MODI: Mobile Deep Inference

Made Efficient by Edge Computing

Samuel S. Ogden
Tian Guo
WPI

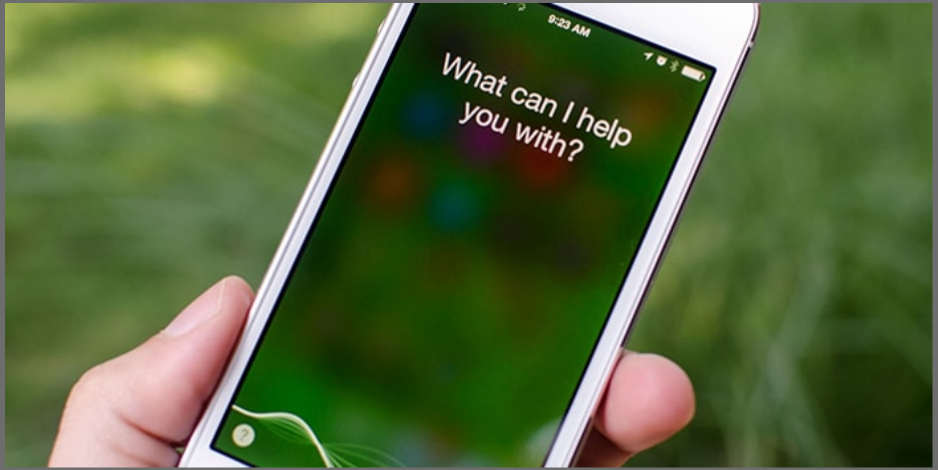
Sneak Peek

- Aim is to be a dynamic solution to a dynamic problem that has previously been solved statically

ssogden@WPI.EDU-MODI

Background – Mobile Inference

- Using deep learning models in mobile application
  - Increasingly common to use deep learning models within mobile applications
    - Image recognition
    - Speech recognition



- Two major metrics
  - Model Accuracy
  - End-to-end latency

- On-device vs. Remote inference

Mobile Deep Inference Limitations

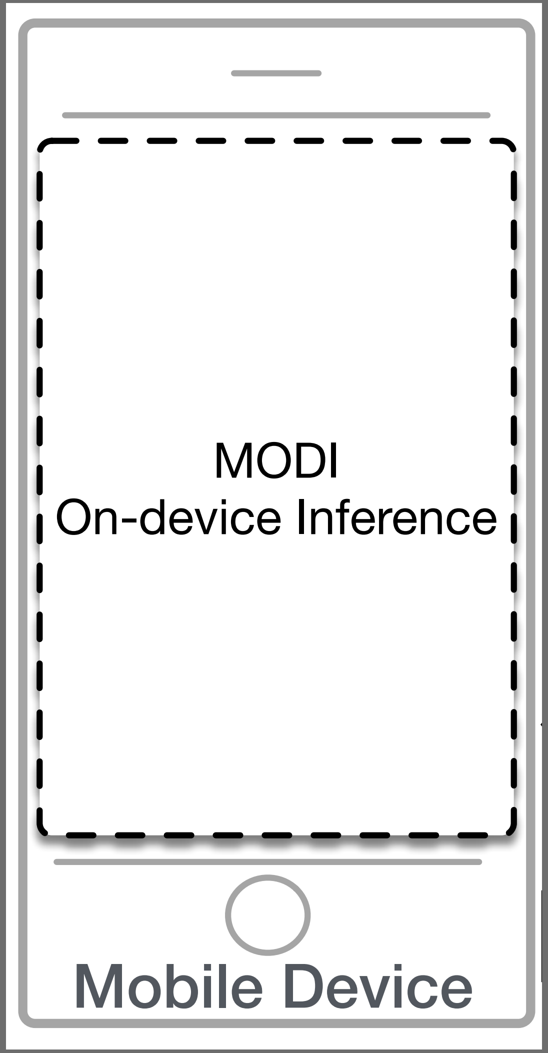
- Highly constrained resource models
  - Battery constraints
  - Lack of limited hardware in general case
- Highly dynamic environment
  - Variable network conditions
- Common approaches are statically applied
  - Choosing a one-size-fits-all model for on-device inference
  - Using the same remote API for all inference requests

Our Vision: MODI

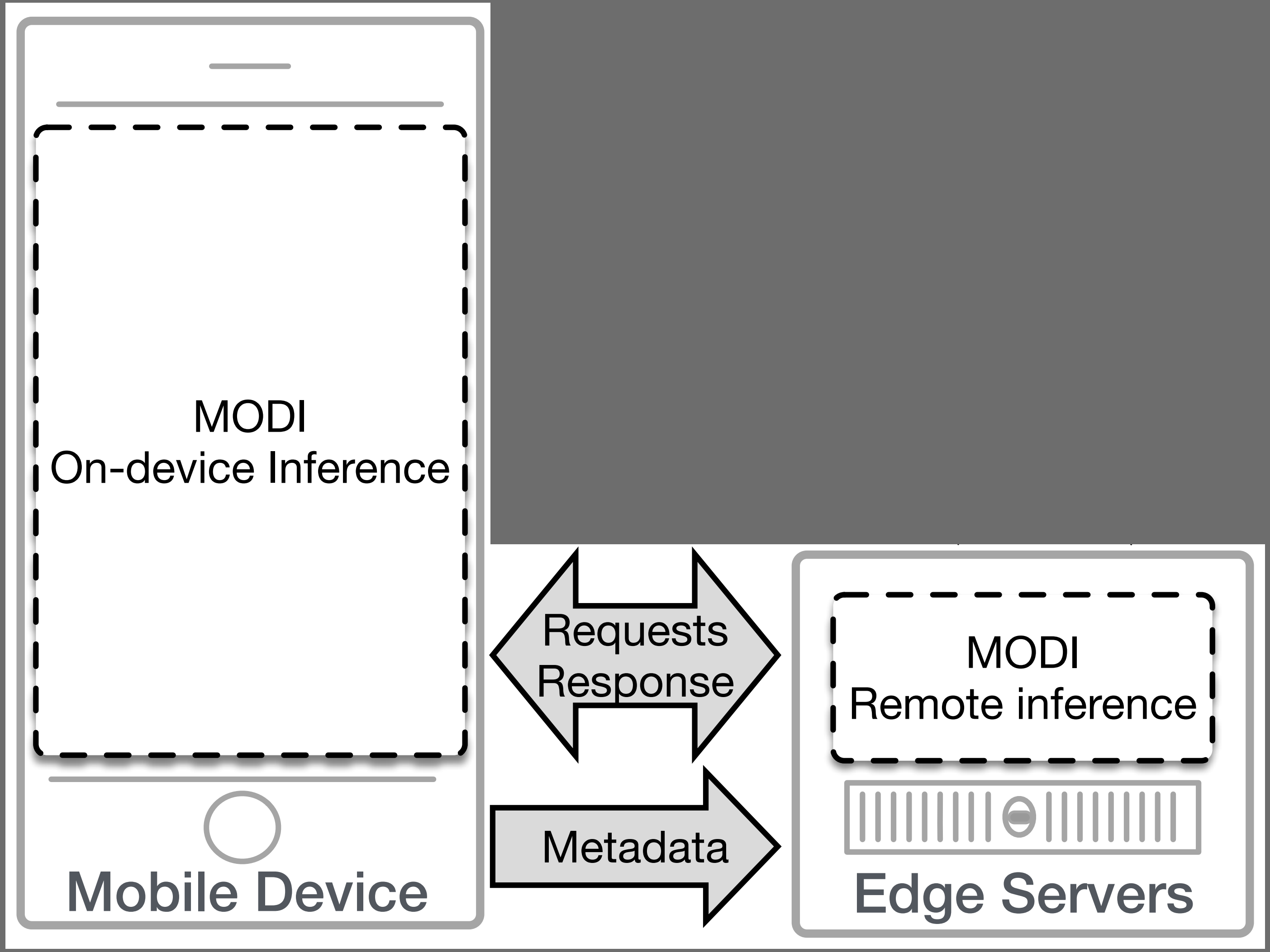
- How do we balance accuracy and latency based on dynamic constraints?

- Provide a wide array of models
  – Model-usage families and derived models
- Dynamically choose inference location and model
  – Make decision based on inference environment
    - e.g., network, power, model availability
- Make the choice transparent

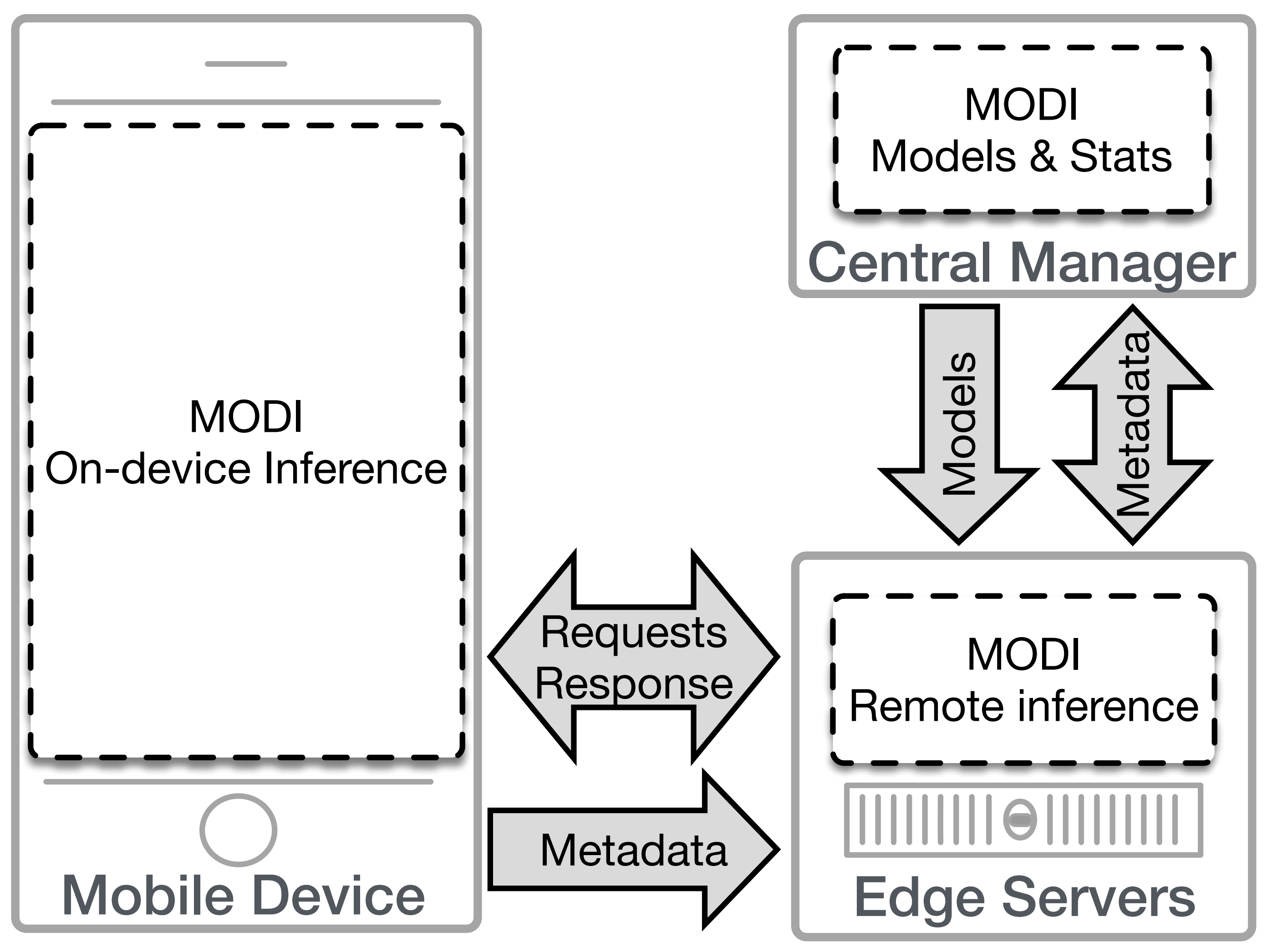
MODI: System design



MODI: System design



MODI: System design



Design Principles

- Maximize usage of on-device resources
- Storage and analysis of metadata
- Dynamic model selection

Design Questions

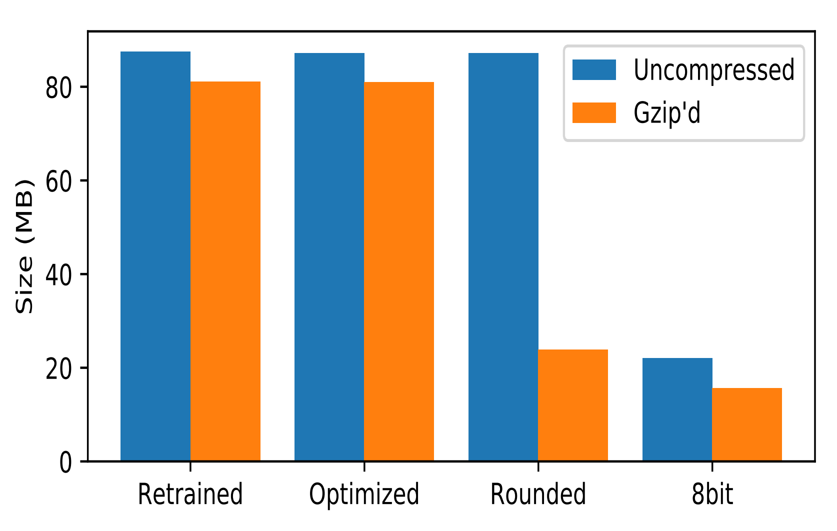
- Which compression techniques are useful?
- Which model versions to store where?
- When to offload to edge servers?

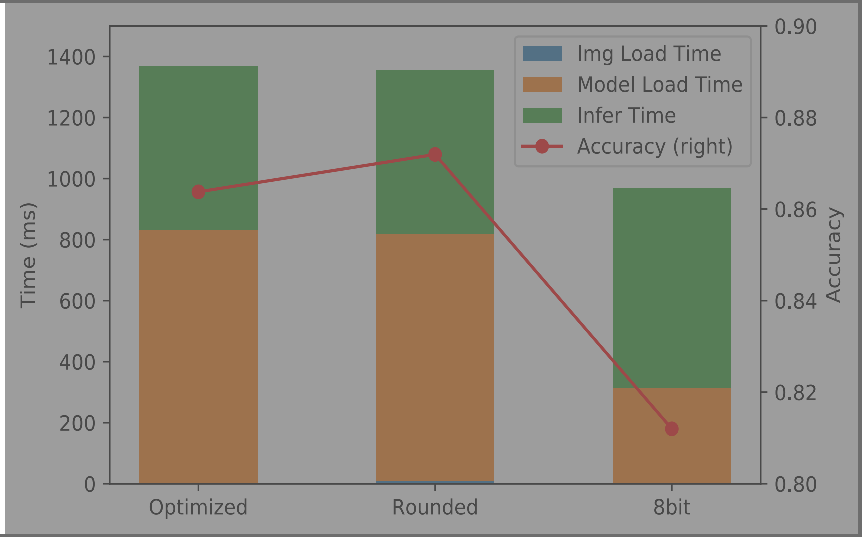
Design Questions

- **Which compression techniques are useful?**
- Which model versions to store where?
- When to offload to edge servers?

Results – Model Compression

InceptionV3 image classification model[1] running on a Google Pixel2 device



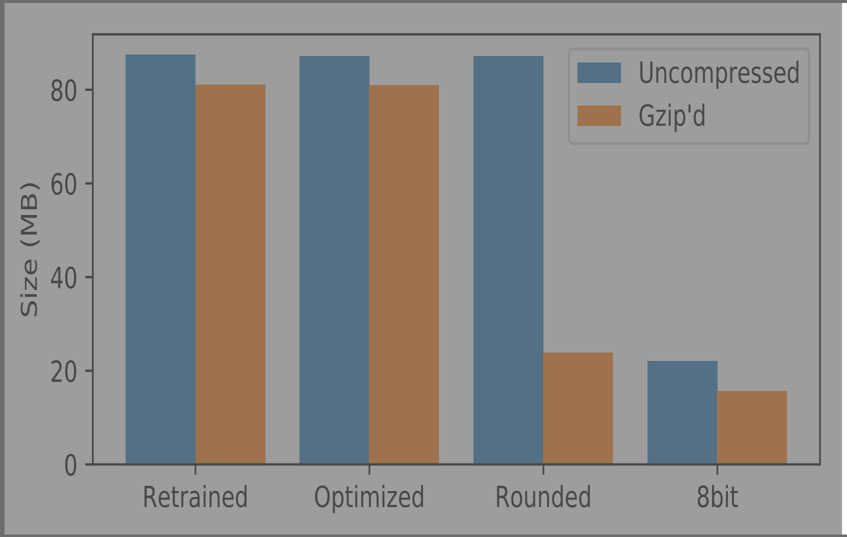


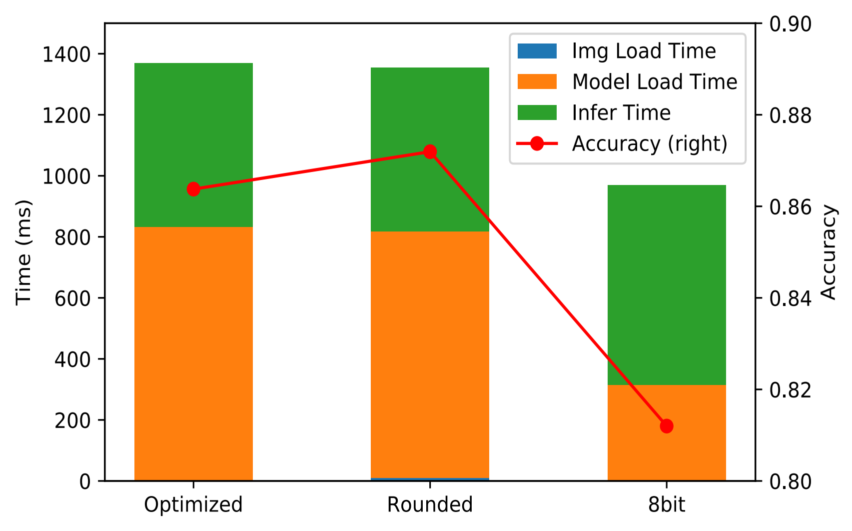
Storage requirements **reduced by 75%** for quantized models

[1]https://arxiv.org/abs/1512.00567

Results – Model Compression

InceptionV3 image classification model[1] running on a Google Pixel2 device





Load time **reduced by up to 66%**

– Leads to ~6% reduction in accuracy

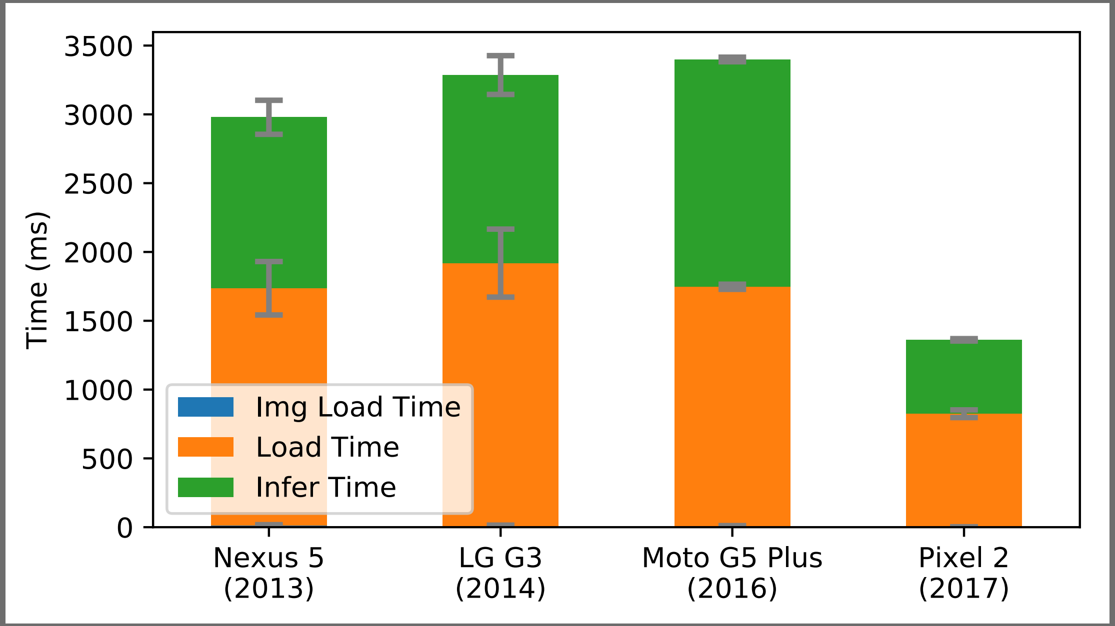
[1]https://arxiv.org/abs/1512.00567

Design Questions

- Which compression techniques are useful?
  - Quantization and gzip significantly reduce model size
- **Which model versions to store where?**
- When to offload to edge servers?

Results – Model Comparison across devices

InceptionV3 image classification model optimized for inference



Pixel2 **over 2.5x faster** than older devices

– Specialized deep-learning hardware

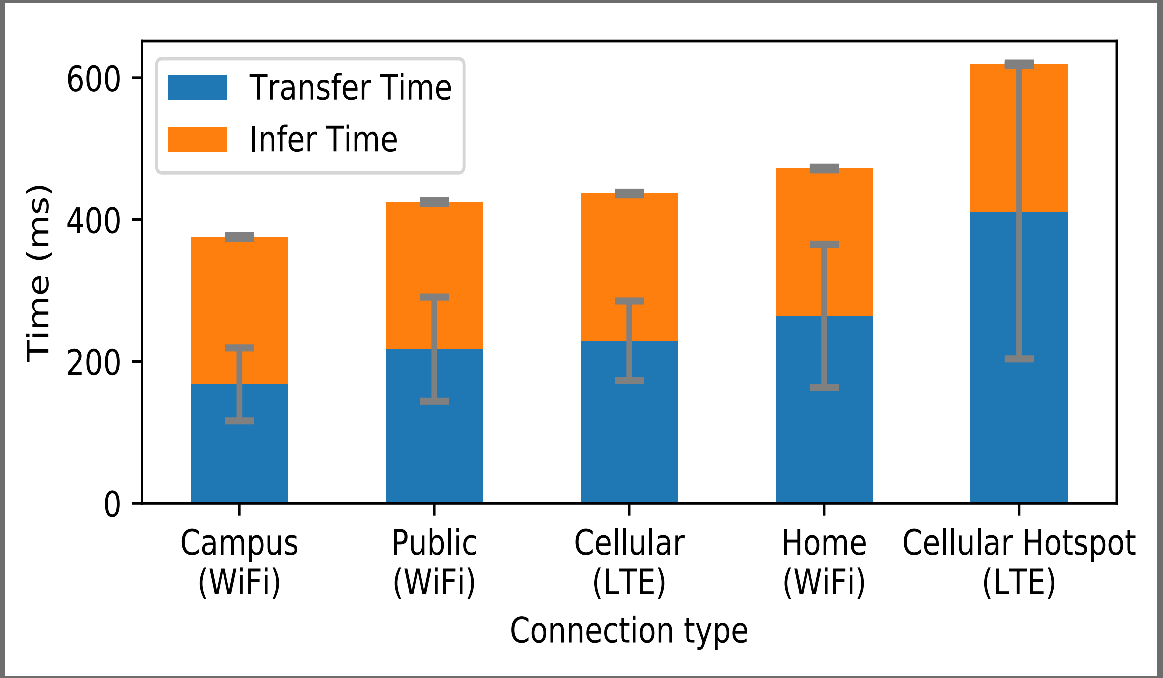
ssogden@wpi.edu - MODI

Design Questions

- Which compression techniques are useful?
  - Quantization and gzip significantly reduce model size
- Which model versions to store where?
  - Mobile devices can reduce runtime up to 2.4x
- **When to offload to edge servers?**

Results – Inference Offloading Feasibility

Used AWS `t2.medium` instance running InceptionV3



Network transfer is **up to 66.7%** of end-to-end time

Design Questions

- Which compression techniques are useful?
  - Quantization and gzip significantly reduce model size

- Which model versions to store where?
  - Mobile devices can reduce runtime up to 2.4x

- When to offload to edge servers?
  - Slower networks would hinder remote inference

Conclusions & Questions

- Key points:
  - MODI allows for dynamic mobile inference model selection through post-training model management
  - Enables greater flexibility for mobile deep inference
- Controversial:
  - Whether using a low-tier AWS instance is similar to edge
- Looking forward:
  - Integrating MODI with existing deep learning frameworks
  - Explore explicit trade-off points between on-device and remote inference
  - Exploring how far in the edge is ideal for remote inference
  - What other devices could this be used for?