ARTIFACT EVALUATED usenix ASSOCIATION AVAILABLE

More is Different: Prototyping and Analyzing a New Form of Edge Server with Massive Mobile SoCs

Li Zhang[1], Zhe Fu[2], Boqing Shi[1], Xiang Li[1], Rujin Lai[3], Chenyang Yang[3]
Ao Zhou[1], Xiao Ma[1], Shangguang Wang[1], Mengwei Xu[1]
[1]*Beijing University of Posts and Telecommunications*
[2]*Tsinghua University,* [3]*vclusters*

Abstract

Huge energy consumption poses a significant challenge for edge clouds. In response to this, we introduce a new type of edge server, namely SoC Cluster, that orchestrates multiple low-power mobile system-on-chips (SoCs) through an on-chip network. For the first time, we have developed a concrete SoC Cluster consisting of 60 Qualcomm Snapdragon 865 SoCs housed in a 2U rack, which has been successfully commercialized and extensively deployed in edge clouds. Cloud gaming emerges as the principal workload on these deployed SoC Clusters, owing to the compatibility between mobile SoCs and native mobile games.

In this study, we aim to demystify whether the SoC Cluster can efficiently serve more generalized, typical edge workloads. Therefore, we developed a benchmark suite that employs state-of-the-art libraries for two critical edge workloads, i.e., video transcoding and deep learning inference. This suite evaluates throughput, latency, power consumption, and other application-specific metrics like video quality. Following this, we conducted a thorough measurement study and directly compared the SoC Cluster with traditional edge servers, with regards to electricity usage and monetary cost. Our results quantitatively reveal when and for which applications mobile SoCs exhibit higher energy efficiency than traditional servers, as well as their ability to proportionally scale power consumption with fluctuating incoming loads. These outcomes provide insightful implications and offer valuable direction for further refinement of the SoC Cluster to facilitate its deployment across wider edge scenarios.

1 Introduction

Energy efficiency has become a crucial factor in the design and operation of data centers [43]. According to reports [4], EU data centers consumed 76.8 TWh of electricity in 2018, which accounted for 2.7% of the total electricity usage within the EU. This power consumption is estimated to increase to 98.52 TWh by 2030. Such high energy usage also strains the cooling infrastructure and leads to significant costs for data center operators [61, 69].

Edge clouds, which provide computing resources in close proximity to users and devices, are becoming integral to daily life. Their increasing deployment aims to reduce latency and enhance the performance of applications. Nevertheless, energy issues are likely to intensify at the edge for several reasons. First, power supplies to the edge are more limited and expensive, since edge servers are often near populated areas, unlike cloud data centers that may be strategically near abundant energy sources like hydro power. Additionally, edge servers face spatial limitations and have a power density that is an order of magnitude higher than cloud servers. This puts additional strain on cooling mechanisms [69]. Moreover, edge workloads fluctuate more than cloud workloads due to the distinct features of edge applications [85]. One critical pathway towards sustainable data centers lies in enhancing the energy efficiency of individual servers. Past efforts mainly focused on software optimizations [60, 65, 67, 74, 88], but hardware redesign may offer greater benefits [75]. However, implementing this can be challenging with established cloud infrastructure. Fortunately, the emerging edge infrastructure presents a timely opportunity – it is still in the final stage of development, and we are on the eve of its inauguration [29, 78].

In this study, we advocate for a new form of edge server, referred to as SoC Cluster, which comprises tens or hundreds of mobile SoCs, as an enhancement to existing edge infrastructure. Mobile SoCs inherently possess higher energy efficiency than traditional datacenter servers, as they are designed for battery operated mobile devices with intermittent usage patterns [70, 71]. The SoC Cluster offers additional benefits, such as the ability to seamlessly run mobile operating systems and applications. This benefit facilitates computation offloading, particularly in cloud gaming scenarios [12, 14, 31] for native mobile games. Furthermore, mobile SoCs are inherently heterogenous. With an optimized software stack, they can efficiently accommodate both general-purpose workloads (e.g., web services), and domain-specific tasks (e.g., deep learning [27, 56] and multimedia processing [20]) by utilizing hardware accelerators such as GPUs and NPUs.

Breaking down a monolithic server into numerous smaller SoCs also caters to the fine-grained resource allocation required by cloud and edge applications. Figure 1 illustrates the resource subscription of 2.7 million VMs from Microsoft Azure [46] and 7,410 VMs from Alibaba ENS [85]. It reveals that most VMs on Azure/Alibaba datacenters require low resources, with up to 66%/36% of VMs having subscription configurations that fit within the hardware limits of a single mobile SoC (8 CPU cores, 12 GB memory, 256 GB storage). The proportion is anticipated to grow in the future as the hardware improvement of mobile SoCs and the ongoing trend of software services shifting towards disaggregation [58].

Hardware prototyping. We have developed a concrete SoC Cluster server, which integrates 60 Qualcomm Snapdragon 865 SoCs into a 2U rack, with detailed specifications described in §2.2. Over the past two years, we have manufactured more than 10,000 such SoC Clusters, most of which have been deployed in edge clouds, primarily to serve cloud gaming workloads. However, according to monitored traces (§2.3), the utilization of the deployed SoC Clusters varies widely and is generally low. Apparently the potential of these SoC Clusters has not been fully realized. To fill the gap, the first and critical step is to demystify whether SoC Clusters can efficiently serve other edge applications.

Measurement methodology. Therefore, we present a first-of-its-kind measurement study to quantitatively assess how our commercial-off-the-shelf SoC Cluster can efficiently support typical edge workloads. This study focuses on two modern, computation-intensive workloads: video transcoding and deep learning (DL) serving. Video transcoding stands as the predominant workload at the edge [85], with our experiments pinpointing two primary scenarios: live streaming transcoding and archive transcoding. DL serving forms the essential component of numerous intelligent applications. To provide a basis for comparison, we utilized a typical edge server with an Intel Xeon CPU and NVIDIA GPUs. We expanded the testing to six mobile phones with high-end Qualcomm SoCs to broaden the findings.

Benchmark suite. We developed a benchmark suite to test application performance on both the SoC Cluster and the traditional edge server. The benchmark suite employs state-of-the-art libraries for each application. For video transcoding, it uses FFmpeg [10] to process six videos with disparate characteristics from vbench [66]. For DL serving, it incorporates TFLite [27] (SoC Cluster), TVM [45] (Intel CPU), and TensorRT [21] (NVIDIA GPU). The NN models used are ResNet-50, ResNet-152 [52], YOLOv5x [57], and BERT [8]. We selected different software stacks for various hardware platforms as no single software optimally performs across heterogeneous processors. The benchmark suite reports comprehensive metrics, including throughput, latency, and energy consumption under constraints like electricity and cost.

Key findings are summarized below.

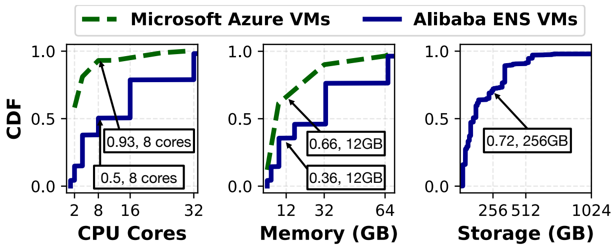


Figure 1: CDF of resource subscription of VMs in Microsoft Azure [46] and Alibaba ENS [85]. Approximately 66% of Azure VMs and 36% of Alibaba ENS VMs can be accommodated within a mobile SoC evaluated in this study (i.e., a Qualcomm Snapdragon 865 chip with 8 CPU cores, 12 GB memory and 256 GB storage).

(1) Energy efficiency. The SoC Cluster demonstrates up to 6.5× higher throughput per unit of energy for serving DL inference workloads compared to the traditional edge server equipped with NVIDIA A40 GPUs. Its energy efficiency is also comparable to high-end NVIDIA A100 GPUs. However, for complex video transcoding workloads, SoC CPUs inside the SoC Cluster underperform to NVIDIA GPUs that are optimized for highly parallel tasks. Additionally, attributing to the discrete SoC organization, the SoC Cluster can proportionally scale its energy consumption with dynamic loads, ensuring minimal degradation in energy efficiency.

(2) Latency. Due to a lack of proper software support for cross-SoC collaboration, the SoC Cluster faces challenges in handling delay-critical workloads. While the inference latency of a single SoC for a medium-sized DNN model is low enough to meet the requirements of most edge applications (8.8 ms on quantized ResNet-50), the latency can be up to hundreds of milliseconds for larger models. Therefore, there is an urgent need for a DL library to enable collaborative inference on multiple SoCs.

(3) Monetary cost. The SoC Cluster offers more than 2.23× greater throughput per monetary cost compared to the traditional edge server for live streaming transcoding. On the other hand, NVIDIA GPUs significantly outperform the SoC Cluster in DL serving. While this outcome might deter investment in new SoC Clusters for DL serving workloads, migrating lightweight or latency-insensitive DL tasks to the already deployed, underutilized SoC Clusters can still enhance energy efficiency.

(4) SoC longitudinal study. Through a longitudinal study, we found that mobile SoCs have demonstrated remarkable performance enhancements over the past six years, with a highest improvement of 8.5× on SoC DSPs. Improvements in mobile co-processors position them as suitable candidates for handling more complex server-side workloads in the future. Regarding the existing manufactured SoC Clusters, optimizing the current software stack is essential to fully utilize the capabilities of mobile co-processors.

Contributions. We made the following contributions.

- We discussed the rationales and potential benefits of organizing mobile SoCs as edge servers, and presented the hardware prototyping and its commercial deployment by edge service providers.
- We designed and implemented a benchmark suite for two typical edge applications to evaluate their performance and power consumption on both the prototyped SoC Cluster server and a traditional edge server.
- We conducted a comprehensive measurement study based on the benchmark suite, and highlighted both the advantages and disadvantages of the SoC Cluster.
- We carried out a longitudinal study on various mobile SoCs to reveal their performance enhancements over time, and the potential to serve complex workloads with their co-processors.

2 Design and Prototyping

2.1 Motivations

The current edge server architecture derives from the legacy of cloud computing that has been entrenched for decades. Typically, an edge server consists of a many-core CPU along with a set of domain-specific accelerators (GPUs, TPUs, FPGAs, etc.). Major edge resource providers worldwide, including Azure, AWS, and Alibaba, adhere to this design philosophy [7, 16, 30]. However, the edge environment presents two distinctive characteristics that set it apart from the cloud: (1) *limited electricity and space availability*, resulting from the necessity to position edge servers close to populated areas, where the cost of electricity is high and physical spaces are confined; (2) *dynamic workloads* of user-oriented applications that challenge the energy efficiency of edge sites. Although various software-level optimizations have been proposed [60, 67, 76], we argue that it is time to reevaluate the architecture of edge servers. In the meanwhile, the growing need to support mobile ecosystems in the cloud (e.g., native mobile games [34] and virtual smartphone [36]) facilitates the adoption of mobile SoC-based servers. In this work, we explore the feasibility and potential benefits of organizing multiple low-end SoCs as an edge server.

There exists some research on SoC servers. Rajovic *et al.* conducted an early analysis of utilizing mobile SoCs for high performance computing [72]. Some studies focused on repurposing decommissioned mobile devices to reduce e-waste [77, 80]. Others explored the use of SoCs for specific applications, such as parallel computing [44], key-value storage [41], web search [55], and video transcoding [64]. A similar vision to ours was presented in [86], but with limited experiments and system design details. Those efforts typically include only theoretical analysis, or they conducted experiments on small-scale implementations involving 2–10 smartphones or Raspberry Pis, or evaluated simplified workloads [59]. Instead, our work aims to materialize the “SoC-as-a-server” concept in a more realistic and industry-relevant context.

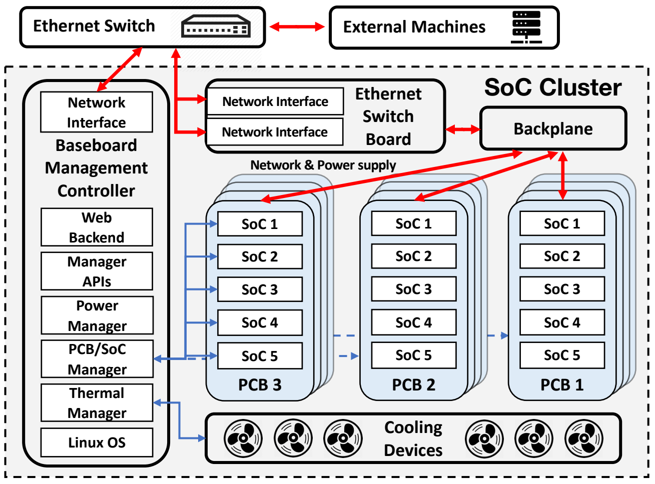


Figure 2: The architecture of SoC Cluster.

2.2 Implementing SoC Cluster

The design space for materializing the concept of an SoC Cluster into a server machine is expansive. In this study, we design and implement a concrete SoC Cluster server that ensures maximal flexibility of each hardware component. For simplicity, we refer to this concrete server as SoC Cluster.
Overall architecture. Figure 2 illustrates the overall architecture of SoC Cluster. The major component of the server is a pool of mobile SoCs, grouped into sets of five and integrated into individual printed circuit boards (PCBs). These PCBs provide dedicated power supplies and network capabilities, and they act as network switches for interconnecting the SoCs. Additionally, an Ethernet Switch Board (ESB) is incorporated into SoC Cluster to connect all SoCs to the external network. The PCBs and ESB are interconnected through a backplane. SoC Cluster also includes a Baseboard Management Controller (BMC) to monitor and manage server status, including power, temperature, and cooling devices. In summary, the architecture of SoC Cluster is modularized, offering great flexibility in the design, manufacture, and upgrading.
Hardware components and functionalities. Figure 3 provides an overview of how the hardware components inside SoC Cluster are managed and connected. Basically communication between two hardware components can be through hardware control messages or network-layer messages. Here, we categorize the hardware components according to their functionalities.

• *Computing.* SoC Cluster’s computing units consist of 60 Qualcomm Snapdragon 865 SoCs [5]. The number of integrated SoCs inside a SoC Cluster is mainly determined

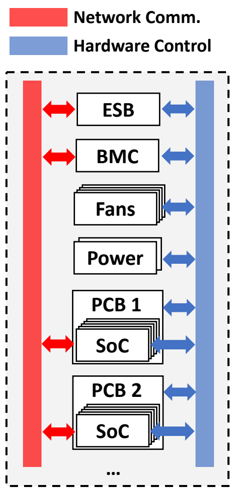


Figure 3: Hardware management in SoC Cluster.

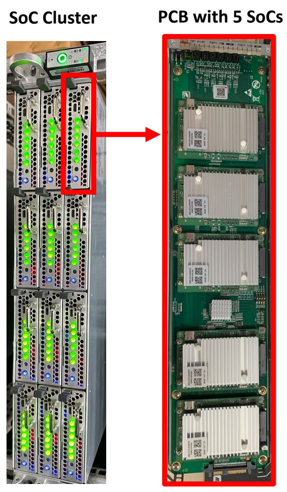


Figure 4: A manufactured SoC Cluster and a PCB.

by the server's physical size and cooling capability. Table 1 summarizes the detailed hardware and OS specifications of each SoC and the entire server.

• *Networking.* The network functionality of SoC Cluster is primarily provided by two core hardware components. The first is the ESB, which exposes the computing units to the external world through its dual SFP+ ports. It supports up to 20 Gbps throughput. The second component consists of 12 PCBs with network switching functionality. The Ethernet connection between the ESB and SoCs is relayed through the corresponding PCBs. When a PCB is plugged into SoC Cluster, it establishes a physical connection with the ESB and builds an Ethernet connection with up to 1 Gbps throughput.

• *Management.* SoC Cluster uses an independent hardware component, BMC, to monitor and control the computing units and all related server status, such as power supplies, temperature, and hardware failures. The hardware control messages are transmitted through a mixture of protocols and technologies, including I2C, USB and UART. The BMC also provides an Ethernet interface that enables external machines to access it.

• *Cooling and power supplies.* To ensure proper cooling, SoC Cluster employs eight fans that circulate air through the mobile SoCs, the ESB, and the BMC, and then expel it from the fan module at the back of the server. The server utilizes two power modules to provide redundant power supplies, with a maximum support of approximately 700 watts.

2.3 Micro-experiments and Trace Analysis

Figure 4 shows the manufactured SoC Cluster evaluated in this study, which has been on sale for over a year. As of August 2023, more than 10,000 SoC Clusters have been shipped and deployed in the wild, primarily to edge service providers. In collaboration with a leading worldwide edge service provider,

Hardware	SoC Cluster		Traditional Edge Server
	Individual SoC	Whole Server	
CPU	Qualcomm Kryo 585	60x Qualcomm Kryo 585	Intel Xeon Gold 5218R Processor
GPU	Qualcomm Adreno 650	60x Qualcomm Adreno 650	8x NVIDIA A40 PCIe 48GB
Memory	12GB LPDDR5	720GB LPDDR5	768GB DDR4
Disk / Flash	256GB Flash	15.36TB Flash	1.92TB SSD, 30TB HDD
OS	Android 10	-	Ubuntu 18.04 LTS
Network	Integrated 1GE Ethernet	2x 10GE SFP+ Port	2x 1GE RJ45 Port, 2x 10GE RJ45 Port
Form Factor	-	2 Rack Units	4 Rack Units

Table 1: Two major hardware platforms used in this study.

Micro Benchmarks	Per-core Performance				Whole Server Performance			
	Ours	Trad.	G2	G3	Ours	Trad.	G2	G3
CPU Score	911	840	762	1,121	194,100	15,450	36,091	51,379
Integer Score	842	800	735	1,039	184,500	16,224	36,653	50,695
Floating Score	948	886	790	1,214	191,820	15,793	35,813	49,885
Text Compress	4.4	4.1	4.2	4.9	906	135	195	206
SQLite Query	257	249	208	279	59,958	9,240	12,200	16,200
PDF Render	52	41	37	66	12,552	710	2,140	3,960

Table 2: A list of micro-benchmark (from Geekbench 5 [11]) results on SoC Cluster and typical edge servers. "Ours": SoC Cluster; "Trad.": the traditional edge server; "G2/3": AWS Graviton 2/3 cloud instances (m6g.metal/m7g.metal; both with 64 cores, 256 GB RAM);

we discovered that *the deployed SoC Clusters are primarily used to serve a specific application: cloud gaming*. Native mobile games, such as Genshin Impact [13], usually release installation packages tailored for mobile platforms (e.g., for the arm64-v8a architecture). While there exists mobile-in-cloud solutions designed for traditional servers with many-core CPUs and monolithic GPUs [25, 33], these games can hardly run as smoothly as they do on mobile SoCs due to incompatibility with the CPU architecture and graphics stacks. Recognized as the optimal hardware setup for native mobile games, the shipped SoC Clusters are estimated to serve millions of game sessions on a daily basis.

Micro-benchmarks on CPU. To initiate an understanding of SoC Cluster's performance, we employed a cross-platform benchmark, Geekbench 5 [11], to run a series of CPU-only micro-benchmarks on four different servers, including SoC Cluster and the traditional edge server as shown in Table 1, and AWS Graviton 2/3 servers with ARM CPUs designed for cloud environments. The results in Table 2 provide the following observations. First, the per-core performance of SoC Cluster aligns closely with that of the Intel Xeon CPU, outperforming the AWS Graviton 2 processor. Second, from a whole server's perspective, the large number of integrated SoCs grants SoC Cluster superior performance compared to other CPU servers. For example, it exhibits 3.8× higher CPU core score and 3.2× faster PDF rendering speed relative to the latest AWS Graviton 3 cloud instance. These results demonstrate the significant potential of SoC Cluster's applicability for general edge workloads.

Network performance. We used ping and iPerf3 to assess network round-trip time (RTT) and TCP/UDP bandwidth between two individual SoCs. The results show that the network is stable, with an RTT of approximately 0.44 ms and TCP and UDP bandwidths of nearly 903 Mbps and 895 Mbps, respectively. However, these results do not truly reflect the performance of the networking system, as they were not measured on real edge applications and did not involve multiple SoCs simultaneously. In subsequent sections, we further investigate the network performance (live streaming transcoding in §4.4 and cross-SoC DL serving in §5.3).

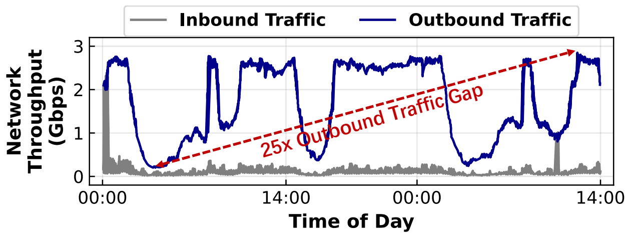


Figure 5: The network throughput of an in-the-wild SoC Cluster that serves cloud gaming workloads over 38 hours. The server is randomly picked from an edge site of one edge service provider. Full network capacity: 20 Gbps.

Trace analysis. Although SoC Cluster exhibits superior performance at the whole server level, our analysis of runtime traces collected from deployed SoC Clusters reveals that *these servers experience workloads with relatively low utilization and high dynamics.* In general, the resource usage of all deployed SoC Clusters remains below 20%. Figure 5 displays the temporal network traffic of a specific SoC Cluster randomly selected from real-world edge sites. As observed, the disparity between its highest and lowest outbound traffic reaches up to 25×. This observation aligns with a recent empirical study on large-scale edge clouds [85], which discovered that edge applications are mainly user-centric, therefore highly dependent on user activities. Motivated by this insight, this study seeks to explore how efficiently SoC Cluster can serve other typical edge workloads beyond cloud gaming.

3 Methodology and Benchmark

In this work, we perform an application-driven measurement study to demystify the performance of SoC Cluster and traditional edge servers. This section elaborates on the methodology used to set up the experiments, with best possible effort to ensure a fair comparison.

Applications. We select two modern, computation-intensive applications: (1) *Video transcoding* [83], which involves converting the format (FPS, resolution, etc.) of a given video stream, is widely adopted in online conferences and live streaming. It is reported to be a dominant workload at the edge [62, 85]. We identify two specific scenarios for this workload: live streaming transcoding and archive transcoding, with their characteristics illustrated in §4. (2) *DL serving* is a key component in intelligent applications like AR, VR, and autonomous driving [22, 26, 28]. A DL serving system receives a stream of input data and executes it using a designated DL model. Substantial academic effort has been directed at optimizing DL serving performance [47, 50, 68, 89]. In summary, the two applications cover the interests of both industry and academia, representing the major workloads at the edge. They are both resource-intensive, indicating that if SoC Cluster performs well on them, it has the potential to efficiently serve other applications as well.

Metrics are divided into two categories: application performance metrics and comparison metrics. The former are mainly about throughput (the number of processed samples or videos per second), latency (model inference time), and power consumption. To ensure a relatively fair comparison between SoC Cluster and the traditional edge server, we further judiciously select two conditional constraints:

• *Energy.* As previously mentioned, energy consumption is a critical constraint for edge sites. Therefore, we use throughput per energy unit (e.g., Joule) to assess application energy efficiency. Besides energy efficiency under a specific (often full) load, we also consider energy proportionality under various load levels, acknowledging that edge servers experience high load variations. An ideal edge server should scale its power consumption proportionally with the load to minimize wasted energy [43].

• *Monetary cost.* We conduct a total cost of ownership (TCO) analysis to provide insight into the relationship between application performance and monthly TCO. This analysis is designed to assist edge operators in making well-informed purchasing and scheduling decisions.

Hardware. The SoC Cluster used in measurements was introduced in §2.2. For comparison, we use a traditional server equipped with an Intel Xeon Gold CPU (4.0 GHz and 40 physical cores), 8 NVIDIA A40 GPUs, and 768 GB DRAM, running Ubuntu 18.04 LTS; its specifications are summarized in Table 1. We confirmed that this type of server is widely used at the edge sites where SoC Clusters are deployed. In our DL serving experiments, we opt for a higher-end NVIDIA A100 GPU from the Google Cloud Platform for a more comprehensive comparison. The decision was made since NVIDIA A40 is not the highest-end server-level GPU released in 2020, the same year as the Qualcomm Snapdragon 865 SoC. We exclude the NVIDIA A100 GPU in video transcoding experiments due to its lack of support for NVENC video encoding as of May 2024 [40].

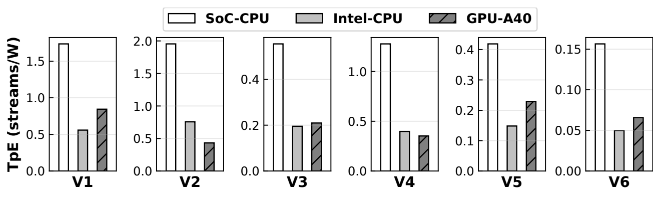
Benchmark suite. One challenge of this study is selecting the appropriate software stack. Given the inherently heterogeneous hardware architectures, we found that there is rarely software compatible with each processor type (i.e., SoC CPU/GPU/DSP, Intel CPU, and NVIDIA GPU). Even if such software exists, its performance could be far from state-of-the-art standards. To obtain meaningful results, we consulted with our industry partners and conducted testing with commonly used software. We then selected best-performing option for each application and hardware configuration.

- *Video transcoding.* We use FFmpeg (v4.4) [10] with libx264 [32] and NVDEC/NVENC [23] support. We cross-compiled FFmpeg to SoC Cluster with ARMv8 NEON acceleration. We employ a popular open-source Android library, LiTr [18], for hardware-accelerated transcoding on SoC Cluster, due to FFmpeg's limited support for Qualcomm SoCs. The benchmark suite is built atop vbench [66], a benchmark tool widely used for cloud video transcoding.
- *DL serving.* We employ TFLite [27] for SoC Cluster, TVM [45] for the Intel CPU, and TensorRT [21] for the NVIDIA GPU. We select one medium-sized DNN (ResNet-50 [52]) and three large DNNs (ResNet-152 [52], YOLOv5x [57], and BERT base [8]), all of which are representative DL serving workloads.

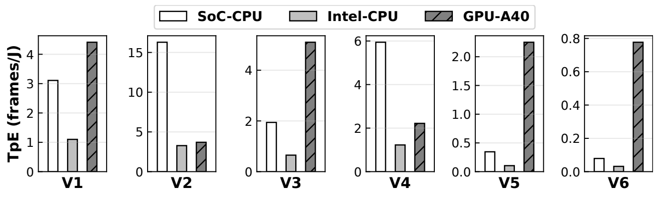
Setups. For experiments on the Intel CPU, we partition the 80 cores (80 hardware threads) into 10 separate 8-core Docker containers. We select 8 cores for two reasons. First, previous edge workloads [85] have shown that 8 is the median number of vCPU cores for edge IaaS VMs, which is also adequate for most edge services. Second, the SoC's CPU also contains 8 cores, making the comparison more direct. We leverage the `turbostat` command to read CPU/RAM power and the `nvidia-smi` command to access GPU power on the traditional edge server. On SoC Cluster, we utilize BMC's API (implemented atop the I2C protocol) to measure power consumption of the whole server. Our report on workload power consumption excludes idle power. By default, our experiments are conducted with hardware fully loaded by batching DL serving or starting multiple transcoding processes. We vary the load levels only when testing the energy proportionality. Furthermore, to reduce power fluctuations during experiments: (1) in live streaming transcoding, we start the maximum number of streams supported by each hardware, ensuring that no stream's performance (FPS) fell below that of the origin video stream; (2) in archive transcoding, we repeat the same transcoding process on a video ten times; (3) in DL serving, we set the DL inference iteration to 1,000 for each test.

4 Video Transcoding Results

In this section, we compare video transcoding performance in two scenarios with distinct characteristics: live streaming transcoding and archive transcoding. The former usually takes video streams with a constant frame rate, widely used in live streaming and online conferences. The latter processes video clips from file storage, typically serving as a preliminary stage before distribution to content providers [24, 66]. Following the prior video benchmark [66], we maintain a constant target bitrate for live streaming transcoding and consistent video quality for archive transcoding across hardware. The metadata for the videos is shown in Table 3.



(a) Live streaming transcoding



(b) Archive transcoding

Figure 6: Transcoding energy efficiency.

4.1 Energy Efficiency

For live streaming transcoding, energy efficiency is measured as the number of video streams supported per watt (streams/W). For archive video transcoding, energy efficiency is measured as how many frames can be processed per Joule (frames/J). We unify the above metrics as throughput per energy unit (TpE). Figure 6a and 6b present the results for live streaming transcoding and archive transcoding, respectively. Our key observation is that *SoC Cluster shows significant energy savings across all live streaming and partial archive transcoding tasks over the Intel CPU and the NVIDIA GPU.*

In live streaming transcoding, SoC Cluster's SoC CPUs are 2.58×–3.21× more energy-efficient than the Intel CPU, and 1.83×–4.53× more energy-efficient than the NVIDIA A40 GPU across different videos. In archive transcoding, SoC CPUs consistently outperform the Intel CPU in energy efficiency, although their advantage over the NVIDIA GPU varies by video. Specifically, the NVIDIA GPU performs worse on videos V2 and V4. We find the common feature of V2 and V4 is that they have low entropy due to minimal motion or rare scene transitions. In such cases, we observe that the NVIDIA GPU stays in a high-power mode with high clock frequencies, while SoC CPUs accomplish these tasks using minimal CPU resources. The similar observation can be proven in live streaming transcoding tasks – SoC Cluster demonstrates superior energy efficiency on videos with lower complexity.

We further explore how energy efficiency scales with dynamic workloads at the edge. To conduct this analysis, we manually adjust the number of simultaneous transcoded video

Video	Video Metadata					Network Bound Analysis of Live Streaming Transcoding		
	Resolution	FPS	Source Entropy	Source Bitrate	Target Bitrate	Max. Stream Num (per SoC)	Max. Network Usage (per PCB, 1 Gbps)	Max. Network Usage (whole server, 20 Gbps)
V1: holi	854x480	30	7.0	2.8 Mbps	819.8 Kbps	13 (CPU) / 16 (HW)	534 Mbps (53.4%)	6,407 Mbps (32.0%)
V2: desktop	1280x720	30	0.2	181 Kbps	90.5 Kbps	15 (CPU) / 16 (HW)	43 Mbps (4.3%)	505 Mbps (2.5%)
V3: game3	1280x720	59	6.1	5.6 Mbps	2.7 Mbps	4 (CPU) / 12 (HW)	673 Mbps (67.3%)	8,072 Mbps (40.3%)
V4: presentation	1920x1080	25	0.2	430 Kbps	215 Kbps	9 (CPU) / 16 (HW)	81 Mbps (8.1%)	968 Mbps (4.8%)
V5: hall	1920x1080	29	7.7	16 Mbps	4.1 Mbps	3 (CPU) / 7 (HW)	1,008 Mbps (100.8%)	12,010 Mbps (60.5%)
V6: chicken	3840x2160	30	5.9	49 Mbps	16.6 Mbps	1 (CPU) / 2 (HW)	985 Mbps (98.5%)	11,821 Mbps (59.1%)

Table 3: The metadata of videos in the transcoding experiments and the network bound analysis of live streaming transcoding on SoC Cluster. The videos are picked from vbench [66] to ensure diverse coverage of resolution, FPS, and entropy. Entropy is calculated by bits per pixel per second and thus relating to the scene complexity. The network usages include both inbound and outbound traffic. CPU and HW represent transcoding on SoC CPU and SoC hardware codec.

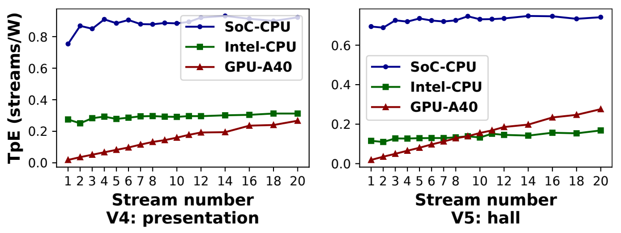
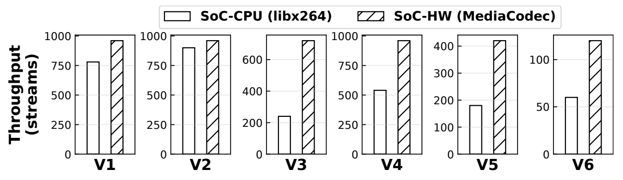


Figure 7: Energy efficiency of live streaming transcoding with different numbers of live video streams being processed simultaneously.

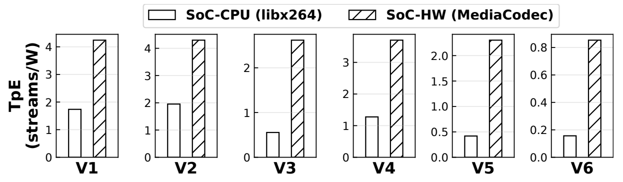
streams from 1 to 20 on all hardware. We use two 1080p videos with diverse scene complexity and present the relationship between throughput per watt and the number of processed video streams. Figure 7 shows a consistent trend for both videos: both SoC CPUs and the Intel CPU display nearly constant energy efficiency as the number of video streams rises, implying a linear increase in power consumption with increasing workloads. As for the NVIDIA GPU, it can only process 0.018 live video streams per watt when transcoding a single video (V4), falling 14.9× behind the Intel CPU and 40.8× behind SoC CPUs. As the number of video streams increases, energy efficiency of the NVIDIA GPU gradually increases but is still lower than that of SoC CPUs. The fine-grained control over SoC CPU cores or SoCs offers better energy scalability for serving dynamic video transcoding workloads on SoC Cluster.

4.2 Hardware-accelerated Video Transcoding on SoCs

Mobile SoCs are equipped with hardware codecs to handle most of the video encoding/decoding tasks on smartphones. Our tests focus on live streaming transcoding because Android MediaCodec's APIs [20] lack the controls for video quality, which is essential for a fair comparison in archive transcoding. We used LiTr [18] for hardware-accelerated video transcoding instead, given that FFmpeg only supports hardware decoding but not encoding on the Android platform [15]. Below, we compare the performance and transcoding behavior of the hardware codec against SoC CPU, focusing on key metrics such as server-side transcoding throughput,



(a) Transcoding throughput of the whole SoC Cluster.



(b) Energy efficiency.

Figure 8: Live streaming transcoding performance of SoC CPU and hardware codec in SoC Cluster.

energy efficiency, output bitrate, and video quality.

Our key observation is that *the hardware codec of the mobile SoC exhibits a significant improvement over its CPU.* As shown in Figure 8a, employing the hardware codec increases the maximum number of supported live video streams by 1.07×–3×. The increase in energy efficiency is even more impressive, as illustrated in Figure 8b. For videos with low-complexity scenes (V1, V2, and V4), the hardware transcoders in SoC Cluster can support a geometric mean of 2.5× more streams per watt compared to SoC CPUs. When transcoding high-entropy and high-resolution videos (i.e., V3, V5, and V6), SoC CPUs consume more power for video encoding, while offloading transcoding workloads to hardware codecs results in a significant boost in energy efficiency, with improvements ranging from 4.7× to 5.5×.

To further clarify the ability of hardware codecs in SoC Cluster to handle live streaming transcoding workloads, we evaluate the output video bitrate, which is one of the most critical metrics affecting user experience. As depicted in Figure 9, we use red dash lines to indicate the target bitrate for each transcoding task. The primary finding reveals that, *in most cases, the hardware codec can meet the bitrate constraint, but it struggles to meet a relatively low bitrate cap.* For example, setting a target bitrate of 90.5 Kbps for V2 will make the encoder create a higher bitrate output (even higher than the

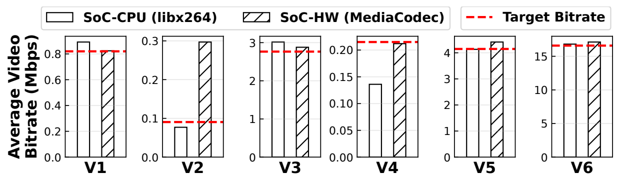


Figure 9: Target/output video bitrate in live streaming transcoding.

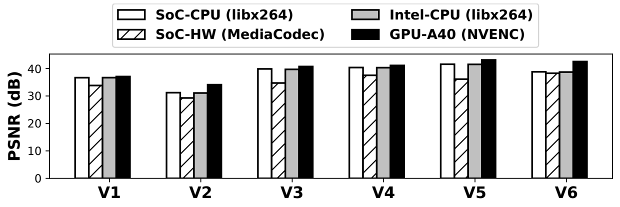


Figure 10: Live streaming transcoding quality of different encoders with the same bitrate constraint.

origin video stream). Such unexpected behavior runs counter to the typical objective of archive transcoding services to compress a video stream. The same behaviors were confirmed by our supplementary experiments on other videos and ultra-low bitrate settings. This may suggest a potential design trade-off made by the mobile SoC vendor with considerations for energy efficiency and chip size.

4.3 Transcoding Quality

Transcoding quality refers to the consumer perception of the output video. The live streaming transcoding experiments were conducted with a fixed target bitrate. However, due to the nuances at both the software (libx264 vs. MediaCodec vs. NVENC) and hardware (CPU vs. GPU vs. ASIC) layers, video quality could vary noticeably even under identical bitrate constraints. To evaluate this, we saved the output of live streaming transcoding to files from previous experiments and used Peak Signal-to-Noise (PSNR) [54] as the metric to represent video quality, where higher values indicate better quality.

As shown in Figure 10, *the software encoder running on SoC CPUs can maintain nearly equivalent video quality to those using the Intel CPU and the NVIDIA GPU, while videos generated by SoC Cluster's hardware codec have slightly poorer quality than others.* The general-purpose computing units (i.e., the SoC CPU and the Intel CPU), paired with the software encoder (i.e., libx264) using identical transcoding configurations, always generate videos with the same quality. In contrast, videos transcoded by MediaCodec exhibit about 1.35%–14.77% lower PSNR values compared to those generated by libx264 using SoC CPUs. This is caused by the less stringent quality and bitrate specifications of mobile encoders [75]. To further explore what bitrate should be set to attain comparable video quality using MediaCodec, we manually tuned the target bitrate and then compared the PSNR value with the original video. Despite these adjustments, videos generated using MediaCodec failed to match the video quality achieved by libx264. As such, if the quality loss incurred by MediaCodec is not bearable, it is better to choose SoC CPUs for video transcoding.

4.4 Network Bound Analysis

If the SoC Cluster is fully occupied with video transcoding workloads, will the network capacity become the bottleneck? Our mathematical analysis in Table 3 indicates that it will not. By multiplying the theoretical maximum number of live video streams supported by a single SoC (SoC CPU + SoC hardware codec) and the total network traffic per stream, we find that among the six videos tested, network usage will slightly exceed the PCB's 1 Gbps capacity only when transcoding V5 video. In practice, this is not feasible as software delegation daemon processes of SoC hardware codecs also consume some CPU resources. For the entire SoC Cluster, the ESB's 20 Gbps capacity will not become a bottleneck. However, with denser SoC integration in future SoC Cluster generations or more complex video transcoding workloads, the network could become a limiting factor and need enhancement.

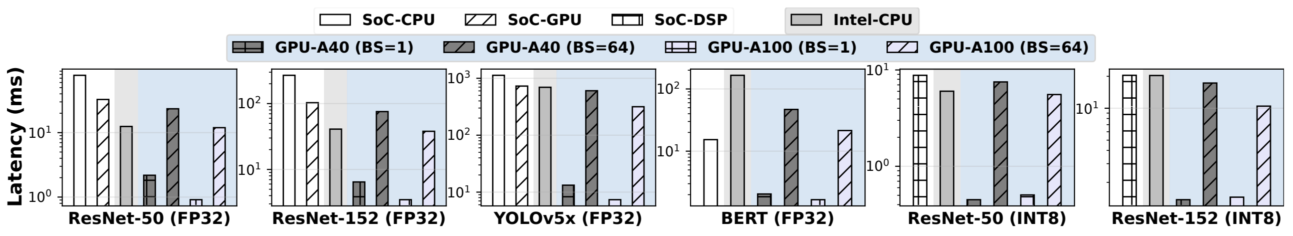
Summary. SoC CPUs in SoC Cluster provide up to 3.2× and 4.5× higher energy efficiency than the traditional Intel CPU and NVIDIA GPU, respectively. The fully-fledged software stack enables a seamless transition of current transcoding services to SoC Clusters. Although hardware codecs of SoCs provide even higher throughput and energy efficiency than SoC CPUs, their differeces in software-level encoding library implementation lead to inconsistent qualities and bitrates in output videos. Nonetheless, SoC CPUs demonstrate high and robust potential for general video transcoding services, and hardware codecs provide even higher profits for relatively narrower video transcoding scenarios.

5 Deep Learning Serving Results

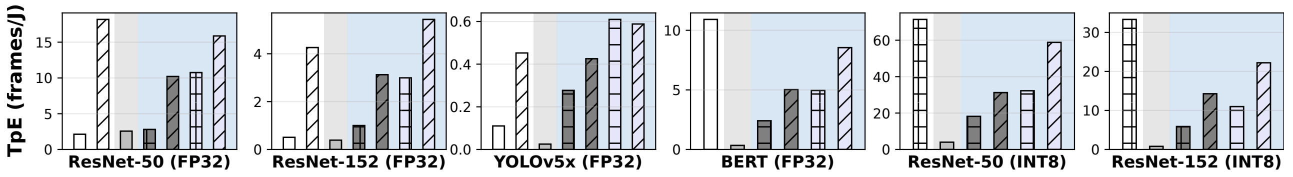
5.1 Inference Latency

Inference latency is a crucial factor in delay-sensitive workloads, directly impacting the user experience. In our study, we conducted tests on NVIDIA GPUs using varying batch sizes to strike a balance between inference latency and energy efficiency. We limited the batch size to 1 on other hardware, as this setup fully utilizes hardware resources; increasing the batch size further only resulted in higher latency while not improving energy efficiency. The results are shown in Figure 11a.

We have made the following observations. (1) The SoC GPUs in SoC Cluster exhibit 1.55×–2.61× lower latency



(a) Deep learning serving latency.



(b) Deep learning serving throughput per Joule.

Figure 11: Deep learning performance on SoC Cluster and the traditional edge server. Background colors are used to distinguish the performance bars/legends of different hardware: SoC Cluster (white), Intel CPU (grey), NVIDIA GPUs (light blue).

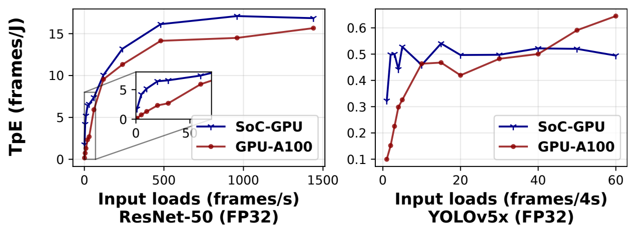


Figure 12: Energy efficiency of SoC Cluster and the traditional edge server under various DL input loads. SoC Cluster delivers higher throughput per energy with light workloads.

compared to SoC CPUs across tasks, and their performance are comparable to the Intel CPU with eight cores used in our experiments. (2) The latency demonstrated by the NVIDIA GPU with batch size 1 is significantly lower than other processors due to its high hardware-level parallelism and well-optimized software (i.e., TensorRT). However, when larger batch sizes are used, the latency on the NVIDIA GPU increases significantly and may even approach or exceed that of SoC Cluster, e.g., performing inference on YOLOv5x (FP32) and BERT (FP32) using the NVIDIA A40 GPU with a batch size of 64. (3) For medium-sized DNNs, such as ResNet-50, SoC GPUs (for FP32 format) or DSPs (for INT8 format) are typically capable of delivering satisfactory inference latency, i.e., 32.7 ms and 8.8 ms, respectively. In comparison, using the NVIDIA GPU for inference only results in marginal speed up (e.g., approximately 8 ms for a INT8-based ResNet-50 model), but comes at the cost of higher energy consumption, as we will discuss below. (4) For large DNNs like ResNet-152, considering both FP32 and INT8 formats, the inference latency of SoC Cluster ranges from 20.4 ms to 269 ms, which is unacceptable for real-time applications [3]. To address this issue, a potential solution could be a cooperative inference framework among multiple SoCs.

5.2 Energy Efficiency

We use throughput per energy unit as the metric to evaluate the energy efficiency of each hardware type, which is calculated as the number of samples processed per Joule. A server with higher energy efficiency can process more samples with the same amount of electricity. The results are presented in Figure 11b. Our key observation is that *SoC Cluster exhibits significantly higher energy efficiency compared to the Intel CPU and mid-end NVIDIA GPU, and is comparable to the high-end NVIDIA GPU.* Especially on ResNet-50 (FP32), SoC GPUs show the ability to process about 18 frames per second per Joule, which is 7.09× higher than the Intel CPU, 1.78× higher than the NVIDIA A40 (BS=64), and 1.15× higher than the NVIDIA A100 (BS=64). The energy efficiency advantage of SoC Cluster is even more significant in quantized models. Taking ResNet-152 with INT8 format as an example, the energy efficiency of SoC DSPs is 42× higher than that of the Intel CPU and 1.5× higher than that of the NVIDIA A100 (BS=64). This can be attributed to the fact that SoC DSPs are designed for low-power data processing, operating at frequencies of $\leq$500MHz.

The above energy efficiency values are primarily based on the measurements taken when the servers are fully loaded. We also measured energy efficiency under varying workloads, as discussed in §4.1. Figure 12 illustrates this analysis of the SoC GPU and the NVIDIA A100 GPU, chosen due to their high energy efficiency on SoC Clusters and traditional edge servers, respectively. For ResNet-50, SoC Cluster shows significant energy efficiency advantages when the workload is lightweight, e.g., 5.71× more energy-efficient than the NVIDIA A100 GPU on average with only five samples per second. This is primarily due to SoC Cluster provides fine-grained scheduling at the level of each SoC for efficient processing of incoming requests. When incoming data can be adequately processed by only a subset of SoCs, the remaining SoCs can be kept in a low-power state or even turned off. In comparison, datacenter-level GPUs have coarser granularity to scale their energy consumption with dynamic workloads.

5.3 SoC-collaborative DL Inference

In this section, we conducted a preliminary analysis on SoC-collaborative DL inference, which aims to mitigate high inference latency. We used the MNN [56] framework and the tensor parallelism algorithm proposed in [87]. Specifically, each participating SoC calculates 1/N of the entire tensor along the width dimension. Intermediate results are transmitted between SoCs via the TCP protocol.

Figure 13 shows inference latencies and their breakdown when using 1–5 SoCs. We observe that involving more SoCs does not proportionally reduce inference latencies. This may stem from imperfections in the software design that incur additional computation and high communication overhead. For example, on the ResNet-50 model, increasing the number of SoCs from one to five reduces the computation time from 80 ms to 34 ms (a 2.35× reduction), while the inference only achieves a 1.38× speedup. When using five SoCs, the data communication time contributes to 41.5% of the total inference latency, where both computation and communication time lengthen the overall latency. We then tried to optimize the data synchronization design between SoCs by transferring computation-required data first, aiming to pipeline computation and communication. Results show that the network communication time still accounts for 22.9% of the total latency with five SoCs involved, indicating network bandwidth could bottleneck the SoC collaboration process. Therefore, software optimizations (e.g., more fine-grained tensor partitioning) and hardware enhancements (e.g., increased network bandwidth) should be jointly utilized to improve performance, especially when a larger number of SoCs are involved.

Summary. SoC GPUs and DSPs in SoC Cluster demonstrate superior energy efficiency compared to traditional server-level CPUs (up to 42×) and GPUs (up to 6.5×). The inference latency of SoC Cluster on medium-sized DNNs, such as ResNet-50, is satisfactory for meeting the requirements of typical edge applications. However, more advanced software that can orchestrate multiple SoCs is urgently demanded to collaboratively serve large DNNs on SoC Clusters efficiently. SoC Cluster could also enhance its network bandwidth to reduce data communication time in cross-SoC DL inference.

6 Cost Analysis

Cost is another critical dimension to evaluate the suitability of new hardware. In this section, we conduct a TCO analysis on

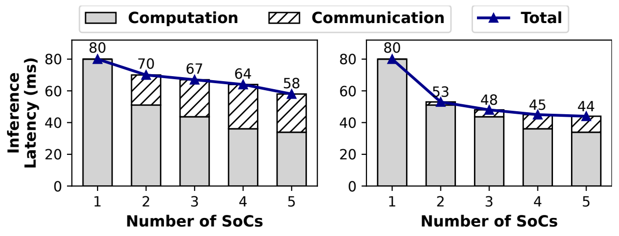


Figure 13: SoC-collaborative DL inference latency using different number of SoCs and breakdown of latencies. Left: Tensor parallelism; Right: Tensor parallelism with computation/communication pipelining.

SoC Cluster and traditional edge servers. The TCO consists of two parts: the capital expenditure needed to purchase the servers (CapEx, with a breakdown of each hardware component) and the operational expenditure (OpEx). We calculate the total CapEx using the retail purchase cost. For OpEx, we consider only the electricity cost as in prior work [59]. In addition to the traditional edge server that consists of an Intel CPU and 8 NVIDIA GPUs, we add a “virtual server” by excluding all 8 NVIDIA GPUs. We use throughput per cost (TpC) as the normalized performance metric. A server with a higher TpC value indicates its ability to process more workloads within the same monetary cost budget.

Capital expenditure. As shown in the top half part of Table 4, GPUs account for a substantial portion of the total CapEx in a traditional edge server. In a CPU-only server, the CapEx is amortized across different hardware components. Within SoC Cluster, 60 SoCs and 12 PCBs constitute almost 87% of the total CapEx. In summary, SoC Cluster has a lower CapEx than the traditional edge server with 8 NVIDIA GPUs but costs about 2.8× more than a CPU-only edge server.

Operational expenditure. We opted not to use complex OpEx models [77] as we observed that CapEx consistently dominated the TCO, as we describe later. This was also reported in a previous study by Google [42]. Therefore, we have chosen to report only the electricity cost in this analysis. The monthly electricity cost is calculated by multiplying the monthly power consumption (kWh) with the electricity unit cost ($/kWh). The monthly power consumption across all workloads correlates with their average power usage. For instance, performing live streaming transcoding when fully utilizing all 8 NVIDIA A40 GPUs has an average power consumption of 1,231 watts. Assuming the servers operate at their average peak power 50% of the time over a month, the monthly power consumption can be calculated as $1231W * 50\% * 24h * 30/1000 = 443kWh$. The electricity unit cost was ascertained by referring to the U.S. industrial average electricity price over one year, from August 2021 to July 2022 [9]. Thus, the monthly electricity cost directly related to computation is $\$0.0786/kWh * 443kWh \approx \$35$. Additionally, the PUE (Power Usage Effectiveness) overhead, reflected by

TCO Component	Parameter	Edge Server Cost	Edge Server (W/O GPU) Cost	Parameter	SoC Cluster Cost
Capital Expenditure (CapEx)	Intel CPU	$2,740 (5.7%)	$2,740 (21.0%)	60× SoC	$24,489 (67.5%)
	DRAM	$3,540 (7.3%)	$3,540 (27.1%)	12× PCB	$7,075 (19.5%)
	Disk	$1,220 (2.5%)	$1,220 (9.4%)	Ethernet Switch Board	$689 (1.9%)
	8× NVIDIA A40 GPU	$35,192 (73.0%)	$0 (0%)	BMC	$1,923 (5.3%)
	Others	$5,544 (11.5%)	$5,544 (42.5%)	Others	$2,104 (5.8%)
	Total CapEx	$48,236	$13,044	Total CapEx	$36,280
	Total CapEx/36 months	$1,340	$363	Total CapEx/36 months	$1,008
Operational Expenditure (OpEx)	Avg. peak power consumption	1,231 watts	633 watts	Avg. peak power consumption	589 watts
	Monthly kWh (50% Util.)	443 kWh	228 kWh	Monthly kWh (50% Util.)	212 kWh
	Electricity unit cost [9]	$0.0786/kWh	$0.0786/kWh	Electricity unit cost [9]	$0.0786/kWh
	Server electricity cost	$35	$18	Server electricity cost	$17
	PUE Overhead (PUE=2.0 [42])	$35	$18	PUE Overhead (PUE=2.0 [42])	$17
	Monthly electricity cost	$70	$36	Monthly electricity cost	$34
Total	Monthly TCO	$1,410	$399	Monthly TCO	$1,042

Table 4: Capital expenditure (CapEx), operational expenditure (OpEx), and resultant monthly TCO of each server. We additionally estimated the TCO of the traditional edge server without NVIDIA GPUs. The monthly electricity cost was calculated by sampling the average peak power consumption when performing live streaming transcoding on V5.

the ratio of total building power consumption to IT infrastructure power consumption, adds to the monthly electricity cost [42]. We used a slightly higher PUE value (2.0) at the edge, compared to 1.5 at cloud data centers [42]. The overall monthly electricity cost is $\$35+\$35*(2.0-1)=\$70$.

Monthly cost. The detailed monthly cost calculation is presented in Table 4. In line with prior work [59], we break down the monthly TCO into the following components:

- *Total CapEx amortized to 36 months.* By assuming a 3-year server lifetime [42,55,59], we amortized the CapEx of each server to 36 months.

- *Monthly OpEx* mainly refers to the electricity cost. It is worth noting that the monthly OpEx is significantly less than the amortized CapEx (e.g., $70 vs. $1,340 for the traditional edge server).

We added these two expenditure figures to get the monthly TCO, then normalized the application throughput (measured in previous sections) to the monthly TCO as the TpC metric. Table 5 shows that SoC Cluster is a cost-efficient option for live streaming transcoding. Specifically, compared to the edge server with NVIDIA A40 GPUs, the SoC CPUs show a geometric mean of TpC that is 4.28× higher than the Intel CPU and 2.23× higher than NVIDIA GPUs. Even without NVIDIA GPUs in the traditional edge server, SoC CPUs achieve a geometric mean of TpC that is 1.22× higher than the Intel CPU. Moreover, the Intel CPU in the non-GPU server shows higher TpC than NVIDIA A40 GPUs for all videos. For archive transcoding, SoC Cluster is less cost-efficient than the traditional edge server due to its low throughput on a single SoC and relatively high CapEx. For most scenarios, processing archive transcoding with the NVIDIA GPUs provides higher TpC compared to other hardware options. Regarding DL serving, the NVIDIA GPUs exhibit a marked increase in cost efficiency over SoC Clusters. This is mainly attributed to their ability to handle batched DL requests and deliver high throughput under full loads.

Server	Hardware	Live Streaming Transcoding TpC (streams/$)					
		V1	V2	V3	V4	V5	V6
Edge (W/ GPU)	Intel CPU	0.180	0.223	0.057	0.101	0.042	0.013
	GPU A40	0.420	0.210	0.102	0.181	0.114	0.034
Edge (W/O GPU)	Intel CPU	0.627	0.777	0.200	0.351	0.146	0.047
SoC Cluster	SoC-CPU	0.748	0.863	0.230	0.519	0.173	0.058

Server	Hardware	Archive Transcoding TpC (frames/s/$)					
		V1	V2	V3	V4	V5	V6
Edge (W/ GPU)	Intel CPU	0.027	0.053	0.020	0.024	0.004	0.001
	GPU A40	0.162	0.140	0.203	0.086	0.091	0.035
Edge (W/O GPU)	Intel CPU	0.094	0.189	0.072	0.085	0.013	0.004
SoC Cluster	SoC-CPU	0.015	0.046	0.010	0.022	0.002	<0.001

Server	Hardware	DL Serving TpC (frames/s/$)					
		R-50 (FP32)	R-152 (FP32)	YOLO (FP32)	BERT (FP32)	R-50 (INT8)	R-152 (INT8)
Edge (W/ GPU)	Intel CPU	0.579	0.176	0.010	0.044	1.201	0.355
	GPU A40	14.631	4.535	0.571	7.311	45.684	19.840
Edge (W/O GPU)	Intel CPU	2.026	0.617	0.036	0.152	4.199	1.242
SoC Cluster	SoC-CPU	0.750	0.131	0.026	1.840	-	
	SoC-GPU	3.210	0.628	0.077	-		
	SoC-DSP	-				6.673	2.871

Table 5: Normalized application throughput to monthly TCO. We highlight the highest throughput per cost among used hardware for each video/model. "Edge": the traditional edge server.

Summary. SoC Cluster's CapEx is comparable to an 8-GPU server and is significantly higher than that of a CPU server. While SoC Cluster has the potential to reduce OpEx through electricity savings, the total cost is still dominated by the CapEx. Regarding specific workloads, SoC Cluster outperforms traditional CPU/GPU servers in cost efficiency for live streaming transcoding. However, it falls behind the NVIDIA GPU for tasks that demand significant computing capacity (archive transcoding), and those involving highly GPU-optimized workloads (DL serving).

Devices	SoC	RAM	OS	Release Date
Xiaomi 12 S	QS 8+Gen1	12 GB	Android 12	May 2022
Xiaomi 11 Pro	QS 888	8 GB	Android 11	Jun. 2021
Meizu 17	QS 865	8 GB	Android 10	Mar. 2020
Meizu 16T	QS 855	6 GB	Android 9	Mar. 2019
Xiaomi 8	QS 845	6 GB	Android 8.1	Feb. 2018
Xiaomi 6	QS 835	6 GB	Android 7.1.1	Mar. 2017

Table 6: Device specifications of six mobile phones used in the SoC longitudinal study. QS: Qualcomm Snapdragon.

7 SoC Longitudinal Study

The previous experiments were conducted on an SoC Cluster consisting of a specific SoC model, i.e., the Qualcomm Snapdragon 865 released in 2020. To expand the observations to more hardware types and, more importantly, to understand SoC performance evolution over time, we performed a longitudinal study on six mobile SoCs released from 2017 to 2022. We selected six smartphones equipped with high-end Qualcomm Snapdragon SoCs, with their specifications listed in Table 6. We repeated our experiments on two workloads and presented the results in Figure 14.

First, we measured the DL serving latency on ResNet-50 using the same experimental settings as in §5. The results reveal a significant performance boost in SoC DSPs, with an 8.4× inference latency reduction from the Snapdragon 845 (2018) to the Snapdragon 8+Gen1 (2022). SoC CPUs and GPUs also show improvements, but not as significantly as DSPs, with latency reductions of 4.8× and 3.2× from 2017 to 2022, respectively. An additional experiment focusing on inference throughput shows that the latest Snapdragon 8+Gen1 phone achieved 1.7× higher throughput on its DSP when setting the batch size to 8, compared to the default setup employing a batch size of 1. Furthermore, the recent incorporation of support for floating-point calculations on Qualcomm's flagship Hexagon DSPs [35] has positioned these processors as suitable candidates to serve increasingly complex AI tasks [37, 81, 84].

In live streaming transcoding experiments, we measured frames processed per second during the transcoding of two fixed-duration videos (V4 and V5, with metadata detailed in Table 3). Our key observation is that live streaming transcoding tasks reflect a pattern of gradual performance improvement similar to that seen in DL serving experiments. When using SoC CPUs, the throughput for video V4 on the Snapdragon 865 is 1.42×, 1.82×, and 2.3× higher than that on 855, 845, and 835, respectively. Furthermore, this value increased by 1.8× on the 8+Gen1 phone. This trend is even more impressive on the hardware codec – the throughput on the Snapdragon 865 was 3.8× and 3.24× greater than that on the 835 for V4 and V5, respectively. These performance improvements are attributed not only to the evolution of hardware but also to the enhancements of software along with the Android OS upgrades [6, 19].

To conclude, mobile SoCs have shown tremendous performance improvements in the past six years, positioning

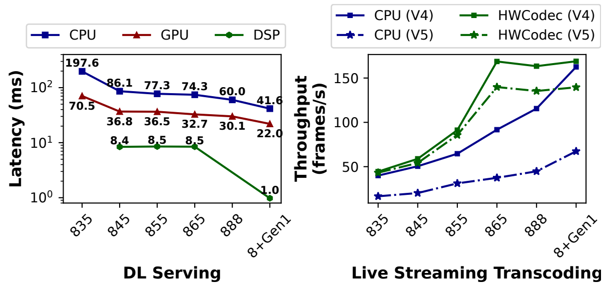


Figure 14: Performance evolution of six high-end Qualcomm Snapdragon SoCs released between 2017–2022.

them as promising candidates for more complex server-side workloads. As Moore's Law approaches its limits, CPU performance evolution might slow down, but the capability of mobile co-processors (GPUs, DSPs, and hardware codecs) continue to grow at a rapid pace, and their performance gains over CPU are also on the rise. To achieve a sustained performance evolution for mobile SoC-based servers, it is pivotal to strategically leverage these co-processors.

8 Discussion and Future Work

Applicability of measurement results. The primary goal of this study is to inspire the edge computing community with a new form of edge server, paving the way towards more efficient computing under constrained deployment environments (space, electricity, etc.). Currently, the measurement results are conducted on a specific SoC Cluster with publicly available commercial services for user access [38]. However, we believe our primary observations could apply to a broader range of SoC Cluster-like hardware models and benefit the community further. First, in terms of performance, the longitudinal study on different SoC models over years (§7) may indicate improved performance of SoC Clusters equipped with advanced SoC models. Second, improving the networking subsystem may significantly speed up data-intensive applications, such as collaborative DL training. The high energy efficiency of mobile SoCs usually makes them good candidates for workloads they can properly serve at the edge.

Overhead of SoC virtualization. The primary results on Intel CPUs in this study are measured within Docker containers, with the same experiments conducted on physical SoCs without virtualization. For a fair comparison, we measure DL serving application performance and hardware resource usage on both physical and virtualized SoCs. Currently, the virtualization solution of SoC Cluster allows a physical SoC to only run the Android Linux kernel. A virtualized SoC uses the same Linux OS and is required to run the Android framework inside Docker containers, which leads to higher memory usage as shown in Table 7. In most cases, there are trivial differences in DL serving performance and hardware resource usage. One exception is that the immature Android containerization solution prevents GPU workloads on virtualized SoCs from achieving the same high level of GPU usage

Model	Metric	SoC CPU		SoC GPU		SoC DSP	
		Phy.	Vir.	Phy.	Vir.	Phy.	Vir.
R50	Latency	81.2/0.2	80.4/0.1	32.5/0.4	33.9/0.1	11.0/0.1	10.5/0.01
	CPU	52.1/0.3	53.1/0.1	9.0/5.7	9.5/0.1	5.2/2.1	5.7/0.3
	GPU	0.7/0.3	0.5/0.1	73.9/1.2	71.3/0.6	0.6/0.1	0.6/0.1
	Mem	32.3/0.7	37.7/0.2	35.2/5.7	37.6/0.3	32.7/1.9	37.4/0.2
R152	Latency	258.3/0.4	257.8/1.0	100.9/0.1	102.8/0.2	21.0/0.04	20.4/0.02
	CPU	53.3/0.1	53.9/0.3	5.9/0.5	8.6/0.1	6.0/0.9	7.1/0.5
	GPU	0.4/0.1	0.6/0.1	81.1/0.5	78.5/0.2	0.7/0.1	0.6/0.1
	Mem	34.9/1.0	40.1/0.1	35.6/2.1	39.8/0.1	33.7/0.8	39.0/0.2
YOLO	Latency	1121.3/13.7	1113.9/2.8	620.6/1.0	683.7/4.1	/	
	CPU	53.9/0.2	54.5/0.1	5.3/0.2	7.6/0.1		
	GPU	0.5/0.1	0.6/0.04	82.5/0.1	77.1/0.4		
	Mem	40.1/0.6	45.9/0.1	39.5/3.3	44.2/0.4		

Table 7: DL inference performance and hardware usages (average/standard deviation) on physical SoCs (Phy.) and virtualized SoCs (Vir.). Latency is measured in milliseconds. CPU, GPU, and memory utilization are represented as percentages.

as on physical SoCs, leading to a slight performance slowdown (e.g., 60 ms on the YOLOv5x model). We believe the SoC virtualization implementation can be further improved to mitigate the potential performance overhead and unnecessary resource usage when the Android framework is not required.

There are also a few directions to further explore to enhance the vision of this study.

• *Killer applications*. SoC Cluster is endowed with ample storage space and high I/O speed, making it well-suited for database systems with compatible design patterns [90]. The SoC-level workload scheduling granularity lends itself to ephemeral serverless workloads [76]. However, mobile SoCs are not typically designed to operate at full speed and 24/7 in clouds, presents a challenge for operating SoC Cluster. The failure of a single SoC subsystem, such as flash, can render the application and entire SoC unusable. Therefore, fault tolerance is crucial for the success of SoC Cluster.

• *Operating system*. Mobile OSs are designed and optimized for interactive scenarios rather than server workloads. Although it is viable to run Linux or Windows [39] on ARM SoCs, simply replacing Android with other OSs may result in the loss of compatibility with native mobile apps, as well as the inability to leverage certain hardware accelerators if their drivers are vendor-specific and proprietary [82]. To get the most from both, the correct approach seems to revise the original Android OS to fit edge workloads.

• *Network infrastructure and topology.* We show in §4 that an overall 20 Gbps network capacity is mostly sufficient to support video transcoding on all SoCs. However, the limited bandwidth between SoCs inside SoC Clusters makes the current network infrastructure is not equipped for workloads requiring high-volume data exchanges across SoCs. High performance datacenter network interfaces and switches, such as InfiniBand [1] and NVLink [2], provide network bandwidth in the hundreds of Gbps, which is two orders of magnitude higher than the 1 Gbps theoretical bandwidth between SoCs in SoC Clusters. To support a wider range of data-intensive applications, SoC Cluster should incorporate recent progress from network research and industry [1, 17, 79].

9 Related Work

Grouping mobile SoCs as servers. Several previous attempts have been made to conceptualize a server consisting of compact SoCs. Some researchers have investigated whether mobile SoCs can provide sufficient performance and reduce costs for HPC [72, 73]. In an effort to reduce e-waste, Shahrad *et al.* [77] constructed computational nodes with used smartphones, but they analyzed server design without evaluating real workloads. Switzer *et al.* created a junkyard cluster [80] comprising just ten smartphones, with an emphasis on reducing carbon footprints. Our previous study proposed a similar vision of renovating mobile SoCs at the edge [86], but limited experiments and evaluation did not fully reveal the capabilities of SoC Clusters. Other studies have employed IoT/mobile SoCs to support specific applications, such as video transcoding [64], key-value storage [41], web search [55], and parallel computing [44]. However, these studies mainly focus on specialized app types and lack a performance comparison with traditional servers. In contrast, our study utilized a commercial-off-the-shelf SoC Cluster, conducted application-driven measurements, and presented extensive performance metrics to showcase its suitability for modern, computation-intensive edge workloads.

Energy-efficient cloud/edge. Energy efficiency has been recognized as a crucial factor in data centers [48]. Various techniques have been explored to advance green data centers, including workload scheduling and management [60, 63, 65, 74, 91], resource under-provisioning [67, 88], greening the data-center network [49, 51, 53], among others. In contrast to these software-level approaches, we propose a redesign of servers to fundamentally enhance energy efficiency. As the edge infrastructure is still in its preliminary stage, we consider such a radical measure to be feasible.

10 Conclusion

In this study, we explored the feasibility and implications of utilizing a novel type of edge server, SoC Cluster, which comprises multiple mobile SoCs. We conducted extensive benchmarking on two typical edge workloads (i.e., video transcoding and DL serving), and quantitatively demonstrated SoC Cluster achieves substantial energy savings compared to conventional edge servers, while also identifying its limitations. These findings highlight the promising potential of SoC Cluster at the edge, and provide guidance for further improvements in both hardware and software.

Acknowledgments

We sincerely thank our shepherd, Mohammad Shahrad, and the anonymous reviewers for their constructive feedback and efforts, which significantly improved our work. This work was supported by National Key R&D Program of China (No.2021ZD0113001), NSFC (62102045, 62032003, 61921003), and China Institute of IoT (Wuxi). Mengwei Xu is the corresponding author of this work.

References

[1] Infiniband - Wikipedia. https://en.wikipedia.org/wiki/InfiniBand.

[2] Nvlink & nvswitch | fastest hpc data center platform | nvidia. https://www.nvidia.com/en-us/data-center/nvlink/.

[3] Cloud ar/vr white paper. https://www.gsma.com/futurenetworks/wiki/cloud-ar-vr-whitepaper/, 2019.

[4] Energy-efficient cloud computing technologies and policies for an eco-friendly cloud market. https://digital-strategy.ec.europa.eu/en/library/energy-efficient-cloud-computing-technologies-and-policies-eco-friendly-cloud-market, 2020.

[5] Hardware | Qualcomm Snapdragon 865 5G Mobile Platform | 5G Mobile Processor | Qualcomm. https://www.qualcomm.com/products/application/smartphones/snapdragon-8-series-mobile-platforms/snapdragon-865-5g-mobile-platform, 2020.

[6] Video encoding improvemens - Android 12. https://developer.android.com/about/versions/12/features#video-encoding, 2021.

[7] Amazon EC2 Instance Types. https://aws.amazon.com/ec2/instance-types/, 2022.

[8] BERT en uncased. https://tfhub.dev/tensorflow/bert_en_uncased_L-12_H-768_A-12/1, 2022.

[9] Electric power monthly - u.s. energy information administration (eia). https://www.eia.gov/electricity/monthly/epm_table_grapher.php?t=table_5_03, 2022.

[10] Ffmpeg. https://ffmpeg.org/, 2022.

[11] Geekbench 5 - cross-platform benchmark. https://www.geekbench.com/, 2022.

[12] Geforce Now. https://www.nvidia.com/en-us/geforce-now/, 2022.

[13] Genshin Impact. https://genshin.hoyoverse.com/en/, 2022.

[14] Google Stadia. https://stadia.google.com/, 2022.

[15] Hwaccelintro - FFmpeg. https://trac.ffmpeg.org/wiki/HWAccelIntro, 2022.

[16] Instance family. https://www.alibabacloud.com/help/en/elastic-compute-service/latest/instance-family, 2022.

[17] Intel tofino. https://www.intel.com/content/www/us/en/products/network-io/programmable-ethernet-switch/tofino-series.html, 2022.

[18] linkedin/litr: Lightweight hardware accelerated video/audio transcoder for Android. https://github.com/linkedin/LiTr, 2022.

[19] Low Latency Decoding in MediaCodec - Android Open Source Project. https://source.android.com/devices/media/low-latency-media, 2022.

[20] Mediacodec. https://developer.android.com/reference/android/media/MediaCodec, 2022.

[21] Nvidia TensorRT. https://developer.nvidia.com/tensorrt, 2022.

[22] Nvidia triton inference server. https://developer.nvidia.com/nvidia-triton-inference-server, 2022.

[23] Nvidia Video Codec SDK. https://developer.nvidia.com/nvidia-video-codec-sdk, 2022.

[24] Recommended upload encoding settings - YouTube Help. https://support.google.com/youtube/answer/1722171?hl=en, 2022.

[25] remote-android/redroid-doc. https://github.com/remote-android/redroid-doc, 2022.

[26] Serving Models | TFX | TensorFlow. https://www.tensorflow.org/tfx/guide/serving, 2022.

[27] Tensorflow Lite. https://www.tensorflow.org/lite, 2022.

[28] Torchserve. https://pytorch.org/serve/, 2022.

[29] Verizon and aws expand edge computing to 19 u.s. metro areas. https://www.verizon.com/about/news/verizon-aws-expand-edge-computing-19-us-metro-areas, 2022.

[30] Virtual machine series | Microsoft Azure. https://azure.microsoft.com/en-us/pricing/details/virtual-machines/series/, 2022.

[31] X-cloud Game Pass. https://www.xbox.com/en-US/xbox-game-pass/cloud-gaming?xr=shellnav, 2022.

[32] x264, the best H.264/AVC encoder - VideoLAN. https://www.videolan.org/developers/x264.html, 2022.

[33] Anbox cloud - scalable android in the cloud. https://anbox-cloud.io/, 2023.

[34] Genshin impact cloud gaming early access: Release, testing, and everything to know. https://www.sportskeeda.com/esports/news-genshin-impact-cloud-gaming-early-access-release-testing-everything-know, 2023.

[35] Snapdragon 8 gen 2 mobile platform. https://www.qualcomm.com/products/mobile/snapdragon/smartphones/snapdragon-8-series-mobile-platforms/snapdragon-8-gen-2-mobile-platform, 2023.

[36] Virtual smartphone | cophone. https://cophone.io/, 2023.

[37] World's first on-device demonstration of stable diffusion on android. https://www.qualcomm.com/news/onq/2023/02/worlds-first-on-device-demonstration-of-stable-diffusion-on-android, 2023.

[38] Elastic cloud phone | alibaba cloud. https://www.alibabacloud.com/help/en/ecp/product-overview/what-is-ecp, 2024.

[39] Snapdragon x elite | our newest laptop processor | qualcomm. https://www.qualcomm.com/products/mobile/snapdragon/pcs-and-tablets/snapdragon-x-elite, 2024.

[40] Video Encode and Decode GPU Support Matrix | NVIDIA Developer. https://developer.nvidia.com/video-encode-and-decode-gpu-support-matrix-new, 2024.

[41] David G Andersen, Jason Franklin, Michael Kaminsky, Amar Phanishayee, Lawrence Tan, and Vijay Vasudevan. Fawn: A fast array of wimpy nodes. In *Proceedings of the ACM SIGOPS 22nd symposium on Operating systems principles*, pages 1–14, 2009.

[42] Luiz André Barroso, Urs Hölzle, and Parthasarathy Ranganathan. The datacenter as a computer: Designing warehouse-scale machines. *Synthesis Lectures on Computer Architecture*, 13(3):i–189, 2018.

[43] Luiz André Barroso and Urs Hölzle. The case for energy-proportional computing. *IEEE Computer*, 40, 2007.

[44] Felix Büsching, Sebastian Schildt, and Lars Wolf. Droidcluster: Towards smartphone cluster computing–the streets are paved with potential computer clusters. In *2012 32nd International Conference on Distributed Computing Systems Workshops*, pages 114–117. IEEE, 2012.

[45] Tianqi Chen, Thierry Moreau, Ziheng Jiang, Lianmin Zheng, Eddie Yan, Haichen Shen, Meghan Cowan, Leyuan Wang, Yuwei Hu, Luis Ceze, Carlos Guestrin, and Arvind Krishnamurthy. Tvm: An automated end-to-end optimizing compiler for deep learning. In *13th USENIX Symposium on Operating Systems Design and Implementation (OSDI 18)*, pages 578–594, Carlsbad, CA, October 2018. USENIX Association.

[46] Eli Cortez, Anand Bonde, Alexandre Muzio, Mark Russinovich, Marcus Fontoura, and Ricardo Bianchini. Resource central: Understanding and predicting workloads for improved resource management in large cloud platforms. In *Proceedings of the 26th Symposium on Operating Systems Principles*, pages 153–167, 2017.

[47] Daniel Crankshaw, Xin Wang, Guilio Zhou, Michael J. Franklin, Joseph E. Gonzalez, and Ion Stoica. Clipper: A Low-Latency online prediction serving system. In *14th USENIX Symposium on Networked Systems Design and Implementation (NSDI 17)*, pages 613–627, Boston, MA, March 2017. USENIX Association.

[48] Wei Deng, Fangming Liu, Hai Jin, Bo Li, and Dan Li. Harnessing renewable energy in cloud datacenters: opportunities and challenges. *iEEE Network*, 28(1):48–55, 2014.

[49] Peter Xiang Gao, Andrew R. Curtis, Bernard Wong, and Srinivasan Keshav. It's not easy being green. *SIGCOMM Comput. Commun. Rev.*, 42(4):211–222, aug 2012.

[50] Arpan Gujarati, Reza Karimi, Safya Alzayat, Wei Hao, Antoine Kaufmann, Ymir Vigfusson, and Jonathan Mace. Serving DNNs like clockwork: Performance predictability from the bottom up. In *14th USENIX Symposium on Operating Systems Design and Implementation (OSDI 20)*, pages 443–462. USENIX Association, November 2020.

[51] Zehua Guo, Yang Xu, Ya-Feng Liu, Sen Liu, H Jonathan Chao, Zhi-Li Zhang, and Yuanqing Xia. Aggreflow: Achieving power efficiency, load balancing, and quality of service in data center networks. *IEEE/ACM Transactions on Networking*, 29(1):17–33, 2020.

[52] Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. Deep residual learning for image recognition. *arXiv preprint arXiv:1512.03385*, 2015.

[53] Brandon Heller, Srini Seetharaman, Priya Mahadevan, Yiannis Yiakoumis, Puneet Sharma, Sujata Banerjee, and Nick McKeown. Elastictree: Saving energy in data center networks. In *Proceedings of the 7th USENIX Conference on Networked Systems Design and Implementation*, NSDI'10, page 17, USA, 2010. USENIX Association.

[54] Alain Horé and Djemel Ziou. Image quality metrics: Psnr vs. ssim. In *2010 20th International Conference on Pattern Recognition*, pages 2366–2369, 2010.

[55] Vijay Janapa Reddi, Benjamin C Lee, Trishul Chilimbi, and Kushagra Vaid. Web search using mobile cores: quantifying and mitigating the price of efficiency. *ACM SIGARCH Computer Architecture News*, 38(3):314–325, 2010.

[56] Xiaotang Jiang, Huan Wang, Yiliu Chen, Ziqi Wu, Lichuan Wang, Bin Zou, Yafeng Yang, Zongyang Cui, Yu Cai, Tianhang Yu, et al. Mnn: A universal and efficient inference engine. *arXiv preprint arXiv:2002.12418*, 2020.

[57] Glenn Jocher, Ayush Chaurasia, Alex Stoken, Jirka Borovec, NanoCode012, Yonghye Kwon, TaoXie, Jiacong Fang, imyhxy, Kalen Michael, Lorna, Abhiram V, Diego Montes, Jebastin Nadar, Laughing, tkianai, yxNONG, Piotr Skalski, Zhiqiang Wang, Adam Hogan, Cristi Fati, Lorenzo Mammana, AlexWang1900, Deep Patel, Ding Yiwei, Felix You, Jan Hajek, Laurentiu Diaconu, and Mai Thanh Minh. ultralytics/yolov5: v6.1 - TensorRT, TensorFlow Edge TPU and OpenVINO Export and Inference, February 2022.

[58] Eric Jonas, Johann Schleier-Smith, Vikram Sreekanti, Chia-che Tsai, Anurag Khandelwal, Qifan Pu, Vaishaal Shankar, João Carreira, Karl Krauth, Neeraja Jayant Yadwadkar, Joseph E. Gonzalez, Raluca Ada Popa, Ion Stoica, and David A. Patterson. Cloud programming simplified: A berkeley view on serverless computing. *CoRR*, abs/1902.03383, 2019.

[59] Willis Lang, Jignesh M Patel, and Srinath Shankar. Wimpy node clusters: What about non-wimpy workloads? In *Proceedings of the Sixth International Workshop on Data Management on New Hardware*, pages 47–55, 2010.

[60] Seunghak Lee, Ki-Dong Kang, Hwanjun Lee, Hyungwon Park, Younghoon Son, Nam Sung Kim, and Daehoon Kim. Greendimm: Os-assisted dram power management for dram with a sub-array granularity power-down state. In *MICRO-54: 54th Annual IEEE/ACM International Symposium on Microarchitecture*, pages 131–142, 2021.

[61] Shaohong Li, Xi Wang, Xiao Zhang, Vasileios Kontorinis, Sreekumar Kodakara, David Lo, and Parthasarathy Ranganathan. Thunderbolt: Throughput-Optimized, Quality-of-Service-Aware power capping at scale. In *14th USENIX Symposium on Operating Systems Design and Implementation (OSDI 20)*, pages 1241–1255. USENIX Association, November 2020.

[62] Yanan Li, Guangqing Deng, Changming Bai, Jingyu Yang, Gang Wang, Hao Zhang, Jin Bai, Haitao Yuan, Mengwei Xu, and Shangguang Wang. Demystifying the qos and qoe of edge-hosted video streaming applications in the wild with snESet. *Proceedings of the ACM on Management of Data*, 1(4):1–29, 2023.

[63] Fangming Liu, Zhi Zhou, Hai Jin, Bo Li, Baochun Li, and Hongbo Jiang. On arbitrating the power-performance tradeoff in saas clouds. *IEEE Transactions on Parallel and Distributed Systems*, 25(10):2648–2658, 2013.

[64] Peng Liu, Jongwon Yoon, Lance Johnson, and Suman Banerjee. Greening the video transcoding service with {Low-Cost} hardware transcoders. In *2016 USENIX Annual Technical Conference (USENIX ATC 16)*, pages 407–419, 2016.

[65] Zhenhua Liu, Yuan Chen, Cullen Bash, Adam Wierman, Daniel Gmach, Zhikui Wang, Manish Marwah, and Chris Hyser. Renewable and cooling aware workload management for sustainable data centers. In *Proceedings of the 12th ACM SIGMETRICS/PERFORMANCE joint international conference on Measurement and Modeling of Computer Systems*, pages 175–186, 2012.

[66] Andrea Lottarini, Alex Ramírez, Joel Coburn, Martha A. Kim, Parthasarathy Ranganathan, Daniel Stodolsky, and Mark Wachsler. vbench: Benchmarking video transcoding in the cloud. In Xipeng Shen, James Tuck, Ricardo Bianchini, and Vivek Sarkar, editors, *Proceedings of the Twenty-Third International Conference on Architectural Support for Programming Languages and Operating Systems, ASPLOS 2018, Williamsburg, VA, USA, March 24-28, 2018*, pages 797–809. ACM, 2018.

[67] Ioannis Manousakis, Íñigo Goiri, Sriram Sankar, Thu D Nguyen, and Ricardo Bianchini. Coolprovision: Underprovisioning datacenter cooling. In *Proceedings of the Sixth ACM Symposium on Cloud Computing*, pages 356–367, 2015.

[68] Jongsoo Park, Maxim Naumov, Protonu Basu, Summer Deng, Aravind Kalaiah, Daya Shanker Khudia, James Law, Parth Malani, Andrey Malevich, Nadathur Satish, Juan Miguel Pino, Martin Schatz, Alexander Sidorov, Viswanath Sivakumar, Andrew Tulloch, Xiaodong Wang, Yiming Wu, Hector Yuen, Utku Diril, Dmytro Dzhulgakov, Kim M. Hazelwood, Bill Jia, Yangqing Jia, Lin Qiao, Vijay Rao, Nadav Rotem, Sungjoo Yoo, and Mikhail Smelyanskiy. Deep learning inference in facebook data centers: Characterization, performance optimizations and hardware implications. *CoRR*, abs/1811.09886, 2018.

[69] Qiangyu Pei, Shutong Chen, Qixia Zhang, Xinhui Zhu, Fangming Liu, Ziyang Jia, Yishuo Wang, and Yongjie Yuan. Cooledge: Hotspot-relievable warm water cooling for energy-efficient edge datacenters. In *Proceedings of the 27th ACM International Conference on Architectural Support for Programming Languages and Operating Systems*, ASPLOS 2022, page 814–829, New York, NY, USA, 2022. Association for Computing Machinery.

[70] Qualcomm. Designing mobile devices for low power and thermal efficiency, 2013.

[71] Qualcomm. How our power-efficient technologies benefit smartphone users and the earth. https://www.qualcomm.com/news/onq/2021/02/how-our-power-efficient-technologies-benefit-smartphone-users-and-earth, 2021.

[72] Nikola Rajovic, Paul M Carpenter, Isaac Gelado, Nikola Puzovic, Alex Ramirez, and Mateo Valero. Supercomputing with commodity cpus: Are mobile socs ready for hpc? In *Proceedings of the International Conference on High Performance Computing, Networking, Storage and Analysis*, pages 1–12, 2013.

[73] Nikola Rajovic, Alejandro Rico, Nikola Puzovic, Chris Adeniyi-Jones, and Alex Ramirez. Tibidabo: Making the case for an arm-based hpc system. *Future Generation Computer Systems*, 36:322–334, 2014.

[74] Yongyi Ran, Han Hu, Xin Zhou, and Yonggang Wen. Deepee: Joint optimization of job scheduling and cooling control for data center energy efficiency using deep reinforcement learning. In *2019 IEEE 39th International Conference on Distributed Computing Systems (ICDCS)*, pages 645–655. IEEE, 2019.

[75] Parthasarathy Ranganathan, Daniel Stodolsky, Jeff Calow, Jeremy Dorfman, Marisabel Guevara, Clinton Wills Smullen IV, Aki Kuusela, Raghu Balasubramanian, Sandeep Bhatia, Prakash Chauhan, Anna Cheung, In Suk Chong, Niranjani Dasharathi, Jia Feng, Brian Fosco, Samuel Foss, Ben Gelb, Sara J. Gwin, Yoshiaki Hase, Da-ke He, C. Richard Ho, Roy W. Huffman Jr., Elisha Indupalli, Indira Jayaram, Poonacha Kongetira, Cho Mon Kyaw, Aaron Laursen, Yuan Li, Fong Lou, Kyle A. Lucke, J. P. Maaninen, Ramon Macias, Maire Mahony, David Alexander Munday, Srikanth Muroor, Narayana Penukonda, Eric Perkins-Argueta, Devin Persaud, Alex Ramírez, Ville-Mikko Rautio, Yolanda Ripley, Amir Salek, Sathish Sekar, Sergey N. Sokolov, Rob Springer, Don Stark, Mercedes Tan, Mark S. Wachsler, Andrew C. Walton, David A. Wickeraad, Alvin Wijaya, and Hon Kwan Wu. Warehouse-scale video acceleration: co-design and deployment in the wild. In Tim Sherwood, Emery D. Berger, and Christos Kozyrakis, editors, *ASPLOS '21: 26th ACM International Conference on Architectural Support for Programming Languages and Operating Systems, Virtual Event, USA, April 19-23, 2021*, pages 600–615. ACM, 2021.

[76] Mohammad Shahrad, Rodrigo Fonseca, Iñigo Goiri, Gohar Chaudhry, Paul Batum, Jason Cooke, Eduardo Laureano, Colby Tresness, Mark Russinovich, and Ricardo Bianchini. Serverless in the wild: Characterizing and optimizing the serverless workload at a large cloud provider. In Ada Gavrilovska and Erez Zadok, editors, *2020 USENIX Annual Technical Conference, USENIX ATC 2020, July 15-17, 2020*, pages 205–218. USENIX Association, 2020.

[77] Mohammad Shahrad and David Wentzlaff. Towards deploying decommissioned mobile devices as cheap energy-efficient compute nodes. In *9th USENIX Workshop on Hot Topics in Cloud Computing (HotCloud 17)*, 2017.

[78] Yizhou Shan, Will Lin, Zhiyuan Guo, and Yiying Zhang. Towards a fully disaggregated and programmable data center. In *Proceedings of the 13th ACM SIGOPS Asia-Pacific Workshop on Systems*, pages 18–28, 2022.

[79] Anirudh Sivaraman, Thomas Mason, Aurojit Panda, Ravi Netravali, and Sai Anirudh Kondaveeti. Network architecture in the age of programmability. *ACM SIGCOMM Computer Communication Review*, 50(1):38–44, 2020.

[80] Jennifer Switzer, Gabriel Marcano, Ryan Kastner, and Pat Pannuto. Junkyard computing: Repurposing discarded smartphones to minimize carbon. In *Proceedings of the 28th ACM International Conference on Architectural Support for Programming Languages and Operating Systems, Volume 2*, ASPLOS 2023, New York, NY, USA, 2023. Association for Computing Machinery.

[81] Hugo Touvron, Thibaut Lavril, Gautier Izacard, Xavier Martinet, Marie-Anne Lachaux, Timothée Lacroix, Baptiste Rozière, Naman Goyal, Eric Hambro, Faisal Azhar, Aurelien Rodriguez, Armand Joulin, Edouard Grave, and Guillaume Lample. Llama: Open and efficient foundation language models, 2023.

[82] Lei Wu, Michael Grace, Yajin Zhou, Chiachih Wu, and Xuxian Jiang. The impact of vendor customizations on android security. In *Proceedings of the 2013 ACM SIGSAC conference on Computer & communications security*, pages 623–634, 2013.

[83] Jun Xin, Chia-Wen Lin, and Ming-Ting Sun. Digital video transcoding. *Proceedings of the IEEE*, 93(1):84–97, 2005.

[84] Daliang Xu, Mengwei Xu, Qipeng Wang, Shangguang Wang, Yun Ma, Kang Huang, Gang Huang, Xin Jin, and Xuanzhe Liu. Mandheling: Mixed-precision on-device dnn training with dsp offloading. In *Proceedings of the 28th Annual International Conference on Mobile Computing And Networking*, pages 214–227, 2022.

[85] Mengwei Xu, Zhe Fu, Xiao Ma, Li Zhang, Yanan Li, Feng Qian, Shangguang Wang, Ke Li, Jingyu Yang, and Xuanzhe Liu. From cloud to edge: a first look at public edge platforms. In *Proceedings of the 21st ACM Internet Measurement Conference*, pages 37–53, 2021.

[86] Mengwei Xu, Li Zhang, and Shangguang Wang. Position paper: Renovating edge servers with arm socs. In *2022 IEEE/ACM 7th Symposium on Edge Computing (SEC)*, pages 216–223, 2022.

[87] Liekang Zeng, Xu Chen, Zhi Zhou, Lei Yang, and Junshan Zhang. Coedge: Cooperative dnn inference with adaptive workload partitioning over heterogeneous edge devices. *IEEE/ACM Transactions on Networking*, 29(2):595–608, 2020.

[88] Chaojie Zhang, Alok Gautam Kumbhare, Ioannis Manousakis, Deli Zhang, Pulkit A Misra, Rod Assis, Kyle Woolcock, Nithish Mahalingam, Brijesh Warrier, David Gauthier, et al. Flex: High-availability datacenters with zero reserved power. In *2021 ACM/IEEE 48th Annual International Symposium on Computer Architecture (ISCA)*, pages 319–332. IEEE, 2021.

[89] Chengliang Zhang, Minchen Yu, Wei Wang, and Feng Yan. MArk: Exploiting cloud services for Cost-Effective, SLO-Aware machine learning inference serving. In *2019 USENIX Annual Technical Conference (USENIX ATC 19)*, pages 1049–1062, Renton, WA, July 2019. USENIX Association.

[90] Tao Zhang, Aviad Zuck, Donald E. Porter, and Dan Tsafrir. Apps can quickly destroy your mobile's flash: Why they don't, and how to keep it that way. In *Proceedings of the 17th Annual International Conference on Mobile Systems, Applications, and Services*, MobiSys '19, page 207–221, New York, NY, USA, 2019. Association for Computing Machinery.

[91] Zhi Zhou, Fangming Liu, Ruolan Zou, Jiangchuan Liu, Hong Xu, and Hai Jin. Carbon-aware online control of geo-distributed cloud services. *IEEE Transactions on Parallel and Distributed Systems*, 27(9):2506–2519, 2015.