
[1]ACCESS – AI Chip Center for Emerging Smart Systems, InnoHK Centers, Hong Kong Science Park
Hong Kong, China

Minyi Guo
guo-my@cs.sjtu.edu.cn
[2]Department of Computer Science and Engineering, Shanghai Jiao Tong University, 800 Donchuan Road
Shanghai, China

ABSTRACT

Multi-modal computing (M^2C) has recently exhibited impressive accuracy improvements in numerous autonomous artificial intelligence of things (AIoT) systems. However, this accuracy gain is often tethered to an incredible increase in energy consumption. Particularly, various highly-developed modality sensors devour most of the energy budget, which would make the deployment of M^2C for real-world AIoT applications a difficult challenge.

To address the above issue, we propose *AMG*, an innovative HW/SW co-design solution tailored to multi-modal AIoT systems. The key behind *AMG* is modality gating (throttling) that allows for adaptively sensing and computing modalities for different tasks. This is non-trivial since we must balance situational awareness, energy conservation, and execution latency. *AMG* achieves our goal with two first-of-its-kind designs. 1) It introduces a novel decoupled modality sensor architecture to support partial throttling of modality sensors. Doing so allows one to greatly save AIoT power but maintains sensor data flow. 2) *AMG* also features a smart power management strategy based on the new architecture, allowing the device to initialize and tune itself with the optimal configuration. It can predict whether a reasonable degree of accuracy will be satisfied during runtime, and react proactively to remediate the gating process. Extensive evaluation based on our prototype system confirms that AMG improves the AIoT lifespan by 74.5% to 133.7% with the same energy budget while meeting the performance requirements.

CCS CONCEPTS

• **Hardware** → **Power estimation and optimization**; • **Computer systems organization** → **Neural networks**.

KEYWORDS

Edge Artificial Intelligence, Autonomous Embedded Systems, Multi-modal Computing, Modality Gating, Energy Efficiency

ACM Reference Format:
Xiaofeng Hou, Jiacheng Liu, Xuehan Tang, Chao Li, Jia Chen, Luhong Liang, Kwang-Ting Cheng, and Minyi Guo. 2023. Architecting Efficient Multi-modal AIoT Systems. In *Proceedings of the 50th Annual International Symposium on Computer Architecture (ISCA '23), June 17–21, 2023, Orlando, FL, USA.* ACM, New York, NY, USA, 13 pages. https://doi.org/10.1145/3579371.3589066

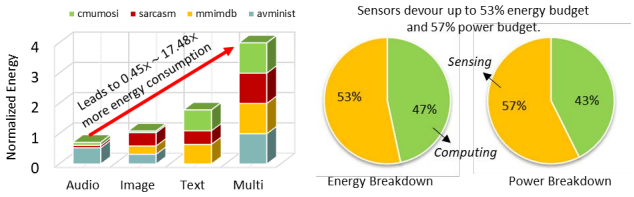
1 INTRODUCTION

Multi-modal computing (M^2C) has attracted tremendous attention [4, 72] in various AIoT domains such as Unmanned Aerial Vehicles (UAVs) [41, 80] and intelligent robots [7]. It leverages a wide variety of modality sensors to capture different environmental data and then feeds the sensing data into perception modules that run multi-modal deep neural networks to generate insightful results [41, 78]. By federating multiple modalities, M^2C has been shown to improve accuracy by up to 30% [4]. Therefore, many multi-modal algorithms and platforms [10, 52, 54] have emerged.

*Chao Li and Kwang-Ting Cheng are the corresponding authors.

ISCA '23, June 17–21, 2023, Orlando, FL, USA

ACM ISBN 979-8-4007-0095-8/23/06.
https://doi.org/10.1145/3579371.3589066



(a) Energy comparison between uni-modal and M^2C.

(b) Energy/power breakdown of sensing and computing phases.

Figure 1: M^2C is power-/energy- hungry.

Despite its accuracy superiority, M^2C incurs incredible energy consumption. As shown in Figure 1-(a), the M^2C task drive power demand higher than its uni-modal counterparts. Particularly, its sensing components devour most of the energy budget, which could impede its deployment in many real-world AIoT scenarios with constrained energy budgets. Typically, a multi-modal AIoT system integrates numerous sensors [10, 64, 78], such as high-resolution cameras and sophisticated microphones to sense the complementary environmental context. These sensors can consume 57% of the total power and 47% of the total energy in a M^2C task as depicted in Figure 1-(b). It has been estimated that a thin-film battery with a form factor of 5.7 mm × 6.1 mm, which stores an electrical charge of 50 µAh or energy of 684 mJ [16], can be easily discharged in about 4 days by a low-power TI MSP430 sensor with a standby power of 0.5 µA [68]. This prompts us to think about this question: *from the perspective of computer architecture and hardware, how can energy-limited multi-modal AIoT systems actually run M^2C tasks in a highly-efficient way*? To date, there is a lack of architectural support and optimization mechanism in this important direction.

One of the most important issues is how to reduce the energy of modality sensors in a way that does not affect situation awareness. Conventional sensors are built with a tightly coupled architecture that consists of three main pipelined modules including a sensor array, an Analog-to-Digital-Converters (ADC) and a digital signal processor (DSP) [13, 15, 48, 69]. They sense and convert environmental signals (e.g., light) into digital data (e.g., images) that can be processed by the perception module without interruption. In order to generate high-quality data, these sensors often implement sophisticated ADCs and DSPs, thus leading to high energy costs [46, 48, 81]. Intuitively we can throttle modality sensors to save energy. However, applying this conventional wisdom to M^2C is ineffective since it loses situation awareness of the associated modalities when turning off a sensor. If we need the modality data and turn on the sensor later, the timestamp of the data will be incorrect. Therefore, new solutions are needed, such as architectures that allow for turning off only the energy-consuming modules while maintaining sensor data flow.

Another critical challenge is how to provide quality M^2C service in complex scenarios in a power-saving mode. Existing M^2C efficiency optimization approaches are based on the fact that modalities exhibit very different importance to a task [4, 80]. They rely on a pre-analysis of all the modality data to select necessary modalities [54, 60], thus reducing the back-end computation effort of multi-modal DNNs. For example, the text modality performs better than visual or auditory modalities in a multimodal language-emotion analysis [2]; in automated vehicle applications, the radar modality may only be needed in extreme weather situations such as fogging and snowing [60]. In a power-saving mode, however, the M^2C system does not have a priori knowledge of all the modality data. Conventional designs would fail if a complex task demands different modality processing to meet accuracy and latency requirements.

In this paper, we argue that the key of energy-efficient M^2C is *modality gating (throttling)* that reduces the energy consumption of M^2C from the sensor side. We propose *AMG*, an adaptive modality gating solution that allows for adaptively sensing and computing modalities for different tasks. At the hardware layer, we introduce a novel decoupled sensor architecture that supports modality semi-gating (i.e., partially throttle the sensor). Instead of turning off the whole sensors, we only gate energy-hungry modules (e.g., ADCs and DSPs) to save energy while maintaining situation awareness. At the system layer, we implement an optimized modality activation mechanism that follows the best modality gating orders and adjusts the M^2C task smartly. It can greatly improve AIoT efficiency while ensuring satisfactory accuracy and latency.

This paper makes the following key contributions:

(1) We present unexploited opportunities for efficiency optimization on multimodal AIoT. We characterize the state-of-the-art `MultiBench` system and show that modality throttling is critical to enable energy-efficient M^2C.
(2) We propose *AMG*, a novel HW/SW co-design solution tailored to M^2C. It introduces a decoupled modality sensor architecture for energy-efficient, non-disruptive modality throttling from the sensor side.
(3) We devise system optimization method for handling complex inference tasks. Our speculative activation scheme can further reduce performance penalty if additional modalities must be progressively activated.
(4) We implement our design as a prototype and construct an adaptive in-situ M^2C system for real-world applications. The extensive evaluation demonstrates that *AMG* can enable highly-efficient in-situ M^2C in a complex environment.

The rest of the paper is organized as follows. First, Section 2 details the background of M^2C. Section 3 introduces the design of *AMG*. Section 4 introduces architectural support. Section 5 introduces system optimization. Section 6 shows experimental methodologies and Section 7 presents evaluation results. Section 8 summarizes related work and finally Section 9 concludes this paper.

2 BACKGROUND AND MOTIVATION

2.1 M^2C Hardware and Software

As shown in Figure 2, M^2C is a complex process that goes beyond the conventional computing stack [60, 78, 80]. A complete M^2C task includes both sensing and computing processes. At the hardware layer, it consists of a set of sensors that collect different modality information from the environment as well as a computing board that processes different modality data. At the software layer, it mainly runs multi-modal DNNs that take the sensing data as input and make inferences to realize different multi-modal AIoT applications.

A very important architectural feature of M^2C is that it relies on multiple modality sensors to obtain the input data [78]. To gain an

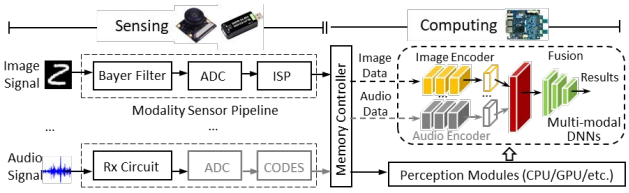


Figure 2: Multi-modal computing hardware and software.

Table 1: In most modality sensing devices, ADC and DSP takes up 84.5% ~ 98.3% of power.

Applications	Power Dissipation (mW)		
	Signal Sensor	ADC	DSP
Image Sensor [13]	~ 0.2uW	~ 10.6uW	~ 9.1uW
General UIS [81]	0.42	4.36	2.17
Doppler UIS [46]	10.56	57.44	
3D UIS [36, 65]	5.68	160	170
PDM Microphone [19]	~ 0.23	~ 0.68	2.16

in-depth understanding of M^2C, we analyze the modality sensor pipeline as shown in Figure 2. A modality sensor is typically built with three main pipelined modules: 1) a sensor array (e.g., Bayer filters for cameras [20], receiver circuits for microphones [53]) that perceives the environmental signals and transfers them into analog electrical signals; 2) an ADC that converts the analog signal into digital signals; 3) a DSP (e.g., ISP for image [48] and CODEC for audio [15]) that pre-processes the digital signals and generates formatted data for the back-end multi-modal DNN inference. It has been reported that the sensor array only takes up a small amount of energy to acquire modality signals [13, 19, 81]. However, modern ADCs and DSPs are typically advanced components designed for producing high-quality data for AIoT applications. According to Table 1, they also consume a significant proportion (84.5% ~ 98.3%) of the overall sensor energy.

2.2 Energy-hungry M^{2}C Task

We evaluate representative M^2C tasks with the latest models and datasets (detailed in Section 6). Our workload encompasses three most common modalities including `image (I)`, `audio (A)` and `text (T)`. We run these tasks on a NVIDIA `Jetson Nano` board.

The M^2C tasks drive power demand higher than the uni-modal counterparts and also significantly increase the total energy consumption. This makes deploying M^2C on edge AIoT devices a difficult challenge. In Figure 3, we collect the average power draw and the total energy consumption of different computing components (including CPUs, GPUs and storages) of each M^2C inference task. We also measure the results of running uni-modal tasks of the same modalities with the same dataset. It shows that the actual power usage of M^2C task is comparable to the results of the uni-modal task. The reason is that the power demand is mainly determined by the number of activated hardware, regardless of the increased computing complexity of the M^2C task. However, the much larger number of multiply-accumulate operations (MACs) in M^2C leads to longer execution duration, thus higher energy consumption. We observe that M^2C shows 0.45× ~ 17.48× more energy compared to the traditional uni-modal tasks.

Figure 3: A comparison of component power/energy consumption under uni-modal ML and M^2C.

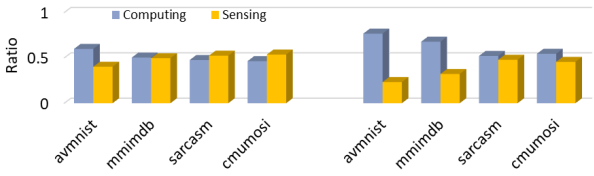


Figure 4: Power and energy distribution of different components for a M^2C inference task.

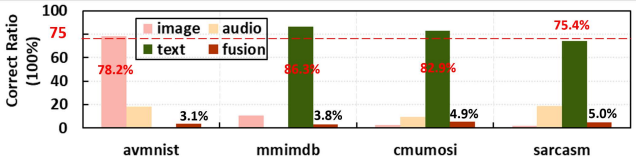


Figure 5: Distribution of mutually exclusive data sample sets correctly processed by different modalities.

2.3 Main Sources of Power Drawn

In Figure 4, we investigate the main sources of power drawn for M^2C tasks. We observe that various sensors for collecting different modalities devour most of the power budget and the energy resource. As shown in the figure, the power consumption of an inference task includes two parts: i) the sensing power and ii) the computing power resulted from MACs and data movements. According to our results, various sensors in a M^2C inference task pull over 40% of the total power of the AIoT system. About half of the energy is consumed by sensors, leaving very few energy for MACs and data movements. This would significantly degrade the productivity of the system. Thus, it can be problematic to turn on all the modality sensors on a energy-limited AIoT device, let alone continuously perform in-situ M^2C tasks.

3 TOWARDS ADAPTIVE MODALITY GATING

The above analysis indicates that throttling modality sensor power drawn is the key to deploying M^2C in piratical AIoTs. While it is intuitive that one can dynamically turn off hardware to save power, applying this conventional wisdom to M^2C is challenging. It requires non-trivial architectural support and system management to achieve the best efficiency, accuracy, and latency. This section provides an overview of our design considerations.

3.1 Opportunities and Challenges

We analyze the characteristics of M^2C and find that one can actually predict the results of a majority of data samples with features learned from a small number of modalities.

	avmnist			mmimdb			cmumosi				sarcasm			
	Image	Audio	Multi	Image	Text	Multi	Image	Audio	Text	Multi	Image	Audio	Text	Multi
Acc	**0.65**	0.42	0.72	0.32	**0.37**	0.56	0.26	0.29	**0.34**	0.35	0.54	0.56	**0.62**	0.62
Fsc	**0.65**	0.42	0.71	0.16	**0.22**	0.48	0.15	0.13	**0.25**	0.32	0.55	0.56	**0.62**	0.62
Eng	**1596**	231.9	1833	1592	**230**	1824	1622	684.2	**2649**	5061	2049	685	**2664**	5434

Figure 6: Accuracy (ACC), F-score (FSC) and energy (NRG) consumption of M^2C and the associated uni-modal models.

We first compare the performance and energy of multi-modal models and their counterparts (uni-modal models). Apparently, fusing features from multiple modalities provides the best accuracy as shown in the *multi* columns of Figure 6. It shows that sometimes a single modality (e.g., `image` for `avmnist` and `text` for `sarcasm`) can yield satisfactory accuracy. This indicates that only not all data samples require a full set of modalities to make correct prediction.

To confirm our observation, we further experiment with a mutually exclusive sets of data samples and count the number of datasets that can be correctly processed with features learned from different modalities. In Figure 5, the `fusion`-bar indicates the number of data samples that can be processed correctly only by fusion. The results show that features learned from a single modality are sufficient for correctly processing most of the data samples (over 75%), only a small fraction of samples (less than 5%) needs a fusion of features. Thus, for a typical data sample, one can safely turn off some modality sensors and omit the associated modality calculations to reduce power and energy consumption without visible accuracy loss.

Nevertheless, whenever we throttle a sensor, it also leads to an embarrassing situation: the AIoT system becomes blind to the associated modality, i.e., the modality data will not be prepared in advance for the current M^2C task. In the absence of complete modality information, one can hardly determine whether or not a typical M^2C task is to be executed. According to Figure 5, failing to fusion enough modalities can cause degraded inference accuracy for some complex M^2C tasks. Although the percentage of such scenarios are relatively low (e.g., 5%), it is unacceptable to ignore them in many mission-critical scenarios such as video surveillance.

3.2 Design Considerations for Efficient Multimodal AIoT

We summarize two key design considerations:

1) **A new modality sensor architecture is necessary, since power throttling on current sensors may disrupt normal M^{2}C execution**. We need to deactivate some sensors for energy conservation. However, we do not want to make the AIoT blind to certain sensitive information, even if the information is associated with the throttled modality sensors. It is desirable to cap sensor power while maintaining situation awareness (raw data streaming).

2) **A new AIoT system management strategy is necessary, since the optimal execution order of each modality is unknown**. One must examine all the modality execution paths to identify the optimal execution order of sensor gating. In addition, dynamic scheduling of modality gating matters for complex tasks. If by any chance the preset configuration does not satisfy results, the cost of resorting to additional modalities can be expensive. It is better to proactively fix the performance penalty issue.

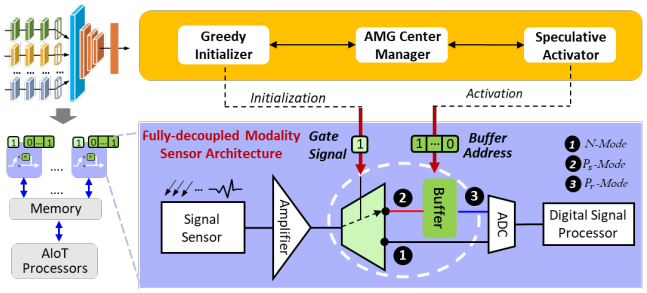


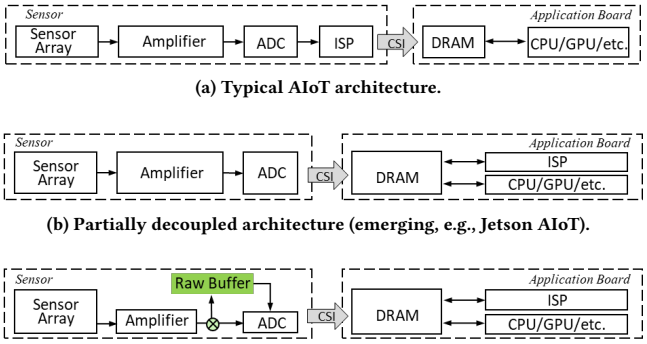
Figure 7: An overview of *AMG*'s SW/HW co-designed modality sensor gating approach.

3.3 Adaptive Modality Gating: An Overview

In this paper we propose adaptive modality gating (*AMG*), a novel HW/SW co-design that augments AIoT systems and enables efficient multi-modal computing. We modulate AIoT system through gating modality sensors. It is different from conventional system power management such as CPU core power gating or DVFS with idle period coalescing. For a given inference task, we neither turn on/off a modality repeatedly in a fine-grained manner, nor delay tasks for energy conservation.

Figure 7 depicts the overall system architecture of our design. *AMG*, for the first time, leverages both hardware and software-based tuning knobs to jointly control the sensing and computing power of each modality for every data sample. At the hardware layer, we introduce architectural support for low-power data sensing. At the software layer, we adaptively orchestrate and control the modality execution paths based on the feedback information from M^2C applications. In total, the proposed design allows power-constrained AIoT devices to execute M^2C tasks while satisfying the accuracy and latency requirements. Specifically, *AMG* addresses the previous design considerations through two innovative approaches.

- **Fully-decoupled Sensor and Modality Semi-Gating**. At the architecture level, *AMG* introduces a fully-decoupled sensor architecture and divides the sensor pipeline into a lightweight frontend that senses the environment and a power-hungry backend that converts the data. We disable the backend of unnecessary modalities and in the meantime temporarily hold the associated analog signal in a buffer. We selectively release backend data streaming based on the M^2C applications' requirement. Such an architecture design allows one to partially throttle the modality sensor pipeline and enable what we call *sensor semi-gating*.
- **Optimized Speculative Sensor Activation**. *AMG* does not activate modality sensor purely on an ad hoc basis. It initializes the AIoT platform with a carefully chosen route of modality activation. We propose a greedy initialization model that could maximize the likelihood of optimal modality gating in a dynamic environment. In case that complex M^2C tasks require additional computation to ensure accuracy, we enhance our design with a speculative activation scheme. *AMG* inserts lightweight probes to check each task's progress. By proactively activating necessary modality sensors and parallelize data processing, we can achieve better design tradeoff with low latency.



(a) Typical AIoT architecture.

(b) Partially decoupled architecture (emerging, e.g., Jetson AIoT).

(c) Fully decoupled architecture (Our design in this work).

Figure 8: Evolution of AIoT architecture.

4 FULLY-DECOUPLED SENSOR ARCHITECTURE

Traditional sensors have a unified, static architecture that does not support dynamic modality gating as shown in Figure 8-(a). In recent years, the trend and benefits of modality pipeline reorganization have been recognized. For example, Figure 8-(b) shows how current AIoT systems move the DSP component from the sensor part to the compute board, which we classify as partially decoupled sensor architecture. We find that doing so provides the AIoT system with much more control over its energy consumption. However, it fails to support adaptive modality throttling since it does not decouple the energy-hungry ADC component. In our *AMG* design, we take one step further and propose *fully-decoupled sensor architecture* as shown in Figure 8-(c). It allows one to store the analog signal of a modality at the sensor array end, and then optionally decides whether to execute the power-consuming ADC/DSP according to the actual requirement of the M^2C task. This architecture supports highly flexible modality gating as shown in Figure 7.

4.1 Modality Semi-Gating Modes

In the fully-decoupled sensor architecture, we use a small buffer to actually decouple the two parts of the modality pipeline. In this way, we can eliminate the high energy consumption of the data sensing process while avoiding raw data loss. By adaptively choosing proper modalities for different data samples, we can greatly reduce the power/energy demand of M^2C tasks.

In our design, we define three sensor operation modes that can be determined by the performed M^2C tasks. Specifically, the sensor has one normal execution mode (❶: N-Mode) and two power semi-gating modes (P-Modes) including the store (❷: P_s-Mode) and restore (❸: P_r-Mode) raw data mode. The N-Mode is the same as the conventional approaches where the sensor acquires and transmits the modality data to the AIoT processor. In the P_s-Mode the sensor temporarily saves the analog signal coming from the signal sensor and amplifier into a buffer. Then, the upper system-level controller (i.e., the AMG central manager) will decide whether to switch the sensor to its P_r-Mode where it performs ADC and DSP processing according to the M^2C application's accuracy requirement.

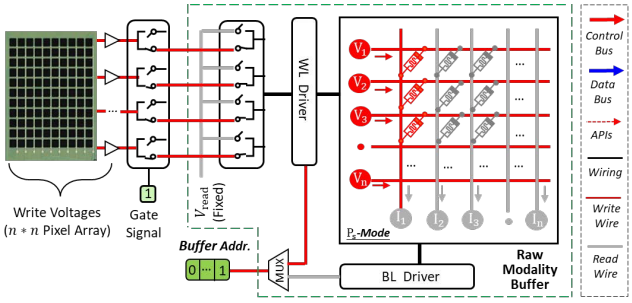


Figure 9: Hardware-based modality sensor modulation.

Table 2: Characteristics of the state-of-the-art non-volatile memory technologies [8, 9, 14, 42]

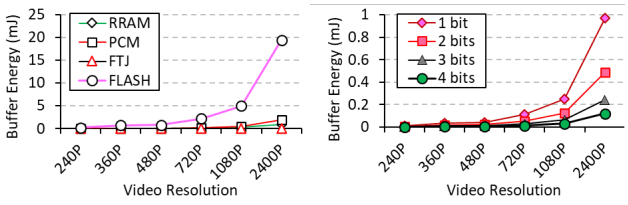
Technologies	RRAM	PCM	FeFET/FTJ	Flash
Cell size(F^2)	4-10	4-10	4-20	4-10
Endurance(Cycles)	10^{12}	10^{10}	10^5	10^5
Speed(ns)	10 ~ 100	10 ~ 100	$\leq$ 10	10^5
Energy(pJ/bit)	1 ~ 10	10	0.01	$\geq$ 100
Bits per cell	1 ~ 8	1 ~ 8	1 ~ 3	1 ~ 4
Price($/Gb)	1000	0.3	$\geq$ 1000	0.014

4.2 Raw Modality Buffer

We leverage a small buffer, i.e., raw modality buffer to preserve the analog signal as shown in Figure 9. The buffer consists of a non-volatile storage array. Each cell of the array is a non-volatile memory device that can hold one or more analog signals. Considering a camera whose sensor is a 256*256 pixel array (each pixel contains 3 RGB signals each of which has 256 states), the buffer needs to represent 256*256*3*8 bits to save the analog signals of a picture. Assuming that a memory cell supports 8 bits, then the buffer size would be 24KB. Generally, the buffer can be implemented with a variety of non-volatile memory devices, as shown in Table 2. In Figures 10 and 11, we estimate the energy and latency overhead of various buffers for video modality with different resolutions with the method provided by prior work [8, 14, 42]. It shows the energy and latency overhead of a buffer relies on its technologies, bits per cell and ADC bits. It is worth noting that the buffer sizes in these two figures can accommodate most modality data as video modality data often requires a larger buffer than other modalities. Basically, it is better to implement the buffer with emerging technologies such as RRAM. Non-volatile storage devices have been increasingly adopted by various AIoT devices [34, 63], making them more attractive.

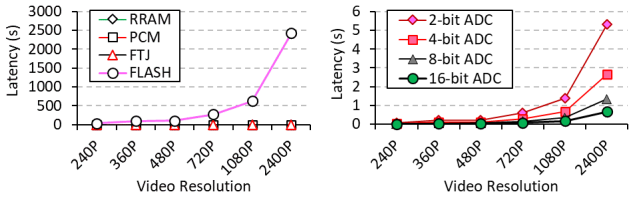
4.3 Sensor Mode Switching

We leverage a 1-bit gate signal register to control the sensor to switch between the N-Mode and P-Mode as shown in Figure 9. The value 0 means working in N-Mode while 1 means P-Mode. The gating signal is transferred to control the 1-to-2 MUX module via the on-chip control bus. When working in the P-Mode, a buffer address register will be used to determine where to store and restore the analog signals. When storing analog signals (in P_s-Mode), the buffer array is programmed with multiple voltage levels generated by the signal sensor. The voltages pass through cells and are converted into different resistance values for storage. If the analog signal needs to



(a) Various technologies. (b) Various bits per cell (RRAM).

Figure 10: Estimated energy overhead of various buffers with different sizes, technology and bits.



(a) Various technologies. (b) Various-bit ADC (RRAM).

Figure 11: Estimated latency of various buffers with different sizes, technology and ADC bits.

be read (in P_r-Mode), a read voltage is added to the buffer array, then the previously stored resistance becomes a current signal, which is then converted into data by the ADC and DSP and transmitted to the AIoT processor to process. In addition, the binary value of the gate register should be larger than the number of sensors, i.e., for n sensors, the gate register should have x bits where $2^x \geq n$. Meanwhile, there should be n buffer address registers for the n sensors. One can implement these registers using `mrs` [5].

5 OPTIMAL MANAGEMENT OF MULTI-MODAL AIOT

Once having the decoupled architecture and the semi-gating mechanism, the next question is how to determine and control the operation modes of sensors. It is not straightforward to realize modality selection since we do not have a priori knowledge of the M^2C task and it requires time-consuming online analysis as well. In this work we devise an optimized sensor activation strategy.

5.1 Greedy Initialization Mechanism

As shown in Figure 7, we make the case for determining the best modality configuration offline and only use online adaptation to further improve performance. Such a design methodology is widely used in embedded system design for its effectiveness and efficiency. The purpose of our initialization is to automatically learn the optimal modality execution orders.

Modality Ordering Strategy: In the initialization phase, our goal is to determine the priority list of hardware modality (initialization configuration) which specifies the order of modality activation during runtime. We adopt a greedy initialization approach in that we want to identify the activation order that benefits the greatest number of data inputs. Figure 12 shows our initialization processes. For AIoT systems with a limited number of modalities, we can

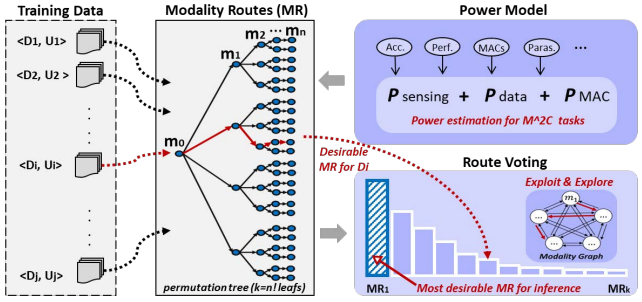


Figure 12: Modality ordering process.

construct a permutation tree that covers all the possible execution orders of modality. We call each execution order as a modality route (MR). Since AIoT applications have different modality preferences, some features play a dominant role in terms of prediction accuracy. During our training process, each data input votes its most desirable MR. Finally, the MR with the most votes will be selected.

If the types of modalities grow, the complexity of the above process (O(n!)) may greatly increase. In this case, we model the problem with a complete graph, where vertices represent different modalities and each edge corresponds to the sequential execution (transition) from one modality to the next. There is a direct transition between any two states that can be executed sequentially. Each transition has a set of metrics that reflect the outcomes of the transition. The graph model provides a resource-conservative representation of the utilities of various modality execution orders. We train the graph model with the classic exploring-and-exploiting approach [44]. To be specific, we develop an evaluator based on `MultiBench` with the extension of an energy model that considers the computing energy of data movement and MAC operations as well as the sensing energy to train our strategy based on the previous work [55]. Each data sample of the training dataset represents an M2C task, denoted by $< D_n, U_n >$, where D_n represents the modality data for this task and U_n is a utility score calculated by the weighted sum of the accuracy, energy and latency (obtained from the evaluator) for the task. The weights are automatically searched using the `optuna` [33] tool. The loss function is the overall loss of the utility score for all the training data samples. After training, the transition with the highest total utility score means that it is the ideal candidate for most of the data samples.

The above ordering process ultimately outputs a modality list, which stores the optimal orders of modalities for different M^2C tasks. Besides, we also obtain a power and performance reference table after the offline profiling process. In the table, we record the empirical power consumption and performance of a modality when it is executed at different layers.

5.2 Speculative Activation Scheme

Because the optimal modality order only statistically satisfies most of the data samples, we must use online adaption to guarantee the accuracy of the inference tasks, particularly for some non-ideal data input. It may cause a key performance issue that we call *modality mismatch*. For some complex tasks, it is not unusual that the first-ranked modality cannot provide the necessary accuracy. Then it would incur a long delay since we need to restore the modality data

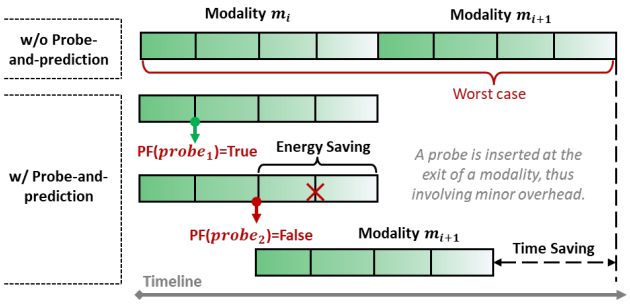


Figure 13: Speculative modality activation.

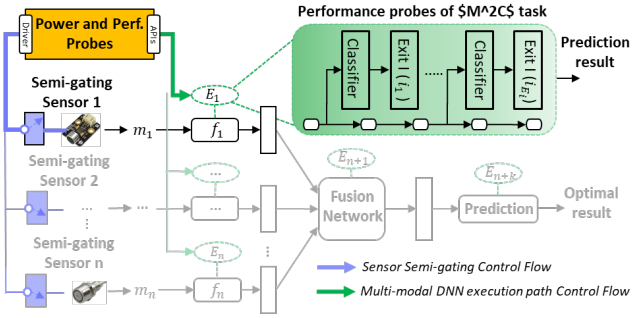


Figure 14: Performance probes of the M^2C task based on the neural network early exit techniques [67].

of more sensors from their raw input buffers and perform extra computation in serial. To address this, *AMG* monitors the current modality's execution progress. Then it uses a speculative activation to proactively restore and compute the data of the next modality that will be computed according to the modality routing list.

As shown in Figure 13, we design an probe-and-prediction controller that determines whether the next modality needs to be computed based on the current performance of the M^2C task and the remaining power budget. If so, the controller will process the data of the next modality in advance. This includes two aspects of optimization. First, it only operates the sensor of the most energy-efficient modality rather than all the sensors in N-Mode which often leads to much higher sensing energy at a time, thus improving energy efficiency and avoiding unnecessary energy waste. Second, it proactively determines whether to restore the raw data of the next sensor and execute the corresponding modality data.

Since DNN applications have a layer-by-layer execution structure, we can easily insert some power and performance probes between different layers. Each probe monitors the current power consumption and the performance such as latency, and accuracy of the M^2C task. To monitor the energy consumption and execution latency of tasks, today's AIoT devices have integrated many system performance monitor tools such as `jtop` [52]. To observe the performance of the M^2C task at the layer level, we add some neural network exits to calculate the accuracy at the observed layer with intermediate features. As shown in Figure 14, we implement the exits by adding a few neural network layers to generate a prediction of the present immediate feature based on the previous work [67].

Algorithm 1: The probe-and-prediction process

Input: A pre-defined multi-modal task M_i, reference table T_{ref}, hierarchical control table T_{ctrl}, and modality routing list R_m.
Initialization: Power on the sensor of the first modality in R_m and power off the other sensors.
for *modality* m_i *in* R_m **do**
  for *Exit* e_j *in* E_{m_i} **do**
    /*Probe-and-prediction strategy*/
    Check the probe flag (PF) at current exit.
    Calculate $PF(probe)$ acc. to Eq. (1).
    /* Hierarchical execution control*/
    if $PF(A) \geq A_{empirical}$ **then**
      Invoke multi-exit DNN APIs.
    if $IF(probe) > 0$ **then**
      if *The last exit of current modality* **then**
        Restore data for the next modality.
      Executing current modality.
    Reserve the state with checkpoint and exit.

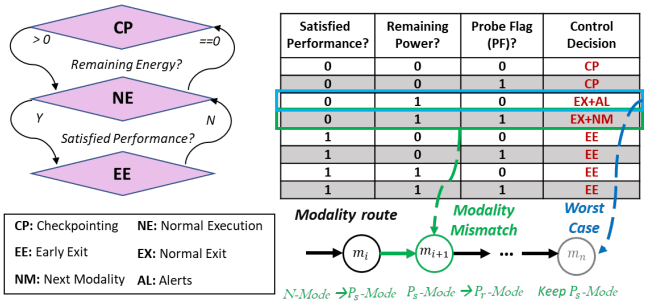


Figure 15: Coordination state diagram for an inference task.

To make the decision, the probe-and-prediction controller compares the data from the probe with the reference table which is built offline. Assume that the power and performance from the probes are respectively denoted by P_{probe} and A_{probe}. The reference power and performance are respectively denoted by P_{ref} and A_{ref}. The power and performance for completing the whole modality are formulated as P_{REF} and A_{REF}. Then, we can predict whether the performance A_{pred} will be satisfied. To this end, we use a probe flag (PF) to illustrate the prediction result and compute its value denoted by $PF(probe)$,

$$PF(probe) = \mathbf{1}(A_{REF} - A_{pred} \leq 0) \tag{1}$$

$$A_{pred} = A_{REF} \times (1 - \frac{(A_{REF} - A_{ref}) \times A_{probe}}{A_{REF}}) \tag{2}$$

where $\mathbf{1}$ is a bool function. We can roughly predict if there is any remaining energy in a similar way. The details of this mechanism are shown in Algorithm 1.

5.3 AMG Central Manager

The *AMG* Central Manager is responsible for coordinating all of them and modulating modality execution. As observed in Section

Table 3: Description of the used M^2C tasks and datasets.

Dataset	Samples	Modality	M^2C Application
`sarcasm`	690	*Language (BERT/GloVe), visual (ResNet), audio (Librosa)*	Affective Computing [12]
`cmumosi`	2,199		
`mmimdb`	25,959	*text (Glove) ; image (VGG)*	Multimedia Computing [26]
`avmnist`	70,000	*image (Raw), audio (Spectogram)*	

Table 4: SOTA M^2C models for performance evaluation.

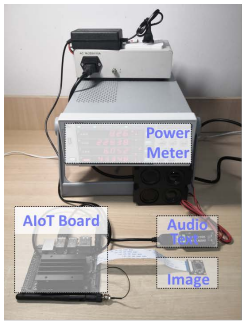
Type	Scheme	Description
Uni-modal models	*image* [24]	Using ResNet to process data from visual modality
	audio [50]	Using the speech processing library Librosa
	text [17, 57]	Processing text modality with BERT/GloVe
SOTA M2C Model	*LF (BL)* [47]	Fusion by concatenation (Baseline)
	TF [79]	Fusion by tensor outer product
	LRTF [49]	Fusion by a modality-specific factors
	MIM [35]	Fusion by inter-modality communication

Table 5: Evaluated power management schemes.

Category	Scheme	Description
Computing Opti. Methods	*CPU-First*	Adjusting CPU frequency at first [40]
	GPU-First	Adjusting GPU frequency at first [22]
	CPU-GPU	Co-adjusting CPU and GPU frequency [39]
	Quan	Using DNN quantization to manage energy [71]
	Prune	Using DNN pruning to manage energy [25]
	Qlearning	Using machine learning to tune different knobs [75]
	OnlySen	Assuming no computing overheads (Ideal)
Sensing Opti. Methods	*Rndm*	Randomly decide modality order (Adaptive acc.)
	Grdy1	Completing current modality (Best modality order)
	Grdy2	Arriving at the final exit (Best modality order)
	AMG	Ours method (Best modality order & adaptive Acc.)

3-A, for most simple data samples, only the first sensor works in N-mode while the others work in P_s-Mode. To handle the very complex execution environment of in-situ M^2C, we might need to switch more sensors into their P_r-Mode. To determine this, *AMG* is further driven by the state diagram shown in Figure 15. In the Central Manager, we define three operating states including checking point, normal execution and early exit. There are three inputs that affect these states, indicating if the execution should be advanced or not. We create checkpoint by interrupting the AIoT execution temporally and save the system states into non-volatile storage. Normal execution means that we compute the modality until the expected performance is achieved. It can be interrupted by a checking point state or an early exit state. Early exit happens when the expected performance has been achieved.

Overall, *AMG* processes multi-modal data stream based on the table shown in Figure 15. It is determined by three parameters, namely, performance objective, remaining energy and the aforementioned probe flag. In our design, a M^2C task is deemed complete if its performance (accuracy and latency) requirement is met; we no longer perform more computation as shown in the last 4 rows of the table. Before the performance is satisfied, we process data based on energy availability. If there is inadequate energy, the system will use a checkpoint to save the execution state and awake the task until the accumulated energy passes a threshold. As shown in the case (i.e., `EX+NM`) marked in green in Figure 15, the modality mismatch occurs if the energy is adequate while the performance is not satisfied. We will restore and compute the data of the next modality using the speculative activation scheme in Section 5-B. If a task has to compute to the last modality which happens less frequently according to our analysis in Section 3-A , i.e., the worst



Power Meter Specifications
High-precision power meter. (Minimum resolution 0.01W, Max sampling rate: 300KS/s)

Modality	Input	Sensor/Software
Visual	Image	Sony's IMX219-77 camera (8 Megapixels)
Vocal	Audio	Waveshare USB sound card (Sampling Rates 48KHz)
Verbal	Text	Audio-to-text converter with DeepSpeeh

Figure 16: Evaluated system prototype and key specs.

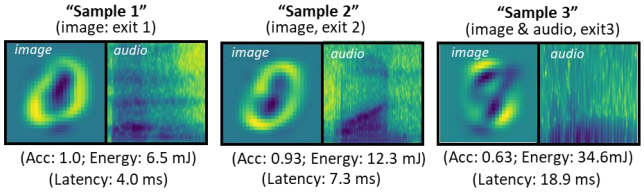


Figure 17: Demonstration of *AMG* when processing data samples of different difficulty from the `avmnist` dataset.

case `EX+AL`, it might result in poor latency for the task. In this case, we need to make a better tradeoff between accuracy and latency, which we do not discuss in this paper.

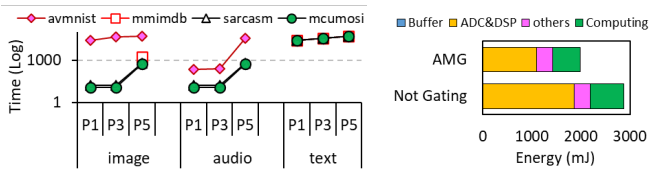
6 EXPERIMENT METHODOLOGIES

6.1 Real-world Applications and Datasets

To validate *AMG*, we use the `MultiBench` [47] which contains a wide range of real-world applications. We choose 2 of the most representative M^2C applications including the multimedia applications and affective computing applications from the `MultiBench`. We use 4 state-of-the-art (SOTA) multi-modal learning datasets as described in Table 3. These 4 datasets have a wide range of data samples from 690 to 70,000 samples. They cover the most common modalities such as `language text`, `image`, `audio`. Among them, the `sarcasm (sa)` is a video corpus used for discovering sarcasm [11]. The `cmumosi (mo)` is a real-world multi-modal dataset for affect recognition which is regularly used in competitions and workshops [70]. The `avmnist (av)` is created by paring the audio of human reading digits from the `FSDD` dataset with written digits in the `MNIST` dataset [58]. The `mmimdb (mm)` is the largest publicly available multi-modal dataset for genre prediction on movies [4].

6.2 Baselines and the State-of-the-arts

We compare *AMG* with representative multi-modal algorithms to verify that it can achieve similar even better performance compared to the SOTA. We consider seven different algorithms, as shown in Table 4. *LF* [47] represents the most common last fusion methods that combine multiple uni-modal representations with the concatenation operation. We also compare *AMG* with both the uni-modal methods and the state-of-the-art multi-modal methods. In each of the *uni-modal models (i.e.,* `image`, `audio` *and* `text`*)*, we only use the



(a) Sensors' activation time. (b) Energy breakdown.

Figure 18: Sensor gating reduces the sensor energy by 40.9% and the inference energy by 19% for a M^2C *cmumosi* task.

encoding network of one modality and connect it to the classification network to obtain the output predictions. *Tensor fusion network (TF)* [79] uses tensor outer product to fuse information from different modalities. *Low rank tensor fusion network (LRTF)* [49] leverages a modality-specific set of low-rank factors to improve the efficiency of tensor fusion. *Multiplicative interaction model (MIM)* [35] further generalizes the tensor products to include learnable parameters for capturing the interactions between different modalities.

To show the superiority of *AMG* in optimizing the energy efficiency of M^2C, we compare our design with two categories of works including 1) various current methods that mainly optimize the computing tasks and 2) derivatives of our design that optimize the sensing components as shown in Table 5.

The existing methods improve the energy efficiency of an autonomous AIoT system from two sides. One is to throttle the frequency of task execution, i.e., sacrificing some latency for efficiency, for example, *CPU-First* [27, 30, 40] and *GPU-First* [22] respectively adjust the execution frequency of CPU and GPU to manipulate energy consumption while *CPU-GPU* [39] jointly adjusts the frequency of both. Another line of baseline is to adjust the amount of DNN computation. For example, *Quan* [71] and *Prune* [25] are the most common DNN compression schemes that reduce the energy consumption of inference by abbreviating network models, while *QLearning* fine-tunes multi-dimensional knobs based on online learning. All the above methods only improve energy efficiency by optimizing the back-end multi-DNN workload. To illustrate their limitations in M^2C, we further implement *OnlySen* which assumes no computational overheads. We also compare *AMG* with several sensing-aware optimization methods including *Rndm*, *Grdy1* and *Grdy2*. Among these approaches, the *Rndm* only uses the confidence score returned by the model to determine the exit threshold, without considering the remaining energy. The *Grdy1* method will complete current modality without exit. The *Grdy2* is a more aggressive method that arrives the final exit.

6.3 Implementation Details

As shown in Figure 16, we implement a prototype bench of *AMG* based on NVIDIA `Jetson Nano`, a representative IoT board with rich interfaces. We add different sensors to the board and use them to collect various signals. We use a Sony's `IMX219-77` camera and a Waveshare's `USB to Audio` to collect signals from the `image` and `audio` modalities, respectively. For the `text` modality, we first use the audio sensor to collect the audio signal and then use the `DeepSpeech` model [3] to convert the audio signal into a text signal. For each modality sensor, We devise a 24 MB raw input buffer (can store 30 frames of 1080P images) which is enough for most modalities. We implement the buffer with an 8-bit per cell RRAM and a 10-bit ADC which is commonly used in current sensors [1]. We simulate it in `NVSim` [18] and observe that the energy and latency of writing/reading the whole buffer are about 0.93mJ and 0.34s respectively. Note that for most modalities, we do not need to read the entire buffer since we design it with a large size. Moreover, with the optimization of the speculation mechanism, the buffer's read latency does not affect the back-end multimodal DNN inference much. We feed the data to our prototype testbench which mimics the behavior of sensor semi-gating. We monitor the full system power with a high-precision meter that follows SPECpower guidelines. We collect the power and energy usage data with the `jtop` tool of `Jetson Nano` during the inference phase.

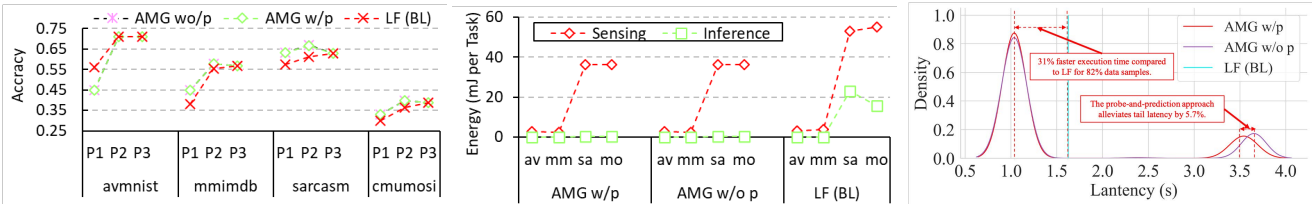
We implement the core components of *AMG* in about 3000 lines of `python` as well as some shell scripts to communicate between different modules. For all the multi-modal models (*LF*, *TF*, *LRTF* and *MIM*), we use the same settings as described in the `MultiBench` [47]. We implement the probes by adding exits to the model architecture of LF. We add 2 exits for `avmnist` and `mmimdb` while 3 exits for `sarcasm` and `cmumosi`. All the evaluated models are implemented in Pytorch and trained in a sever with GeForce RTX 2080Ti GPU. After training, we save them with Pytorch's utilities and deploy these models to the `Jetson Nano` board. We assess task results (accuracy) with the evaluation scripts provided by `MultiBench`. We repeat our experiments multiple times and report the average value.

7 EVALUATION RESULTS

7.1 Effectiveness of AMG

7.1.1 Demonstration of Adaptive Inference. We first illustrate that our proposed *AMG* has the ability to adaptively execute the appropriate modalities for different tasks. In Figure 17, we plot the results for the `avmnist` dataset which classifies data samples based on two modalities including `image` and `audio`. The data samples of `image` are represented in pixels, and the data samples of `audio` are represented with a 112×112 spectrogram. We can see that both Samples 1 & 2 can be classified accurately by exiting from `image` modality. However, for Sample 3, *AMG* has to complete both `image` and `audio` modalities to compute the final prediction. The results indicate that Sample 3 is more complex than Sample 1 & 2, thus requiring a fusion of modalities. *AMG* can predict results for different data samples with optimal computational efforts.

7.1.2 Validation of Sensor Semi-gating Mechanism. We verify the effectiveness of a sensor semi-gating mechanism, in reducing the energy waste of sensors for simple and common data samples. In Figure 18-(a), we count the operation time of different sensors completing the entire sensing pipeline (i.e., in N-Mode or P_r-Mode) under large (P3), medium (P2), and small (P1) capacity of the battery. It shows that *AMG* almost relies on a single modality to accomplish most inference tasks. For example, the activation time percentage when the device should open is 0.14%, 0.14% and 43.31% respectively for `image`, `audio` and `text` under `cmumosi`. In Figure 18-(b), we analyze the energy breakdown of an M^2C `cmumosi` task with/without sensor gating. The result shows that sensor gating reduces the ADC and DSP energy by 40.9% and the inference energy by 19% while the



(a) Accuracy. (b) Per-inference energy. (c) Latency distribution.

Figure 19: *AMG* achieves the same or even better accuracy than traditional M^2C methods, while having a little tail latency.

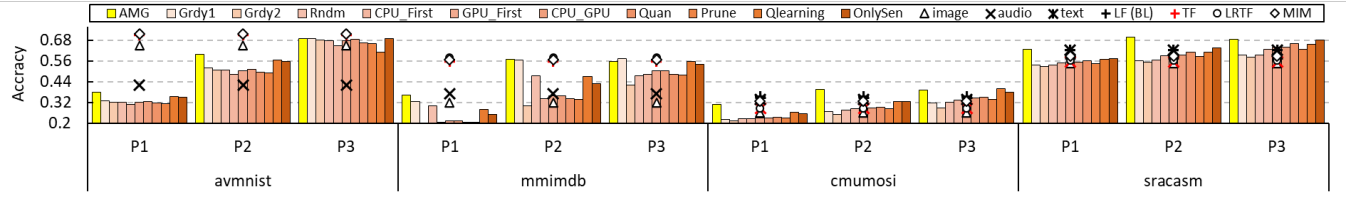


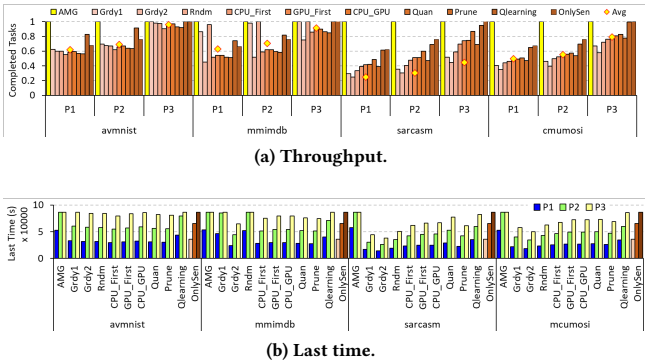
Figure 20: Accuracy comparison of *AMG* with the SOTA energy-efficient methods

write/read energy overhead of the raw data buffer is 0.062mJ. Thus, modality gating can significantly reduce the energy consumption of M^2C with negligible overhead.

7.1.3 Analysis on Speculative Activation. We compare the accuracy, energy consumption and latency of *AMG* to the baseline method, i.e., *LF* in Figure 19. It shows that *AMG* can achieve the same and even higher accuracy compared to *LF* with less energy for each inference. *AMG* can significantly reduce the execution time for over 80% data samples by about 1 second. However, it can also lead to increased tail latency. In this work we consider it to be less critical for IoT scenarios that do not directly interact with users all the time (e.g. wildlife detection). Our probe-and-prediction approach can alleviate the long tail problem to some extent by activating sensors for those complex data samples. To further optimize the long-tail problem, one can use latency as a gating criterion as well.

7.2 Comparison with the State-of-the-art

7.2.1 Accuracy. It is critical to meet the accuracy requirement of M^2C applications. In Figure 20, we compare the accuracy of different datasets under different power management (PM) methods. In order to verify that our proposed algorithm can match the state-of-the-art performance. We use the accuracy of the state-of-the-art (SOTA) M^2C algorithms (described in Section 6.2) as reference points. We can see that *AMG* achieves the best accuracy among all the PM methods. It is remarkable that *AMG* can achieve the same accuracy and even outperform the best SOTA M^2C algorithms in most scenarios. For example, on the `cmumosi` and `sarcasm` datasets, *AMG* shows a significant improvement over existing algorithms. This is because it can effectively balance multi-modal information with multi-exit co-training, and plays a certain regularization effect through multi-modal and unimodal co-feature extraction, which trains a better model under the same architecture as *LF*. Conversely, traditional power optimization methods can severely degrade performance as they blindly manage energy consumption without taking into account the important characteristics of M^2C applications.

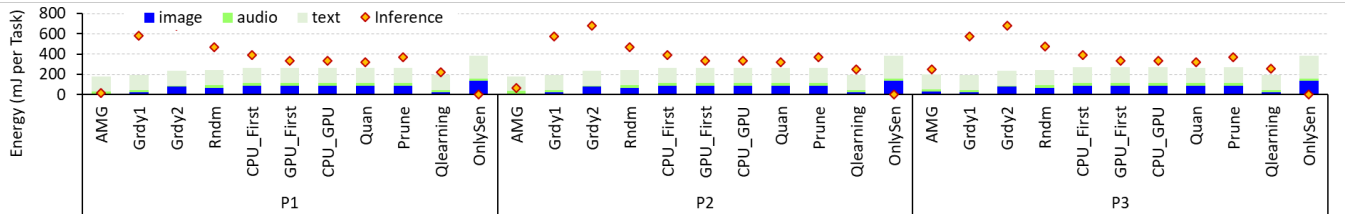


(a) Throughput. (b) Last time.

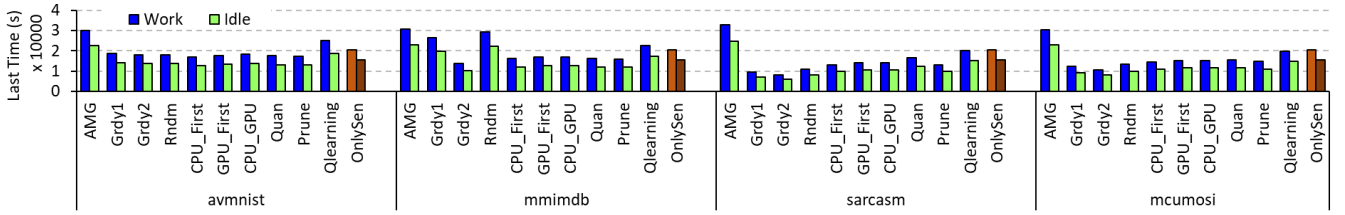
Figure 21: Comparison of throughput (normalized to *AMG*) and system last time under various energy budgets.

7.2.2 Throughput. In Figure 21-(a), we compare the number of completed inference tasks under different PM methods. The x-axis depicts different scenarios of different datasets under various energy budgets, and the y-axis shows the number of completed tasks which is normalized to the number of *AMG*. The results show that *AMG* can accomplish much more inference tasks compared to the other methods with the same energy budget. For example, *AMG* can complete 1.6X ~ 1.8X `avmnist` tasks and 1.8X ~ 3.8X `sarcasm` tasks. One of the main reasons why *AMG* can complete more tasks is that it reduces the waste energy of sensors for most data samples. Overall, *AMG* can have 1.6 ~ 3.8X higher throughput than the SOTA methods with the highest accuracy in our experiment.

In Figure 21-(b), we compare the lifespan (how long the system can sustain its functionality) and energy usage trace of *AMG* with the other methods. The results show that the energy usage time of *AMG* is longer than the others due to its adaptive modality sensing and computing control. In detail, *AMG* can execute 10% ~ 280% more time with the same energy budget. Although the system must face an energy crunch, *AMG* adjusts the accuracy of inference tasks to



(a) Per-task energy breakdown: *AMG* significantly reduces the sensing energy compared to computing-optimized methods, e.g., *OnlySen*.



(b) Last time:*AMG* has the longest last time by optimizing the sensing and computing energy.

Figure 22: Energy consumption and system last time of different M^2C tasks under different energy-efficient approaches.

the remaining energy. In this way, other sensors are gated, so the system can accomplish more tasks without losing much accuracy.

7.2.3 Energy Utilization. In Figure 22, we further analyze the energy usage under different PM schemes. In Figure 22-(a), we show both the average sensing (of different sensors) and computing energy for each inference task. The result illustrates that *AMG* significantly reduces both the sensing and computing energy of the M^2C, i.e., *LF*. It consumes the least energy compared to the other PM schemes. It is notable that the SOTA PM schemes are too blind and thus waste much energy. This further runs out of the energy budgets and leads to shorter last time as shown in 22-(b). Overall, *AMG* is able to employ less power consumption while guaranteeing very high performance due to its ability to efficiently activate the modalities and early exit mechanisms.

8 RELATED WORK

Multi-modal Deep Neural Networks: Multi-modal DNNs [31, 80] are designed to leverage complementary information from multiple heterogeneous modalities. They have been shown to outperform uni-modal networks in many fields [47]. Most of the current studies focus on finding more effective fusion or representation methods of different modalities [7, 72, 79, 80], including two main classes of approaches, i.e., early fusion methods [51, 76] and late fusion methods [6, 74]. *Different from these works focusing on multi-modal fusion algorithm optimization, we improve the efficiency of existing M^2C applications running in AIoT environments.*

Energy-Efficient Edge DNN Inference: Deploying DNN inference tasks on edge devices is gaining popularity due to the advancement of compact DNN models [24, 61] and specialized accelerators [10, 21, 52]. One of the biggest issues of edge DNN is energy management. Besides conventional power optimization techniques such as DVFS [28, 29, 45], power gating [66, 73], researchers also propose to optimize DNNs' energy consumption by alleviating the power dissipation of silicon devices [38, 77] or abbreviating neural network models [56], such as quantization [59, 71], pruning [25, 75]. *AMG is orthogonal to most of the existing power optimization methods. It would be easy to integrate them with AMG to achieve better energy optimization.*

Adaptive DNN Inference: Several prior works have implemented adaptive DNN inference through the early exit technique [23, 43, 62] which provides better trade-offs between inference latency and accuracy. These approaches can also enable faster inference by sacrificing some accuracy [37, 43]. A few works aim to deploy the lightweight and dynamic NNs with early exit on energy harvesting-powered IoT devices, which often have limited computation resources and energy budgets [32, 75]. *To our knowledge, there is no prior work on building effective early exits for various modalities in a multi-modal network. We are the first to apply the early exit approach to multi-modal analysis.*

9 CONCLUSION

It is attractive to implement efficient multi-modal computing for handling various sensory modalities. We introduce *AMG*, a novel design that enables energy-efficient M^2C on power-constrained systems. It synergistically integrates two inventive techniques: 1) at the hardware layer, we introduce decoupled sensor architecture and modality semi-gating mechanism; 2) at the system layer, we devise optimized speculative sensor activation. The proposed design can greatly slash M^2C overhead while maintaining high performance. It can greatly contribute to the wide adoption of multi-modal computing on various AIoT devices and edge micro/nano data centers, thereby benefiting numerous real-life smart applications.

ACKNOWLEDGEMENTS

This research was partially sponsored by ACCESS - AI Chip Center for Emerging Smart Systems, InnoHK funding, Hong Kong SAR. It is also supported in part by the National Natural Science Foundation of China (No.62122053), and Shanghai S&T Committee Rising-Star Program (No.21QA1404400)). We thank all the anonymous reviewers for their valuable feedback.

REFERENCES

[1] Sony IMAX 219. 2022. Diagonal 4.60mm (Type 1/4.0) 8M Pixel CMOS Image Sensor with Square Pixel for Color Cameras. https://www.electronicsdatasheets.com/download/5721ed8ce34e24fd697a913a.pdf.

[2] Md Akhtar, Dushyant Chauhan, Deepanway Ghosal, Soujanya Poria, Asif Ekbal, and Pushpak Bhattacharyya. 2019. Multi-task learning for multi-modal emotion recognition and sentiment analysis. In *Association for Computational Linguistics (ACL)*.

[3] Dario Amodei, Sundaram Ananthanarayanan, Rishita Anubhai, Jingliang Bai, Eric Battenberg, Carl Case, Jared Casper, Bryan Catanzaro, Qiang Cheng, Guoliang Chen, et al. 2016. Deep speech 2: End-to-end speech recognition in english and mandarin. In *International Conference on Machine Learning (ICML)*.

[4] John Arevalo, Thamar Solorio, Manuel Montes-y Gómez, and Fabio González. 2017. Gated Multimodal Units for Information Fusion. In *International Conference on Learning Representations (ICLR)*.

[5] ARM. 2022. MRS (system coprocessor register to ARM register). https://developer.arm.com/documentation/dui0473/j/arm-and-thumb-instructions/mrs--system-coprocessor-register-to-arm-register-.

[6] Francis Bach, Gert Lanckriet, and Michael Jordan. 2004. Multiple kernel learning, conic duality, and the SMO algorithm. In *International Conference on Machine Learning (ICML)*.

[7] Tadas Baltrušaitis, Chaitanya Ahuja, and Louis Morency. 2018. Multimodal machine learning: A survey and taxonomy. In *IEEE Transactions on Pattern Analysis and Machine Intelligence (PAMI)*.

[8] Writam Banerjee. 2020. Challenges and Applications of Emerging Nonvolatile Memory Devices. *Electronics* (2020).

[9] Simone Bertolazzi, Walt Coon, Mike Howard, Ivan Donaldson, Emilie Jolivet, and Santosh Kumar. 2020. Status of the Memory Industry: Market and Technology Report 2020. https://www.i-micronews.com/products/emerging-non-volatile-memory-2020.

[10] Canaan. 2019. KENDRYTE K210 Datasheet. https://tinyurl.com/354tzz4t.

[11] Santiago Castro, Devamanyu Hazarika, Verónica Pérez-Rosas, Roger Zimmermann, Rada Mihalcea, and Soujanya Poria. 2019. Towards multimodal sarcasm detection (an _Obviously_ perfect paper). In *Association for Computational Linguistics (ACL)*.

[12] Oya Celiktutan, Efstratios Skordos, and Hatice Gunes. 2019. Multimodal Human-Human-Robot Interactions (MHHRI) Dataset for Studying Personality and Engagement. *IEEE Transactions on Affective Computing (TAC)* (2019).

[13] Ismail Cevik, Xiwei Huang, Hao Yu, Mei Yan, and Suat U. Ay. 2015. An Ultra-Low Power CMOS Image Sensor with On-Chip Energy Harvesting and Power Management Capability. *Sensors* (2015).

[14] An Chen. 2016. A review of emerging non-volatile memory (NVM) technologies and applications. *Solid-state Electronics* (2016).

[15] Cheng-Ta Chiang, Chih-Hsien Wang, and Chia-Yu Wu. 2012. A CMOS MEMS Audio Transducer Implemented by Silicon Condenser Microphone With Analog Front-End Circuits of Audio Codec. *IEEE Transactions on Very Large Scale Integration (VLSI) Systems* (2012).

[16] CYMBET. 2016. Rechargeable Solid State Bare Die Batteries. In *EnerChip™ Bare Die CBC005 Datasheet*.

[17] Jacob Devlin, Ming Chang, Kenton Lee, and Kristina Toutanova. 2019. BERT: Pre-training of Deep Bidirectional Transformers for Language Understanding. In *Conference of the North American Chapter of the Association for Computational Linguistics (NAACL)*.

[18] Xiangyu Dong, Cong Xu, Yuan Xie, and Norman P. Jouppi. 2012. NVSim: A Circuit-Level Performance, Energy, and Area Model for Emerging Nonvolatile Memory. *IEEE Transactions on Computer-Aided Design of Integrated Circuits and Systems (TCAD)* (2012).

[19] Sensors Expo and Conference. 2017. Low Power Microphone Acquisition and Processing for Always-on Applications Based on Microcontrollers. https://hobbydocbox.com/Radio/80585828-Low-power-microphone-acquisition-and-processing-for-always-on-applications-based-on-microcontrollers.html.

[20] Abbas El Gamal and Helmy Eltoukhy. 2005. CMOS Image Sensors: An Introduction to the Technology, Design and Performance Limits, Presenting Recent Developments and Future Directions. *IEEE Circuits and Devices Magazine* (2005).

[21] Google. 2021. Edge TPU - AI at the Edge. https://cloud.google.com/edge-tpu.

[22] João Guerreiro, Aleksandar Ilic, Nuno Roma, and Pedro Tomás. 2019. Modeling and Decoupling the GPU Power Consumption for Cross-Domain DVFS. In *IEEE Transactions on Parallel and Distributed Systems (TPDS)*.

[23] Yizeng Han, Gao Huang, Shiji Song, Le Yang, Honghui Wang, and Yulin Wang. 2021. Dynamic neural networks: A survey. In *IEEE Transactions on Pattern Analysis and Machine Intelligence (PAMI)*.

[24] Kaiming He, X. Zhang, Shaoqing Ren, and Jian Sun. 2016. Deep Residual Learning for Image Recognition. In *Conference on Computer Vision and Pattern Recognition (CVPR)*.

[25] Yihui He, Ji Lin, Zhijian Liu, Hanrui Wang, Li Li, and Song Han. 2018. AMC: AutoML for Model Compression and Acceleration on Mobile Devices. In *European Conference on Computer Vision (ECCV)*.

[26] Jana Helgath, Philip Braun, Andreas Pritschet, Maximilian Schubert, Patricia Böhm, and Daniel Isemann. 2018. Investigating the Effect of Different Autonomy Levels on User Acceptance and User Experience in Self-driving Cars with a VR Driving Simulator. *Interacción* (2018).

[27] Xiaofeng Hou, Luoyao Hao, Chao Li, Quan Chen, Wenli Zheng, and Minyi Guo. 2018. Power Grab in Aggressively Provisioned Data Centers: What is the Risk and What Can Be Done About It. *International Conference on Computer Design (ICCD)* (2018).

[28] Xiaofeng Hou, Chao Li, Jiacheng Liu, Lu Zhang, Yang Hu, and Minyi Guo. 2020. ANT-man: towards agile power management in the microservice era. In *International Conference for High Performance Computing, Networking, Storage, and Analysis (SC)*.

[29] Xiaofeng Hou, Chao Li, Jiacheng Liu, Lu Zhang, Shaolei Ren, Jingwen Leng, Quan Chen, and Minyi Guo. 2021. AlphaR: Learning-Powered Resource Management for Irregular, Dynamic Microservice Graph. *IEEE International Parallel and Distributed Processing Symposium (IPDPS)* (2021).

[30] Xiaofeng Hou, Mingyu Liang, Chao Li, Wenli Zheng, Quan Chen, and Minyi Guo. 2019. When Power Oversubscription Meets Traffic Flood Attack: Re-Thinking Data Center Peak Load Management. *International Conference on Parallel Processing (ICPP)* (2019).

[31] Xiaofeng Hou, Cheng Xu, Jiacheng Liu, Xuehan Tang, Lingyu Sun, Chao Li, and Kwang-Ting Cheng. 2022. Characterizing and Understanding End-to-End Multi-Modal Neural Networks on GPUs. *IEEE Computer Architecture Letters (CAL)* (2022).

[32] Zhaowu Huang, Fang Dong, Dian Shen, Junxue Zhang, Huitian Wang, Guangxing Cai, and Qiang He. 2021. Enabling Low Latency Edge Intelligence based on Multi-exit DNNs in the Wild. In *International Conference on Distributed Computing Systems (ICDCS)*.

[33] Preferred Networks Inc. 2022. Optimize Your Optimization: An open-source hyperparameter optimization framework to automate hyperparameter search. https://optuna.org/.

[34] Huawei Incorporation. 2019. Huawei GT 2 Smartwatch. https://consumer.huawei.com/en/wearables/watch-gt2/.

[35] Siddhant Jayakumar, Wojciech Czarnecki, Jacob Menick, Jonathan Schwarz, Jack Rae, Simon Osindero, Yee Teh, Tim Harley, and Razvan Pascanu. 2020. Multiplicative Interactions and Where to Find Them. In *International Conference on Learning Representations (ICLR)*.

[36] Kambiz Kaviani, Ömer Oralkan, Butrus T. Khuri-Yakub, and Bruce A. Wooley. 2003. A multichannel pipeline analog-to-digital converter for an integrated 3-D ultrasound imaging system. *IEEE J. Solid State Circuits (JSSCC)* (2003).

[37] Yigitcan Kaya, Sanghyun Hong, and Tudor Dumitras. 2019. Shallow-deep networks: Understanding and mitigating network overthinking. In *International Conference on Machine Learning (ICML)*.

[38] Hyungjun Kim, Hyunmyung Oh, and JaeJoon Kim. 2020. Energy-efficient XNOR-free In-Memory BNN Accelerator with Input Distribution Regularization. In *IEEE/ACM International Conference On Computer Aided Design (ICCAD)*.

[39] Seyeon Kim, Kyungmin Bin, Sangtae Ha, Kyunghan Lee, and Song Chong. 2021. zTT: learning-based DVFS with zero thermal throttling for mobile devices. In *International Conference on Mobile Systems, Applications, and Services (MobiSys)*.

[40] Won Kim, Meeta Gupta, GuYeon Wei, and David Brooks. 2008. System level analysis of fast, per-core DVFS using on-chip switching regulators. In *International Symposium on High Performance Computer Architecture (HPCA)*.

[41] Srivatsan Krishnan, Zishen Wan, Kshitij Bhardwaj, Paul N. Whatmough, Aleksandra Faust, Sabrina M. Neuman, Gu-Yeon Wei, David M. Brooks, and Vijay Janapa Reddi. 2022. Automatic Domain-Specific SoC Design for Autonomous Unmanned Aerial Vehicles. *International Symposium on Microarchitecture (MICRO)* (2022).

[42] Mario Lanza, Abu Sebastian, Wei Dang Lu, Manuel Le Gallo, M. F. Chang, Deji Akinwande, Francesco Maria Puglisi, Husam N. Alshareef, Meilin Liu, and Juan Bautista Roldán. 2022. Memristive technologies for data storage, computation, encryption, and radio-frequency communication. *Science* (2022).

[43] Stefanos Laskaridis, Alexandros Kouris, and Nicholas Lane. 2021. Adaptive Inference through Early-Exit Networks: Design, Challenges and Directions. In *Proceedings of the 5th International Workshop on Embedded and Mobile Deep Learning (EMDL)*.

[44] Erwan Lecarpentier and Emmanuel Rachelson. 2019. Non-Stationary Markov Decision Processes a Worst-Case Approach using Model-Based Reinforcement Learning. In *Conference on Neural Information Processing Systems (NeurIPS)*.

[45] Chao Li, Zhenhua Wang, Xiaofeng Hou, Hao peng Chen, Xiaoyao Liang, and Minyi Guo. 2016. Power Attack Defense: Securing Battery-Backed Data Centers. *International Symposium on Computer Architecture (ISCA)* (2016).

[46] Di Li, Xuejun Qian, Runze Li, Chunlong Fei, Laiming Jiang, Xuyuan Chen, Yintang Yang, and Qifa Zhou. 2020. High Resolution ADC for Ultrasound Color Doppler Imaging Based on MASH Sigma-Delta Modulator. *IEEE Transactions on Biomedical Engineering (TBE)* (2020).

[47] Paul Liang, Yiwei Lyu, Xiang Fan, Zetian Wu, Yun Cheng, Jason Wu, Leslie Chen, Peter Wu, Michelle Lee, and Yuke Zhu. 2021. MultiBench: Multiscale Benchmarks for Multimodal Representation Learning. In *Conference on Neural Information Processing Systems (NeurIPS)*.

[48] Robert Likamwa, Bodhi Priyantha, Matthai Philipose, Lin Zhong, and Paramvir Bahl. 2013. Energy characterization and optimization of image sensing toward continuous mobile vision. *International Conference on Mobile Systems, Applications, and Services (MobiSys)* (2013).

[49] Zhun Liu, Ying Shen, Bharadhwaj Lakshminarasimhan, Pu Liang, Amir Zadeh, and LouiPhilippe Morency. 2018. Efficient low-rank multimodal fusion with modality-specific factors. In *Association for Computational Linguistics (ACL)*.

[50] Brian McFee, Colin Raffel, Dawen Liang, Daniel Ellis, Matt McVicar, Eric Battenberg, and Oriol Nieto. 2015. librosa: Audio and Music Signal Analysis in Python.

[51] Neverova Natalia, Wolf Christian, W Graham, and Nebout Florian. 2014. Multi-scale deep learning for gesture detection and localization. In *European Conference on Computer Vision (ECCV)*.

[52] Nvidia. 2020. NVIDIA JETSON NANO DEVELOPER KIT. https://developer.nvidia.com/zh-cn/embedded/learn/get-started-jetson-nano-devkit.

[53] Hiroshi Ono and Kazutoshi Mizoi. 1983. Wireless transmitting and receiving systems including ear microphones. *Journal of the Acoustical Society of America* (1983).

[54] Rameswar Panda, Chun-Fu Chen, Quanfu Fan, Ximeng Sun, Kate Saenko, Aude Oliva, and Rogério Schmidt Feris. 2021. AdaMML: Adaptive Multi-Modal Learning for Efficient Video Recognition. *International Conference on Computer Vision (ICCV)* (2021).

[55] Angshuman Parashar, Priyanka Raina, Yakun Sophia Shao, Yu hsin Chen, Victor A. Ying, Anurag Mukkara, Rangharajan Venkatesan, Brucek Khailany, Stephen W. Keckler, and Joel S. Emer. 2019. Timeloop: A Systematic Approach to DNN Accelerator Evaluation. *International Symposium on Performance Analysis of Systems and Software (ISPASS)* (2019).

[56] Eunhyeok Park, Dongyoung Kim, and Sungjoo Yoo. 2018. Energy-Efficient Neural Network Accelerator Based on Outlier-Aware Low-Precision Computation. In *International Symposium on Computer Architecture (ISCA)*.

[57] Jeffrey Pennington, Richard Socher, and Christopher Manning. 2014. GloVe: Global Vectors for Word Representation. In *Empirical Methods in Natural Language Processing (EMNLP)*.

[58] Hai Pham, Paul Liang, Thomas Manzini, Louis Morency, and Barnabás Póczos. 2019. Found in translation: Learning robust joint representations by cyclic translations between modalities. In *AAAI Conference on Artificial Intelligence (AAAI)*.

[59] Mohammad Rastegari, Vicente Ordonez, Joseph Redmon, and Ali Farhadi. 2016. XNOR-Net: ImageNet Classification Using Binary Convolutional Neural Networks. In *European Conference on Computer Vision (ECCV)*.

[60] Debashri Roy, Yuanyuan Li, Tong Jian, Peng Tian, Kaushik Roy Chowdhury, and Stratis Ioannidis. 2022. Multi-modality Sensing and Data Fusion for Multi-vehicle Detection. *IEEE Transactions on Multimedia (ToM)* (2022).

[61] Mark Sandler, Andrew Howard, Menglong Zhu, Andrey Zhmoginov, and Liang-Chieh Chen. 2018. Mobilenetv2: Inverted residuals and linear bottlenecks. In *Conference on Computer Vision and Pattern Recognition (CVPR)*.

[62] Simone Scardapane, Michele Scarpiniti, Enzo Baccarelli, and Aurelio Uncini. 2020. Why should we add early exits to neural networks?. In *Cognitive Computation*.

[63] Muhammad Akmal Shafique, Theocharis Theocharides, Vijay Janapa Reddi, and Boris Murmann. 2021. TinyML: Current Progress, Research Challenges, and Future Roadmap. *Design Automation Conference (DAC)* (2021).

[64] Sipeed. [n. d.]. Sipeed MAIX-II Dock (V831). https://www.seeedstudio.com/Sipeed-MAIX-Dock-p-4815.html.

[65] Sheng Sun, Jianyuan Wang, Menglun Zhang, Yuan Ning, Dong Ma, Yi Yuan, Pengfei Niu, Zhicong Rong, Zhuochen Wang, and Wei Pang. 2022. MEMS ultrasonic transducers for safe, low-power and portable eye-blinking monitoring. *Microsystems & Nanoengineering* (2022).

[66] Hamed Tabkhi and Gunar Schirner. 2014. Application-Guided Power Gating Reducing Register File Static Power. In *IEEE Transactions on Very Large Scale Integration Systems (VLSI)*.

[67] Surat Teerapittayanon, Bradley McDanel, and H Kung. 2016. BranchyNet: Fast inference via early exiting from deep neural networks. In *International Conference on Pattern Recognition (ICPR)*.

[68] Texas. 2013. Mixed Signal Microcontroller. In *MSP430G2x11 MSP430G2x01 Datasheet*.

[69] Marcelo Urbina, Tatiana Acosta, Jesús Lázaro, Armando Astarloa, and Unai Bidarte. 2019. Smart sensor: SoC architecture for the Industrial Internet of Things. In *IEEE Internet of Things Journal (IoTJ)*.

[70] Oriol Vinyals, Alexander Toshev, Samy Bengio, and Dumitru Erhan. 2016. Show and tell: Lessons learned from the 2015 mscoco image captioning challenge. In *IEEE Transactions on Pattern Analysis and Machine Intelligence (PAMI)*.

[71] Kuan Wang, Zhijian Liu, Yujun Lin, Ji Lin, and Song Han. 2019. Haq: Hardware-aware automated quantization with mixed precision. In *Conference on Computer Vision and Pattern Recognition (CVPR)*.

[72] Weiyao Wang, Du Tran, and Matt Feiszli. 2020. What makes training multi-modal classification networks hard?. In *Conference on Computer Vision and Pattern Recognition (CVPR)*.

[73] Kai Wu, IngLin, Yao Wang, and Shuen Yang. 2014. BTI-Aware Sleep Transistor Sizing Algorithm for Reliable Power Gating Designs. In *IEEE Transactions on Computer-Aided Design of Integrated Circuits and Systems (TCAD)*.

[74] Yi Wu, Edward Chang, Kevin Chang, and John Smith. 2004. Optimal multimodal fusion for multimedia data analysis. In *ACM International Conference on Multimedia (ACM MM)*.

[75] Yawen Wu, Zhepeng Wang, Zhenge Jia, Yiyu Shi, and Jingtong Hu. 2020. Intermittent inference with nonuniformly compressed multi-exit neural network for energy harvesting powered devices. In *ACM/IEEE Design Automation Conference (DAC)*.

[76] Zhou Xiaoli and Bhanu Bir. 2008. Feature fusion of side face and gait for video-based human identification. In *Pattern Recognition*.

[77] Shihui Yin, Zhewei Jiang, Minkyu Kim, Tushar Gupta, Mingoo Seok, and Jaesun Seo. 2020. Vesti: Energy-Efficient In-Memory Computing Accelerator for Deep Neural Networks. In *IEEE Transactions on Very Large Scale Integration Systems (VLSI)*.

[78] Bo Yu, Wei Hu, Leimeng Xu, Jie Tang, Shaoshan Liu, and Yuhao Zhu. 2020. Building the Computing System for Autonomous Micromobility Vehicles: Design Constraints and Architectural Optimizations. *International Symposium on Microarchitecture (MICRO)* (2020).

[79] Amir Zadeh, Minghai Chen, Soujanya Poria, Erik Cambria, and Louis Morency. 2017. Tensor fusion network for multimodal sentiment analysis. In *Association for Computational Linguistics (ACL)*.

[80] Chao Zhang, Zichao Yang, Xiaodong He, and Li Deng. 2020. Multimodal intelligence: Representation learning, information fusion, and applications. In *IEEE Journal of Selected Topics in Signal Processing*.

[81] Yi Zhang, Chia-Hung Chen, Tao He, and Gábor C. Temes. 2015. A Continuous-Time Delta-Sigma Modulator for Biomedical Ultrasound Beamformer Using Digital ELD Compensation and FIR Feedback. *IEEE Transactions on Circuits and Systems (TCS)* (2015).