Stony Brook University

Evaluating the energy impact of device parameters for DNN inference on edge

Anurag Dutt, *Sri Pramodh Rachuri, Ashley Lobo, Nazeer Shaik, Anshul Gandhi, Zhenhua Liu*

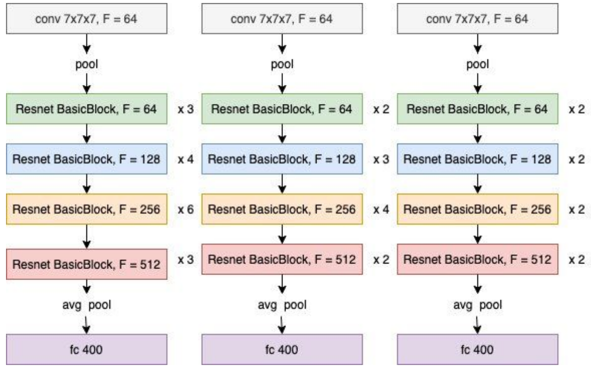
FAR BEYOND

Stony Brook University

Chat GPT Google Bard

Motivation

- Deployment of large DNN models
- Edge Computing
    - Examples - Jetson lineup
    - Battery-Operated
    - Deployed in resource scarce environments
- Large parameter space to optimize
    - Hardware parameters



FAR BEYOND

Introduction

- Sustainable DNN workload deployments on the Edge
- Study the impact of **hardware** parameters
    - CPU frequency
    - GPU frequency

- Compare the performance of multiple Deep Learning workloads on default device configuration vis-a-vis adjusting CPU and GPU frequencies for optimal energy usage

Stony Brook University

Experimental Setup

- Power readings for each edge device are polled at 100ms intervals
    - Overhead for 100 ms < 0.5%; Overhead for more frequent polling (10 ms or 1 ms) > 2%
- PyTorch for all the workloads except for YOLOv4 (OpenCV)
- I2C interface is polled in a separate thread for more granular readings
- Each experiment on a given model
    - One out of x CPU+GPU Freq combinations
    - Fixed workload - 3200 inferences inputs
    - 10 reruns; variance was less than 5%

Experimental Setup - Device Specifications

Specification	Jetson Nano	Jetson Xavier NX
CPU	4-core ARM A57	8-core Nvidia Carmel
CPU Freq. range	102 MHz – 1.48 GHz	115 MHz – 1.9 GHz
CPU Freq. step	100 MHz (15 steps)	77 MHz (25 steps)
GPU	Nvidia Maxwell	NVIDIA Volta
CUDA Cores	128	384
Tensor Cores	-	48
Memory	4 GB LPDDR4	8 GB LPDDR4
GPU Freq. range	76 MHz – 921 MHz	114 MHz – 1.1 GHz
GPU Freq. steps	77 MHz (count 12)	90 MHz (count 15)
Throughput	472 GFLOPs	21 TOPs
Power Modes	5W, 10W	10W, 15W
Libraries	CUDA 10.2 + cuDNN 8.2.1	CUDA 10.2 + cuDNN 8.0.0

Stony Brook University

Experimental Setup - Workloads

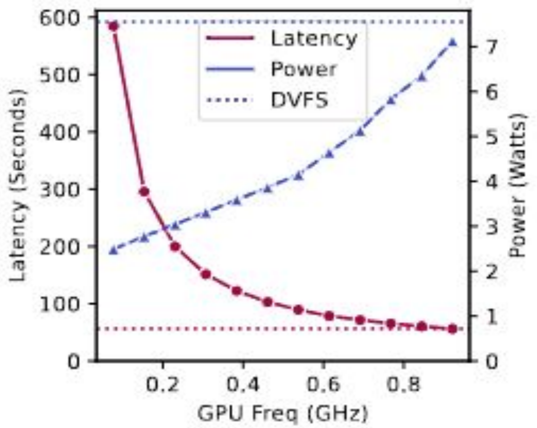
- The workloads chosen span across 3 different categories of Deep Learning use-cases:
    - Image Classification (AlexNet, ResNet-18, MobileNet)
    - Object Detection (YOLOv4 - Tiny)
    - Natural Language Classification (BERT-Tiny, distilBERT)
- MobileNet, BERT-Tiny, DistilBERT, and YOLOv4-Tiny are tailored for edge devices, characterized by their lightweight architecture and efficient performance.

Evaluation - Frequency Sweeps on Nano

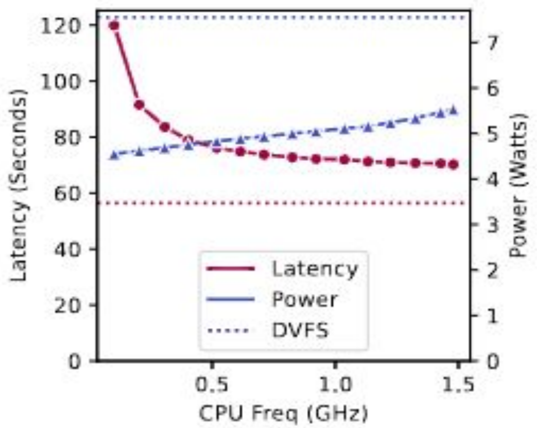
- DVFS Governor
  - CPU Default - “schedutil”
  - GPU Default - “nvhost_podgov”
  - Highest freq - CPU 89%; GPU 83%
  - Other governors < 1% variation

Monotonic relation with freq

Impact of CPU Freq < GPU Freq

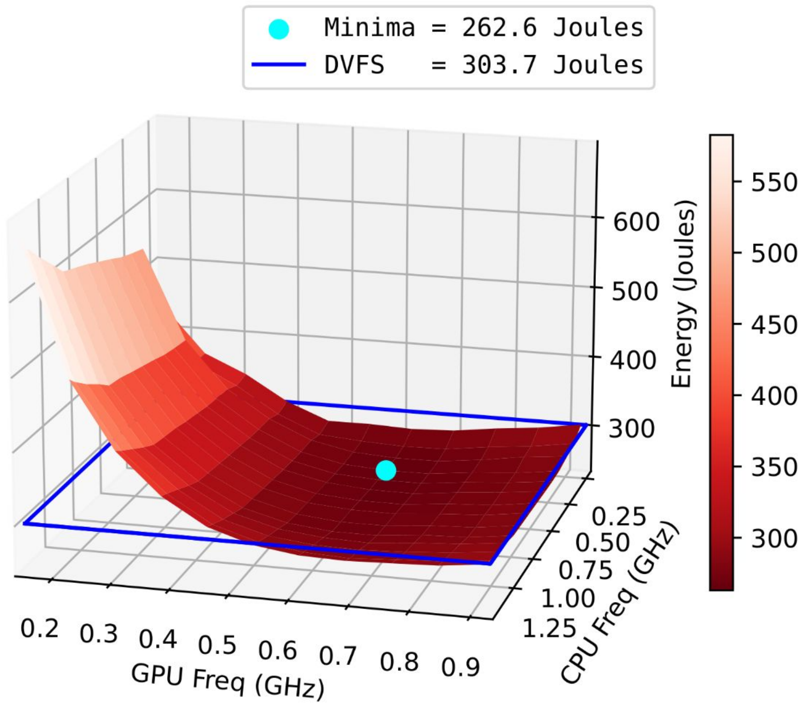


(a) Changing GPU frequency



(b) Changing CPU frequency

Evaluation - Energy usage trends on Nano

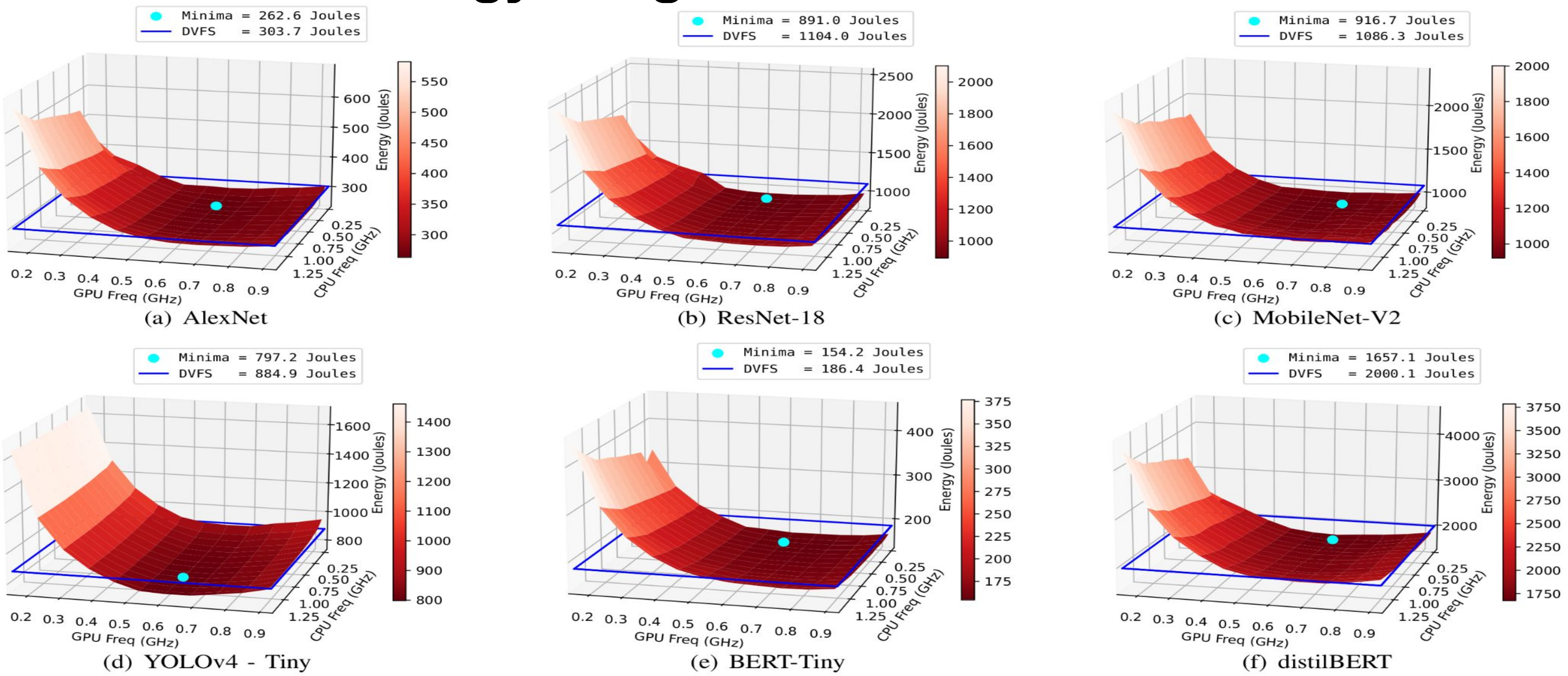


(a) AlexNet

GPU Freq substantially impacts Energy but not monotonic

Minima consumes 13% lower than DVFS

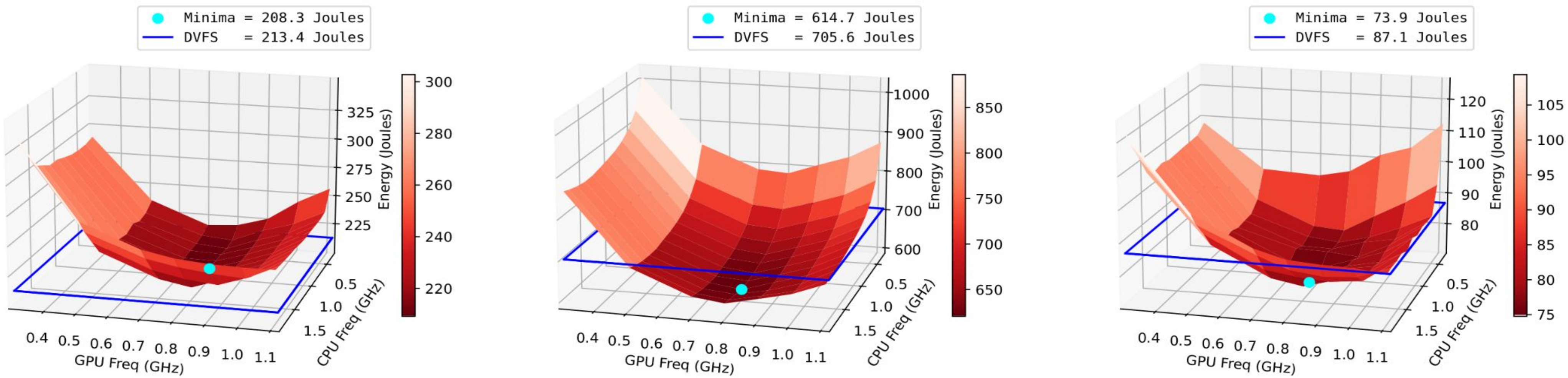
Evaluation - Energy usage trends on Nano



Minima consumes 13%, 19%, 15%, 9%, 17%, 17% lower energy than DVFS

Stony Brook University

Evaluation - Energy usage trends on Xavier NX



(a) ResNet-18 (b) YOLOv4-Tiny (c) BERT-Tiny

Not monotonicity of GPU and CPU Freq is more prominent

Minima consumes 2%, 13%, 15% lower energy than DVFS

FAR BEYOND

IMPACT OF OTHER FACTORS:

- Compared to LibTorch implementation in C++, python implementation in PyTorch consumes 16% more energy due to python overheads
- However, LibTorch is not commonly used during prototyping due to ease of usage of Python and compilation overheads
- LibTorch is commonly used in high performance systems due to its energy and latency benefits
- Impact of turning off lazy-loading and garbage collection were found to to 1.82% and 1.65% reduction in total energy, respectively

Conclusion

- Selecting optimal freq configuration gives upto 19% savings in energy for Jetson Nano as compared to DVFS
- Selecting optimal freq configuration gives upto 15% savings in energy for Xavier NX as compared to DVFS
- Energy savings at the are not free!!! Latency trade-offs to the tune of 28% - 35% are observed

THANK YOU

Q&A

Experimental Setup

- Power readings for each device are polled at 100ms intervals
    - Overhead for 100 ms < 0.5%; Overhead for more frequent polling (10 ms or 1 ms) > 2%
- PyTorch for all the workloads except for YOLOv4 (OpenCV)
- 150 CPU/GPU Freq combinations for Nano and 375 combinations for Xavier NX
- Each experiment consisted of running a DNN inference workload with a specified batch size under a specific CPU and GPU frequency setting
- Each workload in a given experimental configuration was repeated 3200 times and each experiment was repeated 10 times
- Variance in energy readings for the all the experimental configurations was much less than 5%