2021 IEEE 27th International Conference on Parallel and Distributed Systems (ICPADS) | 978-1-6654-0878-3/21/$31.00 ©2021 IEEE | DOI: 10.1109/ICPADS53394.2021.00031

Choosing Appropriate AI-enabled Edge Devices, Not the Costly Ones

Ziyang Zhang[1], Feng Li[1], Changyao Lin[1], Shihui Wen[3], Xiangyu Liu[3], and Jie Liu[2]
[1]School of Computer Science and Technology, Harbin Institute of Technology, Harbin, China
[2]Institute for Artificial Intelligence, Harbin Institute of Technology, Shenzhen, China
[3]School of Astronautics, Harbin Institute of Technology, Harbin, China
Email: {20B903026, 20S003095, 21B304003}@stu.hit.edu.cn, wenshihui_jiayou@163.com
{feng.li, jieliu}@hit.edu.cn

***Abstract*—Advances in Edge AI make it possible to achieve inference deep learning for emerging applications, e.g., smart transportation and smart city on the edge in real-time. Nowadays, different industry companies have developed several edge AI devices with various architectures. However, it is hard for application users to justify how to choose the appropriate edge-AI, due to the lack of benchmark testing results and testbeds specifically used to evaluate the system performance for those edge-AI systems. In this paper, we attempt to design a benchmark test platform for the edge-AI devices and evaluate six mainstream edge devices that are equipped with different computing powers and AI chip architectures. Throughput, power consumption ratio, and cost-effectiveness are chosen as the performance metrics for the evaluation process. Three classic deep learning workloads: object detection, image classification, and natural language processing are adopted with different batch sizes. The results show that under different batch sizes, compared with traditional edge devices, edge devices equipped with AI chips have outperformance in throughput, power consumption ratio, and cost-effectiveness by 134×, 57×, and 32×, respectively. From system perspective, our work not only demonstrates the effective AI capabilities of those edge AI devices, but also provide suggestions for AI optimization at edge in details.**

***Index Terms*—Edge AI, System Performance, Deep Learning**

I. Introduction

Recent advances in the Internet of Things have driven more and more applications, such as smart city and smart grid, etc. to reality. In those environments, one of the new challenges is how to make an intelligent decision in real-time with low latency, close to the site at the edge.

To solve the problem, some companies, including NVIDIA, Google, Intel, etc. have developed novel embedded devices by integrating dedicated neural network processing units, or so-called AI chips that empower the process and data analysis capabilities even when there is no network connection. Running deep learning workloads on edge devices with AI chips can reduce the inference delay to a few milliseconds or even lower, and because there is no transmission delay, the entire quality of service (QoS) is improved [1]. In addition, through the local processing and analysis of data at the edge device, there is no need to upload the data to the data center, and thus provides security and privacy protection.

The edge AI devices/systems are very promising in many applications, for instance, a smart camera with machine learning capabilities can perform video analysis locally to determine relevant video clips, and only send these clips to the cloud, thereby reducing privacy risks. In addition, edge AI devices can deploy deep learning workloads in a low-power mode of several watts. With more and more emerging demands for such AI paradigm, including object detection in autonomous driving, multi-sensor data fusion in smart agriculture, and machine vision in smart factories, some companies also provide deep learning inference engines such as NVIDIA's TensorRT [2], Google's Tensorflow Lite [3], Microsoft's ONNX [4] and Intel's OpenVINO [5], as software supporting suit on the system, to take full use of the advantages of the specific chip designs. Although in general, the embedded system with AI accelerator and inference engine can improve the performance dramatically at the edge, for different application scenarios, how to choose appropriate edge computing ability to satisfy various constraints is still open and hard for application developers.

Unlike the mature evaluation standards for the x86 architecture server-side, the current edge devices are normally based on the ARM architecture and have not yet established a standard architecture and evaluation benchmark from the edge AI perspective. On the other hand, due to the limited resource constraints, it is not wise to just directly borrow those metrics or standards for AI at the edge evaluation.

In this paper, we seek to evaluate the system performance on edge devices for typical AI algorithms. Our contributions are as follows:

- We set up an experimental environment to benefit the evaluation process. Specifically, we choose six mainstream edge AI devices, including Raspberry Pi, Nvidia's Jetson Nano/TX2/NX, Rockchip's RK3399Pro, and Bitmain's SE5 to build an edge device performance testing platform, and deploy three classic deep learning workloads, including object detection, image classification, and natural language processing on each device. Throughput, power consumption ratio, and cost-effectiveness are used to evaluate the comprehensive performance for each device.
- We conduct various experiments and the results show: 1) For YOLO-v3 workloads, compared to traditional edge devices such as Raspberry Pi, the average throughput

2690-5965/21/$31.00 ©2021 IEEE
DOI 10.1109/ICPADS53394.2021.00031

of edge AI equipment equipped with a dedicated neural network processing unit under different batch sizes can produce up to 373× performance improvement. 2) Followed by ResNet-50, edge AI devices also have a performance improvement of 27×. 3) In Tiny-BERT's benchmark test, we observe Edge AI devices only gain 2× improvement. 4) For the average throughput per unit power consumption and unit price, edge AI devices also have the standardized performance of 57× and 22×, respectively. In conclusion, in general, edge devices with AI abilities demonstrate the out-performance for deep learning workloads, and at the same time, the system proves to be cost-effective.

- We give an in-depth analysis from both application and system levels for the explanation of the evaluation results. Based on such, finally, considering the edge computing and model efficiency, we provide recommendations for the deployment of deep learning workloads on edge devices to promote future research on edge AI.

The rest of the paper is organized as follows: Section II introduces related work. Section III presents our experimental platform, workload, and test metrics. Section IV shows the results of our proposed benchmark test and provides performance analysis. Section V provides recommendations for model optimization and deployment. Finally, in Section VI, we summarize our work.

II. Related work

To our knowledge, most of the current benchmark tests mainly focus on cloud servers, which are few for edge devices. Next, we will divide the existing benchmark tests into four categories and discuss them separately.

AI Bench: AI Bench include both micro-benchmarks and component and application benchmarks. Gao et al. [6] summarized the differences on several previous works. Benchmarks like MLPerf [7], AIIA DNN Benchmark [8], TBD [9] and Dawnbench [10] focus on the component and application benchmarks, while DNNMark [11] focus on the micro and application benchmarks.

AIoT Bench: Luo et al. [12] proposed an AIoT benchmark suite, which can evaluate the AI ability of mobile and embedded devices. The AIoT Bench not only covers a variety of application domains like image and speech recognition but also include different edge devices, such as Android mobile device and Raspberry Pi. BenchCouncil published a standard specification of AIoT Bench [13], proposed a unified metric, Valid Flops Per Second (VFPS), to evaluate the AI inference accuracy and the speed of AI devices.

Edge AI Bench: Edge AI Bench [14] was proposed by the BenchCouncil, which built an edge computing tested platform and took an end-to-end view and focus on data distribution and workload collaboration on clientside devices layer, edge computing layer, and cloud servers layer. In 2019, a standard specification of Edge AI Bench [15] was published to test the metrics of inference stage and training stage of the above four edge AI scenarios.

Edge Bench: Das et al. [16] provided a suite of performance metrics as a new benchmark, then tested and compared two platforms, Amazon AWS Greengrass and Microsoft Azure IoT Edge, as well as cloud-only implementations available in their respective cloud ecosystems, and analyzed the differences in key types of workloads used in edge applications. Qirui Yang, et al. [17] evaluated two representative edge workflows, an IoT hub workflow, and a video analytics workflow, using the workflow-level and function-level metrics reported by EdgeBench, to illustrate the performance bottlenecks of the edge systems and the edge workloads.

III. Experimental Design: Background and Methodology

In this section, we will first present related background, then we discuss the testing system design methodology.

A. Background

The current computing engines in the AI chip field are mainly divided into three types: Application Specific Processor, Dedicated Hardware Accelerators, and Programming Fabric. The AI chips in the edge devices we tested here belong to the first two types. This section will introduce these AI chips.

1) Application Specific Processor: The edge AI devices of the NVIDIA series we tested are equipped with GPU with different architectures. Specifically, Jetson Nano is equipped with 128 CUDA cores GPU based on the Maxwell architecture. Jetson TX2 is equipped with 256 CUDA cores GPU based on the Pascal architecture. Xavier NX is equipped with 384 CUDA cores GPU and 48 Tensor cores based on the Volta architecture. Fig.1 is an architectural diagram of NVIDIA GPU most of the GPU's core graphics functions are performed inside the Graphics Processing Cluster (GPC), within the GPC there are multiple Streaming Multiprocessor (SM) units.

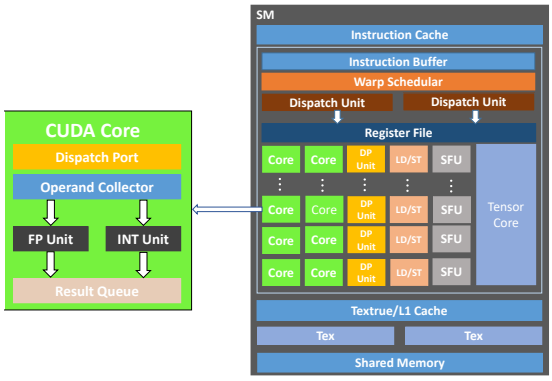

Fig. 1: NVIDIA GPU architecture.

2) Dedicated HardWare Accelerators: The NPU of Rockchip RK3399Pro is a processing unit dedicated to neural networks. Bitmain's SE5 is equipped with the TPU chip BM1684, it has a built-in tensor calculation module TPU. The

TPU module contains 64 NPU arithmetic units. Each NPU includes 16 Processing Element(PE), for a total of 1024 PE.

Fig.2 is the architecture diagram of the NPU [18]. The computing units of the RK3399Pro's NPU and SE5's TPU draw lessons from the systolic array structure [19] of Google's TPU. It realizes the direct transmission of calculation data and weight data between each PE, and thus improves the overall data reuse rate.

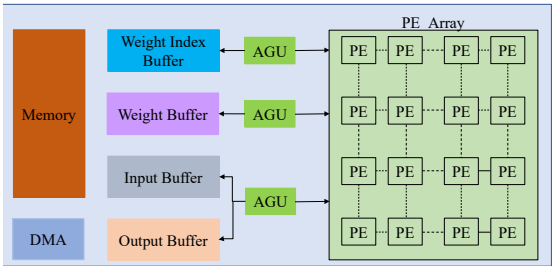

Fig. 2: NPU architecture.

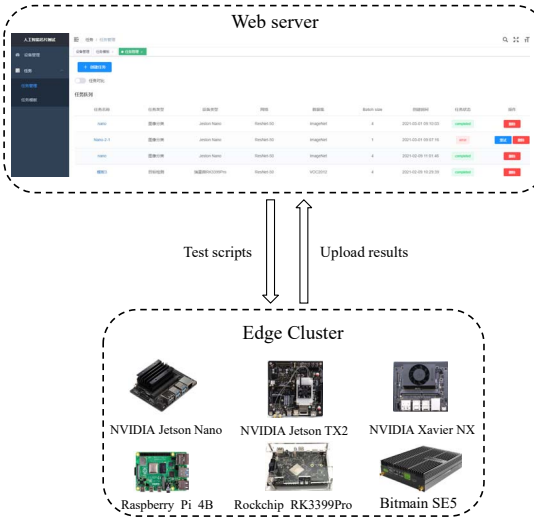

Fig. 3: Test platform.

B. Experimental System Design

The purpose of our experiment is to compare existing edge AI devices with traditional edge devices (such as Raspberry Pi without dedicated hardware accelerators) in terms of throughput, power consumption ratio, and cost-effectiveness when deploying different deep learning workloads. We built an edge AI testing platform and propose a set of benchmarks to investigate the system-level performance.

As shown in Fig.3, the test platform consists of two parts: edge device clusters and Web servers. The edge device cluster is used as the test-bed, which consists of six edge devices, including Raspberry Pi 4B, Nvidia Jetson Nano, Nvidia Jetson TX2, Nvidia Xavier NX, Rockchip RK3399Pro, and Bitmain SE5. We developed a test result display software on the Web server, and the test commands can be easily sent out to edge device remotely by scripts.

Table I lists the detailed parameters of various edge devices. Specifically, the Raspberry Pi 4B is used as a traditional edge device, and its test results are used as a baseline to compare with other edge AI devices.

TABLE I: Performance parameters of edge devices

Edge Devices	Computing Capabilities	Cost	Memory	Power
Raspberry Pi 4B	-	$55	4GB	3.5/7.5W
Nvidia Jetson Nano	0.47TFLOPS	$99	4GB	5/10W
Nvidia Jetson TX2	1.33TFLOPS	$249	8GB	7.5/15W
Nvidia Xavier NX	21TOPS	$399	8GB	10/15W
Rockchip RK3399Pro	3TOPS	$299	6GB	8W
Bitmain SE5	17.6TOPS	$937	12GB	20W

C. Workloads

The workload we used includes three categories: object detection, image classification, and machine question answering. It covers image processing and natural language processing. Table II lists the three models and datasets used in our experiment. All the above models are pre-trained, and the input is cut to the corresponding size.

Object Detection: Object detection is more difficult than image classification. It needs to determine the position of all objects in the image, and then assign a category label to each of them. The best comprehensive object detection model belongs to the YOLO series. Here we use YOLO-v3 [20] and the most popular object detection dataset VOC2012 [21]. To save test time, we randomly select 500 images in the validation set.

Image Classification: Image classification labels each image to indicate which category the object in the image belongs to. We use the most representative ResNet-50 [22] network and ImageNet [23] dataset. Because the dataset is too large, we randomly selected 5000 images from it for testing.

Machine Question Answering: Machine question answering is a very important task in natural language processing. It outputs the corresponding answers according to the given questions. Because the BERT-base model is too large to run on our edge nodes, we use a smaller version, that is, TinyBERT [24]. We also adopted the SQuAD1.1 [25] dataset, the one often used in intelligent Q&A tasks.

D. Metrics

To evaluate the performance of edge devices equipped with different AI chips, throughput, power consumption ratios, and cost-effectiveness are selected as indicators here.

Throughput: Throughput refers to the number of images or texts inferred per unit time between the loading of the model and the completion of the inference under the input of different batch sizes by the edge device. The purpose of using batch processing is to fully tap the parallel processing potential of AI chips and maximize resource utilization.

TABLE II: Benchmarking workloads

Workload	Model name	Input Type	Model size	Dataset	Dataset size
Object Detection	YOLO-v3	Image	98MB	VOC2012	500
Image Classification	ResNet-50	Image	236MB	ImageNet	4000
Question Answering	Tiny-BERT	Text	17MB	SQuAD1.1	12000

Power consumption ratios: We use the USB power meter to test the power consumption of the edge device during the inference period. The power consumption ratio is defined here to measure the power consumption of the edge device under different batch sizes. Power consumption ratio is defined as the inference throughput per unit power.

Cost-effectiveness: The cost of edge devices is also not negligible. Generally speaking, the higher the cost, the more abundant computing resources the edge device has, but the cost-effectiveness is not necessarily the highest. We defined the throughput per unit price as an indicator to fairly compare the cost-effectiveness for different edge devices.

IV. Performance Evaluation and Result Analysis

To fairly compare the throughput, power consumption ratios, and cost-effectiveness indicators for edge devices, we run a unified workload under the same framework and use the same data set for inference. Each workload is repeatedly measured 5 times under different batch sizes. Specifically, we set the batch size to increase from 1 to 128. To alleviate the bias, the maximum and minimum values are removed, and then the average value is adopted as the test result.

A. Throughput

1) Object detection workload test: In the object detection workload test, we take use of YOLO-v3 model based on the Darknet framework to benchmark edge devices equipped with AI chips, and 500 pictures in VOC2012 are taken as the test set. Fig.4 presents the average throughput under different batch sizes, and we have the following observations:

Traditional edge devices are not able to support delay-sensitive workloads. Raspberry Pi without a neural network processing unit can only reason about a single picture when the batch size is 1 and does not have parallel processing capabilities. Because it only uses CPU as the computing unit, which results in very low throughput and no support for real-time needs. Therefore, it is not recommended to deploy delay-sensitive tasks on traditional edge devices. On the other hand, edge devices equipped with dedicated neural network processing units have significant performance improvements. Specifically, even when the batch size is 1, the average throughput is increased by 69 to 953×.

There is no direct linear relationship between computing capabilities and throughput. Since the Raspberry Pi does not have batch processing capabilities, here we use Nano with the smallest computing resources as the baseline. We can observe that compared to the baseline, the average throughput of other edge devices under batch processing has increased by 2.4 to 13.8×. Surprisingly, the throughput of various edge AI devices under batch processing has no obvious linear relationship with the computing capabilities, as shown in Table I. We think there are two reasons for such results: 1) The peak computing capability of the AI chip does not mean the same effectiveness of the computing capabilities in real scenarios; 2) The hardware structure, relative software tools, and ecosystem, and algorithm optimization will deeply affect the performance even for the same deep learning workloads. Compared with NVIDIA that has classic GPUs, the edge devices such as RK3399Pro and SE5 which are equipped with ASIC-based dedicated neural network processing units have shown more powerful processing ability. This is because AI chip designers customize the inference engine, which converts the native model into a model format matching the specific AI chip design to optimize the inference performance.

Batch size has little impact on the throughput for edge devices. The reason is that object detection requires high memory and computing resources. But in most cases, edge devices are fully loaded even when the batch size is 1. Therefore, increasing the batch size will not lead to higher throughput. This also works for SE5 which has more memory and computing resources than others: the throughput will not increase after initial slow growth, as batch size increases.

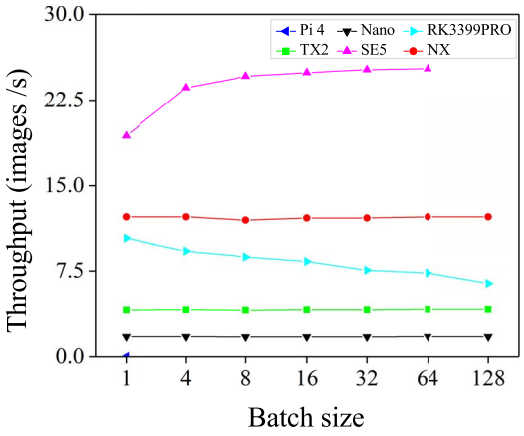

Fig. 4: Throughput of YOLO-v3 workload.

2) Image classification test: In the image classification workload benchmark test, we choose the ResNet-50 model, based on the Tensorflow framework, with 4000 images in ImageNet as the test set. Thanks to the residual module,

ResNet-50 dramatically reduces the number of parameters and thus leads to about one-third of the YOLO-v3 model scale, or YOLO-v3 has 2.7× the number of network layers. Fig.5 illustrates the average throughput for different batch sizes under ResNet-50. The observations are as follows.

The Raspberry Pi can perform ResNet-50 inference. Because it is a lighter model, it works on Raspberry Pi. However, it has no dedicated neural network processing unit, the inference throughput is the lowest among all edge devices, and it does not have real-time inference capability. This follows the same reason as in the previous discussion.

All edge devices equipped with AI chips outperform traditional edge devices in image classification workloads. Specifically, SE5 performed the best, and the average throughput under different batch size is 83× that of Raspberry Pi. The average throughput of Nano, which has the smallest computing resources, is 4.5× that of Raspberry Pi. The results further demonstrate the effectiveness of edge AI devices for reasoning on deep learning workloads.

Batch size does not have a significant influence. Taking SE5 as an example, the throughput reaches its peak when the batch size is 32, and then the throughput decreases sharply with the increase of batch size. As discussed before, in addition to the computing ability, memory also matters. The mismatch of the computing capability and memory constraint may lead to performance degradation, because of the long queuing delay.

In conclusion, edge devices equipped with dedicated neural network processing units have better performance than general-purpose GPU devices. When the average throughput is the same, the computing capabilities of RK3399Pro is only one-seventh of NX. Under the same computing capabilities, the average throughput of SE5 is 4.5× that of NX. All these results demonstrate the advantages of ASIC-based dedicated neural network processing units.

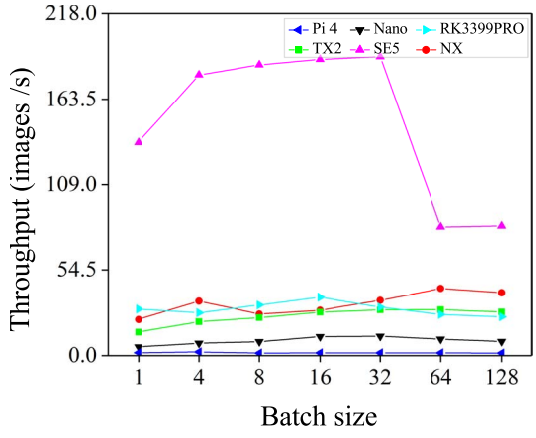

Fig. 5: Throughput of ResNet-50 workload.

3) Question answering workloads test: In machine question answering workloads, pre-trained models such as BERT usually require high computing capabilities and cannot be inferred on resource-constrained edge devices. Therefore, we use TinyBERT for edge device inference workloads and use SQuAD1 which contains 12,000 intelligent question and answer texts. We test the performance of edge devices under different batch sizes. The following results are summarized from the test in Fig.6:

There is no significant performance improvement for edge AI devices. The average throughput of edge AI devices is 1.2× to 3.5× that of Raspberry Pi. The reason for this result is that most of the existing edge AI devices are oriented to the field of object detection and image classification, and a large number of convolution operations in the model network structure are optimized, but they do not support natural language processing models such as machine question answering well, which is difficult to perform efficient reasoning on resource-constrained edge devices.

Edge devices with more computing capabilities do not lead to higher throughput. From the Fig.6, it can obviously observe that the NVIDIA series of edge devices perform the best. Even the Nano with the smallest computing capabilities has 1.5× the throughput of SE5, although the former has only one-fifth of the latter's computing capabilities. This is because NVIDIA's general-purpose GPU has a relatively high degree of algorithm support and can well support common algorithms in various fields, even natural language processing. In contrast, dedicated AI chips are only for application scenarios in specific fields. For example, RK3399Pro and SE5 are only for target detection and image classification and do not support models in the field of natural language processing. This also shows from the side that the existing dedicated AI chips lack versatility, and how to build dedicated AI chips that can support natural language processing models is a promising direction in the future.

Batch size has little effect. Like the previous benchmark tests for the two image processing workloads, in the natural language processing model, the increase in batch size has a very limited impact on throughput changes. The throughput of most edge devices is saturated when the batch size is 4. The reason for this phenomenon is also because the size of the model exceeds the limited memory and computing resources of the edge device.

B. Power consumption ratio

Fig.7(a)-(c) plots the power consumption ratios for the tested edge devices under different batch sizes for the three workloads. The metric we use here is the *inference throughput per joule*, which means the larger result of the power consumption ratio, the better performance of the edge device. The USB power meter is set up to measure the power consumption for all edge devices. In addition, all edge devices turn on the maximum power mode by default and turn off the graphical interface to save memory consumption.

Fig.7(a) presents the results for the YOLO-v3 workload. The performance of the average power consumption ratio is RK3399Pro>NX>SE5>TX2>Nano>Raspberry Pi. The results indicate that the dedicated AI chips outperform gen-

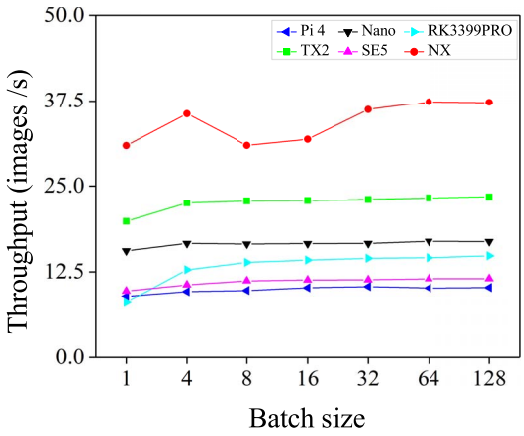

Fig. 6: Throughput of Tiny-BERT workload.

eral GPU devices and traditional edge devices. Specifically, RK3399Pro has the best value, and the number of inferred pictures per joule is 1.5, which is 144× that of Raspberry Pi. Even the Nano, the edge AI device with the smallest computing power, has a power consumption ratio of 48× that of the Raspberry Pi. Not surprisingly, the Raspberry Pi has the lowest power consumption ratio. Although the power consumption is the lowest of all devices, Raspberry Pi does not have an advantage in the power consumption ratio because it has no dedicated neural network processing unit. The other interesting observation is the batch size has little impact on the power consumption ratio. The reason is the same as before, i.e., even when the batch size is relatively small, the memory has become saturated, which seriously affects the batch processing capabilities of edge devices.

The results for ResNet-50 are shown in Fig.7(b). The average power consumption ratio from high to low is SE5>RK3399Pro>NX>TX2>Nano>Raspberry Pi. Among them, SE5, which has the largest average power consumption ratio, is 13.4× that of Raspberry Pi. Nano, which has the smallest power consumption ratio among edge AI devices, is also 5.6× the latter. This also demonstrates the powerful processing and energy-saving abilities of dedicated AI chips. This time, the power consumption ratio changes significantly with the increase of batch size. This is because the model structure of ResNet-50 is simpler than YOLO-v3, so it requires less memory and computing resources, and thus the batch size will affect the performance.

Fig.7(c) illustrates the results for Tiny-BERT workloads under different batch sizes. Unlike the previous two image processing workloads, the performance is NX>TX2>Nano>RK3399Pro>Raspberry Pi>SE5. The edge devices of the NVIDIA series equipped with general-purpose GPU perform the best, and the power consumption ratio is much higher than that of edge devices equipped with dedicated AI chips. Specifically, the average power consumption ratio of NX is 8.7× and 14.6× that of 3399Pro and SE5, respectively. What surprised us more is that the power consumption ratio of the SE5 with the highest computing power is only half that of the Raspberry Pi. The reason is: edge devices equipped with dedicated AI chips do not support workloads in the field of natural language processing, resulting in performance similar to ordinary edge devices.

C. Cost

Generally speaking, the cost of edge devices is directly proportional to the computing capability. However, unlike traditional PC system that has only one common architecture, in the emerging new edge AI field, different kind of specific accelerators or architectures for different AI model optimization are designed. That's why although in traditional system performance evaluation metrics, the cost is rarely considered. Here, we think it is an important metric for AI-needed applications. Similar to the power benchmark test, to fairly compare the advantages and disadvantages of the edge devices in terms of cost, we normalize the cost and use the throughput per unit price as the measurement metric. Fig.8(a)-(c) presents three types of cost-effectiveness for edge devices under different batch sizes.

Fig.8(a) shows the results under the YOLO-v3 workload. The order of cost-effectiveness from high to low is RK3399Pro>SE5>NX>Nano>TX2>Raspberry Pi. RK3399Pro has the best performance in terms of cost-effectiveness, i.e., 2.7× to 148× that of other devices; SE5 is the second. The results indicate that the dedicated AI chips are extremely cost-effective. We also notice that the change in batch size has almost no impact on the performance.

For ResNet-50, the performance results in Fig.8(b) keep the same order as for YOLO-v3, This is because the network model structures of these two workloads are similar, and both contain a large number of convolution operations, which can essentially be converted to general matrix multiplication (GEMM). The main difference is that the complexity of the model and the required computing power are different. This is why the performance will decrease with the increase of batch size, since that the memory and computing resources of edge devices will reach saturation at some point. When it reaches saturation, continuing to increase the batch size at this time will cause additional waiting delays. Here, the cost-effectiveness of NVIDIA edge devices with general-purpose GPUs is still not as good as RK3399Pro and SE5 that integrate dedicated AI chips.

Interestingly, as shown for Tiny-BERT in Fig.8(c), Nano's cost-effectiveness is the best, followed by Raspberry Pi; and there is no direct relationship between rich computing resources and the cost-effectiveness for the other four edge devices. Specifically, SE5 has the highest unit price but performs the worst. RK3399pro, NX and TX2 seem also to have a similar situation. We attempt to explain from the following two features: unit price and algorithm support. On the one hand, although Nano and Raspberry Pi do not have rich computing resources compared with other edge devices,

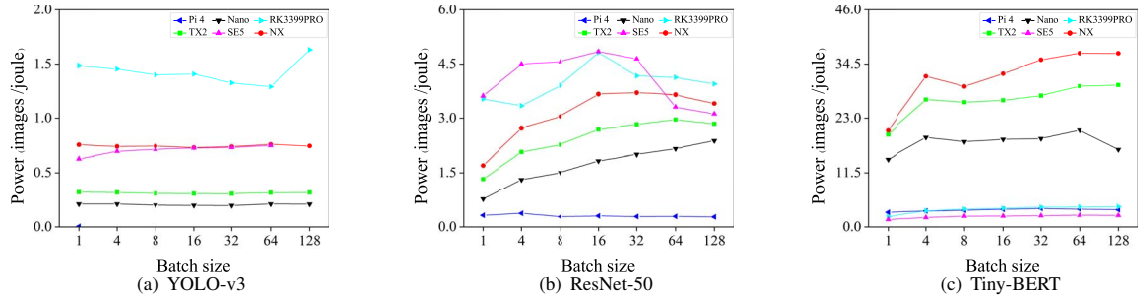

Fig. 7: Power consumption ratio of different workloads.

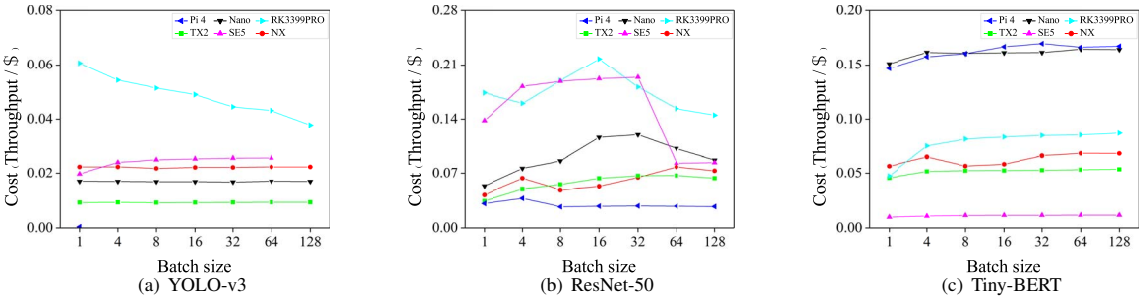

Fig. 8: Cost-effectiveness of different workloads.

their low price makes up for the shortcomings of insufficient computing power and can make a good trade-off between unit price and performance. On the contrary, the unit price of RK3399Pro and SE5 with dedicated AI chips is expensive, 5.5× to 17× that of Raspberry Pi. More importantly, the algorithm support of these two devices is relatively narrow:- currently only supports workloads in the image processing field, therefore, it has no advantage in the cost-effectiveness test for the natural language processing workloads.

V. Recommendations

From the test results we can observed that traditional deep learning workloads have a large amount of calculation due to the complexity of the model structure, and deployment to resource-constrained edge devices will cause high latency and power consumption. In order to enable edge devices to deploy real-time inference deep learning workloads, we give suggestions from the perspective of edge computing and model lightweighting.

A. Edge Computing

Combining Edge AI [26] with edge computing is a feasible method. Edge computing [27], as an extension of cloud computing, deploys edge servers on the side close to the source of data, and meets the high computing and low latency requirements of deep learning workloads on edge devices through "cloud-edge collaboration" with cloud servers to improves User's QoS. Even if complex deep learning workloads are deployed on edge devices, we can use edge computing to offload local models to cloud servers, use the powerful computing capabilities of cloud servers to complete real-time inferences and return the results to edge devices, while greatly reducing power consumption of the edge device.

B. Model Lightweighting

Model compression and acceleration [28] [29] [30] are two different topics. The former focuses on reducing the amount of network parameters, while the latter focuses on reducing the computational complexity and improving the parallel ability. We think the model compression and acceleration can be divided into three levels:

- **Algorithm layer compression acceleration:** This dimension is mainly in the algorithm application layer, which is also the work scope of most Algorithm Engineers. It mainly includes structure optimization, model quantification and model distillation.
- **Frame layer acceleration:** This dimension is mainly in the algorithm framework layer. For example, TF-Lite, NCNN, MNN and so on. It mainly includes compiler optimization, cache optimization, sparse storage and computation, NEON instruction application, operator optimization and so on.

- **Hardware layer acceleration:** This dimension is mainly in the AI hardware chip layer. At present, there are GPU, FPGA, ASIC and other solutions, and various TPU and NPU are ASIC solutions. Through chip customization for deep learning, the running speed of the model is greatly accelerated.

VI. Conclusions

In this work, we build a testing platform for systematically evaluate the AI-enabled edge devices. We choose six mainstream edge AI devices that are off-the-shelf in the market for the benchmarking purpose. Experimental results show that under the workload of object detection and image classification, compared with traditional edge devices, edge devices equipped with AI chips can dramatically improve the performance, e.g., throughput, power consumption ratio, and cost efficiency. Unfortunately, existing edge AI devices have not yet provided enough support for natural language processing workloads:- the performance improvement is limited. These results can be used to direct our future research, such as how to design the edge-cloud AI binding system architecture to optimize the performance for different applications, or where the edge AI should go, generic or specific? We leave those thoughts as our future work.

Acknowledgment

This work is supported in part by China Academy of Industrial Internet under Grant RQQQ2394000121 and Heilongjiang Provincial Department of Science and Technology under Grant KMQQ2440400120.

References

[1] L. Zeng, B. Benatallah, A. H. Ngu, M. Dumas, J. Kalagnanam, and H. Chang, "Qos-aware middleware for web services composition," *IEEE Transactions on software engineering*, vol. 30, no. 5, pp. 311–327, 2004.
[2] "Tensorrt," https://github.com/NVIDIA/TensorRT.
[3] "Tensorflow-lite," https://tensorflow.google.cn/lite.
[4] "Onnx," https://github.com/onnx/onnx.
[5] "Openvino," https://docs.openvinotoolkit.org.
[6] W. Gao, C. Luo, L. Wang, X. Xiong, J. Chen, T. Hao, Z. Jiang, F. Fan, M. Du, Y. Huang *et al.*, "Aibench: towards scalable and comprehensive datacenter ai benchmarking," in *International Symposium on Benchmarking, Measuring and Optimization*. Springer, 2018, pp. 3–9.
[7] "Mlperf," https://mlcommons.org/zh/.
[8] "Aiia dnn benchmark," http://www.aiiaorg.cn/benchmark.
[9] H. Zhu, M. Akrout, B. Zheng, A. Pelegris, A. Phanishayee, B. Schroeder, and G. Pekhimenko, "Tbd: Benchmarking and analyzing deep neural network training," *arXiv preprint arXiv:1803.06905*, 2018.
[10] C. Coleman, D. Narayanan, D. Kang, T. Zhao, J. Zhang, L. Nardi, P. Bailis, K. Olukotun, C. Ré, and M. Zaharia, "Dawnbench: An end-to-end deep learning benchmark and competition," *Training*, vol. 100, no. 101, p. 102, 2017.
[11] S. Dong and D. Kaeli, "Dnnmark: A deep neural network benchmark suite for gpus," in *Proceedings of the General Purpose GPUs*, 2017, pp. 63–72.
[12] C. Luo, F. Zhang, C. Huang, X. Xiong, J. Chen, L. Wang, W. Gao, H. Ye, T. Wu, R. Zhou *et al.*, "Aiot bench: towards comprehensive benchmarking mobile and embedded device intelligence," in *International Symposium on Benchmarking, Measuring and Optimization*. Springer, 2018, pp. 31–35.
[13] "Aiot bench," http://www.benchcouncil.org/AIoTBench.
[14] T. Hao, Y. Huang, X. Wen, W. Gao, F. Zhang, C. Zheng, L. Wang, H. Ye, K. Hwang, Z. Ren *et al.*, "Edge aibench: towards comprehensive end-to-end edge computing benchmarking," in *International Symposium on Benchmarking, Measuring and Optimization*. Springer, 2018, pp. 23–30.
[15] "Edge ai bench," http://www.benchcouncil.org/EdgeAIBench.
[16] A. Das, S. Patterson, and M. Wittie, "Edgebench: Benchmarking edge computing platforms," in *2018 IEEE/ACM International Conference on Utility and Cloud Computing Companion (UCC Companion)*. IEEE, 2018, pp. 175–180.
[17] Q. Yang, R. Jin, N. Gandhi, X. Ge, H. A. Khouzani, and M. Zhao, "Edgebench: A workflow-based benchmark for edge computing," *arXiv preprint arXiv:2010.14027*, 2020.
[18] N. P. Jouppi, C. Young, N. Patil, D. Patterson, G. Agrawal, R. Bajwa, S. Bates, S. Bhatia, N. Boden, A. Borchers *et al.*, "In-datacenter performance analysis of a tensor processing unit," in *Proceedings of the 44th annual international symposium on computer architecture*, 2017, pp. 1–12.
[19] H.-T. Kung, "Why systolic architectures?" *Computer*, vol. 15, no. 01, pp. 37–46, 1982.
[20] J. Redmon, S. Divvala, R. Girshick, and A. Farhadi, "You only look once: Unified, real-time object detection," in *Proceedings of the IEEE conference on computer vision and pattern recognition*, 2016, pp. 779–788.
[21] M. Everingham, S. A. Eslami, L. Van Gool, C. K. Williams, J. Winn, and A. Zisserman, "The pascal visual object classes challenge: A retrospective," *International journal of computer vision*, vol. 111, no. 1, pp. 98–136, 2015.
[22] K. He, X. Zhang, S. Ren, and J. Sun, "Deep residual learning for image recognition," in *Proceedings of the IEEE conference on computer vision and pattern recognition*, 2016, pp. 770–778.
[23] A. Krizhevsky, I. Sutskever, and G. E. Hinton, "Imagenet classification with deep convolutional neural networks," *Advances in neural information processing systems*, vol. 25, pp. 1097–1105, 2012.
[24] X. Jiao, Y. Yin, L. Shang, X. Jiang, X. Chen, L. Li, F. Wang, and Q. Liu, "Tinybert: Distilling bert for natural language understanding," *arXiv preprint arXiv:1909.10351*, 2019.
[25] P. Rajpurkar, R. Jia, and P. Liang, "Know what you don't know: Unanswerable questions for squad," *arXiv preprint arXiv:1806.03822*, 2018.
[26] E. Li, L. Zeng, Z. Zhou, and X. Chen, "Edge ai: On-demand accelerating deep neural network inference via edge computing," *IEEE Transactions on Wireless Communications*, vol. 19, no. 1, pp. 447–457, 2019.
[27] W. Shi, J. Cao, Q. Zhang, Y. Li, and L. Xu, "Edge computing: Vision and challenges," *IEEE internet of things journal*, vol. 3, no. 5, pp. 637–646, 2016.
[28] Y. Cheng, D. Wang, P. Zhou, and T. Zhang, "A survey of model compression and acceleration for deep neural networks," *arXiv preprint arXiv:1710.09282*, 2017.
[29] T. Choudhary, V. Mishra, A. Goswami, and J. Sarangapani, "A comprehensive survey on model compression and acceleration," *Artificial Intelligence Review*, vol. 53, no. 7, pp. 5113–5155, 2020.
[30] Y. Cheng, D. Wang, P. Zhou, and T. Zhang, "Model compression and acceleration for deep neural networks: The principles, progress, and challenges," *IEEE Signal Processing Magazine*, vol. 35, no. 1, pp. 126–136, 2018.