Down, down, do your dance, do your dance: CupidShuffle

Matt Krzus Alex Horn
matt@lastfrontiertechnologies.com nanodust@gmail.com

Abstract

With the recent popularity of Vision Transformers, much of the focus has been on the novel aspect of the field. This paper looks to simply expand on one of the influential edge-focused models, The ShuffleNet. We propose CupidShuffle, a ShuffleNetV2 variant replacing the initial entry layer with a Window-based Transformer and also removing 7 ShuffleV2Unit layers while still keeping the round-trip shuffle intact by switching from [3, 7, 3] to [1, 4, 1]. The proposed architecture keeps ShuffleNetV2's predictive power, while decreasing the total model size. And probably the most important contribution, this paper also attempts to keep it short and concise.

1 Introduction

Vision Transformers have become incredibly popular following advancements in Generative Networks field. Some have incredible utility to solve very pressing problems for those of us without tres commas worth of GPUs. It has become incredibly difficult to reinvent the wheel, and rather than doing so, the attempt of this paper is simply to utilize the novelty of the embedding space of the transformer while combining brutal compute efficiency of ShuffleNetv2.

2 Related Work

ShuffleNet[1] is a computation-efficient architecture designed specially for devices with very limited computing power. The novelty is applying group convolutions and then shuffling the groups so that information flows across channels. This group splitting followed by cross-channel information flow is what makes the network so efficient. In **ShuffleNetV2** [2], the authors present a number of guidelines for computation efficiency: G1) Equal channel width minimizes memory access cost (MAC) G2) Excessive group convolution increases MAC G3) Network fragmentation reduces degree of parallelism and G4) element-wise operations are non-negligible. Based on those 4 principles, the authors conclude that: d 1) use equal channel width convolutions; 2) avoid group convolutions if possible, 3) reduce network fragmentation; and 4) reduce element-wise operations. The authors then improve upon the ShuffleNet model and introduce an operator called *channel-split* for their ShuffleNetV2 architecture. This channel-split works such that at the beginning of each unit, the input feature channels are split into two channel-based branches; where one branch remains as an identity and the other branch consists of three convolutions with the same input and output channels. The major change here is that the two 1x1 convolutions of ShuffleNet are now no longer group-wise. The two branches are then concatenated where the *channel-shuffle* operation is then used to enable communication between the two branches.

Other ideas exist in the efficiency domain, like [5][6][7] - but interestingly, the authors of Shuffle-NasNet paper[8] use the NAS framework combined with ShuffleNetV2 to build an efficient network, that in their own words, violates G1, but still results in a fast network showing that despite the theoretical framework, sometimes violations unexpectedly increase speed. In [9], the authors build

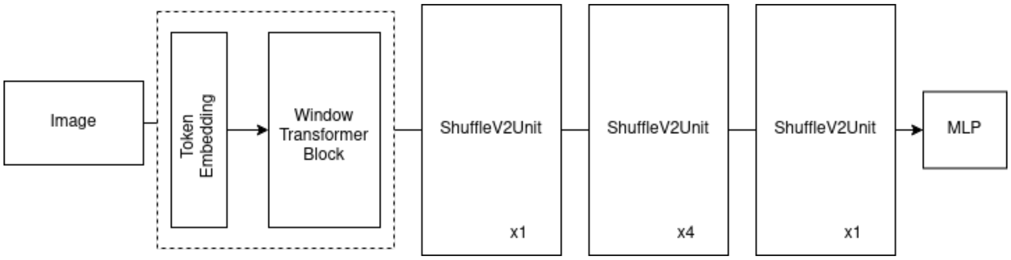

Figure 1: CupidShuffle architecture: token embedding into a single Window-based Multihead Self-Attention (WMSA) where the token-embedding is the same size as the first channel of the first shuffleunit layer to adhere to G1 and 3 total ShuffleV2Unit blocks of a 1-4-1 size

thier network using a similar block to MobileNetV1's blocks and demonstrate that the vast majority of the speedups attained are through heavily optimizing the deployment target. MnasNet[11] introduces attention modules and [10] MobileNetV3, combines [12, 13, 14, 15, 16, 17] to reduce latency nearly 20% compared to MobileNetV2. Overall, there have been a ton of light-weight CNNs [23][24][25][26][27][28][29][30][31]

While ShuffleNetV2 looked to exploit MAC specific efficiencies, **Shuffle Transformer**[3] looked to focus simply on the *shuffle* operation entirely. The authors replace the ShuffleUnit with a Window-based Multi-head Self-Attention (WMSA) unit. The WMSA structure uses windows that are arranged to evenly partition the image in a non-overlapping manner, and while more computation friendly than standard Multi-head Self-Attention (MSA), the receptive field is limited within the window. To address the lack of information flow, the authors propose a spatial shuffle operation which mimics the channel shuffle operation of the original ShuffleNet paper. The result is a much more computationally efficient transformer network, but not adherent to the goals set forth in the **ShuffleNetV2** paper[2]. LeViT[18] the authors bring a number of things to light that shrinks ViT[19] models down, among them a computationally efficient patch extractor to shrink the number of features in the first layer. Other combinations of ViT and CNN have shown promise [32][33][34][35]. In MobileFormer[20], they introduce a structure that parallelizes MobileNet and Transformer together.

3 CupidShuffle

In order to abide by the practical guidelines proposed in [2] We replace the initial convolutional block of ShuffleNetV2 with a Window-based Multi-head Self-Attention (WMSA) module from the Shuffle Transformer[2]. Here, in order to adhere to G1, we propose augmenting the tokenization scheme from [2] by changing the inter-channel and embedding dimensions to have the same grouping (HWC), where this conveniently becomes same grouping as the input to the first ShuffleV2Unit. The standard ShuffleNetV2 Unit size is then used according the the 1.0 ratio from the paper, but instead of using a 3 blocks in the first layer, 7 blocks in the second layer, and 3 blocks in the third layer, we use a 1-4-1 format which allows for a near replication in ShuffleNetV2's output, which reducing the number of parameters and operations the original network needed. The intuition here was to take as many good things that very thoughtful people have done and simply make an attempt to limit the size of the network so that it would be applicable for low power scenarios.

4 Formats

Provided is the training code and a tvm, compiled shufflenet variant that will do forward inference. One thing to keep in mind is that on some variants of CUDA, we won't have access to THRUST and thus, some of the efficiencies of TVM won't be of use to us (like when we want to perform NMS for YOLO for example); so we completely remove NMS from the architecture and allow NMS to be executed outside TVM's compiler. Secondly, we want to provide a framework with a focus on CPU/ARM deployment and not GPU deployment. Thirdly, none of this really matters, go spend time with your kids. Thank you for coming to my TED talk. Code available at: https://github.com/KayneWest/cupidshuffle

Table 1: Small model family performance on CIFAR 100

network	params	top1 err	top5 err
mobilenet	3.3M	34.02	10.56
mobilenetv2	2.36M	31.92	09.02
squeezenet	**0.78M**	30.59	8.36
shufflenet	1.0M	29.94	8.35
shufflenetv2	1.3M	30.49	8.49
cupidshuffle128	1.3M	29.75	8.46
cupidshuffle64	1.1M	**29.54**	8.61
cupidshuffle28	1.03M	29.55	**8.34**

References

[1] Xiangyu Zhang, Xinyu Zhou, Mengxiao Lin, and Jian Sun. Shufflenet: An extremely efficient convolutionalneural network for mobile devices. In Proceedings of the IEEE conference on computer vision and patternrecognition, pages 6848–6856, 2018. 1, 3, 4

[2] Ningning Ma, Xiangyu Zhang, Hai-Tao Zheng, Jian Sun. ShuffleNet V2: Practical Guidelines for Efficient CNN Architecture Design

[3] Zilong Huang, Youcheng Ben, Guozhong Luo, Pei Cheng, Gang Yu, Bin Fu. Shuffle Transformer: Rethinking Spatial Shuffle for Vision Transformer

[4] Tianqi Chen, Thierry Moreau, Ziheng Jiang, Lianmin Zheng, Eddie Yan, Meghan Cowan, Haichen Shen, Leyuan Wang, Yuwei Hu, Luis Ceze, Carlos Guestrin, Arvind Krishnamurthy. TVM: An Automated End-to-End Optimizing Compiler for Deep Learning https://arxiv.org/abs/1802.04799

[5] Han Cai, Ligeng Zhu, Song Han. ProxylessNAS: Direct Neural Architecture Search on Target Task and Hardware. https://arxiv.org/abs/1812.00332

[6] Andrew G. Howard, Menglong Zhu, Bo Chen, Dmitry Kalenichenko, Weijun Wang, Tobias Weyand, Marco Andreetto, Hartwig Adam. MobileNets: Efficient Convolutional Neural Networks for Mobile Vision Applications

[7] Mark Sandler, Andrew Howard, Menglong Zhu, Andrey Zhmoginov, Liang-Chieh Chen. MobileNetV2: Inverted Residuals and Linear Bottlenecks. https://arxiv.org/abs/1801.04381

[8] Kevin Alexander Laube, Andreas Zell. ShuffleNASNets: Efficient CNN models through modified Efficient Neural Architecture Search. https://arxiv.org/abs/1812.02975

[9] Cheng Cui, Tingquan Gao, Shengyu Wei, Yuning Du, Ruoyu Guo, Shuilong Dong, Bin Lu, Ying Zhou, Xueying Lv, Qiwen Liu, Xiaoguang Hu, Dianhai Yu, Yanjun Ma. PP-LCNet: A Lightweight CPU Convolutional Neural Network. https://arxiv.org/abs/2109.15099

[10] Andrew Howard, Mark Sandler, Grace Chu, Liang-Chieh Chen, Bo Chen, Mingxing Tan, Weijun Wang, Yukun Zhu, Ruoming Pang, Vijay Vasudevan, Quoc V. Le, Hartwig Adam. Searching for MobileNetV3. https://arxiv.org/abs/1905.02244

[11] Mingxing Tan, Bo Chen, Ruoming Pang, Vijay Vasudevan and Quoc V. Le. Mnasnet: Platform-aware neural architec ture search for mobile. CoRR, abs/1807.11626, 2018. 2, 3, 5, 6

[12] Prajit Ramachandran, Barret Zoph, and Quoc V. Le. Searching for activation functions. CoRR, abs/1710.05941, 2017. 2, 4

[13] Stefan Elfwing, Eiji Uchibe, and Kenji Doya. Sigmoid weighted linear units for neural network function approxi mation in reinforcement learning. CoRR, abs/1702.03118, 2017. 2, 4

[14] Dan Hendrycks and Kevin Gimpel. Bridging nonlinearities and stochastic regularizers with gaussian error linear units. CoRR, abs/1606.08415, 2016. 2, 4

[15] R. Avenash and P. Vishawanth. Semantic segmentation of satellite images using a modified cnn with hard-swish activation function. In VISIGRAPP, 2019. 2, 4

[16] Matthieu Courbariaux, Yoshua Bengio, and Jean-Pierre David. Binaryconnect: Training deep neural networks with binary weights during propagations. CoRR, abs/1511.00363, 2015. 2, 4

[17] J. Hu, L. Shen, and G. Sun. Squeeze-and-Excitation Networks. ArXiv e-prints, Sept. 2017. 2, 3, 7

[18] Benjamin Graham Alaaeldin El-Nouby Hugo Touvron Pierre Stock Armand Joulin Herve Jegou Matthijs Douze. LeViT: a Vision Transformer in ConvNet's Clothing for Faster Inference

[19] Alexey Dosovitskiy, Lucas Beyer, Alexander Kolesnikov, Dirk Weissenborn, Xiaohua Zhai, Thomas Unterthiner, Mostafa Dehghani, Matthias Minderer, Georg Heigold, Sylvain Gelly et al., "An image is worth 16x16 words: Transformers for image recognition at scale," arXiv preprint arXiv:2010.11929, 2020. 1, 2, 3, 4, 5, 7

[20] Mobile-Former: Bridging MobileNet and Transformer Yinpeng Chen, Xiyang Dai, Dongdong Chen, Mengchen Liu, Xiaoyi Dong, Lu Yuan, Zicheng Liu

[21] EfficientFormer: Vision Transformers at MobileNet Speed Yanyu Li, Geng Yuan, Yang Wen, Ju Hu, Georgios Evangelidis, Sergey Tulyakov, Yanzhi Wang, Jian Ren

[22] Yinpeng Chen, Xiyang Dai, Dongdong Chen, Mengchen Liu, Xiaoyi Dong, Lu Yuan, Zicheng Liu Mobile-Former: Bridging MobileNet and Transformer

[23] Yunsheng Li, Yinpeng Chen, Xiyang Dai, Dongdong Chen, Mengchen Liu, Lu Yuan, Zicheng Liu, Lei Zhang, and Nuno Vasconcelos. Micronet: Improving image recognition with extremely low flops. In International Conference on Computer Vision, 2021. 1, 2, 4, 11, 12

[24] Kai Han, Yunhe Wang, Qiulin Zhang, Wei Zhang, Chunjing XU, and Tong Zhang. Model rubiks cube: Twisting resolu tion, depth and width for tinynets. In H. Larochelle, M. Ranzato, R. Hadsell, M. F. Balcan, and H. Lin, editors, Advances in Neural Information Processing Systems, volume 33, pages 19353–19364. Curran Associates, Inc., 2020

[25] Kai Han, Yunhe Wang, Qi Tian, Jianyuan Guo, Chunjing Xu, and Chang Xu. Ghostnet: More features from cheap opera tions. In IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR), June 2020. 1, 2, 6

[26] Hanting Chen, Yunhe Wang, Chunjing Xu, Boxin Shi, Chao Xu, Qi Tian, and Chang Xu. Addernet: Do we really need multiplications in deep learning? In Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR), June 2020.

[27] Mingxing Tan and Quoc V. Le. Mixconv: Mixed depthwise convolutional kernels. In 30th British Machine Vision Conference 2019, 2019

[28] Daquan Zhou, Qi-Bin Hou, Y. Chen, Jiashi Feng, and S. Yan. Rethinking bottleneck structure for efficient mobile network design. In ECCV, August 2020.

[29] Mingxing Tan and Quoc Le. Efficientnet: Rethinking model scaling for convolutional neural networks. In ICML, pages 6105–6114, Long Beach, California, USA, 09–15 Jun 2019. 2, 5, 6

[30] Mingxing Tan, Ruoming Pang, and Quoc V. Le. Efficientdet: Scalable and efficient object detection. In Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR), June 2020

[31] ai Han, Yunhe Wang, Qiulin Zhang, Wei Zhang, Chunjing XU, and Tong Zhang. Model rubiks cube: Twisting resolution, depth and width for tinynets. In H. Larochelle, M. Ranzato, R. Hadsell, M. F. Balcan, and H. Lin, editors, Advances in Neural Information Processing Systems, volume 33, pages 19353–19364. Curran Associates, Inc., 2020

[32] Aravind Srinivas, Tsung-Yi Lin, Niki Parmar, Jonathon Shlens, Pieter Abbeel, and Ashish Vaswani. Bottleneck transformers for visual recognition. In Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR), pages 16519–16529, June 2021

[33] Stephane d'Ascoli, Hugo Touvron, Matthew Leavitt, Ari Morcos, Giulio Biroli, and Levent Sagun. Convit: Improving vision transformers with soft convolutional inductive biases. arXiv preprint arXiv:2103.10697, 2021

[34] Haiping Wu, Bin Xiao, Noel Codella, Mengchen Liu, Xiyang Dai, Lu Yuan, and Lei Zhang. Cvt: Introducing convolutions to vision transformers, 2021

[35] Tete Xiao, Mannat Singh, Eric Mintun, Trevor Darrell, Piotr Dollar, and Ross B. Girshick. Early convolutions help trans formers see better. CoRR, abs/2106.14881, 2021. 1, 2, 3, 4, 5, 6