AIEnergy: An energy benchmark for AI-empowered mobile and IoT devices

***Xiaolong Tu*[1], *Anik Mallik*[2], *Haoxin Wang*[1], *Jiang Xie*[3]**

[1] Georgia State University, USA, [2] Towson University, USA, [3] University of North Carolina at Charlotte, USA

Corresponding author: Haoxin Wang, haoxinwang@gsu.edu

This paper presents AIEnergy, the first energy benchmark suite and benchmarking methodology to allow accurate energy measurement and performance evaluation of AI-empowered mobile and IoT devices with diverse AI chipsets and software stacks. We first discuss the design principles and the key challenges for developing an accurate, interpretable, and adoptable energy benchmark. We address these design challenges by developing an energy measurement methodology that incorporates three strategies and an end-user understandable scoring system. AIEnergy collects over 8.8 GB measurement data from 264 configuration combinations of eight commercial AI-empowered mobile and IoT devices with diverse chipsets, six deep learning applications with unique end-to-end processing pipelines and 12 deep neural network models under CPU, GPU, and Neural Networks API (NNAPI) delegates. AIEnergy will evolve and serve as a ready-to-adopt benchmark that is accessible by both mobile and IoT end users with non-technical backgrounds and researchers with varying levels of expertise.

Keywords: Edge computing, energy benchmark, energy efficiency, Internet of Things (IoT), measurement

1. INTRODUCTION

The Internet of Things (IoT) is increasingly penetrating into sectors, such as healthcare, agriculture, and smart cities, leading to an exponential growth in machine-generated data. This surge in data naturally necessitates the integration of Artificial Intelligence (AI) and Machine Learning (ML) with IoT, which enables the real-time processing and analysis of data from mobile and IoT platforms to optimize systems, predict needs, and automate complex tasks across these vital sectors. AI-empowered IoT and mobile devices are driving a shift from traditional centralized cloud-based computing to a decentralized, pervasive, and scalable computing paradigm. Such a shift significantly enhances data privacy by reducing the need to upload sensitive information to cloud platforms. It also seamlessly incorporates computational capabilities into everyday objects and delivers the ultra-low latency required for a range of emerging AI applications, including those based on vision [1, 2, 3], voice [4, 5], and language [6, 7, 8]. Despite significant improvements in hardware device capabilities, including computational power, functionality, and connectivity, the limited energy capacity remains a major bottleneck in advancing AI applications on diverse mobile and IoT devices.

First, mobile and IoT devices typically rely solely on embedded batteries for power, making their energy capacity heavily dependent on factors such as form factor constraints, safety requirements, production costs, and the environmental implications of the battery technology employed. Second, AI/ML algorithms are often computation-intensive and consume substantial energy, as seen in mobile Augmented Reality (AR) applications [9, 10, 11]. Third, a device's energy efficiency or the battery usage of its applications frequently dictates end-user satisfaction and the practical usability of the technology in daily operations. For instance, a survey [12] revealed that over 55% of respondents would leave a negative review for an application that significantly drains the device's battery, underscoring that energy efficiency is an essential aspect of user experience and cannot be overlooked. Consequently, it is imperative to prioritize the energy efficiency of AI-powered mobile and IoT devices to enhance overall performance and secure user approval. To this end, the very first step is to *systematically identify the energy bottlenecks* in these devices and AI applications to enable further optimization.

Unfortunately, this is non-trivial because of the lack of comprehensive understanding of how AI impacts energy use on diverse physical mobile and IoT devices. On one hand, we cannot improve energy efficiency without conducting on-device measurements. The energy efficiency of an AI-powered mobile or IoT device involves more than just its hardware capabilities. It is intrinsically linked with the AI software stack, yet the overall performance of this integration is obscured by the complexity of Deep Neural Network (DNN) models and the comprehensive processing pipeline in AI applications. Although efforts have been made to develop software-based energy profilers for general applications on mobile and IoT devices [13], adapting these tools to AI applications is complex. This complexity arises from the challenge of discerning the unique energy consumption patterns within the AI software stack. On the other hand, we cannot optimize AI's energy efficiency on mobile and IoT devices without acknowledging and addressing its role during the design phase. Current research primarily focuses on enhancing the performance of AI techniques to increase accuracy or reduce latency, often overlooking their impact on system overheads, such as energy cost. Hence, there is a pressing need to prioritize energy efficiency alongside performance improvements in AI, particularly for resource-constrained mobile and IoT devices. Consequently, the current lack of a holistic understanding of energy consumption motivates the urgent necessity for an energy benchmark for AI-empowered mobile and IoT devices that is *accurate, interpretable, and adoptable.*

To this end, in this paper, we present AIEnergy, the first energy benchmark suite and benchmarking methodology for mobile and IoT with AI applications, to accurately measure and fairly compare the energy efficiency of various hardware devices. To ensure the accuracy, interpretability, and adoptability of the AIEnergy benchmark, its design is informed by the following five principles:

P1. The benchmark should **accurately measure the energy consumption** of AI applications running on mobile and IoT devices in real scenarios, to provide a trustworthy foundation for energy-aware optimization and fair device comparison.

P2. The benchmark should offer **insights that are easy to interpret** for identifying energy bottlenecks, so developers and researchers can pinpoint inefficiencies and improve energy-performance trade-offs.

P3. The benchmark should identify a range of **representative hardware and software stacks** for AI-empowered mobile and IoT devices, ensuring coverage of real-world deployment diversity and enabling meaningful evaluation.

P4. The benchmark should be **extensible to new mobile and IoT hardware and software stacks** and continuously updated to reflect technology advances and AI evolution, ensuring it stays relevant and usable over time.

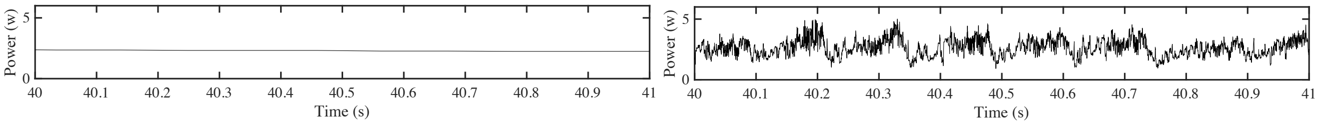
P5. The benchmark results should be **easily understandable by device end users** and **readily adoptable by our research community**, bridging the gap between practical deployment and academic innovation to promote widespread impact.

The comparison of the benchmark results collected from 264 configuration combinations of eight commercial mobile and IoT devices with diverse AI chipsets, six AI applications with unique end-to-end processing pipelines and 12 DNN models under Central Processing Unit (CPU), Graphics Processing Unit (GPU), and Neural Networks API (NNAPI) delegates, reveals the following key takeaways:

- *Scoring metrics matter for benchmarking.* Developing holistic scoring systems is crucial to foster fair evaluation of AI performance on mobile and IoT devices.
- *The energy efficiency of AI frameworks is overlooked.* NNAPI consumes more energy than other delegates in approximately 50% of testing configurations, or worse.
- *Software-hardware co-design play a crucial role.* Improving the AI energy efficiency on mobile and IoT devices requires synergistic co-designs between software and hardware.

2. AIENERGY BENCHMARK DESIGN CHALLENGES

Developing an AI energy benchmark for modern mobile and IoT devices that can incorporate principles P1-P5 is challenging. In this section, we describe the challenges in addressing these principles.



(a) Power consumption trace recorded by a built-in current sensor

(b) Power consumption trace recorded by an external power monitor

Figure 1 – Comparison of recorded power traces in terms of time granularity. Measured device: Huawei P40 Lite; application: object detection; DNN model: MobileNet-V2, FP32; delegate: CPU with 1 thread; input image resolution: 300 × 300 pixels.

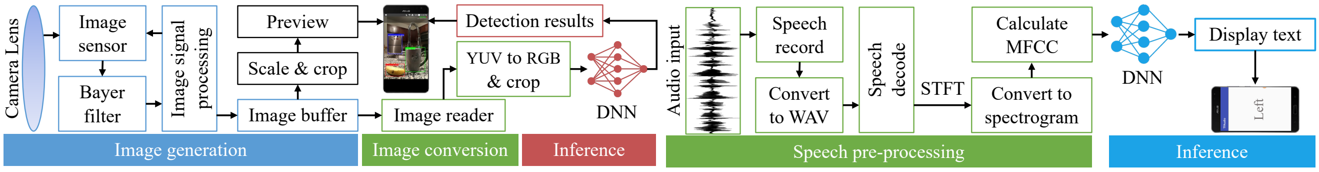
P1: Accuracy. DNN models typically form the backbone of AI applications and consume a significant share of the device's computational resources. Hence, accurately identifying energy bottlenecks within a DNN model is crucial. However, comprehending energy consumption characteristics within mobile-specific DNN models through measurement is highly challenging due to their complex and intricate architectures, as well as the low time granularity of built-in current sensors on mobile and IoT devices (e.g., fuel gauge).

Fig. 1 presents a comparison of power consumption traces recorded by an external power monitor (we use Monsoon power monitor in this work [14]) and the device's built-in current sensor. It is obvious that the built-in current sensor is not capable of capturing accurate and fine-grained power measurements within the DNN model due to its low power sampling frequency. We observe that the built-in current sensors in many mobile and IoT devices typically support a sampling period ranging from a hundred milliseconds to seconds, which limits the frequency of current measurements to between 1 and 10 readings per second. However, tracing the power consumption trend within a DNN model on a specific hardware device, including changes in power consumption of individual layers or operators, requires a time granularity of less than 1 millisecond. As a result, most current energy profiling solutions that depend on the built-in current sensors fall short in providing precise energy measurements for AI applications [13, 15, 16, 17].

Fig. 1(b) demonstrates that using an external power monitor such as the Monsoon power monitor with a sampling frequency of 5000 Hz shows promise in capturing accurate and fine-grained power measurements for AI and identifying energy bottlenecks within DNN models. Nonetheless, interfacing recent commercial mobile and IoT devices, especially those introduced post-2017, with an external power monitor proves to be difficult due to the more complex integration of their electronic components. Taking modern smartphones as examples, the battery connector, which links the battery to the circuit board, has evolved significantly over the years. Previous-generation smartphones commonly used a "snap-type connector" with four metal prongs, which simplified the identification of positive and negative terminals and made connections to external power monitors straightforward. In contrast, modern smartphones often utilize a small, proprietary, and fragile Flexible Printed Circuit (FPC) battery connector. The diminutive size and fragile nature of the FPC connector complicate handling, necessitating specialized tools and expertise for successful connection to an external power monitor that can deliver enhanced accuracy [18]. Given these complexities with the FPC battery connector, recent studies generally depend on the built-in current sensor to measure power consumption of modern mobile and IoT devices, although this method only provides coarse-grained results.

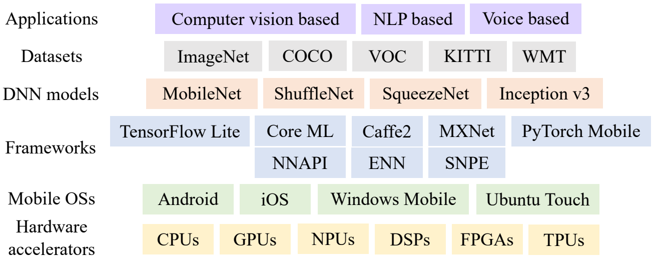
P2: Interpretability. In addition to DNN models, the software stack for mobile and IoT devices is distinctly characterized by the end-to-end pipeline for processing AI applications. The end-to-end energy efficiency is important as it includes pre and post-processing overheads. AI applications typically involve complex and distinctive end-to-end processing pipelines that consist of several energy-consuming phases. For instance, Fig. 2 compares two processing pipelines for mobile/IoT-based object detection and speech recognition. Measuring the overall device's energy consumption in isolation is insufficient for obtaining informative insights. Instead, it is crucial to break down the power and energy consumption based on either the involved device hardware components or the processing phases in order to achieve interpretable insights on the inner workings and energy bottlenecks of different AI applications. However, due to the high level of hardware integration in commercial mobile and IoT devices, it is not feasible to directly measure and isolate the power and energy consumption of individual hardware components such as the camera, display, and System-on-Chip (SoC). Furthermore, the interconnection and dependencies between different phases in the end-to-end processing pipeline for AI applications make it difficult to isolate the power and energy consumption of individual phases, such as image generation, image conversion, and inference.

P3: Representativeness. AI energy efficiency is shrouded behind the diversity of the mobile and IoT ecosystem, including both hardware and software stack [21]. As shown in Fig. 3, the possible combinations of AI hardware accelerators, Operating Systems (OS), frameworks, DNN models, datasets, and applications are numerous, and each of these factors can contribute to the variability in energy efficiency. For example, advanced SoCs typically consist of various components, such as CPU, GPU, Digital



(a) Object detection processing pipeline for mobile/IoT devices [19] (b) Speech recognition processing pipeline for mobile/IoT devices [20]

Figure 2 – Comparison between two end-to-end processing pipelines for AI applications. WAV: waveform audio file format; STFT: short-time Fourier transform; MFCC: Mel frequency cepstral coefficients.



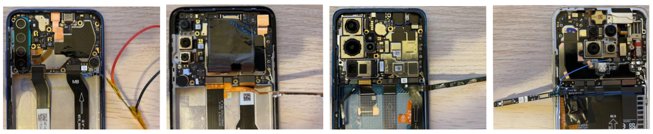
Applications	Computer vision based	NLP based	Voice based			
Datasets	ImageNet	COCO	VOC	KITTI	WMT	
DNN models	MobileNet	ShuffleNet	SqueezeNet	Inception v3		
Frameworks	TensorFlow Lite	Core ML	Caffe2	MXNet	PyTorch Mobile	
		NNAPI	ENN	SNPE		
Mobile OSs	Android	iOS	Windows Mobile	Ubuntu Touch		
Hardware accelerators	CPUs	GPUs	NPUs	DSPs	FPGAs	TPUs

Figure 3 – The diversity of the mobile and IoT ecosystem presents challenges in achieving benchmarking representativeness (information source partially from [22]). TPU: tensor processing unit; NNAPI: neural networks API; SNPE: Snapdragon neural processing engine.

Signal Processor (DSP), and a Neural Processing Unit (NPU), among others. These components can be utilized to support AI inference on mobile and IoT devices, either individually or in combination. Hence, the diversity of the mobile and IoT ecosystem provides a rich set of choices and opportunities for AI deployment, but it also presents challenges in achieving benchmarking representativeness.

P4: Extensibility. The mobile and IoT ecosystem, including hardware and software stack, are rapidly evolving. As a result, the benchmark needs to be regularly updated to accommodate these changes and ensure its relevance to the latest AI technologies. This requires significant resources and ongoing effort from benchmarking researchers to stay up-to-date with the latest developments in AI and adapt the benchmark accordingly. We discuss and present our plans to extend the AIEnergy benchmark suite over time in Section 6.

P5: Understandability and adoptability. The effectiveness of a benchmark hinges on its broad accessibility to diverse audiences, including AI end users without technical backgrounds and researchers of varying expertise levels. As discussed in Section 1, users usually regard the power/energy efficiency of device and AI application as a pivotal consideration. Clear and understandable benchmark results empower end users make well-informed decisions. For example, they can compare different devices based on the balance between AI performance and energy efficiency to select the one that best meets their requirements. Conversely, an easily adoptable energy benchmark can significantly propel advancements in AI research. For instance, the benchmark can provide a standardized and precise method for evaluating the energy efficiency of various System on Chips (SoCs) in the development of AI accelerators. Additionally, researchers can utilize the benchmark to test their approaches for enhancing end-to-end pipelines and refining DNN models to optimize AI applications. Hence, the benchmark must strike a balance between accessibility and technical depth.



(a) Redmi Note8 (b) One Macro (c) P40 Pro (d) K30 Ultra

Figure 4 – Examples of tested mobile and IoT devices with segregated BMS chips.

3. AIENERGY BENCHMARKS

To address these challenges, we developed AIEnergy, the first AI energy benchmark suite and benchmarking methodology for modern mobile and IoT devices. In this section, we present our proposed approaches for accurate and interpretable energy measurements, representative AI hardware and software stack, and an end-user understandable scoring system, respectively.

3.1 Accurate and interpretable energy measurements

To tackle the challenges presented in Section 2 and adhere to principles P1 and P2, we develop an energy measurement methodology that incorporates three strategies.

Proposed strategy 1. As discussed in Section 2 and Fig. 1, the built-in current sensor in commercial mobile and IoT devices fails to capture accurate and fine-grained power measurements because of their low power sampling frequency. While connecting to power monitors demonstrates promising accuracy for tracing fine-grained power fluctuations, it is challenging to connect them to modern device hardware with FPC battery connectors. To tackle this issue, we initially utilized off-the-shelf mechanical device DC power cables that are compatible with various devices with FPC connectors. It needs minimal effort from benchmarking researchers to connect the measured device to the external power monitor. Unfortunately, the devices we measured fail to boot because they lack the proprietary Battery Management System (BMS) chipsets, which are commonly affixed to the embedded batteries in modern mobile and IoT platforms.

The BMS is a crucial electronic system that is responsible for monitoring and managing the safety and performance of the battery. The operating system requires communication with the proprietary BMS to verify the battery's status and safety before the device can be powered on. Consequently, the device will fail to boot if a non-authorized battery is installed or if its battery is disconnected. After evaluating various solutions, we find that the most efficient approach involves detaching the BMS chipset from the device's battery without complete disassembly, then using it as the bridge for the connection between the measured device and the power monitor, as shown in Fig. 4. Firstly, the device is fully powered off. It is then carefully disassembled using specialized tools, such as plastic pry instruments and precision screwdrivers, to remove the rear panel and any protective hardware. Once the embedded battery pack is exposed, the BMS chipset, typically located adjacent to the battery connector and often concealed beneath insulating or shielding materials, is identified. Any insulating tape or adhesive securing the battery contacts is gently removed. Precision tweezers and a fine-tip soldering iron are then used to carefully desolder the BMS chipset from the battery cells, specifically detaching the positive (+) and negative (-) terminals. Insulated electrical wires are attached to these desoldered terminals on the detached BMS chipset, with the opposite ends securely connected to the respective terminals of an external power monitoring device. This setup enables the external power monitor to both power the device and simultaneously measure its power consumption. This approach has been validated on more than ten different modern devices, all of which successfully powered on with full functionality. It provides a promising reproducibility and can be extended to other mobile and IoT devices with FPC with minimal effort.

Proposed strategy 2. We establish a strict set of run rules to enhance the accuracy and reproducibility of the AIEnergy benchmark. These rules are designed to minimize any interference from the measurement environment and background activities. Each device must meet the following conditions before any measurement is taken [18, 23, 24, 25]:

- Disable connectivity features such as cellular, Bluetooth, WiFi, and NFC to minimize measurement inaccuracies caused by these interfaces.
- Terminate and disable any non-essential background applications and services to further reduce measurement interference. Potential activities and services can be identified by observing the power curve of the device when it is idle. For example, a flat curve typically indicates that no extraneous processes are running.
- Set the screen refresh rate to 60 Hz.
- Adjust the display to its minimum brightness level and disable adaptive brightness.
- If the benchmarking AI application involves the device's camera, adjust its sampling frequency to 15 fps.
- Perform measurements at a controlled room temperature, maintained between 20 and 25 degrees Celsius.
- Ensure proper ventilation and maintain an air gap around the device to manage its temperature and avoid thermal throttling during measurements.
- Ensure a break setting of 0 to 5 minutes between individual tests to allow the measured device to return to its cooldown state before starting the next one.

Proposed strategy 3. We develop a ready-to-adopt approach for breaking down the power and energy consumption of individual phases in an end-to-end processing pipeline. This enables the identification of energy bottlenecks and facilitates the interpretation of insights. The main idea is to synchronize the timestamps between the log files recorded by the tested mobile and IoT device (e.g., `getTimeInMillis()`) and the power trace sampled by the Monsoon power monitor. However, this is nontrivial, as the tested device and the power monitor do not share the same global clock. To tackle this challenge, we develop a method of creating a flag event that can be precisely and consistently identified in both the timestamps recorded by the device and the power trace captured by the Monsoon power monitor. After trying different events, we choose the "touch event" that activates the AI application as the flag event for synchronization. Once the touch event has been identified, it can be marked in both the device logs and the power monitor data by recording the timestamp at which it occurs. This allows for precise synchronization of the two data sources, and ensures that any changes in power consumption can be correlated with specific actions or events on the mobile and IoT device. Fig. 5 illustrates an example of the synchronization between the power trace and the device's recorded timestamps.

As an illustration of the touch event identification and synchronization method, suppose an AI application is launched and it first displays a User Interface (UI) containing a button to activate the application. Initially, the power consumption of the mobile and IoT device would be low and stable because there is no activity within the application until the button is touched. As soon as the button is pressed, a sudden increase in power consumption would be registered by the power monitor, and the device would record the timestamp of the touch event. This allows for easy and precise synchronization of the local clocks in the device and the power monitor.

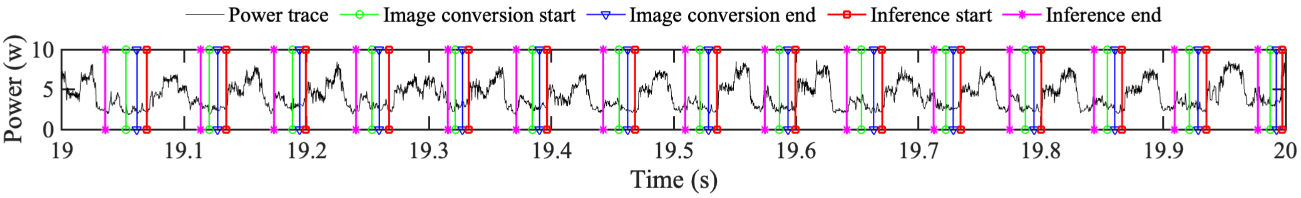


Figure 5 – An example of the synchronization between the power monitor trace and device recorded timestamps. Tested device: Huawei P40 Lite; application: object detection; DNN model: MobileNet-V2, FP32; delegate: CPU, 1 thread; image resolution: 300 × 300 pixels.

3.2 Representative hardware and software stack

We address the principle P3 from the perspectives of both mobile and IoT hardware and AI software stack.

Devices and chipsets. AIEnergy selects eight modern devices, each featuring a distinct SoC from the world-leading chip manufacturers, including MediaTek, Qualcomm, and HiSilicon. Table 1 presents the detailed specifications. These selected SoCs can represent the advanced and widely used AI silicon present in modern mobile and IoT devices.

Qualcomm is one of the first mobile SoC vendors that launch specialized AI silicon. The first generation Qualcomm AI engine, Snapdragon 820, was released in 2016. In 2020, Qualcomm launched its fifth generation AI engine, Snapdragon 865, containing eight Kryo CPU cores, a Hexagon 698 processor and an Adreno 650 GPU. It brings the total Qualcomm AI Engine performance up to 15 Tera Operations Per Second (TOPS). The Hexagon 698 processor consists of a tensor accelerator, vector eXtensions, and scalar accelerator, which is optimized for a deep learning workload. Furthermore, Adreno GPU is responsible for Floating-Point (FP) models, while the Hexagon processor is used for quantized inference.

HiSilicon launched its first mobile AI SoC that is named as Kirin 970 with a dedicated NPU, in 2017. At the end of 2020, HiSilicon released its mobile AI SoC, Kirin 9000, containing four Cortex-A77 CPUs, four Cortex-A55 CPUs, a Mali-G78 GPU with 24 cores, and a Da Vinci 2.0 NPU with two big cores and one tiny core. The triple-core NPU can achieve better energy efficiency, where the two big cores are responsible for heavy AI computations and the tiny core is for low-power AI computations.

MediaTek launched its first mobile AI SoC with a dedicated AI Processing Unit (APU) in early 2018, named Helio P60. The design of the APU was optimized for operations intensively used in DNN models. In 2020, MediaTek released its mobile AI SoC, Dimensity 1000 plus, containing four Cortex-A77 CPUs, four Cortex-A55 CPUs, a Mali-G77 GPU with nine cores, and a hexa-core MediaTek 3.0 APU that can offer up to 4.5 TOPS performance and support popular data precision, including 16-bit floating-point (FP16), 16-bit integer (INT16), and 8-bit integer (INT8).

Software stack. As summarized in Table 2, AIEnergy includes six of the most widely used AI applications, spanning three primary categories: vision-based, language-based, and voice-based applications. All those applications are developed based on TensorFlow Lite (TFLite) [26], a generic AI framework for mobile and IoT platforms. In addition, AIEnergy provides 12 reference DNN models with various architectures, which are available in both FP32 and quantized INT8 weight formats. These reference DNN models can represent the most commonly used deep learning architectures for mobile and IoT devices.

Moreover, AIEnergy leverages TFLite Delegates [27], including the GPU and NNAPI delegates, to enable hardware acceleration of inference on mobile and IoT devices. By default, TFLite uses CPU kernels that are optimized for the ARM Neon instruction set. But it is a common perception that the CPU is not optimized for running DNN models due to the heavy arithmetic. TFLite Delegates allow TFLite to delegate the execution of certain operations to specialized hardware accelerators such as GPUs, DSPs, or custom accelerators. AIEnergy evaluates each reference DNN model's power consumption, latency, and energy consumption across four different processing configurations: CPU with 1 or 4 threads, GPU, and NNAPI.

3.3 End-user understandable scoring system

We then address the principle P5, understandability and adoptability, to ensure that both end users without a technical background and researchers with varying levels of expertise can effectively utilize AIEnergy. Specifically, we develop two scoring metrics in addition to detailed measurement results, which can help to convey the power/energy efficiency of diverse AI-empowered mobile and IoT devices in an understandable and clear manner.

Power Efficiency Rating (PER). The PER measures the Power Efficiency (PE) of a device when executing all six applications utilizing 12 DNNs (listed in Table 2) in AIEnergy. Given an AI application with a reference DNN model, the PE of the measured device is defined as:

$$PE = \left(1 - \frac{APC}{TDP}\right) \times 100, \qquad (1)$$

Table 1 – Specifications of mobile and IoT device hardware and chipsets in AIEnergy

Device		Xiaomi Redmi K30 Ultra	OnePlus 8 Pro	Huawei Mate40 Pro	Xiaomi Redmi Note8	Huawei P40 Pro	Motorola One Macro	Huawei P40 Lite	Huawei P40 Lite E
SoC		Dimensity 1000+	Snapdragon 865	Kirin 9000	Snapdragon 665	Kirin 990 5G	Helio P70	Kirin 810	Kirin 710F
Manufacturer		MediaTek	Qualcomm	HiSilicon	Qualcomm	HiSilicon	MediaTek	HiSilicon	HiSilicon
CPU	MA #C HF	ARM A77+A55 4+4 2.6 GHz	A77+A55 4+4 2.84 GHz	A77+A55 4+4 3.13 GHz	A73+A53 4+4 2.0 GHz	A76+A55 4+4 2.86 GHz	A73+A53 2+6 2.1 GHz	A76+A55 4+4 2.27 GHz	A73+A53 4+4 2.2 GHz
GPU		Mali-G77	Adreno 650	Mali-G78	Adreno-610	Mali-G76	Mali-G72	Mali-G52	Mali-G51
Dedicated AI accelerator		MediaTek 3.0 APU	Hexagon 698 DSP	Ascend Lite+Tiny NPU Da Vinci 2.0	Hexagon 686 DSP	Lite+Tiny NPU	MediaTek APU Da Vinci	D100 Lite NPU Da Vinci	None
RAM		6 GB LPDDR4x	8 GB LPDDR5	8 GB LPDDR5	4 GB LPDDR4X	8 GB LPDDR4X	4 GB LPDDR4x	8 GB LPDDR4X	4 GB LPDDR4
Process		7 nm	7 nm	5 nm	11 nm	7 nm	12 nm	7 nm	12 nm
OS		Android 10	Android 10	Android 10	Android 10	Android 10	Android 9	Android 10	Android 10
NNAPI support		Yes	Yes	Yes	Yes	Yes	Yes	Yes	Yes
Display	S FR	6.67 inches 120/60 Hz	6.78 inches 120/60 Hz	6.76 inches 90/60 Hz	6.67 inches 60 Hz	6.58 inches 90/60 Hz	6.2 inches 60 Hz	6.40 inches 90/60 Hz	6.39 inches 60 Hz
Battery	BC R	4500 mAh No	4510 mAh No	4400 mAh No	4500 mAh No	4200 mAh No	4000 mAh No	4200 mAh No	4000 mAh No

[*] MA: Micro-architecture; #C: number of cores; HF: Highest CPU frequency; S: Size of display; FR: Display fresh rate; BC: Battery capacity; and R: Removable battery.

where APC denotes the average power consumption during inference, and TDP refers to the thermal design power, the maximum sustained power the device can dissipate under typical operating conditions without overheating. TDP is used as the denominator because it represents a standardized, hardware-defined upper bound of sustainable power drawn across mobile and IoT platforms. In contrast to peak power, which captures short-term power spikes and can vary significantly due to dynamic workload fluctuations or DVFS policies, TDP reflects the long-term thermal and power budget of the device. This makes it more appropriate for characterizing energy efficiency under real deployment conditions. Additionally, TDP is widely documented across commercial hardware, enhancing benchmarking consistency and cross-platform reproducibility. The ratio $\frac{APC}{TDP}$ reflects how effectively a device utilizes its power budget, with a lower value indicating better efficiency. The overall PER is computed by averaging the PE across all models and configurations, including CPU (1-thread and 4-thread), GPU, and NNAPI:

$$PER = \frac{\sum_{i=1}^{n} PE_i}{n}, \tag{2}$$

where n is the number of DNNs in the benchmark. A higher PER suggests the device can support longer battery life under AI workloads.

However, PER does not account for inference performance metrics such as latency. Devices may achieve high PER scores by underclocking or using low-performance configurations, which can degrade real-time responsiveness. Therefore, an additional metric is needed to capture the trade-off between power efficiency and AI performance.

Inference Energy Performance Rating (IEPR). The IEPR is developed to capture the energy efficiency of a mobile and IoT device when running AI applications. We first design a new metric named Inference Efficiency Rate (IER), which is the number of inferences a device can perform while consuming one unit of energy. The IER is calculated as

$$IER = \frac{NI}{EC}, \tag{3}$$

where NI and EC denotes the number of inferences and the device's energy consumption within a certain time period (e.g., 1 second), respectively. The IEPR of a mobile and IoT device is the overall IER when running all applications and reference DNNs designed in the benchmark across different processing configurations, including CPU with 1 or 4 threads, GPU, and NNAPI. It is defined as:

$$IEPR = \sum_{i=1}^{n} IER_i. \tag{4}$$

An AI-empowered mobile and IoT device is considered more energy-efficient when it acquires a higher IEPR.

4. AIENERGY BENCHMARK RESULTS

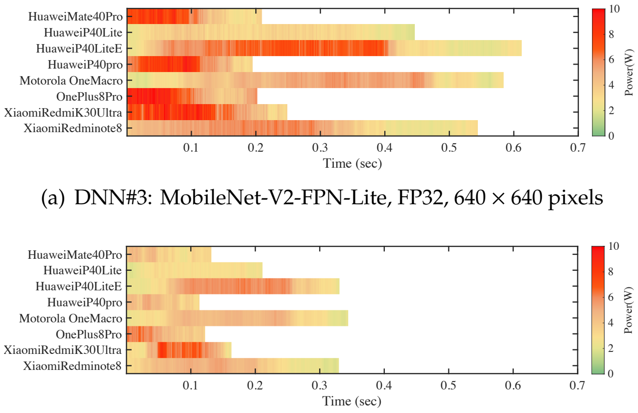
In this section, we present quantitative benchmark results obtained from 264 configuration combinations of eight commercial mobile and IoT devices with diverse AI chipsets (summarized in Table 1) and six AI applications with 12 DNN models (summarized in Table 2) under CPU, GPU, and NNAPI delegates. In addition, existing work [28] found that a single energy measurement can be misleading due to the variability in energy consumption. Hence, each benchmark result presented in tables 3 and 4 is the average of at least 200 inferences[1]. *The total amount of measurement data in AIEnergy is over* 8.8 *GB. The data will be made publicly available.*

[1] We observed that the variance in the average measured power and energy consumption becomes negligible after at least 200 inferences.

Table 2 – Summary of software stacks implemented in AIEnergy

Category	Application	Reference DNN model			Delegate				ID
		Model	Data Precision	Input (pixels)	CPU1	CPU4	GPU	NNAPI	
Vision-based	Object detection	MobileNet-V2	FP32	300 × 300	▼	▼		▼	DNN#1
		MobileNet-V2	INT8	300 × 300	▼	▼		▼	DNN#2
		MobileNet-V2-FPN-Lite	FP32	640 × 640	▼	▼		▼	DNN#3
		MobileNet-V2-FPN-Lite	INT8	640 × 640	▼	▼		▼	DNN#4
	Image classification	EfficientNet	FP32	224 × 224	▼	▼	▼	▼	DNN#5
		EfficientNet	INT8	224 × 224	▼	▼		▼	DNN#6
		MobileNet-V1	FP32	224 × 224	▼	▼	▼	▼	DNN#7
		MobileNet-V1	INT8	224 × 224	▼	▼		▼	DNN#8
	Super resolution	ESRGAN	FP32	50 × 50	▼		▼		DNN#9
	Image segmentation	DeepLab-V3	FP32	257 × 257		▼			DNN#10
Language-based	Natural language processing	Mobile Bert	FP32	-	▼	▼		▼	DNN#11
Voice-based	Speech recognition	Conv-Actions-Frozen	FP32	-	▼	▼		▼	DNN#12

PER and IEPR results. Table 3 and Table 4 summarize the benchmark results in terms of power and energy efficiency. The entries in these two tables are color-coded to rank the devices running with each reference DNN model according to their APC and Average Energy consumption per Inference (AEI). AEI is defined as the amount of energy consumed to perform a single inference. In the last row, each device is provided with a final PER and IEPR.



(a) DNN#3: MobileNet-V2-FPN-Lite, FP32, 640 × 640 pixels

(b) DNN#4: MobileNet-V2-FPN-Lite, INT8, 640 × 640 pixels

Figure 6 – Examples of energy bottleneck identification within DNN models (CPU with 1 thread).

DNN results. Figures 6 and 7 show the power consumption fluctuations of devices while running different DNN models, which demonstrates that AIEnergy is capable of identifying energy bottlenecks within a DNN model by coordinating with the start and end timestamps of individual operators. In Fig. 6, a comparison is made between the power consumption characteristics of MobileNet-V2-FPN-Lite and its quantized version, where the quantized MobileNet-V2-FPN-Lite achieves significant power and energy reductions. Fig. 7 demonstrates that AIEnergy can identify energy bottlenecks within a DNN model across various mobile and IoT hardware devices and delegates.

End-to-end processing pipeline results. Fig. 8 presents the breakdown of energy consumption across different phases in the processing pipeline of object detection. This demonstrates that AIEnergy can offer clear and interpretable insights into identifying the primary energy drivers within an end-to-end processing pipeline for an AI application.

5. INSIGHTS FROM BENCHMARK RESULTS

5.1 Insight 1: scoring metrics matter for benchmarking

The scoring metric can significantly affect the evaluation of AI performance and efficiency, as well as the results of a benchmarking study on mobile and IoT devices. Fig. 9 visualizes the quantitative benchmarking results with four different metrics for individual devices, including the AI inference score developed in AI Benchmark [29, 30], PER, IEPR, and the reciprocal of selling price. We choose the radar charts due to their exceptional ability to display multivariate data in a comprehensible format. Each axis on a radar chart represents one of the trade-offs, which easily visualizes how multiple variables compare and contrast for each item being analyzed. It is straightforward to identify strengths, weaknesses, and balance across various factors in a single, cohesive visual representation. In Fig. 9, a larger area within the radar chart generally suggests better overall performance across the included benchmarking metrics. Additionally, the shape of the area is equally crucial. A device may not exhibit the largest coverage but could extend significantly in key metrics like PER or AI score, which may hold greater importance for specific applications or user preferences. For instance, the Mate40 Pro secures the highest AI performance score, yet it ranks second lowest in terms of PER. In contrast, the Redmi Note8, while leading in PER, finds itself at the lower end with the second lowest AI performance score. This disparity highlights that the AI score does not account for power or energy efficiency, underscoring the need for a balanced metric. The introduction of the IEPR in AIEnergy addresses this by considering the balance between PE and AI inference performance, enabling a more comprehensive and fair evaluation of mobile and IoT devices. The P40 Pro, for instance, exem-

Table 3 – AIEnergy benchmark result I - PER

Hardware Devices			Xiaomi Redmi Note8	Huawei P40 Pro	OnePlus 8 Pro	Huawei P40 Lite	Huawei P40 Lite E	Xiaomi Redmi K30 Ultra	Huawei Mate40 Pro	Motorola One Macro
SoCs			Snapdragon665	Kirin990-5G	Snapdragon865	Kirin810	Kirin710F	Dimensity1000+	Kirin9000	Helio-P70
TDP (watt)			6	15	10	12	8	10	15	8
DNN#1	CPU1	APC	4.1483	4.4851	4.7495	2.6663	5.4068	3.7958	4.3433	3.7454
	CPU4	APC	3.5806	4.3876	4.8083	2.5310	4.2025	3.2196	3.7868	3.3152
	NNAPI	APC	3.6269	4.3285	4.8055	2.5199	3.9966	3.0397	3.8100	3.3499
DNN#2	CPU1	APC	3.4319	3.2151	3.9760	2.0250	2.7139	2.4900	2.5743	3.0976
	CPU4	APC	/	2.9271	3.7985	2.0250	2.8956	2.6120	2.7118	2.9373
	NNAPI	APC	2.9326	2.8651	3.7337	1.9390	2.7319	2.6699	2.5841	2.9116
DNN#3	CPU1	APC	4.4445	6.4258	6.5488	3.1547	5.0260	6.5780	5.6446	4.2065
	CPU4	APC	4.3751	6.0799	6.4656	3.1507	4.3356	6.3693	5.0218	3.9965
	NNAPI	APC	4.4895	6.4144	6.5409	2.9834	3.9964	6.8599	4.9898	3.8940
DNN#4	CPU1	APC	3.8352	4.1216	4.8723	2.7973	4.6100	5.2172	3.5964	3.8147
	CPU4	APC	3.7075	4.2976	4.8153	2.6839	4.2709	4.3025	3.7172	3.6432
	NNAPI	APC	3.7466	4.3168	5.0417	2.5156	4.1743	4.2974	3.7576	3.4983
DNN#5	CPU1	APC	2.5334	3.1219	3.5176	1.9588	3.4052	2.4806	2.4226	2.7977
	CPU4	APC	2.3515	3.1186	3.9474	2.4127	3.0808	2.2546	2.4624	2.6370
	GPU	APC	2.0625	2.7089	3.1602	1.6984	1.7393	2.3600	2.2066	2.0687
	NNAPI	APC	2.1710	3.0256	3.3014	2.2599	3.1470	5.2780	2.2543	2.3403
DNN#6	CPU1	APC	1.9324	2.7704	3.3375	1.8010	2.6868	2.3745	2.3723	2.1569
	CPU4	APC	2.0795	2.8345	3.2977	1.9455	2.4146	2.3747	3.1146	2.2178
	NNAPI	APC	2.5139	3.5347	3.4051	2.1766	2.6293	4.5144	3.2671	2.2252
DNN#7	CPU1	APC	2.7262	3.8398	4.0808	2.3526	2.9596	2.4317	2.6128	2.8080
	CPU4	APC	2.2724	3.3248	3.6885	2.1872	2.9300	2.3699	2.5668	3.0107
	GPU	APC	2.0794	2.5343	3.2397	1.6904	1.6904	2.1920	1.9101	2.1333
	NNAPI	APC	2.1638	2.6743	3.2527	1.7317	3.2639	5.2395	2.2492	2.1315
DNN#8	CPU1	APC	2.3942	2.8653	2.8792	1.8143	2.0548	2.3008	2.2488	2.4511
	CPU4	APC	2.1764	2.6944	3.1023	1.8362	2.6199	2.3291	2.2371	2.4170
	NNAPI	APC	1.8802	2.8766	3.3555	1.6034	1.9420	4.6366	2.7255	1.8699
DNN#9	CPU1	APC	2.3405	3.1425	4.0743	2.6565	3.7004	3.2577	4.3030	2.7834
	GPU	APC	0.4792	0.4492	0.5375	0.4847	0.3985	0.6442	2.8579	0.3609
DNN#10	CPU4	APC	1.4373	4.0929	2.8932	2.6162	3.2849	4.2790	3.4070	2.6808
DNN#11	CPU1	APC	1.7440	3.0304	3.1120	2.3063	2.2359	3.2577	3.1010	2.7834
	CPU4	APC	2.5173	4.1072	4.4738	1.2681	3.4850	3.3303	4.3165	2.5897
	NNAPI	APC	/	0.6434	1.6490	0.5901	1.0675	2.4449	2.3391	1.4254
DNN#12	CPU1	APC	1.1727	1.1058	1.6463	1.4339	2.5533	1.4339	1.1154	1.2112
	CPU4	APC	1.0677	1.3548	1.5411	1.1863	1.5975	1.3714	1.2546	1.1878
	NNAPI	APC	1.1054	1.3478	1.5671	1.1937	1.6169	1.4394	1.1449	1.1907
PER			78	78	75	74	70	67	62	56

[*] APC (watt): average power consumption; TDP (watt): thermal design power.

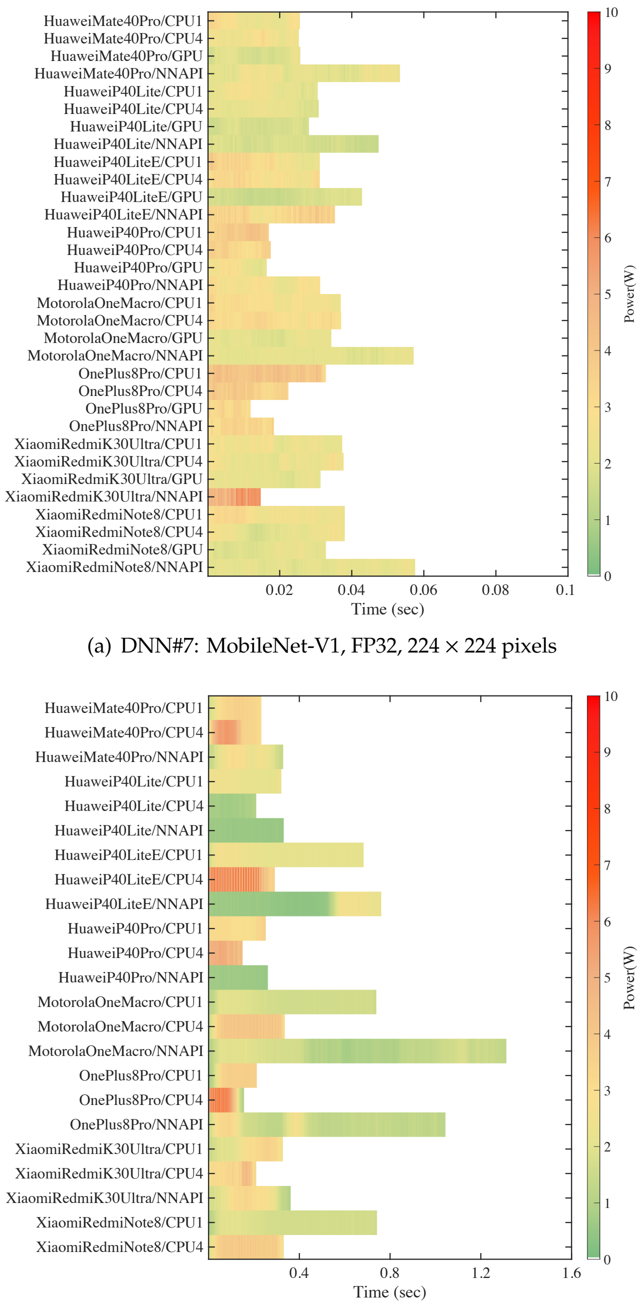
Ranking: ■ 1 ■ 2 ■ 3 ■ 4 ■ 5 ■ 6 ■ 7 ■ 8

Table 4 – AIEnergy benchmark result II - IEPR

Hardware Devices			Huawei P40 Pro	Huawei Mate40 Pro	Huawei P40 Lite	OnePlus 8 Pro	Xiaomi Redmi K30 Ultra	Xiaomi Redmi Note8	Huawei P40 Lite E	Motorola One Macro
SoCs			Kirin990-5G	Kirin9000	Kirin810	Snapdragon865	Dimensity1000+	Snapdragon665	Kirin710F	Helio-P70
DNN#1	CPU1	AEI	0.1715	0.2421	0.2649	0.3209	0.4640	0.4421	0.5132	0.4481
	CPU4	AEI	0.1854	0.2439	0.2790	0.2858	0.4735	0.4656	0.5418	0.4441
	NNAPI	AEI	0.2030	0.2221	0.2419	0.3130	0.4421	0.4844	0.4148	0.4237
DNN#2	CPU1	AEI	0.1069	0.1228	0.1469	0.2228	0.2158	0.2592	0.2452	0.2270
	CPU4	AEI	0.1021	0.1186	0.1497	0.2034	0.1880	/	0.2375	0.2334
	NNAPI	AEI	0.0933	0.1128	0.1361	0.2149	0.3032	0.0265	0.1623	0.2443
DNN#3	CPU1	AEI	1.2684	1.2222	1.3527	1.3727	1.6245	2.3948	3.0454	2.4968
	CPU4	AEI	1.3390	1.1794	1.3869	1.3868	1.6299	2.4183	3.1238	2.4786
	NNAPI	AEI	1.2899	1.1156	1.3407	1.3905	1.7859	2.4250	2.7439	2.4988
DNN#4	CPU1	AEI	0.5117	0.5194	0.6088	0.6314	0.6435	1.2265	1.4706	1.2757
	CPU4	AEI	0.5424	0.5437	0.5692	0.6150	0.7760	1.2402	1.5131	1.2519
	NNAPI	AEI	0.5082	0.5174	0.5942	0.6120	0.7759	1.2311	1.3755	1.3334
DNN#5	CPU1	AEI	0.1103	0.1055	0.1301	0.1792	0.2261	0.2009	0.2058	0.2123
	CPU4	AEI	0.1154	0.1249	0.1419	0.1778	0.2252	0.2041	0.1875	0.2144
	GPU	AEI	0.0964	0.0954	0.0979	0.1118	0.1615	0.1432	0.1714	0.1504
	NNAPI	AEI	0.1916	0.1238	0.2295	0.2350	0.4128	0.3672	0.3287	0.2351
DNN#6	CPU1	AEI	0.0797	0.1015	0.1079	0.1923	0.1928	0.1678	0.1675	0.1657
	CPU4	AEI	0.0806	0.1191	0.1061	0.1870	0.1962	0.1619	0.1403	0.1457
	NNAPI	AEI	1.6405	1.3280	1.3641	0.9329	0.0819	0.2756	0.1267	2.9408
DNN#7	CPU1	AEI	0.1261	0.1475	0.1449	0.3309	0.2803	0.2139	0.2247	0.2302
	CPU4	AEI	0.1228	0.1448	0.1414	0.1854	0.2572	0.2348	0.2429	0.2348
	GPU	AEI	0.0816	0.1461	0.0943	0.0803	0.1586	0.1326	0.1536	0.1650
	NNAPI	AEI	0.1652	0.2247	0.1758	0.1277	0.1631	0.2623	0.2246	0.2644
DNN#8	CPU1	AEI	0.0814	0.0956	0.0887	0.1633	0.1796	0.1635	0.1248	0.1943
	CPU4	AEI	0.0764	0.0884	0.0887	0.1555	0.1743	0.1811	0.1250	0.1614
	NNAPI	AEI	0.0466	0.0457	0.0772	0.0711	0.0758	0.0678	0.0996	0.0789
DNN#9	CPU1	AEI	1.5211	1.4124	1.5064	1.6846	2.0102	3.0001	3.9297	3.1099
	GPU	AEI	0.1502	0.9453	0.3426	0.1369	0.2993	0.5716	0.4584	0.4053
DNN#10	CPU4	AEI	0.3819	0.3943	0.3451	0.3411	0.6042	0.3481	0.7609	0.7436
DNN#11	CPU1	AEI	1.8226	1.9101	1.8624	1.5330	2.2376	3.1282	3.7850	3.2543
	CPU4	AEI	1.8889	1.8117	0.3534	2.1198	2.0826	2.8539	3.9754	2.9359
	NNAPI	AEI	0.4334	2.0306	0.4605	4.5117	2.4897	/	2.0196	7.0580
DNN#12	CPU1	AEI	0.1681	0.1094	0.1209	0.3367	0.6353	0.3038	0.3491	0.4105
	CPU4	AEI	0.1530	0.2489	0.1220	0.3029	0.4117	0.2610	0.3222	0.3606
	NNAPI	AEI	0.1695	0.2264	0.0861	0.3241	0.3785	0.2798	0.2988	0.3535
IEPR			260	224	202	173	158	140	138	129

[*] AEI (Joule): represents the average energy cost per inference, evaluating the amount of energy needed to perform a single inference on a mobile and IoT device.

Ranking: ■ 1 ■ 2 ■ 3 ■ 4 ■ 5 ■ 6 ■ 7 ■ 8



(a) DNN#7: MobileNet-V1, FP32, 224 × 224 pixels

(b) DNN#11: Mobile Bert, FP32

Figure 7 – Energy bottleneck identification across various mobile and IoT devices and delegates.

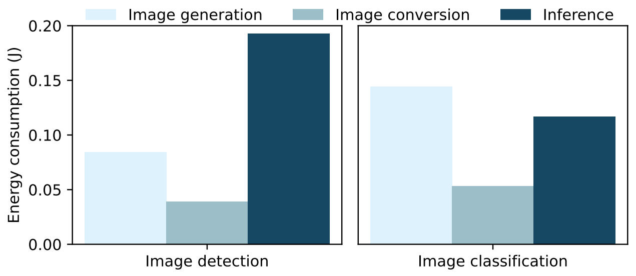


Figure 8 – Energy consumption breakdown for end-to-end processing pipelines. Measured device: Huawei Mate40 Pro; delegate: CPU, 1 thread.

plifies an optimal balance between AI performance and power efficiency, as reflected in its IEPR and positioning in the chart. This approach illuminates the trade-offs between efficiency and AI capabilities, essential for making informed decisions in technology deployment.

5.2 Insight 2: energy efficiency of AI framework is overlooked

The Android NNAPI is designed to provide a base layer of functionality for higher-level AI frameworks such as TFLite. In TFLite, the NNAPI delegate supports the acceleration of DNN models on mobile devices by distributing the workload across CPUs, GPUs, DSPs, and NPUs. However, we observe that the energy efficiency of NNAPI is overlooked. For object detection, Natural Language Processing (NLP), and speech recognition, NNAPI consumes more energy than other delegates in about 50% of the cases, as demonstrated by the AIEnergy benchmark results in Table 4. Its energy efficiency exacerbates further in image classification. This is because many TFLite operations are not supported by the NNAPI delegate. To address this issue, TFLite initially checks which operations in the input DNN model can be performed using the delegate. It then divides the original graph into several subgraphs and substitutes each subgraph that can be handled by the delegate with a delegate node. The delegate is then responsible for carrying out subgraphs in the corresponding nodes. Unsupported operations are by default computed by the CPU, which could result in a significant increase in power and energy consumption due to the overhead of transferring results from the subgraph to the main graph.

5.3 Insight 3: software-hardware co-design plays a crucial role

The benchmark results demonstrate that the hierarchy of mobile and IoT ecosystems, including both hardware and AI software stack, complicates the energy efficiency optimization for devices. No mobile AI chipset dominates all the reference DNN models in the AIEnergy benchmark. In Table 4, each SoC obtains a disparate AEI when running the same DNN model. HiSilicon's Kirin 990 5G consumed the least AEIs when running most reference DNN models with CPU and GPU, while Qualcomm's Snapdragon 865 and MediaTek's Dimensity 1000+ are competitive when running DNN#6 and DNN#7 for image classification with NNAPI. Therefore, improving the AI energy efficiency on modern mobile and IoT devices requires synergistic co-designs between software and hardware (e.g., co-design for DNN architecture and AI hardware acceleration). A low-power Convolutional Neural Network (CNN) processor for face recognition was designed for mobile devices in [31, 32],

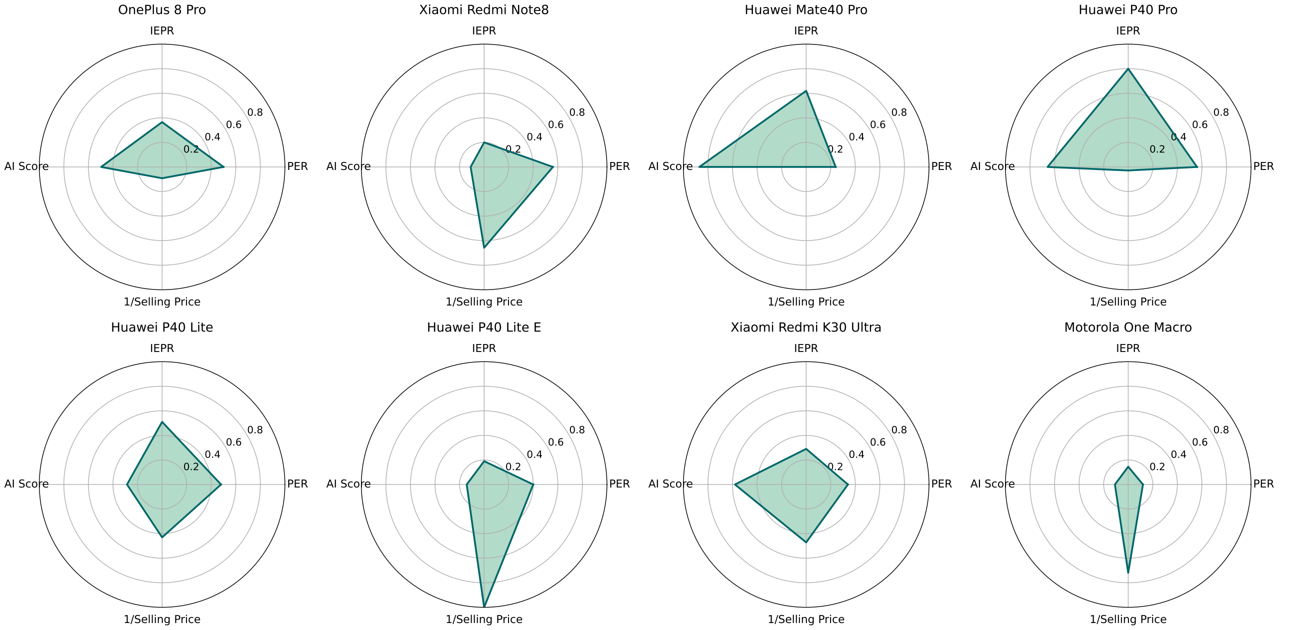


Figure 9 – Visualization of benchmarking results across four distinct metrics: AI performance score [29, 30], IEPR, PER, and the reciprocal of selling price between devices. and the reciprocal of selling price. Each radar chart illustrates the trade-offs between AI performance, power efficiency, energy efficiency, and the cost-effectiveness of the devices.

an example of application-specific software-hardware co-design. However, as modern mobile and IoT devices are multi-model and support a variety of AI applications, it is essential to consider co-designs that can accommodate both specialized and general AI applications. *AIEnergy offers software-hardware co-design researchers a reproducible approach to evaluate and detect energy bottlenecks in real devices running various AI applications.*

6. DISCUSSION AND ONGOING WORK

We are currently involved in multiple efforts to improve the extensibility (principle P4) of our AIEnergy benchmark.

Expanding the benchmark suite. We are working to expand our AIEnergy benchmark beyond three dimensions: *regular device updating, on-device training, and AI frameworks from a variety of vendors.* First, AIEnergy can be easily extended to support new modern devices, and we are committed to regularly updating our benchmark suite to include the latest devices as they become available. In particular, Apple's iOS is widely recognized as a major player in mobile AI and has a strong reputation for energy efficiency. Therefore, we plan to include the results from iOS devices to further diversify our benchmark suite in the near future. Second, on-device training is energy-intensive, but becoming increasingly prevalent on modern mobile and IoT devices. As AIEnergy currently only focuses on AI inference, we are expanding it to include on-device training as a new dimension. Third, SoC vendors often provide proprietary AI frameworks that are optimized to run on their specific hardware. Some examples of these frameworks include Qualcomm's SNPE [33], HiSilicon's HiAI [34], MediaTek's NeuroPilot [35], and Samsung's ENN [36]. These frameworks are designed to accelerate the execution of DNNs on the SoC's specialized processing units. AIEnergy can be readily extended to measure the energy efficiency of these vendor proprietary AI frameworks.

Developing online energy estimation techniques for AI-empowered mobile and IoT platforms. AIEnergy currently requires physical access to external power monitors for accurate measurements. Although we have developed a ready-to-adopt benchmark methodology and detailed documentation with step-by-step instructions for implementing this methodology, it may still be difficult for AI end users or software developers to perform AIEnergy measurements on their own devices or applications due to the lack of access to external power monitors. In our ongoing work, we are investigating online energy estimation techniques that do not require the external power monitor and support automated energy consumption assessment. Although the online energy estimation techniques may be more accessible, AIEnergy with physical measurements is indispensable for achieving accurate online energy estimation, as it will serve as the ground truth against which the online techniques can be validated and calibrated.

7. RELATED WORK

AI benchmark for mobile and IoT devices. Recent studies have developed mobile AI benchmarks to measure the on-device learning performance. For instance, MLPerf Mobile [22, 21] is the first industry-standard, open-source benchmark for evaluating the performance and accuracy of mobile devices. AI Benchmark [29, 30] is one of the first benchmark suites primarily targeting Android smartphones and measuring only latency. Additionally, AIoTBench [37] includes a broader range of DNN architectures and AI frameworks, focusing on assessing the inference capabilities of embedded and mobile devices. However, none of these AI benchmarks prioritize energy efficiency or aim to create an energy benchmark that accounts for the diverse hardware and software stacks in the mobile and IoT ecosystem.

Energy measurement for mobile and IoT devices. A few research efforts have developed various methodologies and frameworks to measure energy consumption of mobile, embedded, and IoT hardware. The Green Miner [38] is capable of physically measuring the device's energy consumption and automating application testing. The study in [39] examines the energy consumption of GUI colors on OLED displays. GfxDoctor, developed in [40], systematically diagnoses energy inefficiencies in app graphics at the source-code level. However, none of these pieces of work have focused on on-device energy evaluation for mobile and IoT devices with AI applications. Moreover, extending these methods to create an energy benchmark for modern mobile and IoT devices is challenging due to the need to address principles P1-P5 and the challenges discussed in Section 2.

8. CONCLUSION

This paper outlines the principles, challenges, strategies, and opportunities for developing the first AI energy benchmark, AIEnergy, for modern mobile and IoT devices. We collected over 8.8 GB measurement data from 264 configuration combinations of eight commercial devices with diverse AI chipsets, six AI applications with unique end-to-end processing pipelines, and 12 DNN models under CPU, GPU, and NNAPI delegates. Overall, the benchmark results with the developed scoring system can provide interpretable insights and guidelines for mobile AI optimization in terms of energy efficiency. Additionally, several ongoing pieces of work were presented to improve the extensibility of the AIEnergy benchmark, such as expanding the benchmark suite and developing online energy estimation techniques. We believe the benchmark results and insights provided by AIEnergy will help researchers and developers to optimize energy efficiency for AI-empowered mobile and IoT platforms, enable end users to make informed decisions which leads to better consumer awareness and responsible consumption habits, and encourage mobile SoC vendors to invest in more greener technologies.

REFERENCES

[1] Zhengxia Zou, Keyan Chen, Zhenwei Shi, Yuhong Guo, and Jieping Ye. "Object detection in 20 years: A survey". In: *Proceedings of the IEEE* 111.3 (2023), pp. 257–276.

[2] Kai Han, Yunhe Wang, Hanting Chen, Xinghao Chen, Jianyuan Guo, Zhenhua Liu, Yehui Tang, An Xiao, Chunjing Xu, Yixing Xu, et al. "A survey on vision transformer". In: *IEEE transactions on pattern analysis and machine intelligence* 45.1 (2022), pp. 87–110.

[3] Chenyu Yang, Yuntao Chen, Hao Tian, Chenxin Tao, Xizhou Zhu, Zhaoxiang Zhang, Gao Huang, Hongyang Li, Yu Qiao, Lewei Lu, et al. "BEVFormer v2: Adapting modern image backbones to bird's-eye-view recognition via perspective supervision". In: *Proc. the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*. 2023, pp. 17830–17839.

[4] Ziqi Yang, Xuhai Xu, Bingsheng Yao, Ethan Rogers, Shao Zhang, Stephen Intille, Nawar Shara, Guodong Gordon Gao, and Dakuo Wang. "Talk2Care: An LLM-based Voice Assistant for Communication between Healthcare Providers and Older Adults". In: *Proc. the ACM on Interactive, Mobile, Wearable and Ubiquitous Technologies* 8.2 (2024), pp. 1–35.

[5] Veton Kepuska and Gamal Bohouta. "Next-generation of virtual personal assistants (microsoft cortana, apple siri, amazon alexa and google home)". In: *Proc. IEEE 8th Annual Computing and Communication Workshop and Conference (CCWC)*. 2018, pp. 99–103.

[6] Arun James Thirunavukarasu, Darren Shu Jeng Ting, Kabilan Elangovan, Laura Gutierrez, Ting Fang Tan, and Daniel Shu Wei Ting. "Large language models in medicine". In: *Nature medicine* 29.8 (2023), pp. 1930–1940.

[7] Haotian Liu, Chunyuan Li, Qingyang Wu, and Yong Jae Lee. "Visual Instruction Tuning". In: *Proc. NeurIPS*. 2023.

[8] Iulian V Serban, Chinnadhurai Sankar, Mathieu Germain, Saizheng Zhang, Zhouhan Lin, Sandeep Subramanian, Taesup Kim, Michael Pieper, Sarath Chandar, Nan Rosemary Ke, et al. "A deep reinforcement learning chatbot". In: *arXiv preprint arXiv:1709.02349* (2017).

[9] Haoxin Wang, BaekGyu Kim, Jiang Xie, and Zhu Han. "LEAF+AIO: Edge-assisted energy-aware object detection for mobile augmented reality". In: *IEEE Transactions on Mobile Computing* 22.10 (2022), pp. 5933–5948.

[10] Haoxin Wang and Jiang Xie. "User Preference Based Energy-Aware Mobile AR System with Edge Computing". In: *IEEE INFOCOM 2020*. 2020, pp. 1379–1388.

[11] Hongyu Ke, Wanxin Jin, and Haoxin Wang. "CarbonCP: Carbon-aware DNN partitioning with conformal prediction for sustainable edge intelligence". In: *arXiv preprint arXiv:2404.16970* (2024).

[12] *Users Reveal Top Frustrations That Lead to Bad Mobile App Reviews*. https://finance.yahoo.com/news/apigee-survey-users-reveal-top-120200656. Accessed on May 2024.

[13] Abhilash Jindal and Y Charlie Hu. "Experience: developing a usable battery drain testing and diagnostic tool for the mobile industry". In: *Proc. the 27th Annual International Conference on Mobile Computing and Networking (MobiCom)*. 2021, pp. 804–815.

[14] *Monsoon Power Monitor*. https://www.msoon.com/specifications. Accessed on May 2024.

[15] Xukan Ran, Haolianz Chen, Xiaodan Zhu, Zhenming Liu, and Jiasi Chen. "Deepdecision: A mobile deep learning framework for edge video analytics". In: *Proc. IEEE Conference on Computer Communications (INFOCOM)*. 2018, pp. 1421–1429.

[16] Xiaomeng Chen, Abhilash Jindal, Ning Ding, Yu Charlie Hu, Maruti Gupta, and Rath Vannithamby. "Smartphone background activities in the wild: Origin, energy drain, and optimization". In: *Proc. the 21st Annual International Conference on Mobile Computing and Networking*. 2015, pp. 40–52.

[17] Kittipat Apicharttrisorn, Xukan Ran, Jiasi Chen, Srikanth V Krishnamurthy, and Amit K Roy-Chowdhury. "Frugal following: Power thrifty object detection and tracking for mobile augmented reality". In: *Proc. the 17th Conference on Embedded Networked Sensor Systems (SenSys)*. 2019, pp. 96–109.

[18] Xiaolong Tu, Anik Mallik, Dawei Chen, Kyungtae Han, Onur Altintas, Haoxin Wang, and Jiang Xie. "Unveiling Energy Efficiency in Deep Learning: Measurement, Prediction, and Scoring across Edge Devices". In: *Proceedings of the Eighth ACM/IEEE Symposium on Edge Computing (SEC 2023)*, pp. 80–93.

[19] *TensorFlow Lite Object Detection*. https://www.tensorflow.org/lite/examples/object_detection/overview. Accessed on May 2024.

[20] *TensorFlow Lite Speech Recognition*. https://www.tensorflow.org/lite/examples/audio_classification/overview. Accessed on May 2024.

[21] Vijay Janapa Reddi, David Kanter, Peter Mattson, Jared Duke, Thai Nguyen, Ramesh Chukka, Ken Shiring, Koan-Sin Tan, Mark Charlebois, William Chou, et al. "Mlperf mobile inference benchmark: An industry-standard open-source machine learning benchmark for on-device AI". In: *Proc. Machine Learning and Systems (MLSys)*. Vol. 4. 2022, pp. 352–369.

[22] Vijay Janapa Reddi, Christine Cheng, David Kanter, Peter Mattson, Guenther Schmuelling, Carole-Jean Wu, Brian Anderson, Maximilien Breughe, Mark Charlebois, William Chou, et al. "Mlperf inference benchmark". In: *Proc. ACM/IEEE 47th Annual International Symposium on Computer Architecture (ISCA)*. 2020, pp. 446–459.

[23] Xiaolong Tu, Dawei Chen, Kyungtae Han, Onur Altintas, and Haoxin Wang. "GreenAuto: An Automated Platform for Sustainable AI Model Design on Edge Devices". In: *Proceedings of the 26th International Workshop on Mobile Computing Systems and Applications (HotMobile)*. 2025, pp. 7–12.

[24] Xiaolong Tu, Anik Mallik, Haoxin Wang, and Jiang Xie. "Deepen2023: Energy datasets for edge artificial intelligence". In: *arXiv preprint arXiv:2312.00103* (2023).

[25] Anik Mallik, Haoxin Wang, Jiang Xie, Dawei Chen, and Kyungtae Han. "EPAM: A predictive energy model for mobile AI". In: *IEEE International Conference on Communications*. 2023, pp. 954–959.

[26] *TensorFlow Lite*. https://www.tensorflow.org/lite/guide. Accessed on May 2024.

[27] *TensorFlow Lite Delegates*. https://www.tensorflow.org/lite/performance/delegates. Accessed on May 2024.

[28] Shaiful Chowdhury, Stephanie Borle, Stephen Romansky, and Abram Hindle. "Greenscaler: training software energy models with automatic test generation". In: *Empirical Software Engineering* 24 (2019), pp. 1649–1692.

[29] Andrey Ignatov, Radu Timofte, William Chou, Ke Wang, Max Wu, Tim Hartley, and Luc Van Gool. "AI benchmark: Running deep neural networks on android smartphones". In: *Proc. the European Conference on Computer Vision (ECCV) Workshops*. 2018.

[30] Andrey Ignatov, Radu Timofte, Andrei Kulik, Seungsoo Yang, Ke Wang, Felix Baum, Max Wu, Lirong Xu, and Luc Van Gool. "AI benchmark: All about deep learning on smartphones in 2019". In: *Proc. 2019 IEEE/CVF International Conference on Computer Vision Workshop (ICCVW)*. 2019, pp. 3617–3635.

[31] Kyeongryeol Bong, Sungpill Choi, Changhyeon Kim, and Hoi-Jun Yoo. "Low-power convolutional neural network processor for a face-recognition system". In: *IEEE Micro* 37.6 (2017), pp. 30–38.

[32] Jinsu Lee, Sanghoon Kang, Jinmook Lee, Dongjoo Shin, Donghyeon Han, and Hoi-Jun Yoo. "The hardware and algorithm co-design for energy-efficient DNN processor on edge/mobile devices". In: *IEEE Transactions on Circuits and Systems I: Regular Papers* 67.10 (2020), pp. 3458–3470.

[33] *Qualcomm Neural Processing SDK*. https://developer.qualcomm.com/qualcomm-robotics-rb5-kit/software-reference-manual/machine-learning/snpe. Accessed on May 2024.

[34] *HUAWEI HiAI Engine*. https://developer.huawei.com/consumer/en/doc/2020315. Accessed on May 2024.

[35] *NeuroPilot: MediaTek's Ecosystem for AI Development*. https://neuropilot.mediatek.com/. Accessed on May 2024.

[36] *Samsung Neural SDK*. https://developer.samsung.com/neural/overview.html. Accessed on May 2024.

[37] Chunjie Luo, Xiwen He, Jianfeng Zhan, Lei Wang, Wanling Gao, and Jiahui Dai. "Comparison and benchmarking of AI models and frameworks on mobile devices". In: *arXiv preprint arXiv:2005.05085* (2020).

[38] Abram Hindle, Alex Wilson, Kent Rasmussen, E Jed Barlow, Joshua Charles Campbell, and Stephen Romansky. "Greenminer: A hardware based mining software repositories software energy consumption framework". In: *Proc. the 11th ACM Working Conference on Mining Software Repositories*. 2014, pp. 12–21.

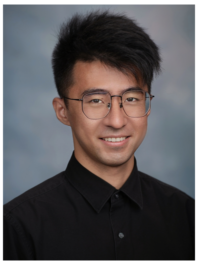
[39] Tedis Agolli, Lori Pollock, and James Clause. "Investigating decreasing energy usage in mobile apps via indistinguishable color changes". In: *Proc. IEEE/ACM 4th International Conference on Mobile Software Engineering and Systems (MOBILESoft)*. 2017, pp. 30–34.

[40] Ning Ding and Y Charlie Hu. "GfxDoctor: A holistic graphics energy profiler for mobile devices". In: *Proc. the Twelfth European Conference on Computer Systems*. 2017, pp. 359–373.

AUTHORS

Xiaolong Tu received a B.S. degree in communication engineering from Hubei University, Wuhan, China. He is currently pursuing a Ph.D. with the Department of Computer Science at Georgia State University. His research interests include mobile computing, energy-efficient deep learning, and on-device learning. Before starting his Ph.D., Xiaolong worked in the industry for over 10 years at Ericsson, Huawei, Qualcomm, and Apple. He is a student member of IEEE.

Anik Mallik is an assistant professor in the Department of Computer and Information Sciences at Towson University. He received his Ph.D. degree in electrical and computer engineering from the University of North Carolina at Charlotte, where he researched augmented reality and AI applications in mobile devices using edge computing. His current work involves developing energy-efficient and resilient mobility systems for AI applications. Before starting his Ph.D., Anik worked in the telecommunications industry for five years, gaining practical experience in the design and deployment of wireless networks and their energy infrastructure. During this time, he also developed an interest in emerging technologies such as edge computing and 5G networks, which led him to pursue further research in these areas. Anik received his Bachelor of Science degree in electronics and communication engineering from Khulna University of Engineering and Technology in Bangladesh. As an undergraduate student, he participated in several research projects on wireless communications, metamaterials, antennas, and automated systems. He is a member of the IEEE and ACM.



Haoxin Wang received a Ph.D. degree in electrical and computer engineering from The University of North Carolina at Charlotte in 2020, and a B.S. degree in control science and engineering from Harbin Institute of Technology in China in 2015. From 2020 to 2022, he was a research scientist at Toyota Motor North America, InfoTech Labs. He is currently an assistant professor in the Department of Computer Science at Georgia State University, and leads the Advanced Mobility & Augmented Intelligence (AMAI) Lab. His current research interests include sustainable edge AI, mobile AR/VR, and digital twins.

Jiang Xie received a B.E. degree from Tsinghua University, Beijing, China, an MPhil degree from the Hong Kong University of Science and Technology, and M.S. and PhD degrees from Georgia Institute of Technology, all in electrical and computer engineering. She joined the Department of Electrical and Computer Engineering at the University of North Carolina at Charlotte (UNC Charlotte) as an assistant professor in August 2004, where she is currently a full professor. Her current research interests include resource and mobility management in wireless networks, mobile computing, Internet of Things, and cloud/edge computing. She is on the editorial boards of the IEEE Transactions on Wireless Communications, IEEE Transactions on Sustainable Computing, and Journal of Network and Computer Applications (Elsevier). She received the US National Science Foundation (NSF) Faculty Early Career Development (CAREER) Award in 2010, a Best Paper Award from IEEE Global Communications Conference (Globecom 2017), a Best Paper Award from IEEE/WIC/ACM International Conference on Intelligent Agent Technology (IAT 2010), and a Graduate Teaching Excellence Award from the College of Engineering at UNC Charlotte in 2007. She is a fellow of the IEEE and a senior member of the ACM.