# Characterizing Resource Heterogeneity in Edge Devices for Deep Learning Inferences

Jianwei Hao, Piyush Subedi, Dr. In Kee Kim, Dr. Lakshmish Ramaswamy

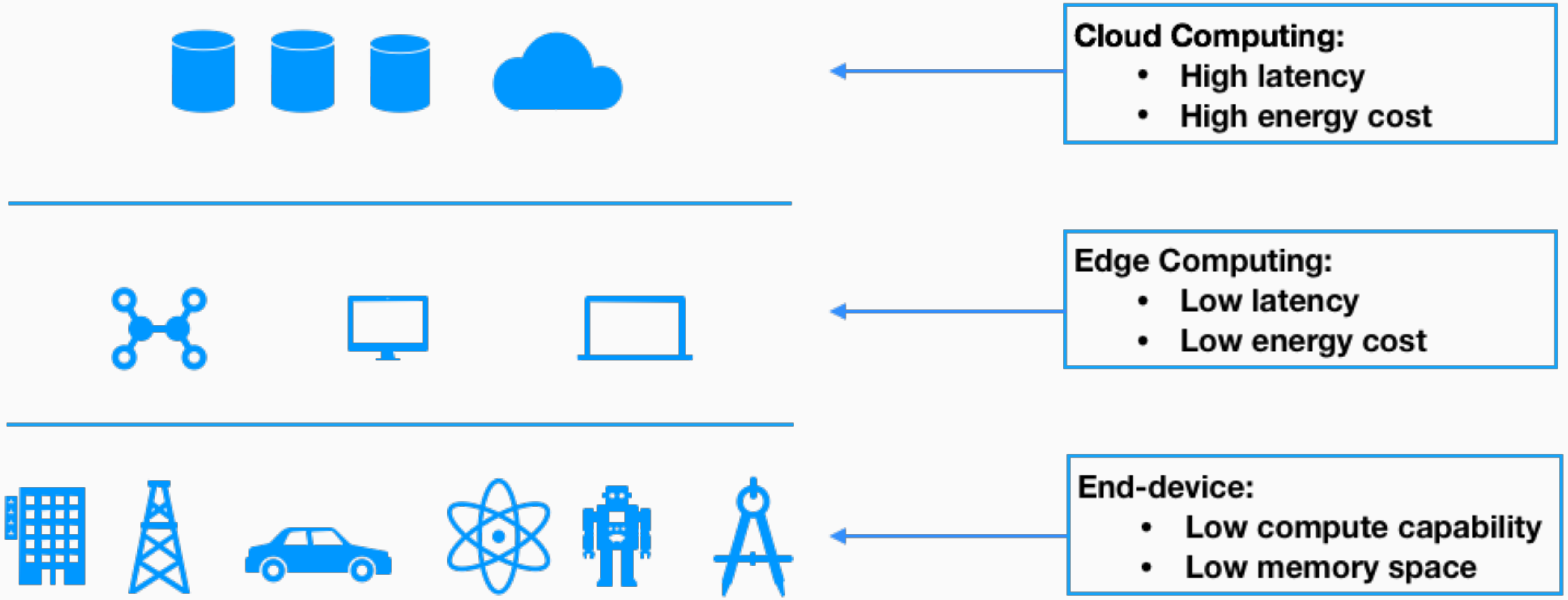
# Agenda

- Problem Statement
- Related Works
- Evaluation Setup
- Results
- Conclusion and Future Work

I

# Problem Statement

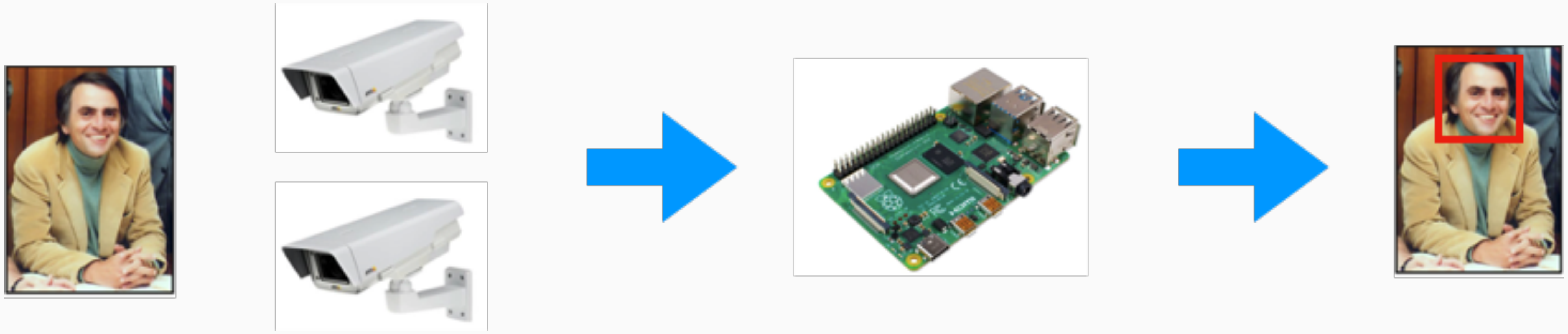
Edge Computing:



I

## Problem Statement

AI on Edge
Computing:



I

# Problem Statement

- Wide variety of DNN (Deep Neural Networks) architectures + Diverse set of heterogeneous edge devices => *Which model and device to use for my DL tasks?*
- Also, which ML Framework would be more suitable at the edge?
- How does batching input affect the performance?

II

# Related Works

| Authors | Paper | Year | Focus | Metrics | Frameworks | Models | Devices | Batching |
|---|---|---|---|---|---|---|---|---|
| Zhang et al. | pCAMP: Performance Comparison of Machine Learning Packages on the Edges | 2018 | Edge | Latency, Memory, Energy | TF, PyTorch, MXNet, Caffe2, TFLite | AlexNet, SqueezeNet | Jetson TX2, Raspberry Pi, Macbook Pro, Intel FogNode, Nexus 6P | NO |
| Antonini et al. | Resource Characterisation of Personal-Scale Sensing Models on Edge Accelerators | 2019 | Edge | Latency, Memory, Energy | TF | 5 vision based models, 2 audio based models and 1 motion based model | Jetson Nano, Raspberry Pi 4B, Coral Dev Board, Coral USB Accelerator | NO |
| Ross et al. | EdgeInsight: Characterizing and Modeling the Performance of Machine Learning Inference on the Edge and Cloud | 2019 | Edge-Cloud | Latency, Network, CPU | TFLite | MobileNetV2 quantized | OnePlus 6T | NO |
| Süzen et al. | Benchmark Analysis of Jetson TX2, Jetson Nano and Raspberry PI using Deep-CNN | 2020 | Edge | Latency, Memory, Energy | NVIDIA's cuDNN | Custom | Jetson TX2, Jetson Nano, Raspberry Pi 4 | NO |
| Jo et al. | Benchmarking GPU-Accelerated Edge Devices | 2020 | Edge-Cloud | Throughput (frames/sec) | NVIDIA's TensorRT | AlexNet, ResNet50 | Jetson Nano, Jetson TX2 | NO |

III

# Evaluation Setup

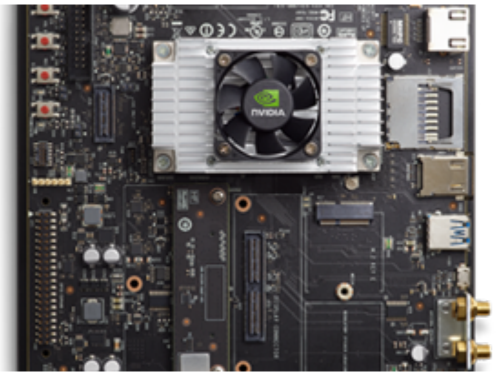
III

# Evaluation Setup -- Devices

Raspberry Pi 4B
GPU : N/A
CPU : Quad core Cortex-A72
Memory : 4 GB

Odroid N2
GPU : Mali-G52 GPU
CPU : Quad-core ARM Cortex-A73 +
Dual core Cortex-A53
Memory : 4 GB

Jetson Nano
GPU: 128-core Maxwell
CPU: Quad-core ARM A57
Memory: 4 GB



Jetson TX2
GPU: 256-core
CPU: Dual-Core NVIDIA Denver +
Quad-Core ARM® Cortex®-A57r
Memory: 8 GB

Jetson Xavier NX
GPU: 384-core
CPU: 6-core
Memory: 8 GB

III

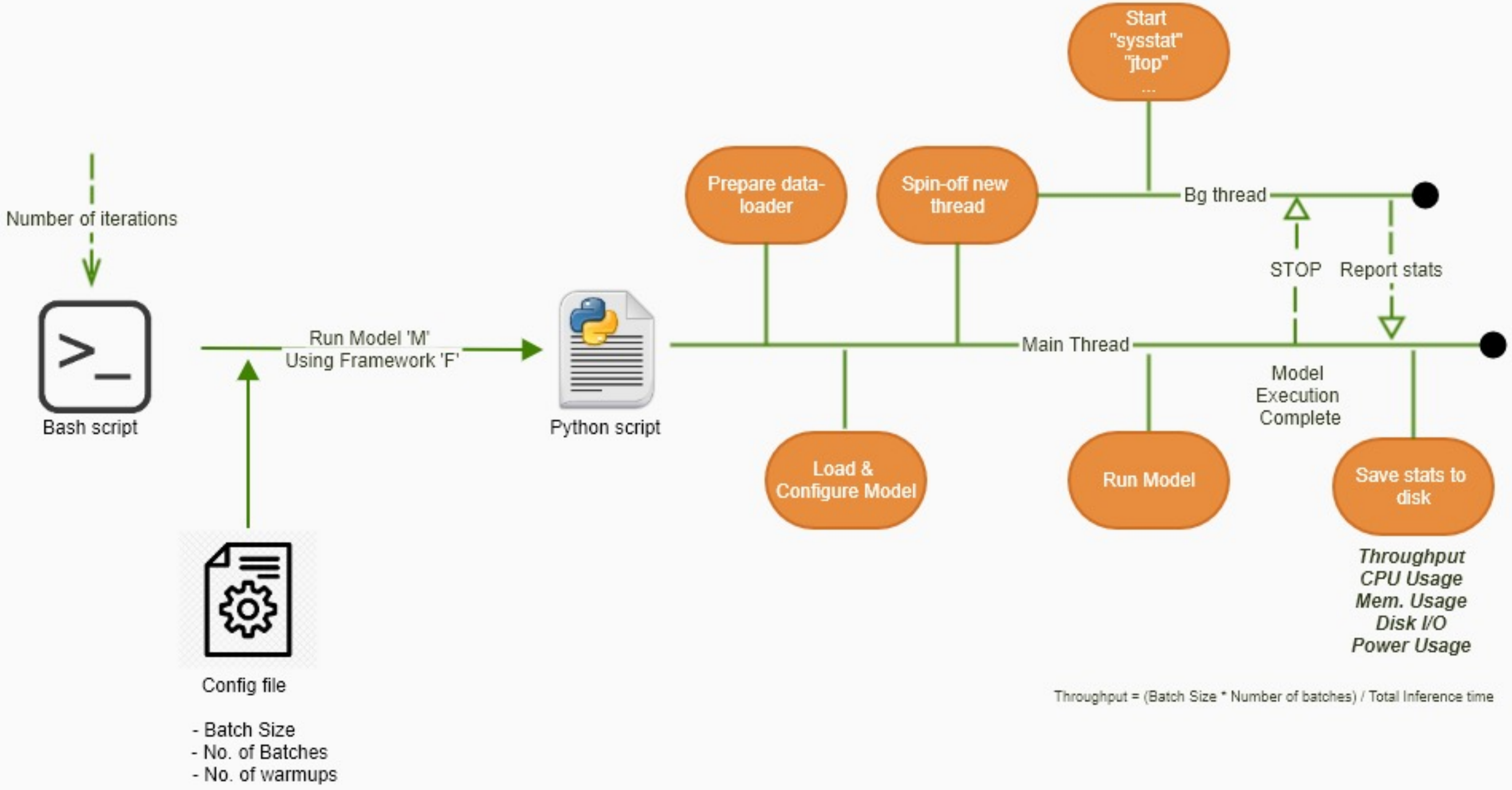
# Evaluation Setup -- DNN

mxnet TensorFlow PyTorch

| Models | Year | Input Size | Num. Layers | Billion FLOPS | # Params (Million) |
|---|---|---|---|---|---|
| AlexNet | 2012 | 224 X 224 | 8 | 0.7 | 61 |
| SqueezeNet | 2016 | 224 X 224 | 15 | 0.4 | 1.2 |
| ResNet18 | 2015 | 224 X 224 | 18 | 1.8 | 11.7 |
| ResNet50 | 2015 | 224 X 224 | 50 | 4.1 | 25.6 |
| DenseNet | 2016 | 224 X 224 | 161 | 7.9 | 28.7 |
| VGG16 | 2014 | 224 X 224 | 16 | 15.4 | 138.36 |

III

# Evaluation Setup -- Process

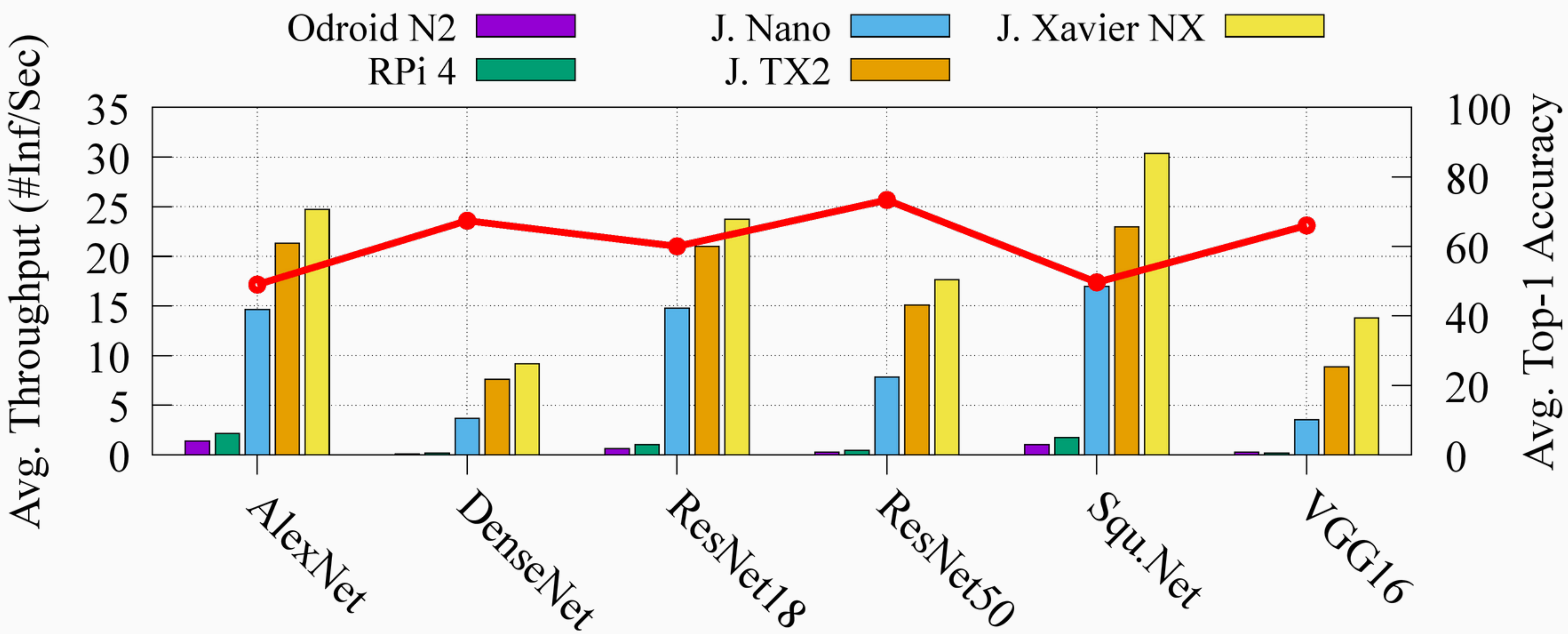


IV

# Results and Discussions

IV

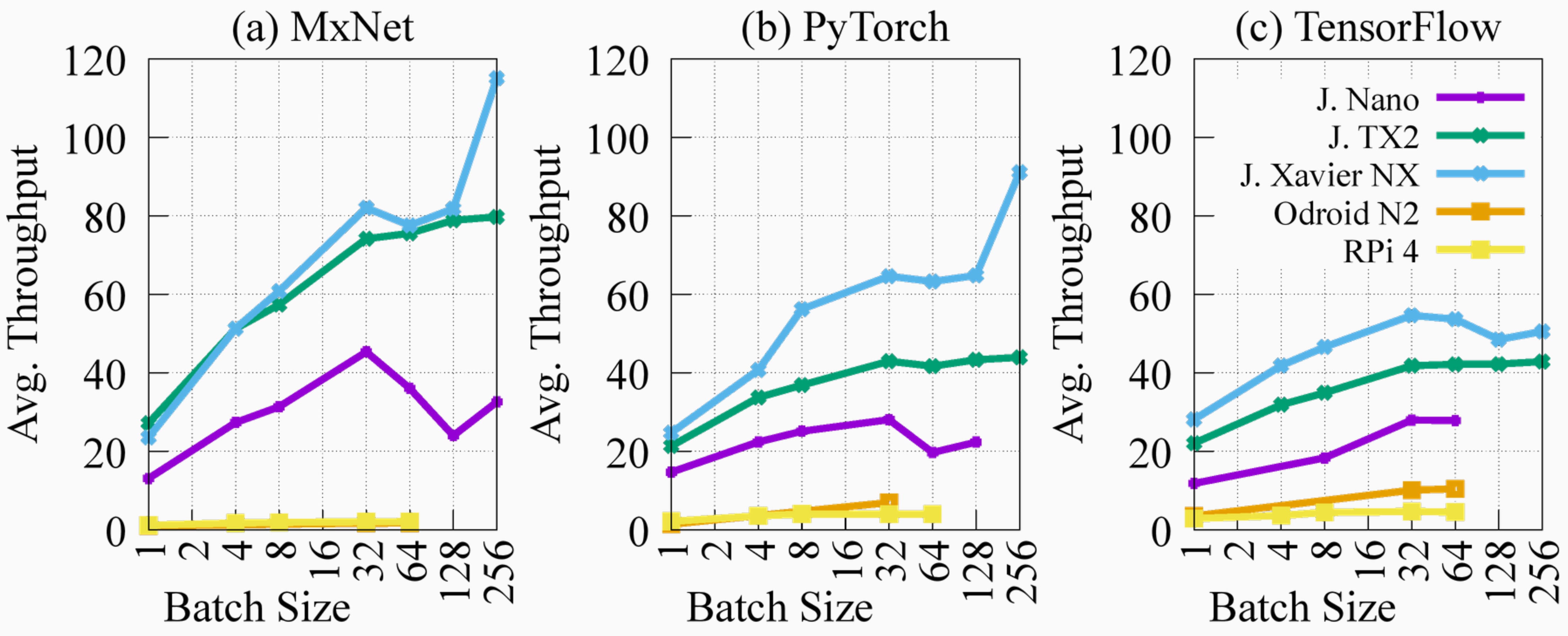
# Device Vs Throughput (Batch Size = 1)




- Results are based on the best performing framework - **PyTorch**
- GPU-device have higher throughput.
- AlexNet and SqueezeNet are fast
- ResNet-18 has decent accuracy and throughput

VI

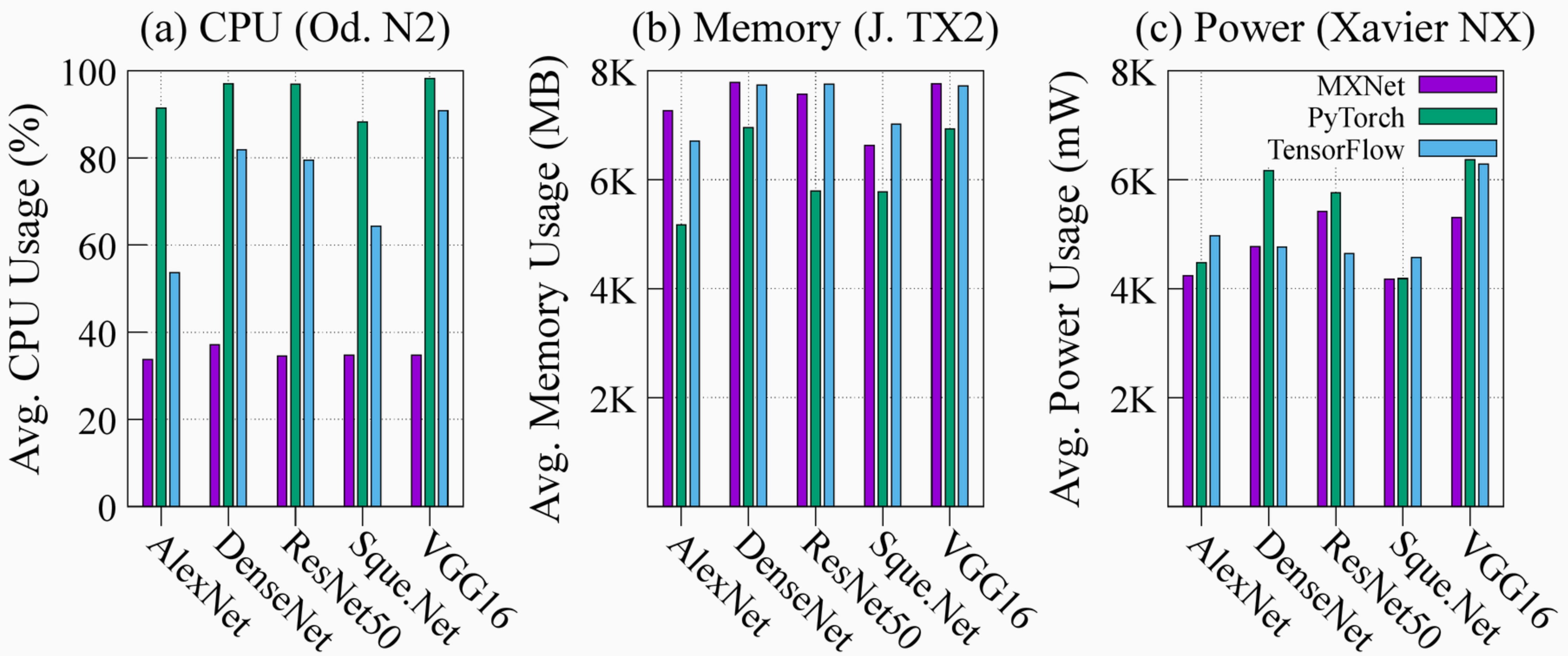
# Impact of batch size (AlexNet)




- GPU-based devices show a significant increase (as much as 2x) in throughput with increasing batch size (expected!!)
- Odroid-N2 and Raspberry Pi4 show minor improvement.
- *Insight:* MxNet slightly better at batched inferencing
- *Bottleneck:* Both CPU and GPU share the same DRAM memory, meaning that increasing batch size can quickly saturate the memory.

IV

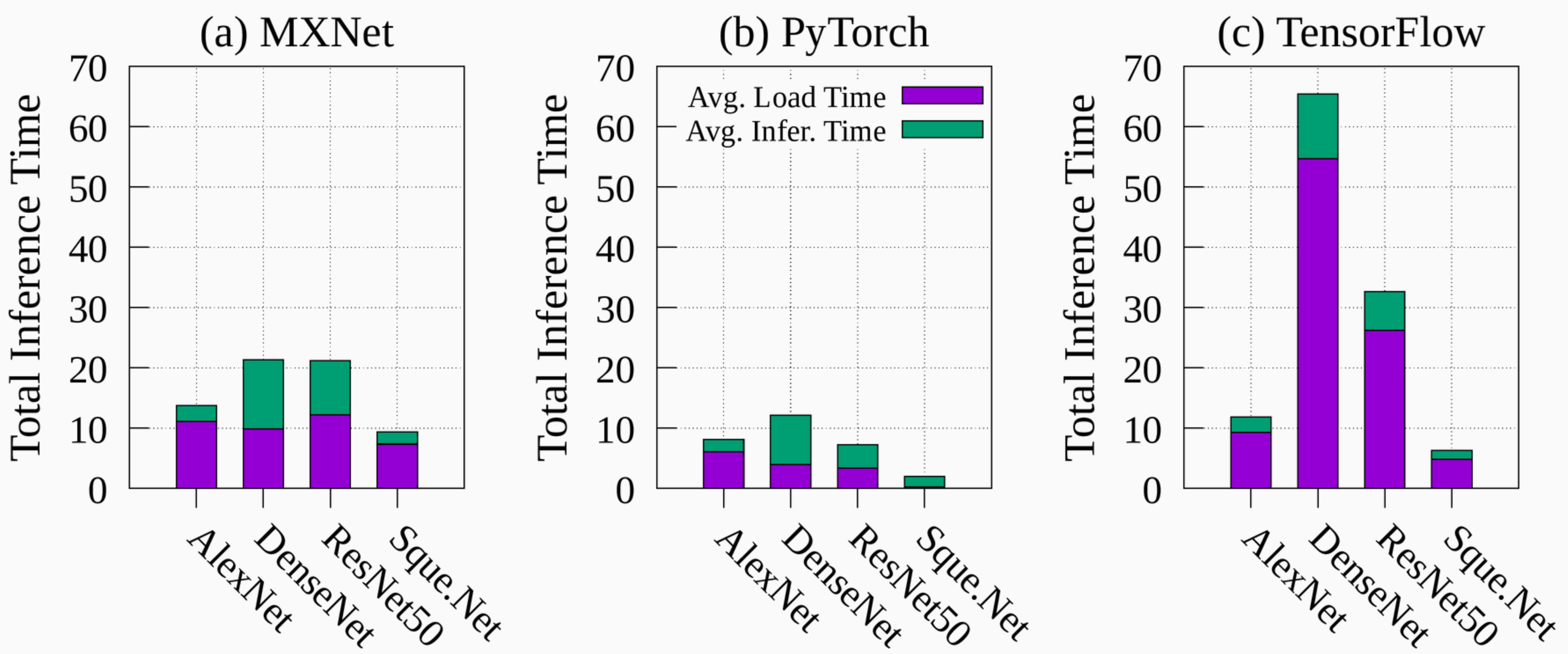
# Resource Usage




- Heavier DNN models like DenseNet and VGG16 consume resources extensively (expected!!)
- PyTorch - least memory consumption; high CPU and Power usage
- MXNet - high memory utilization; low CPU and power usage

IV

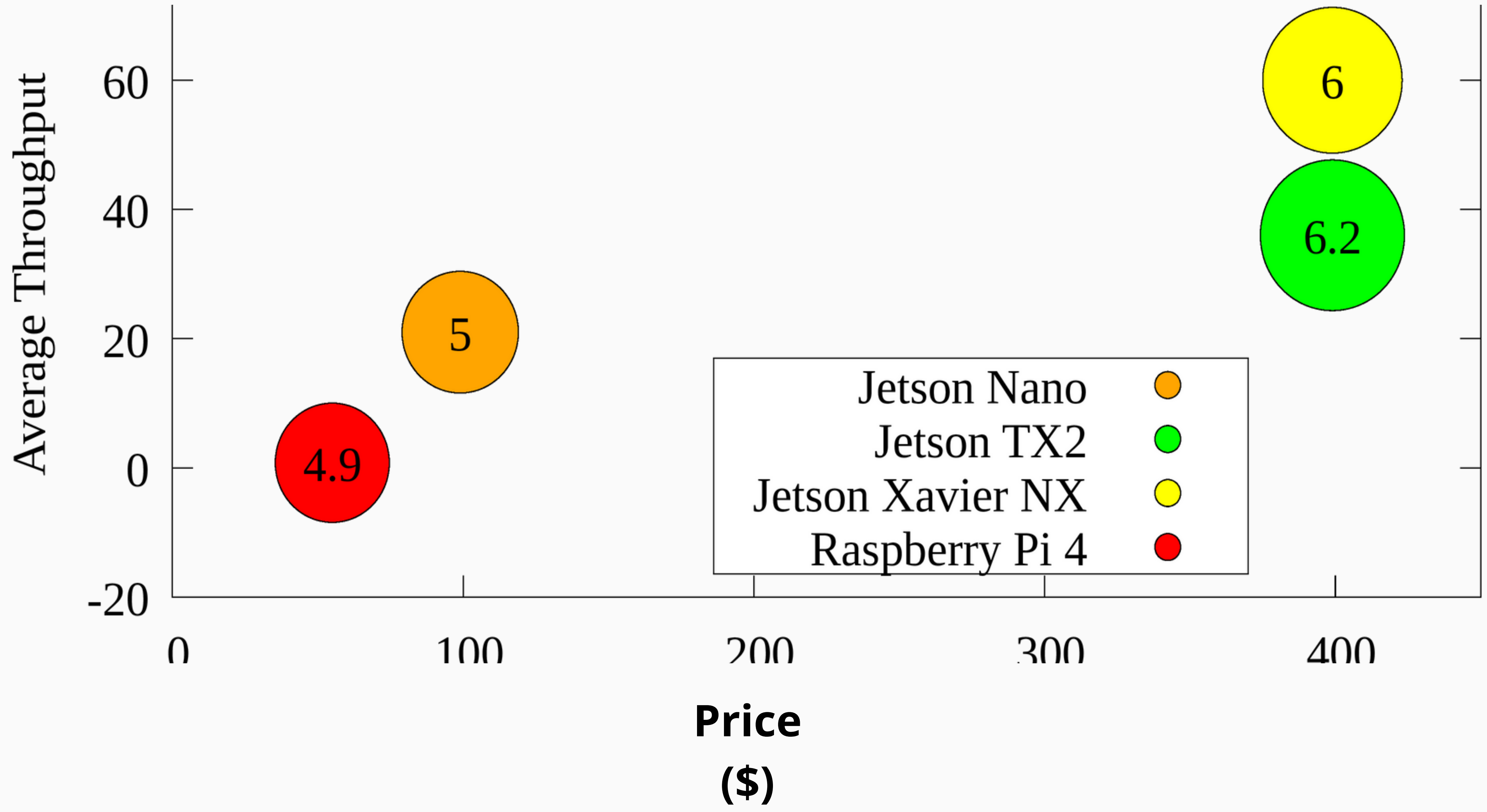
# Impact of loading time (J. Nano)




- Loading time - Overhead associated with initializing DNN models and loading parameters into the memory.
- TF is poor; PyTorch is exceptionally efficient
- Insight: On average, the loading time is 2.4× (MXNet) – 4× (TF) larger than the inference time.

IV

# Cost-Performance analysis




- Cost is also a critical factor when selecting edge devices for DL inference tasks
- Jetson TX2 and Xavier NX are on the pricier side but demonstrate higher performance
- Insight: In terms of power usage, the difference is negligible between the CPU only (e.g., Raspberry Pi4) and GPU-equipped edge devices (e.g., Jetson devices).
- Note- Odroid-N2's results have been excluded due to unreliable results from the power measurement circuit (INA219)

V

# Conclusion

HW specification, batch size and DL framework all affect DL inference performance considerably

GPU resources are critical to increasing the performance

Claim - Pytorch is the most efficient framework in terms of throughput (and latency) and system utilization

Framework-specific optimizations were left out to give all the frameworks a common ground for evaluation

## Future Work

- Future work #1 - Apply all possible optimizations.
- Future work #2 - Investigate TPU (Tensor Processing Unit) based devices like Google's Coral USB Accelerator, Intel's Neural Compute Stick.