AOWS: adaptive and optimal network width search with latency constraints

Supplementary

A. Latency model: biased sampling

We describe the biased sampling strategy for the latency model, as described in section 4. Using the notations of section 4, the latency model is the least-square solution of a linear system $\boldsymbol{Ax} = \boldsymbol{l}$. The variables of the system are the individual latency of every layer configuration $L_{\Theta_i}(c_i, c_{i+1})$. To ensure that the system is complete, each of these layer configurations must be present at least once among the model configurations benchmarked in order to establish the latency model. Instead of relying on uniform sampling of the channel configurations, we can bias the sampling in order to ensure that the variable of the latency model $L_{\Theta_i}(c_i, c_{i+1})$ that has been sampled the least amount of time is present.

As in AOWS, we rely on a Viterbi algorithm in order to determine the next channel configuration to be benchmarked. Let $N(c_i, c_{i+1})$ be the number of times variable $L_{\Theta_i}(c_i, c_{i+1})$ has already been seen in the benchmarked configurations, and

$$M = \min_{i \in [0, n-1]} \min_{\substack{c_i \in C_i \\ c_{i+1} \in C_{i+1}}} N(c_i, c_{i+1}) \tag{A.1}$$

the minimum value taken by N. The channel configuration we choose for the next benchmarking is the solution minimizing the pairwise decomposable energy

$$\min_{c_0, \dots, c_n} \sum_{i=0}^{n-1} -[N(c_i, c_{i+1}) = M]. \tag{A.2}$$

using the Iverson bracket notation. This energy ensures that at least one of the least sampled layer configurations is present in the sampled configuration.

This procedure allows to set a lower bound on the count of all variables among the benchmarked configurations. The sampling can be stopped when the latency model has reached an adequate validation accuracy.

B. Optimization hyperparameters

For the temperature parameter, we used a piece-wise exponential decay schedule, with values 1 at epoch 5, to 10^{-2} at epoch 6, 10^{-3} at epoch 10, and $5 \cdot 10^{-4}$ at epoch 20.

C. Framework versions and CPU/GPU models

We detail the frameworks and hardware used in the experiments of sections 7.1 and 7.2. Although we report latencies in terms of ms/frame, the latency models are estimated with batches of size bigger than 1. In general, we want to stick to realistic operating settings: GPUs are more efficient for bigger batches, and the batch choice impacts the latency/throughput tradeoff.

The *TRT experiments* are done on an NVIDIA V100 GPU with TensorRT 5.1.5 driven by MXNet v1.5, CUDA 10.1, CUDNN 7.6, with batches of size 64. The *CPU inference experiments* are done on an Intel Xeon® Platinum 8175 with batches of size 1, under PyTorch 1.3.0. The *GPU inference experiments* are done on an NVIDIA V100 GPU with batches of size 16, under PyTorch 1.3.0 and CUDA 10.1.

D. Layer channel numbers and final configuration numbers found

In table D.1, we detail the search space in the channel numbers described in section 7.

Table D.1: Search space: channel configurations for all 14 layers in MobileNet-v1. The first layer always has an input with 3 channels; the last layer always outputs 1000 channels for ImageNet classification. The bold values indicate the initial MobileNet-v1 configuration numbers.

i	C_i
1	8, 16, 24, **32**, 40, 48
2	16, 24, 32, 40, 48, 56, **64**, 72, 80, 88, 96
3	24, 40, 48, 64, 80, 88, 104, 112, **128**, 144, 152, 168, 176, 192
4	24, 40, 48, 64, 80, 88, 104, 112, **128**, 144, 152, 168, 176, 192
5	48, 80, 104, 128, 152, 176, 208, 232, **256**, 280, 304, 336, 360, 384
6	48, 80, 104, 128, 152, 176, 208, 232, **256**, 280, 304, 336, 360, 384
7	104, 152, 208, 256, 304, 360, 408, 464, **512**, 560, 616, 664, 720, 768
8	104, 152, 208, 256, 304, 360, 408, 464, **512**, 560, 616, 664, 720, 768
9	104, 152, 208, 256, 304, 360, 408, 464, **512**, 560, 616, 664, 720, 768
10	104, 152, 208, 256, 304, 360, 408, 464, **512**, 560, 616, 664, 720, 768
11	104, 152, 208, 256, 304, 360, 408, 464, **512**, 560, 616, 664, 720, 768
12	104, 152, 208, 256, 304, 360, 408, 464, **512**, 560, 616, 664, 720, 768
13	208, 304, 408, 512, 616, 720, 816, 920, **1024**, 1128, 1232, 1328, 1432, 1536
14	208, 304, 408, 512, 616, 720, 816, 920, **1024**, 1128, 1232, 1328, 1432, 1536

Table D.2: Channel configurations found in the TRT optimization (section 7.1), visualized in fig. 5, and with top-1 errors given in table 2 in the paper. Results are compared to the original Mobilenet-v1 [11] channels.

method	configuration
greedy	8, 24, 40, 48, 104, 128, 208, 304, 768, 360, 720, 616, 1536, 1128
OWS	8, 32, 64, 80, 128, 232, 408, 464, 512, 512, 464, 464, 1024, 1328
AOWS	8, 16, 48, 64, 128, 256, 512, 512, 512, 512, 464, 512, 1536, 1536
Mobilenet-v1 [11]	32, 64, 128, 128, 256, 256, 512, 512, 512, 512, 512, 512, 1024, 1024