CSE3000 Research Project

Partial Loading of Deep Network Layers for Multi-models Edge Inference

Yohan Runhaar - Amirmasoud Ghiassi - Bart Cox

TU Delft

June 23, 2020

Who am I?

- TU Delft BSc student, Computer Science, and Engineering
- Responsible Professor : Dr. Lydia Y. Chen
- Supervisors : Amirmasoud Ghiassi and Bart Cox

1. PARADIGM SHIFT

Internet of Things (IoT)

- Expanding at an astonishing rate
- Smart products
- Market valued at US$ 190.0 Bn in 2018, is anticipated to reach US$ 1,102.6 Bn by 2026

Cloud Computing -> Edge Computing

- Real-time computation
- Decrease in cost pressures
- More robust privacy

AI at the Edge

- AI applications can benefit from the paradigm shift to Edge Computing
- Shorter latency
- Privacy advantages

2. BACKGROUND OF RESEARCH

Resource-constrained devices

Computational and memory heavy algorithms not feasible on these devices

Optimize inference jobs of Deep Neural Networks (DNN)

3. RESEARCH QUESTION

During the loading of deep neural network layers on edge devices, what are the most effective loading techniques to apply in order to optimize the overall inference time?

4. Empirical Research

EdgeCaffe

- Built around Caffe, a deep learning framework
- Partial execution of layers of deep neural networks
  - Splits a deep neural model into smaller parts
  - Load and execute the split layers according to the chosen policy
  - Four different scheduling policies: Bulk, DeepEye, Linear and Partial

Loading policies

- Bulk: loads all the layers upfront before executing them
- DeepEye: interleaves the loading of fully connected layers with the execution of convolutional layers
- Linear: loads and executes a layer at a time
- Partial: loads all the layers greedily over multiple workers, while a single worker executes the loaded layers

5. Results

Impact of swap memory usage

- Memory overhead: loading of DNN layers
- Swap space located on disk requires a lot more time to be accessed than RAM
  -> Loss in inference time

If available memory is large enough to handle greedy loading
-> Partial loading

If in a resource-constrained environment
-> Linear loading