A. Training Details

A.1. Sensitivity of Convolutional Filters to StrassenNets

We generate a training set containing 100k pairs (A_i, B_i) with values i.i.d. uniform on $[-1, 1]$ in A_i, and values of a given convolutional filter in B_i. The SPN is then trained using different number of hidden units. We begin training with full-precision weights (initialized i.i.d. uniform on $[-1, 1]$) for one epoch with SGD (learning rate 0.1, momentum 0.9, mini-batch size 4), activate quantization, and train for few epochs with initial learning rate of 0.01 and progressively smaller learning rates. Once the training converges after activation of the quantization, we collect the L2-loss.

A.2. Hyperparameters Settings for Training Hybrid MobileNets

Table 3: Hyperparameters for training Hybrid MobileNets

Training phase	Hyperparameters
Train using full-precision strassen matrices	Batch size per GPU: 128 Number of GPUs used: 4 Optimizer: Nesterov accelerated gradient (NAG) (Momentum: 0.9, Weight decay: 0.0001) Number of epochs: 200 Weight initialization: Xavier Initial, final learning rate: 0.2, 0.0 Learning rate schedule: cosine decay Number of warmup epochs: 5 Starting warmup learning rate: 0.0 Size of the input image: 224 x 224 x 3
Activate quantization for strassen matrices	Batch size per GPU: 128 Number of GPUs used: 4 Optimizer: Nesterov accelerated gradient (NAG) (Momentum: 0.9, Weight decay: 0.0001) Number of epochs: 75 Initial, final learning rate: 0.02, 0.0 Learning rate schedule: cosine decay
Freeze strassen matrices to ternary values	Batch size per GPU: 128 Number of GPUs used: 4 Optimizer: Nesterov accelerated gradient (NAG) (Momentum: 0.9, Weight decay: 0.0001) Number of epochs: 25 Initial, final learning rate: 0.002, 0.0 Learning rate schedule: cosine decay

The training images from ImageNet are preprocessed by using mean and standard deviation. These images are resized such that the shorter side has length of 256 and are then randomly cropped to 224×224 pixels. Random horizontal flips are applied for data augmentation. The center 224×224 crop of the images are used for evaluation.

Table 3 shows the hyperparameters values used for training Hybrid MobileNets. Similar hyperparameters values are used for training baseline full-precision MobileNets and ST-MobileNets also. The learning rate scheduling involves a 'warm up' period in which the learning rate is annealed from zero to 0.2 over the first 5 epochs, after which it is gradually reduced following a cosine decay function.

B. Group of Filters with Sub-Filter Similarities

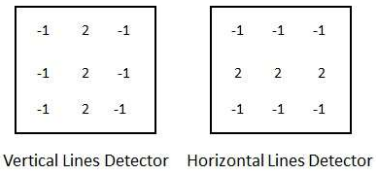

Figure 5: A 3×3 vertical lines detector and a horizontal lines detector sharing common values at all the corners and at the center.

C. Performance of Per-Layer Hybrid Filter Banks on the ResNet-20 Architecture

Table 4: Accuracy along with the computational costs, model size, and energy per inference for baseline ResNet-20, ST-ResNet-20, and ResNet-20 with hybrid filter banks on the CIFAR-10 dataset. α is the fraction of channels generated by the full-precision weight filters at each layer, c_{out} is the number of remaining channels generated by the ternary strassen filters at the corresponding convolutional layer, r is the hidden layer width of the strassenified convolutions. The last column shows the throughput of proposed models on an area-equivalent hardware accelerator comprising both MAC and adder units when compared to the throughput of baseline MobileNets with 16-bit floating-point weights on a MAC-only accelerator.

Network	Alpha (α)	r	Acc. (%)	Muls, Adds	MACs	Model size	Energy/inference (normalized)	Throughput (normalized)
ResNet-20	-	-	92.1	-	40.81M	530.64KB	1	1
ST-ResNet-20 [41]	0	$0.25c_{out}$	85.46	0.05M, 11.78M	-	21.97KB	0.05	6.92
		$0.5c_{out}$	88.63	0.1M, 23.35M	-	41.15KB	0.11	3.49
		$0.75c_{out}$	90.62	0.15M, 34.93M	-	60.32KB	0.17	2.33
		c_{out}	91.24	0.2M, 46.51M	-	79.5KB	0.23	1.75
ResNet-20 (Hybrid filter banks)	0.125	$0.25c_{out}$	88.5	0.04M, 10.18M	5.1M	85.30KB	0.17	4.0
		$0.5c_{out}$	90.39	0.09M, 20.16M	5.1M	101.83KB	0.22	2.68
		$0.75c_{out}$	91.03	0.13M, 30.14M	5.1M	118.36KB	0.27	2.02
		c_{out}	91.45	0.18M, 40.12M	5.1M	134.88KB	0.32	1.62
ResNet-20 (Hybrid filter banks)	0.25	$0.25c_{out}$	89.83	0.04M, 8.62M	10.2M	148.71KB	0.29	2.81
		$0.5c_{out}$	90.79	0.08M, 17.05M	10.2M	162.66KB	0.33	2.17
		$0.75c_{out}$	91.55	0.11M, 25.48M	10.2M	176.61KB	0.37	1.77
		c_{out}	91.79	0.15M, 33.9M	10.2M	190.56KB	0.41	1.5
ResNet-20 (Hybrid filter banks)	0.375	$0.25c_{out}$	90.36	0.03M, 7.11M	15.3M	212.18KB	0.4	2.16
		$0.5c_{out}$	91.19	0.06M, 14.02M	15.3M	223.63KB	0.44	1.82
		$0.75c_{out}$	91.38	0.09M, 20.94M	15.3M	235.07KB	0.47	1.58
		c_{out}	91.88	0.13M, 27.85M	15.3M	246.52KB	0.51	1.39

D. Comparison against Prior Works

Table 5: Top-1 and top-5 accuracy (%) of Mobilenet (full resolution and multiplier of 0.5) on Imagenet for different number of bits per weight and activation.

Method	#bits per weight/activation	Top-1 Acc. (%)	Top-5 Acc. (%)
Baseline MobileNets[4]	32/32	65.53	86.48
Baseline MobileNets[5]	16/16	65.2	86.34
ST-MobileNets ($r = 0.5c_{out}$)	2/16	48.92	73.68
ST-MobileNets ($r = 0.75c_{out}$)	2/16	56.95	80.25
ST-MobileNets ($r = c_{out}$)	2/16	61.8	83.97
ST-MobileNets ($r = 2c_{out}$)	2/16	65.14	86.26
Hybrid MobileNets ($\alpha = 0.25, r = c_{out}$)	2,16/16	63.62	84.98
Hybrid MobileNets ($\alpha = 0.25, r = 1.33c_{out}$)	2,16/16	63.47	85.11
Hybrid MobileNets ($\alpha = 0.25, r = 2c_{out}$)	2,16/16	64.84	85.86
Hybrid MobileNets ($\alpha = 0.375, r = c_{out}$)	2,16/16	64.13	85.4
Hybrid MobileNets ($\alpha = 0.375, r = 1.6c_{out}$)	2,16/16	64.17	85.38
Hybrid MobileNets ($\alpha = 0.375, r = 2c_{out}$)	2,16/16	65.2	86.05
Hybrid MobileNets ($\alpha = 0.5, r = c_{out}$)	2,16/16	64.69	85.66
Hybrid MobileNets ($\alpha = 0.5, r = 2c_{out}$)	2,16/16	65.17	85.98
Baseline MobileNets[6]	32/32	63.3	84.9
Baseline MobileNets[7]	8/8	62.2	-
Alpha-blending [29]	8/8	63	-
Alpha-blending [29]	4/8	58.4	-
HAQ [42]	2/-	57.14	81.87
Relaxed Quantization [30]	Does not demonstrate potential for MobileNets with ternary weights		
Quantization Networks [45]	Does not demonstrate potential for MobileNets with ternary weights		
Differentiable Soft Quantization [11]	Does not demonstrate potential for MobileNets with ternary weights		
Quantization Intervals [26]	Does not demonstrate potential for MobileNets with ternary weights		
Post-training 4-bit quantization [4]	Does not demonstrate potential for MobileNets with ternary weights		
Low-bit Quantization [9]	Does not demonstrate potential for MobileNets with ternary weights		