11767 Project Proporsal

Emily Cheng-hsin WUU, Kwun Fung Lau
cwuu@andrew.cmu.edu, kwunfunl@andrew.cmu.edu

1 Motivation

It is a known issue that the quality of a picture taken under low-light conditions is not satisfactory. Due to the small size of the CMOS in the mobile phone, the signal-to-noise ratio is low and hence causes poor image quality. With the state-of-the-art deep learning algorithms and the dedicated Neural Processing Unit (NPU) embedded in the phones, we would like to leverage neural networks to address this problem. The network will finally post-process the image and restore the appearance.

2 Related Work and Baselines

2.1 Baseline: Learning to See in the Dark

This paper [1] introduces a dataset of raw short-exposure low-light images, corresponding long-exposure reference images and develops a pipeline for processing low-light images based on end-to-end training of a fully convolutional network (UNet). Since the proposed method shows promising results compared to many traditional methods, we will use it as the baseline model to complete our tasks.

2.2 Model Optimization

This paper [2] documents many different approaches on optimization UNet and provides comprehensive ablation study on the network structure of how it benefit or deter the performance of the original UNet. One thing we find it particularly useful will be they identify that depthwise separable convolution can efficiently reduce the size of original model.

3 Hypotheses (Key Ideas)

1. Pre-trained model is provided, and we will use it as our baseline inference model.
2. Dataset containing raw images taken in the extra low-light condition is provided, and we will use this dataset as the input of inference model.
3. We will run the baseline inference model with at least one input raw image from the dataset

on Jetson Nano (2GB).

4. We will be able to apply different model optimization skills, including changing the conventional CNN to depthwise separable convolution, model quantization, and model distillation.

4 Methodology

4.1 How you will test those hypotheses: datasets, ablations, and other experiments or analyses.

The essential hypothesis to verify is whether we can fit our baseline inference model with at least one raw image on the Jetson Nano (2GB). Moreover, we also want to know the reliability of the baseline inference model. Hence, we need to conduct experiments in two steps to ensure we have the correct hypotheses.

4.1.1 Memory Usage

We will verify it by checking the memory usage during model inference with a batch size of 1 on both the computer and the edge (Jetson Nano) since there is no guarantee that the monitoring result (by running command top) will be the same on different devices.

4.1.2 Baseline Reliability

After successfully mount the baseline inference model and run at least one raw image on the Jetson Nano. We will manually check the output images with the sample provided from the official implementation to see how well the baseline model performs. Similarly, to inspect the memory usage, we will also need to conduct experiments on both the computer and the edge (Jetson Nano).

4.2 I/O

Input: Image in .RAW/RAW format

Output: Image in .JPEG format

Tools for converting device inputs: Rawpy, OpenCV, PyTorch/Tensorflow

5 Equipment

5.1 Hardware

To verify our hypothesis, we have checked the memory usage on our current Jetson Nano (2GB) and found out that for loading the model (using PyTorch and related libraries) with one raw image (1024x1024) on CPU, it took 4.5 GB of memory in total, while 1.9 GB was

running on Jetson Nano's 2GB main memory and 2.5GB was running on SD card memory. Since the transmit speed on swapping memory between SD card and Jetson Nano's main memory is only around 20MB/s, this significantly impacts the performance of inference speed.
Further checked that the memory usage when system idle is around 800MB in the main Jetson Nano's main memory while 600MB in SD card. If we want to mount our model on GPU, this will double our memory usage.
Based on the above findings, to have our inference model squeezed into Jetson Nano main memory instead of using the time-consuming paging memory (SD card), we kindly request to replace our 2GB Jetson Nano with the 4GB version. It will significantly facilitate the process and lead to more comprehensive experiments on model optimization.

If for a live demo, a camera that can output RAW data is required. Otherwise, we can use the RAW files stored in the SD card.

5.2 Model Pre-train

Yes, training in GPU servers is required. We have our GPU servers for the training process. No cloud compute credit is needed.

6 Challenges

The original implementation [1] uses a large and complex model to generate satisfying results. It is likely that if we significantly compress the model, the final results are not well. Hence it is better to use an iterative approach to achieve that and find the limit. In addition, we had tested the network on Jetson Nano (2GB) already. It was observed that around 4.5GB of memory (including PyTorch and other necessary system background tasks) was used. Memory usage might be a challenge.

7 Project Extensions

There are two possible directions.
The first direction is to port the compressed model to IPhones/Android phones.
Another feasible choice is to combine the information from the LiDAR camera.

8 Ethical Implication

There is no significant risk since the complete inference process is within the offline embedded system (Jetson Nano).

9 Timeline and Milestones

Oct 14 - Implement the reference model using PyTorch

Oct 21 - Adapt the progressive sink method (NAS) to the network

Oct 28 - Apply techniques of EfficientNet / MobileNet

Nov 4 - Explore the possibility of applying distillation technique (e.g.: Teacher-Student network)

Nov 11 - Explore the possibility of applying distillation technique (e.g.: Teacher-Student network) - II

Nov 18 - Apply quantization techniques to the model

Dec 2 - Summarize the techniques and find the best combinations

Referenser

[1] Chen, C., Chen, Q., Xu, J., Koltun, V.: Learning to see in the dark. In: Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition. pp. 3291–3300 (2018)

[2] Szegedy, C., Liu, W., Jia, Y., Sermanet, P., Reed, S., Anguelov, D., Erhan, D., Vanhoucke, V., Rabinovich, A.: Going deeper with convolutions. In: CVPR (2015)