Frenzy: ARM ML Research MicroNet Submission

Ramon Matas, Igor Fedorov, Hokchhay Tann, Chuteng Zhou, Paul Whatmough, Matthew Mattina

Arm ML Research

1 Submission Score

Our dataset target is CIFAR100 and our submission achieves a score of 0.2. The model size (MS) is 8.43MB, the operation count (OC) is $828.928e6$, and the test set accuracy on CIFAR100 is 80.34%. We name our submitted architecture Frenzy.

1.1 Accounting system

We follow the scoring system posted in the challenge instructions. Our model is not quantized. The submission repository contains a visualization of the submitted network in

```
graph_visualization.png
```

which includes the MS and OC of each node in the graph, along with the kernel size, stride, and width of each convolution. In addition,

```
winner.py
```

contains a complete description of each node, in case the calculations need to be verified by hand.

1.2 Reproducing the results

The results can be reproduced by cloning the shared repository

```
https://github.com/ifed-ucsd/arm_micronet
```

starting docker image

```
fedorov1/tf2:latest
```

and running

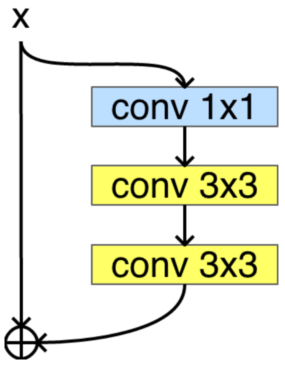

Figure 1: Residual block used in the submission.

```
python standalone_inference.py
```

The code may be launched using

```
docker run -it -v $PATH_TO_REPO:/workspace
  --rm fedorov1/tf2:latest python /workspace/
      standalone_inference.py --load_folder /workspace/
      SavedModel/
```

2 Methodology

We begin with a wide residual network (WRN) backbone [3], using the variant in [1]. In the notation introduced in [3], our backbone is WRN 28-10, with widths $\{\omega_1 = 100, \omega_2 = 200, \omega_3 = 400\}$ for the three groups that comprise the network. The network begins with a 3×3 convolutional layer with 100 output channels, which are followed by three groups of three residual blocks each. In [1], the i'th residual block consists of two 3×3 convolutions with width w_i along with a bypass connection. The first convolution in the second and third groups has a stride of two. All convolutions are preceded by batch normalization and rectified linear unit (Relu) non-linearity [3]. We introduce a slight modification to the residual structure whereby the convolutions in the residual branch are preceded by a 1×1 convolution with width equal to the width of the preceding residual block, as shown in Fig. 1. The reason for the additional convolution in the residual branch is it allows us to modulate the number of input channels to the first 3×3 convolution without affecting information flow to the other residual blocks. We employ the pruning approach from [2] to reduce network complexity by learning the kernel height, kernel width, and number of output channels for each convolution.

We would be happy to discuss this model in more detail.

References

[1] Mark D McDonnell. Training wide residual networks for deployment using a single bit for each weight. *International Conference on Learning Representations (ICLR)*, 2018.

[2] Dimitrios Stamoulis, Ruizhou Ding, Di Wang, Dimitrios Lymberopoulos, Bodhi Priyantha, Jie Liu, and Diana Marculescu. Single-path nas: Designing hardware-efficient convnets in less than 4 hours. *Joint Workshop on On-Device Machine Learning & Compact Deep Neural Network Representations*, 2019.

[3] Sergey Zagoruyko and Nikos Komodakis. Wide residual networks. *Proceedings of the British Machine Vision Conference (BMVC)*, 2016.