IMAGE CLASSIFICATION

MACHINE LEARNING ON MICROCONTROLLERS

Carolina Lopez and Victor Sainz
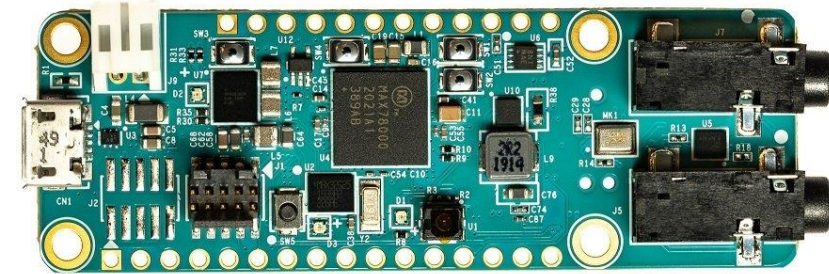
OUR GOAL

IMAGE CLASSIFICATION ON MULTIPLE DATASETS
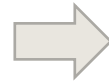BENCHMARKING AND ANALYSIS

CIFAR 100
60.000 32x32

100 classes

600 images / class

50.000 training (500 per class)

10.000 testing (100 per class)

DATA AUGMENTATION
AND ANALYSIS OF ITS IMPACT

- ✓ `RandomCrop`
- ✓ `RandomHorizontalFlip`
- ✓ `RandomRotation`
- ✓ `ColorJitter`
- ✓ `Added Gaussian Noise with std = 0.1`
- ✓ `Normalize`

CIFAR 10
60.000 32x32
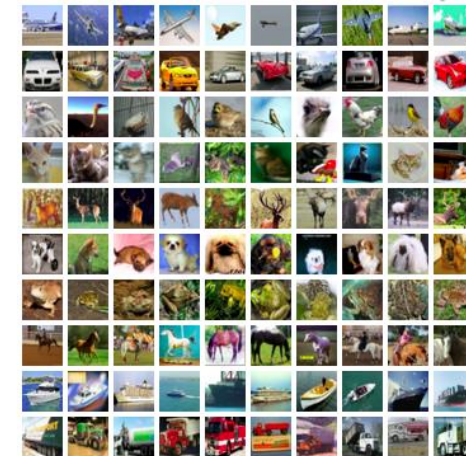
10 classes

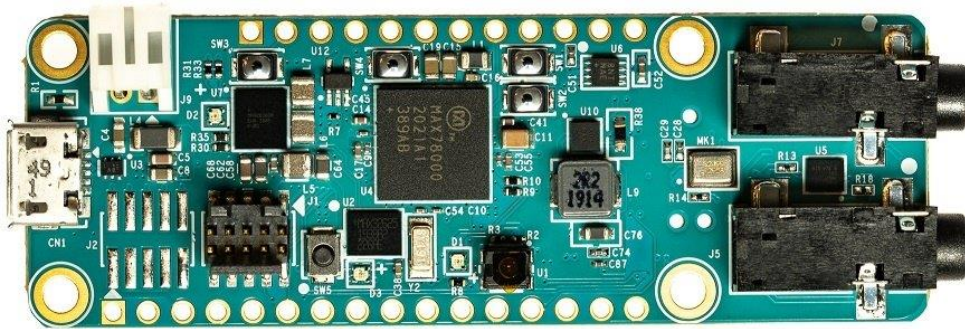**6.000 images / class**

50.000 training (5000 per class)

10.000 testing (1000 per class)

OUR MICROCONTROLLER | MAX78000FTHR BOARD

- Arm Cortex-M4 Processor with FPU up to 100MHz
- 512KB Flash
- 128KB SRAM
- Image Sensor
- DAPLINK Interface

STATE OF THE ART

MULTIPLE PAPERS FOUND TESTED IN MCUs

SIMPLENET

[1] Seyyed Hossein Hasanpour, Mohammad Rouhani, Mohsen Fayyaz, Mohammad Sabokrou "Lets keep it simple, Using simple architectures to outperform deeper and more complex architectures."
https://arxiv.org/abs/1608.06037

CIFAR 10	
Original parameters	5.4 M
SimpleNet-Arch1 Accuracy	94.75 %
SimpleNet-Arch2 Accuracy	95.32 %
Reduced parameters	310K – 460K
SimpleNet Accuracy	91.98 – 92.33 %
CIFAR 100	
Original parameters	5.4 M
SimpleNet-Arch1 Accuracy	73.45 %
SimpleNet-Arch2 Accuracy	74.86 %
Reduced parameters	310K – 460K
SimpleNet Accuracy	64.68 – 66.82 %

MODEL FOR STM32

[2] Lai, Liangzhen, Naveen Suda, and Vikas Chandra. "CMSIS-NN: Efficient Neural Network Kernels for Arm Cortex-M CPUs."
https://arxiv.org/abs/1801.06601

CIFAR 10	
Platform	Arm Cortex-M7	STM32
Time/Inference	99.1 ms (10.1 images)
Throughput (MAC/Cycle)	249 MOps
Memory footprint	133 kB
Accuracy	79.9 %

NUCLEO-F746ZG The STM32 Nucleo-144 – 216 MHz
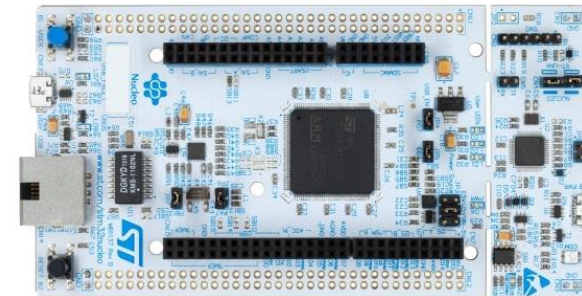
EXISTING MODELS

CIFAR100
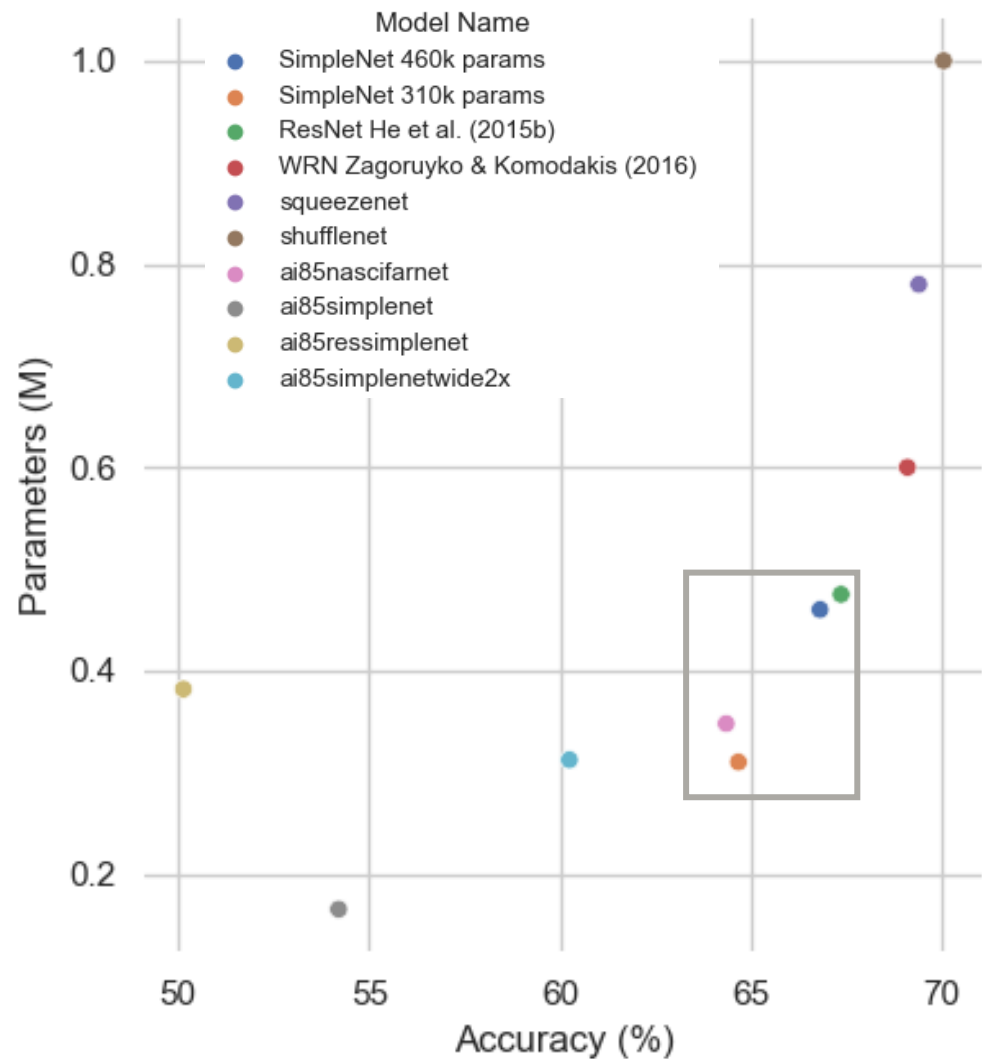
CIFAR10
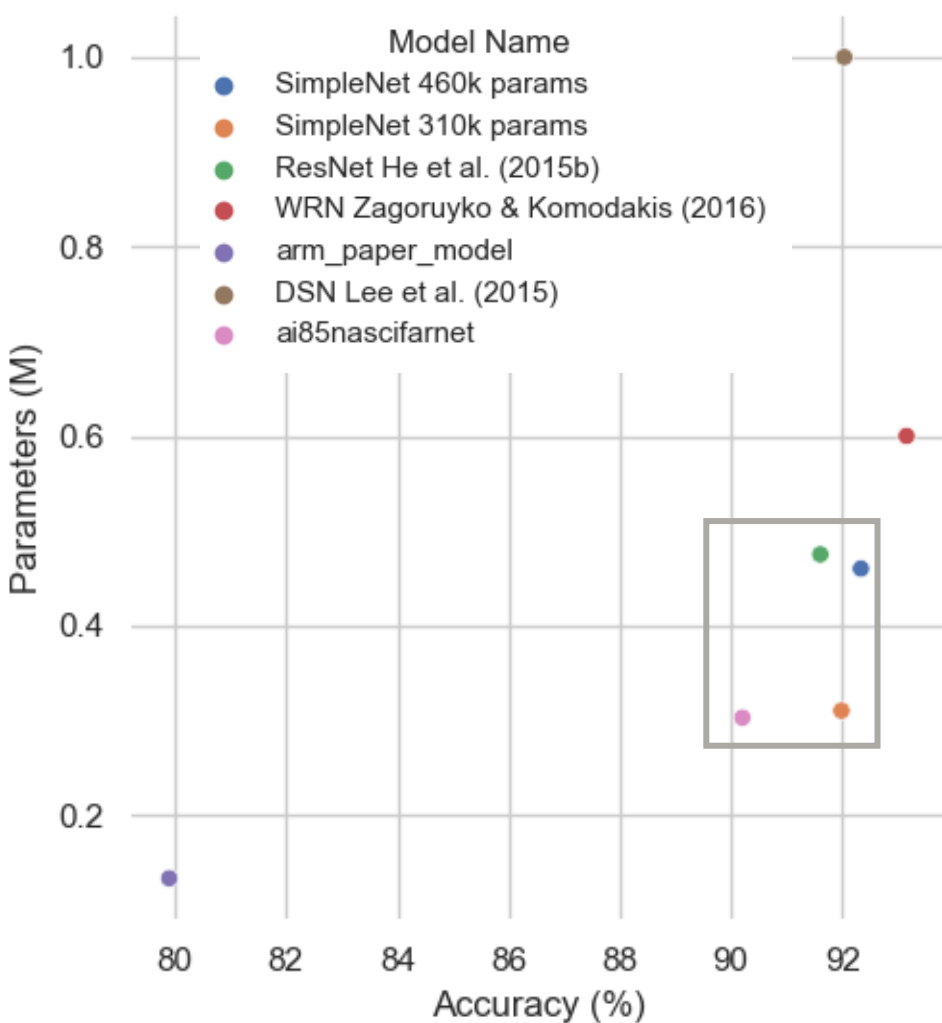
Legend:
- ResNet He et al. (2015b)
- SimpleNet 460K
- SimpleNet 310K
- ai85nascifarnet

CIFARNAS

Best performing model from maxim

FusedConv2dBNReLU(3, 64, 3, stride=1, padding=1)

FusedConv2dBNReLU(64, 32, 1, stride=1, padding=0)

FusedConv2dBNReLU(32, 64, 3, stride=1, padding=1)

FusedMaxPoolConv2dBNReLU(64, 32, 3, stride=1, padding=1)

FusedConv2dBNReLU(32, 64, 1, stride=1, padding=0)

FusedMaxPoolConv2dBNReLU(64, 128, 3, stride=1, padding=1)

FusedConv2dBNReLU(128, 128, 1, stride=1, padding=0)

FusedMaxPoolConv2dBNReLU(128, 64, 3, stride=1, padding=1)

FusedConv2dBNReLU(64, 128, 3, stride=1, padding=1)

FusedMaxPoolConv2dBNReLU(128, 128, 1, stride=1, padding=1)

Linear (2x2x128, 10)

SIMPLENET

Inspired from the layer architecture of the SimpleNet paper

FusedConv2dBNReLU(3, 64, 3, stride=1, padding=1)

FusedConv2dBNReLU(64, 32, 3, stride=1, padding=1)

FusedConv2dBNReLU(32, 32, 3, stride=1, padding=1)

FusedConv2dBNReLU(32, 32, 3, stride=1, padding=1)

MaxPool2d (kernel_size=2, pool_stride=2)

Dropout (0.2)

FusedConv2dBNReLU(32, 32, 3, stride=1, padding=1)

FusedConv2dBNReLU(32, 32, 3, stride=1, padding=1)

FusedConv2dBNReLU(32, 32, 3, stride=1, padding=1)

MaxPool2d (kernel_size=2, pool_stride=2)

Dropout (0.3)

FusedConv2dBNReLU(32, 32, 3, stride=1, padding=1)

FusedConv2dBNReLU(32, 32, 3, stride=1, padding=1)

MaxPool2d (kernel_size=2, pool_stride=2)

Dropout (0.3)

FusedConv2dBNReLU(32, 32, 3, stride=1, padding=1)

MaxPool2d (kernel_size=2, pool_stride=2)

Dropout (0.3)

FusedConv2dBNReLU(32, 64, 3, stride=1, padding=1)

FusedConv2dBNReLU(64, 128, 3, stride=1, padding=1)

MaxPool2d (kernel_size=2, pool_stride=2)

Dropout (0.3)

FusedConv2dBNReLU(128, 256, 3, stride=1, padding=1)

Linear (1x1x256, 10)

CMSISNNET

Inspired from the architecture of the paper evaluated in STM32

FusedConv2dBNReLU(3, 32, 3, stride=1, padding=1)

FusedConv2dBNReLU(32, 32, 3, stride=1, padding=1)

FusedMaxPoolConv2dBNReLU(32, 32, 3, stride=1, padding=1)

FusedConv2dBNReLU(32, 32, 3, stride=1, padding=1)

FusedMaxPoolConv2dBNReLU(32, 64, 3, stride=1, padding=1)

FusedConv2dBNReLU(64, 64, 3, stride=1, padding=1)

MaxPool2d (kernel_size=2, pool_stride=2)

Linear (4x4x64, 10)
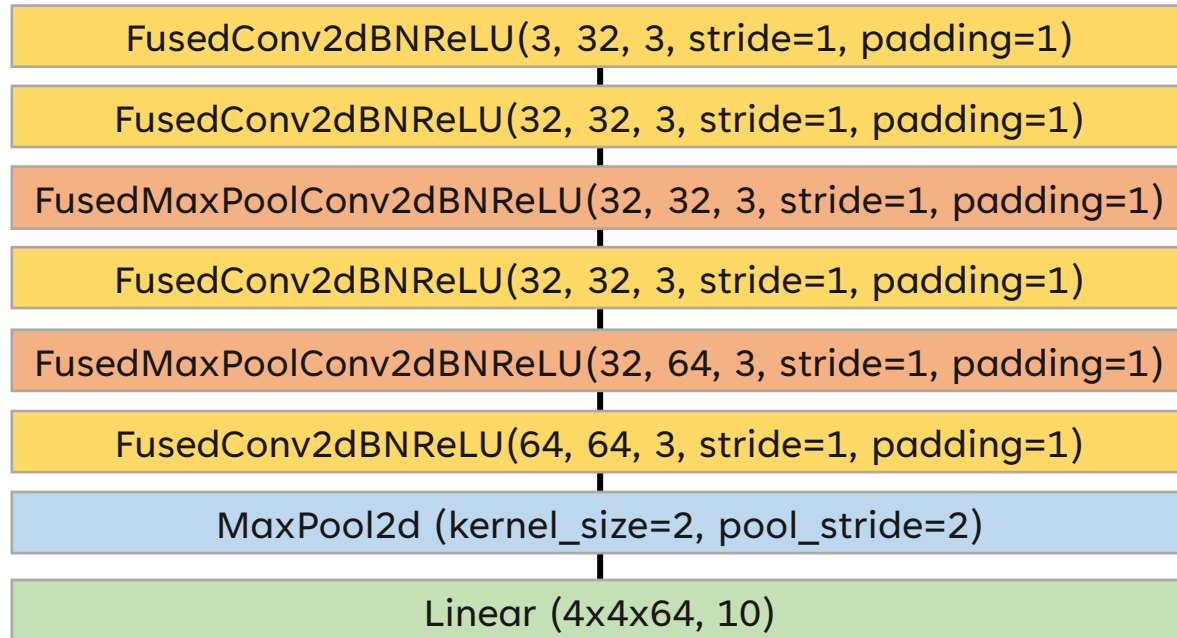
	Layer Type	Filter Shape	Output Shape
Layer 1	Convolution	5x5x3x32 (2.3 KB)	32x32x32 (32 KB)
Layer 2	Max Pooling	N.A.	16x16x32 (8 KB)
Layer 3	Convolution	5x5x32x32 (25 KB)	16x16x32 (8 KB)
Layer 4	Max Pooling	N.A.	8x8x32 (2 KB)
Layer 5	Convolution	5x5x32x64 (50 KB)	8x8x64 (4 KB)
Layer 6	Max Pooling	N.A.	4x4x64 (1 KB)
Layer 7	Fully-connected	4x4x64x10 (10 KB)	10

RESULTS

CIFAR10 – PAPERS COMPARISON

	CmsisnNet		SimpleNet	
	STM32 - REFERENCE	**MAXIM**	**PAPER - REFERENCE**	**MAXIM**
Input Size	32x32x3	32x32x3	32x32x3	32x32x3
Model parameters		94,314	310K – 460K	401,599
Weight + Bias memory	133 kB	94.30 kB	310 kB – 460 kB	401.60 kB
MACs	-	16,820,224	-	-
Time/inference	99.1 ms **10.1 images**	1.97 ms **507.6 images** **X 50.30 times faster**	-	-
Throughput	249.00 MOps	170.76 MOps	-	-
Accuracy	79.9 %	87.15 % **+ 7.25%**	91.98 – 92.33 %	87.57 %

RESULTS

CIFAR10 – AI85NASCIFARNET COMPARISON

	ai85nascifarnet	Nascifarnet		
	REFERENCE BY MAXIM	QAT	WO/Q	PTQ
Input Size	32x32x3	32x32x3	32x32x3	32x32x3
Model parameters	302,602	302,602	302,602	302,602
Weight + Bias memory	302.60 kB	302.60 kB	1210.4 kB	302.60 kB
MACs	36,180,992	36,180,992	✓	36,180,992
Time/inference	4.57 ms **218.81 (images/sec)**	4.57 ms	✓	4.58 ms
MACs/Cycle	158.34 MOps	158.34 MOps	✓	157.99 MOps
Accuracy	90.03% ***600 epochs**	90.13% ***150 epochs**	85.88%	85.76%

COMPARISON OF SYNTHESIZED MODELS WITH CIFAR10
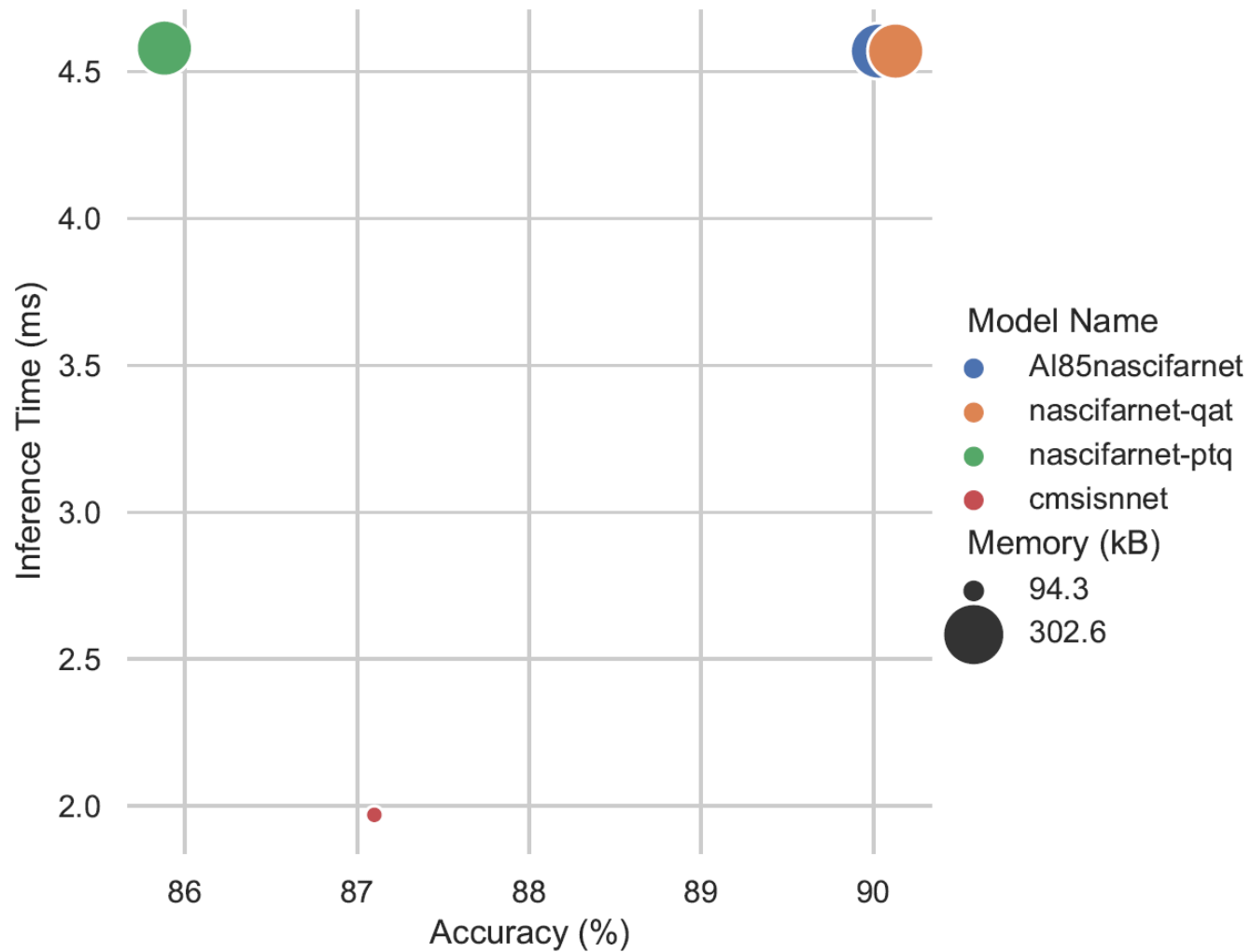
- Similar accuracy

- Cmsisnnet – lowest inference

- Best accuracy – cifarnas (data aug)

STM32
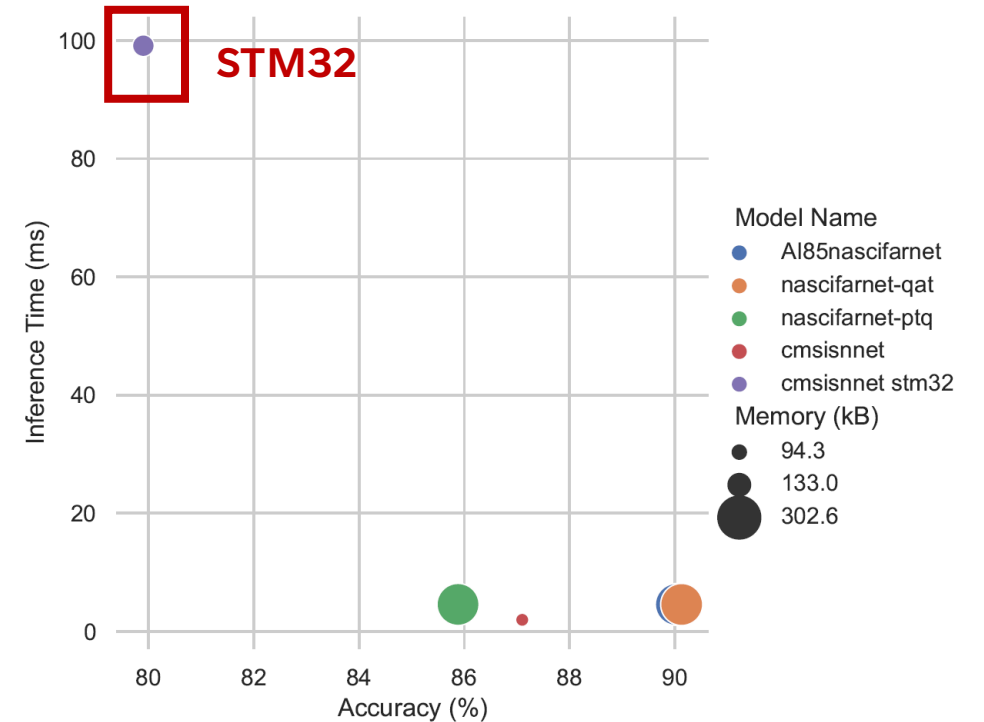
RESULTS

CIFAR100

	Nascifarnet	
	REFERENCE BY MAXIM	**OUR TRAINING**
Input Size	32x32x3	32x32x3
Model parameters	347840	347840
Weight + Bias memory	348.77 kB	348.77 kB
MACs	36,227,072	36,227,072
Time/inference	4584 us	4584 us
Throughput	158.05 MOps	158.05 MOps
Accuracy	64.36% ***600 epochs**	49.18% ***150 epochs**

CONCLUSIONS

CIFAR10

CMSISNNET is 50.3 times faster ran in MAXIM78000 than in the STM32 Nucleo-144 used by the paper.
- We are able to classify **507.6 images/s** with an accuracy of **87.15 %**
- Where the paper presented **10.1 images/s** with an accuracy of **79.9%**

NASCIFARNET showed an improvement in accuracy after performing data augmentation and modifying the learning rate of **0.10% with -450 epochs**
- We confirmed **QAT yields a better accuracy** compared to PTQ, specifically **+4.25%**

SIMPLENET showed lower accuracy than the model presented by the paper **-4%**.
We have used 50 epochs. There is no information about the paper's model.

CIFAR100

NASCIFARNET showed reduced accuracy but there isn't a fair comparison.
150 epochs << 600

CHALLENGES WITH THE CAMERA

MAIN CHALLENGE:

THE IMAGE <u>FORMAT OF THE CAMERA </u>IS DIFFERENT TO THE <u>ONE THE NEURAL NETWORK </u>USES ->
we had to find that conversion to:

1. Visualize the images from the camera in PNG format
2. Convert the sample inputs given to PNG
3. Change the output given by the camera to the format required by the NN

DEMO!
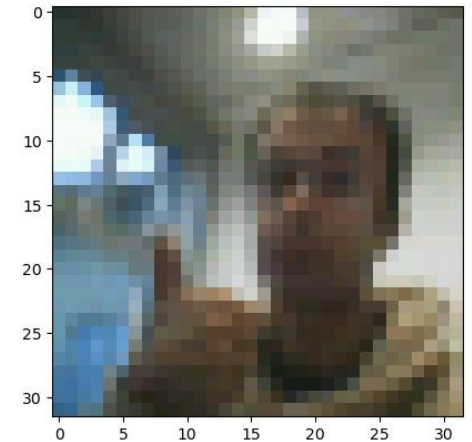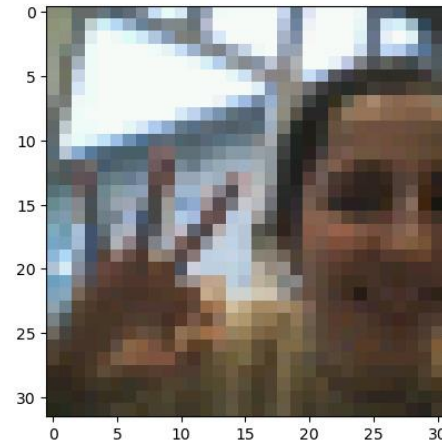
QUESTIONS?

THANK YOU FOR YOUR ATTENTION