A. Profiling in PyTorch

This section details the practical implementation of data collection for the attributes described in Sec.4. Code to support the description here is open-sourced in the tool associated with this work. PyTorch forward and backward hooks are attached to the network's convolution layers and at each hook the following values are recorded for pruning levels $5x|x \in [0, 18]$ and batch sizes {2,4,8,16,32,64,70,80,90,100,110,120,128,140,150,160,170, 180,190,200,210,220,230,240,256}.

Results provided in the paper were for both the NVIDIA Jetson TX2 and the NVIDIA RTX 2080Ti. Φ is profiled in the same manner for the two systems (embedded and server):

Mini-batch training latency (Φ) is profiled using the `torch.cuda.Events` API as this allows to time asynchronous GPU events.

The total training memory attribute is profiled differently for embedded (NVIDIA Jetson TX2) and server (NVIDIA RTX 2080Ti) systems due to the difference in memory management where the embedded device has a unified memory space, where the server system has an independent memory space for the GPU.

Memory Used (Γ)

- **NVIDIA Jetson TX2** - Profiled using information from the `/proc/meminfo` system file. As the Jetson has a unified memory, information in this file accounts for both GPU and CPU memory.

  $$TotalMemUsed = MemTotal - MemFree - Buffers - Cached$$

  The terms on the RHS of the equation correspond directly to values found in `/proc/meminfo` file.

- **NVIDIA RTX 2080Ti** - Profiled using the `pynvml` tool with the command `nvmlDeviceGetMemoryInfo.used`.

The system records the maximum value of Γ observed until the point of profiling.

B. Model Construction

B.1. Initial Profiling

Graphs corresponding to the profiling described in Sec.5.2 for ResNet18, MobileNetV2, SqueezeNet and MnasNet for the modelled attributes are shown in Fig.5. As discussed, they display linearity with batch size and varying linear fit dependent on the pruning level.

B.2. Complete list of model features

This section lists the features described in Sec.5.2.1 along with various summations of these features that are used to develop the performance model.

Consider a CNN where each convolutional layer $l \in \mathcal{L}$ has n_l filters of size $m_l \times k_l \times k_l$. Let layer l have stride s_l, padding p_l and groups g_l. Let the IFM to this layer have dimensions $bs \times m_l \times ip_l \times ip_l$, the weights $n_l \times \frac{m_l}{g_l} \times k_l \times k_l$ and the OFM $bs \times n_l \times op_l \times op_l$ where bs is the batch size of training. The OFM spatial dimensions op_l can be calculated from other variables using the equation $op_l = 1 + \lfloor \frac{ip_l + 2p_l - k_l}{s_l} \rfloor$.

B.2.1 Tensor Allocations

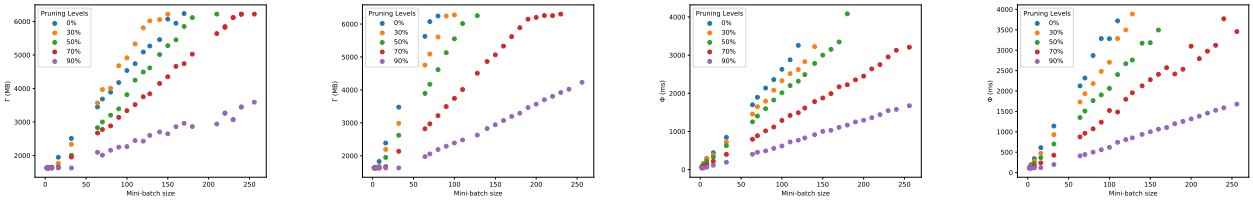
The features in this section describe the sizes of tensors that constitute the IFM, weights and OFM; and each of their gradients on a per-layer basis.

1. $mem_w = n_l \cdot \frac{m_l}{g_l} \cdot k_l^2$
2. $mem_{w_{grad}} = bs \cdot n_l \cdot \frac{m_l}{g_l} \cdot k_l^2$
3. $mem_{ifm_{grad}} = mem_{ifm} = bs \cdot m_l \cdot ip_l^2$
4. $mem_{ofm_{grad}} = mem_{ofm} = bs \cdot n_l \cdot op_l^2$
5. $mem_w + mem_{w_{grad}} + mem_{ifm_{grad}} + mem_{ofm_{grad}}$

The following sections model the memory consumption and operations for the Matrix Multiplication, FFT and Winograd based convolution algorithms on a per layer basis. Each feature models one of Eq.1 (fwd), the forward pass; Eq.2 (bwd_x), the computation of the gradient w.r.t input; or Eq.3, the computation of the gradient w.r.t weights.

B.2.2 Matrix Multiplication based convolution

The features in this section are obtained by modelling the sizes of the matrices that are used to perform a convolution through matrix multiplications, and calculating the memory and operations required to store these matrices and perform the multiplication. The $total$ features correspond to the MATMUL strategy where the entire $im2col$ matrix is stored in memory, while idx features correspond to the strategy of storing only the necessary indices as described in Sec.5.2.1.



(a) ResNet18 Γ **(b)** MobileNetV2 Γ **(c)** SqueezeNet Φ **(d)** MnasNet Φ

Figure 5: Γ and Φ for 4 different networks that were profiled when training 3x224x224 sized inputs on the NVIDIA Jetson TX2. x-axis is mini-batch size and y-axis is the attribute value.

As an example, the forward pass (Eq.1) involves a multiplication between the $im2col$ IFM matrix of size $((bs \times op_l^2) \times (k_l^2 \times m_l))$ and a reshaped weights matrix of size $((k_l^2 \times m_l) \times (n_l))$. Using $k_l^2 \times m_l$ as the common dimension and calculating the memory and operations for a matrix multiplication gives the features $mem\_i2c^{mm}_{fwd^{total}}$ and $ops^{mm}_{bwd_x}$. As discussed in Sec.5.2.1, the index of each of the op_l^2 output pixels needs to be stored thus giving the $mem\_i2c^{mm}_{fwd^{index}}$ feature. Similar arguments applied to Eq.2 and 3 give the remaining features in this section.

6. $mem\_i2c^{mm}_{fwd^{total}} = bs \cdot op_l^2 \cdot k_l^2 \cdot m_l$
7. $mem\_i2c^{mm}_{bwd_w^{total}} = bs \cdot op_l^2 \cdot k_l^2 \cdot \frac{m_l}{g_l}$
8. $mem\_i2c^{mm}_{fwd^{index}} = i2c^{mm}_{bwd_w^{index}} = bs \cdot op_l^2$
9. $mem\_i2c^{mm}_{bwd_x^{total}} = bs \cdot ip_l^2 \cdot k_l^2 \cdot m_l$
10. $mem\_i2c^{mm}_{bwd_x^{index}} = bs \cdot ip_l^2$
11. $mem\_i2c^{mm}_{fwd^{total}} + mem\_i2c^{mm}_{bwd_w^{total}} + mem\_i2c^{mm}_{bwd_x^{total}}$
12. $2 \times mem\_i2c^{mm}_{fwd^{index}} + mem\_i2c^{mm}_{bwd_x^{index}}$
13. $ops^{mm}_{fwd} = ops^{mm}_{bwd_w} = bs \cdot n_l \cdot op_l^2 \cdot k_l^2 \cdot \frac{m_l}{g_l}$
14. $ops^{mm}_{bwd_x} = bs \cdot m_l \cdot ip_l^2 \cdot k_l^2 \cdot n_l$
15. $2 \times ops^{mm}_{fwd} + ops^{mm}_{bwd_x}$

B.2.3 FFT based convolution

The features in the section, apart from the summations, have been obtained from [16] where there is a detailed breakdown of the memory consumption and operations of using the FFT algorithm for matrix multiplication.

16. $mem\_w^{fft}_{fwd} = n_l \cdot \frac{m_l}{g_l} \cdot ip_l \cdot (1 + ip_l)$
17. $mem\_ifm^{fft}_{fwd} = ifm^{fft}_{bwd_w} = bs \cdot m_l \cdot ip_l \cdot (1 + ip_l)$
18. $mem\_ofm^{fft}_{bwd_w} = bs \cdot n_l \cdot ip_l \cdot (1 + ip_l)$
19. $mem\_w^{fft}_{bwd_x} = n_l \cdot \frac{m_l}{g_l} \cdot op_l \cdot (1 + op_l)$
20. $mem\_ofm^{fft}_{bwd_x} = bs \cdot n_l \cdot op_l \cdot (1 + op_l)$
21. $mem\_w^{fft}_{fwd} + mem\_ifm^{fft}_{fwd}$
22. $mem\_ofm^{fft}_{bwd_x} + mem\_ofm^{fft}_{bwd_w}$
23. $mem\_ofm^{fft}_{bwd_w} + mem\_ifm^{fft}_{fwd}$
24. $21 + 22 + 23$
25. $ops^{fft}_{fwd} = ip_l^2 \cdot log(ip_l) \cdot (bs \cdot (m_l + n_l) + n_l \cdot \frac{m_l}{g_l}) + bs \cdot n_l \cdot m_l \cdot ip_l^2$
26. $ops^{fft}_{bwd_x} = op_l^2 \cdot log(op_l) \cdot (bs \cdot (m_l + n_l) + n_l \cdot \frac{m_l}{g_l}) + bs \cdot n_l \cdot m_l \cdot op_l^2$
27. $ops^{fft}_{bwd_w} = ip_l \cdot log(ip_l^2) \cdot (bs \cdot (m_l + n_l) + n_l \cdot \frac{m_l}{g_l}) + bs \cdot n_l \cdot m_l \cdot ip_l^2$
28. $ops^{fft}_{fwd} + ops^{fft}_{bwd_x} + ops^{fft}_{bwd_w}$

B.2.4 Winograd convolution

Each of the features described in this section model Eq.4 for each of Eq.1,2,3. Consider the case of Eq.2. $\frac{\delta L}{\delta y}$ is a matrix of size $(bs \times n_l \times op_l \times op_l)$ and w_{nm} is a matrix of size $(n_l \times m_l \times k_l \times k_l)$. The term d in Eq.4, is one of the $\lceil \frac{op_l}{q} \rceil^2$ tiles of $\frac{\delta L}{\delta y}$ and g is one of the $\lceil \frac{k_l}{r} \rceil^2$ tiles of w. Assuming parallelism over bs, m_l and $\lceil \frac{op_l}{q} \rceil^2$ gives the feature $mem^{wino}_{bwd_x}$ and accounting for $n_l \cdot \lceil \frac{k_l}{r} \rceil^2$ accumulations gives the feature ops^{wino}_{fwd}. Applying similar arguments to Eq.1 and 3 gives the remainder of the features.

The following features are applied twice for $(q \times r)$ of (4×3) and (3×2) which both profiling and [8] showed to be the most commonly used sizes of winograd convolutions by CuDNN

29. $mem^{wino}_{fwd} = bs \cdot n_l \cdot \lceil \frac{ip_l}{q} \rceil^2 \cdot 3 \cdot (q + r - 1)^2$

30. $mem^{wino}_{bwd_x} = bs \cdot m_l \cdot \lceil \frac{op_l}{q} \rceil^2 \cdot 3 \cdot (q + r - 1)^2$

31. $mem^{wino}_{bwd_w} = bs \cdot n_l \cdot \frac{m_l}{g_l} \cdot \lceil \frac{ip_l}{q} \rceil^2 \cdot 3 \cdot (q + r - 1)^2$

32. $mem^{wino}_{fwd} + mem^{wino}_{bwd_x}$

33. $mem^{wino}_{fwd} + mem^{wino}_{bwd_w}$

34. $mem^{wino}_{bwd_w} + mem^{wino}_{bwd_x}$

35. $32 + 33 + 34$

36. $ops^{wino}_{fwd} = bs \cdot n_l \cdot \frac{m_l}{g_l} \cdot \lceil \frac{ip_l}{q} \rceil^2 \cdot \lceil \frac{k}{r} \rceil^2 \cdot (q + r - 1)^2$

37. $ops^{wino}_{bwd_x} = bs \cdot m_l \cdot n_l \cdot \lceil \frac{op_l}{q} \rceil^2 \cdot \lceil \frac{k}{r} \rceil^2 \cdot (q + r - 1)^2$

38. $ops^{wino}_{bwd_w} = bs \cdot n_l \cdot \frac{m_l}{g_l} \cdot \frac{m_l}{g_l} \cdot \lceil \frac{ip_l}{q} \rceil^2 \cdot \lceil \frac{op_l}{r} \rceil^2 \cdot (q + r - 1)^2$

39. $ops^{wino}_{fwd} + ops^{wino}_{bwd_x}$

40. $ops^{wino}_{fwd} + ops^{wino}_{bwd_w}$

41. $ops^{wino}_{bwd_x} + ops^{wino}_{bwd_w}$

42. $39 + 40 + 41$

BUILDING BLOCK	NETWORKS
RESIDUAL	RESNET18, RESNET50
DEPTH-WISE SEPARABLE	MOBILENETV2, MNASNET
FIRE	SQUEEZENET
INCEPTION	GOOGLENET

Table 3: Summary of architectural building blocks and networks that utilise them

C. Training on a "basis" of networks

This section provides additional results for Sec.6.3 which investigated if training could be performed on data from a representative "basis" of networks and predict attributes on other networks not present in the basis but with similar building blocks to those in the basis. The following discussion illustrates the similarity of building blocks between networks in the basis and those not present in it.

Tab.3 details various commonly used building blocks and the networks that utilise them. Both the Fire and Inception modules employ a "branch-and-concatenate" computation structure and have 1×1 and 3×3 convolutions. The Fire module has 2 branches, whereas the Inception module has 4 branches and a 5×5 convolution. ResNet18 is made of "Basic-block residuals" which have two 3×3 convolutions, whereas the deeper ResNet50 is made of "Bottleneck residuals" which have 3 convolutions (one 3×3 and two 1×1). MobileNetV2 and MnasNet are made up of the same depth-wise separable inverted residual module, but have different depths. Thus architecture pairs from most to least similar are: MobileNetV2 and MnasNet; ResNet18 and ResNet50; and SqueezeNet and GoogLeNet. The results of this investigation are inline with this observation where MnasNet performed the best followed by ResNet50 and then GoogLeNet.

D. On-device OFA Case Study Subsets

This section provides details on the classes present in each of the 4 subsets used in the on-device OFA case study presented in Sec.6.4. The subsets were extracted from sub classes of the ImageNet dataset and were created to emulate objects, people, buildings and animals that might observed in different environments that an autonomous vehicle could encounter. The details of the classes present in each subset are as follows:

- **City** (185 classes) : ambulance, trash can, station wagon, tandem bike, taxi, car mirror, car wheel, convertible, crane, electric locomotive, fire engine, fire truck, garbage truck, petrol pump, radiator grille, jeep landrover, rickshaw, lawn mower, limousine, mailbox, manhole cover, minibus, minivan, Model T, moped, scooter, mountain bike, moving van, parking meter, passenger car / coach, pay-phone, pickup truck, police car, race car, school bus, shopping cart, snowplow, sports car, steel arch bridge, tram, suspension bridge, tow truck, trolley bus, street sign, traffic light, palace, mosque, church building, castle, boathouse, tirumphal arch, academic gown/robe, cardigan, fur coat, gown, jersey / t-shirt, suit, sunglasses, sweatshirt, trench coat, umbrella, swan, dogs, red fox, cats
- **Motorway** (26 classes) : station wagon, bullet train, car mirror, car wheel, convertible, electric locomotive, petrol pump, jeep landrover, minibus, minivan, mobile home, Model T, moving van, passenger car / coach, pay-phone, pickup truck, police car, race car, recreational vehicle, snowplow, sports car, tow truck, trailer truck, street sign, water tower
- **Country-side** (204 classes) : wheelbarrow, station wagon, bullet train, taxi, car mirror, car wheel, convertible, electric locomotive, freight car, garbage truck, petrol pump, radiator grille, half track, horse cart, jeep landrover, lawn mower, mailbox, manhole cover, minibus, minivan, mobile home, Model T, moped, scooter, mountain bike, moving van, oxcart, parking meter, passenger car / coach, pay-phone, picket-fence, pickup truck, plough, police car, race car, recreational vehicle, school bus, snowmobile, snowplow, sports car, steel arch bridge, tank, thatched roof, tile roof, tow truck, tractor, trailer truck, worm fence, street sign, traffic light, hay, palace, mosque, church building, castle, lighthouse, barn, viaduct, water tower, cardigan, fur coat, gown, sarong, jersey / t-shirt, suit, sunglasses, sweatshirt, swimming trunks, trench coat, umbrella, cock, hen, quail, goose, swan, dogs, red fox, cats, rabbits, ram, sheep
- **Off-road** (26 classes) : mobile home, mountain bike, oxcart, pickup truck, plough, snowmobile, tank, tractor, hay, ostrich, iguana, alligator, wallaby, koala, wombat, brown bear, black bear, hog, wild boar, ox, water buffalo, bison, wild deer

There are fewer written categories than the number of classes stated as some categories such as ”dogs” have many ILSVRC’12 classes within them. There are also overlap of classes between the subsets as would be expected.