Foundations of Edge AI

Lecture14

MCUNet: TinyML on Microcontrollers

Lanyu (Lori) Xu

Email: lxu@oakland.edu
Homepage: https://lori930.github.io/
Office: EC 524

Lecture Plan

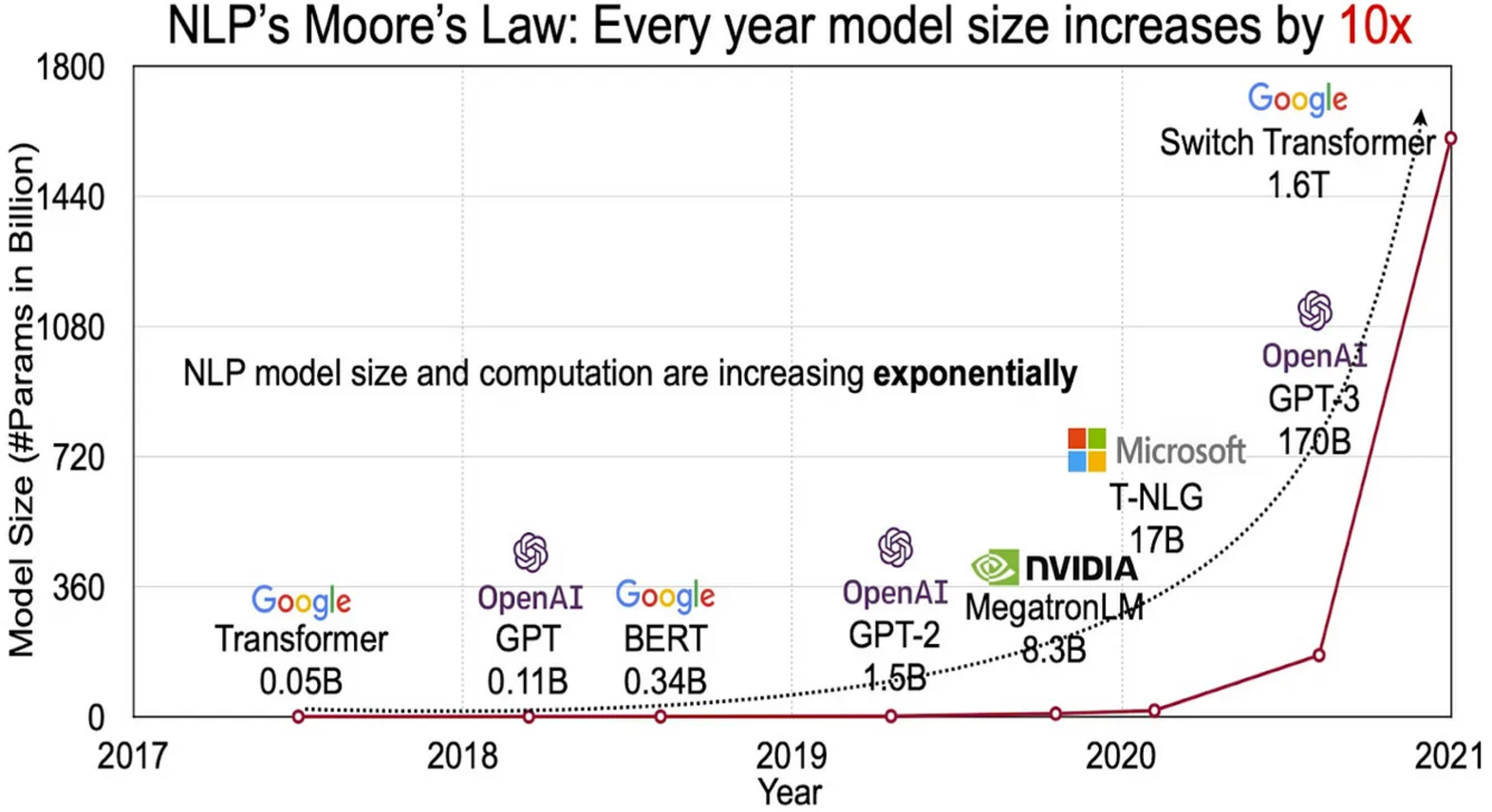
Today we will

- Introduce TinyML
- Understand the challenges of TinyML
- Present tiny neural network design (MCUNet, MCUNetV2)
- Explore MCUNet applications

What is TinyML?

Today's AI is too BIG

Need new algorithms and hardware for TinyML and GreenAI



Source

Deep Learning for Games

AlphaGo, AlphaZero, and AlphaMaster

AlphaGo

AlphaZero

AlphaMaster

- AlphaGo: 1,920 CPUs and 2,80 GPUs (**$3,000 electric bill** per game)
- AlphaZero: 5,000 TPUs for training over several days.

Deep Learning for Scientific Discovery

AlphaFold reveals the structure of the protein universe

The Nobel Prize in Chemistry 2024

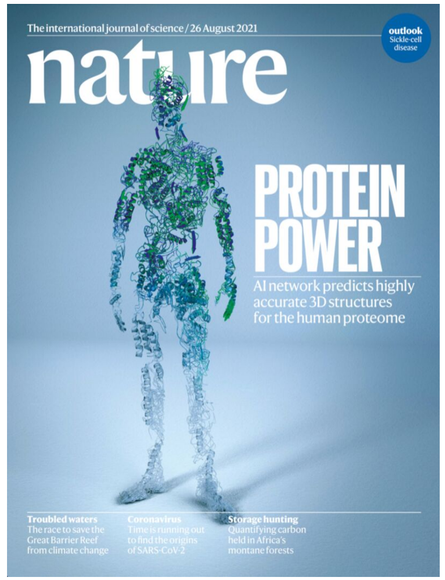
The Royal Swedish Academy of Sciences has decided to award the Nobel Prize in Chemistry 2024 with one half to and the other half jointly to

David Baker
University of Washington, Seattle, WA, USA
Howard Hughes Medical Institute, USA.
"for computational protein design"

Demis Hassabis
Google DeepMind, London, UK

John Jumper
Google DeepMind, London, UK
"for protein structure prediction"

They cracked the code for proteins' amazing structures



AlphaFold (Nature 2021)

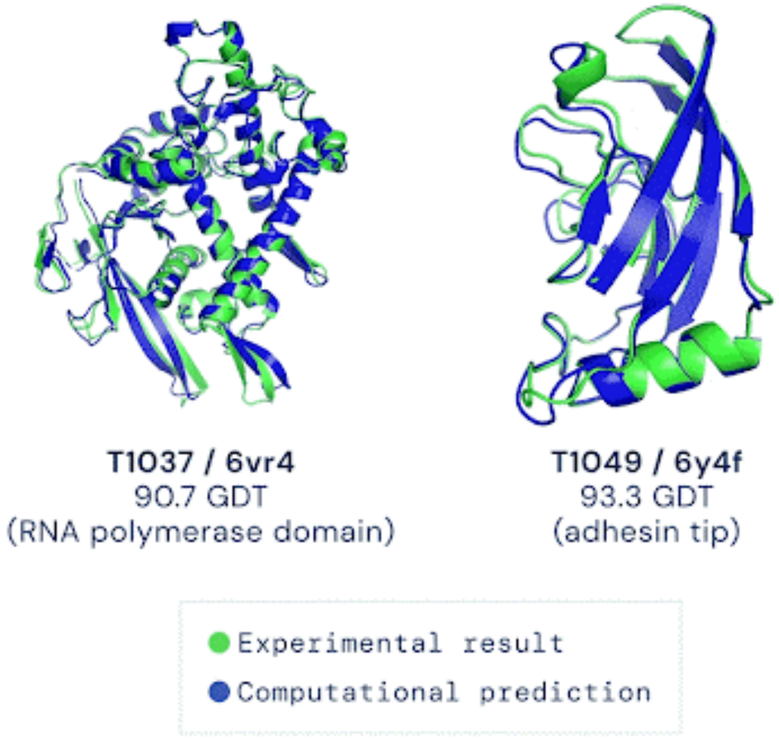
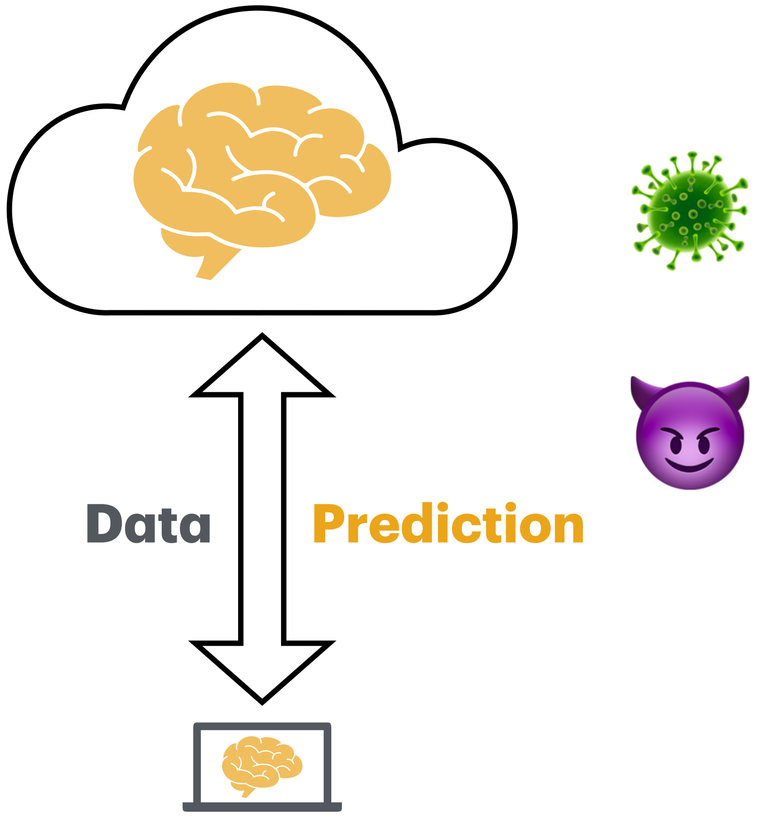


Image source

- 16 TPUv3s (**128 TPUv3 cores**) run over a few weeks

Deep Learning Going “Tiny”

AI-as-a-Service is easy to be attacked



Deep Learning Going “Tiny”

40 billion connected IoT devices by 2030

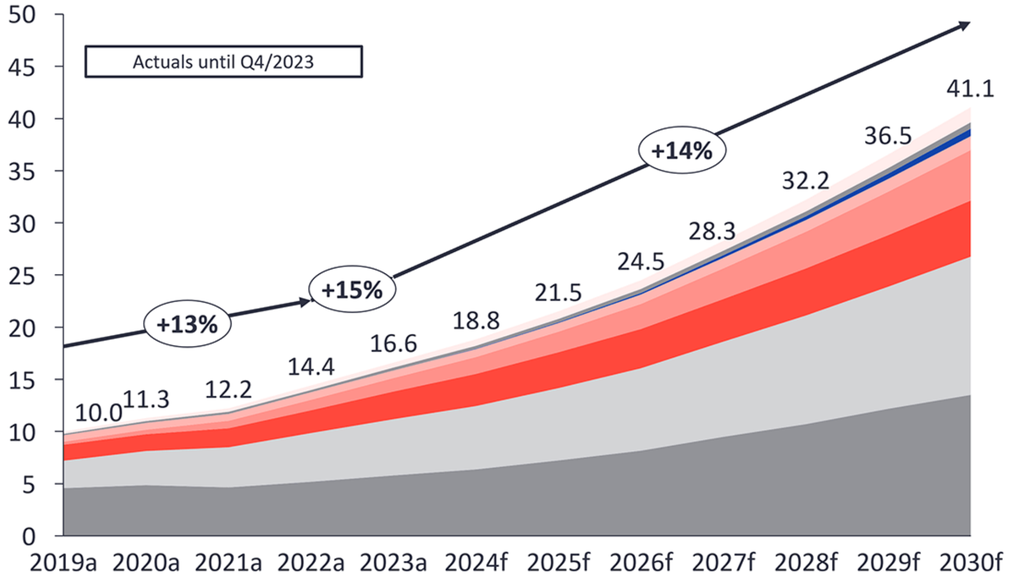
IOT ANALYTICS

September 2024

Your Global IoT Market Research Partner

Global IoT market forecast (in billions of connected IoT devices)

Number of global active IoT connections (installed base) in billions



Connectivity type	CAGR 21–23	CAGR 23–30
Other	21%	17%
Wireless neighborhood area networks (WNAN)	15%	14%
Cellular 5G IoT	147%	62%
Wired IoT	4%	9%
LPWA	35%	21%
Cellular IoT (excl. 5G, LPWA)	21%	11%
Wireless local area networks (WLAN)	18%	14%
Wireless personal area networks (WPAN)	12%	13%

XX% = CAGR

Note: IoT connections do not include any computers, laptops, fixed phones, cellphones, or consumers tablets. Counted are active nodes/devices or gateways that concentrate the end-sensors, not every sensor/actuator. Simple one-directional communications technology not considered (e.g., RFID, NFC). Wired includes ethernet and fieldbuses (e.g., connected industrial PLCs or I/O modules); Cellular includes 2G, 3G, 4G, 5G; LPWA includes unlicensed and licensed low-power networks; WPAN includes Bluetooth, Zigbee, Z-Wave or similar; WLAN includes Wi-Fi and related protocols; WNAN includes non-short-range mesh, such as Wi-SUN; Other includes satellite and unclassified proprietary networks with any range.
Source: IoT Analytics Research 2024-State of IoT Summer 2024. We welcome resharing: Please attribute this image to its original source and include a link back to the original article.

IoT Analytics

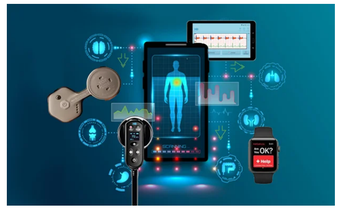
Deep Learning Going “Tiny”

Cloud → Mobile → Tiny: Squeeze deep learning into IoT devices

- Billions of IoT devices around the world are based on **microcontrollers**
- **Low-cost** for democratizing AI: normal people can afford access
- **Low-power** for Green AI: reduce carbon
- Privacy-preserving for secure AI: no data sharing
- Wide range of applications

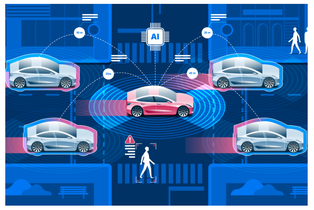

Smart Manufacturing


Disaster Relief


Connected Health


Smart City

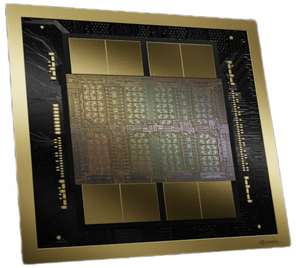

Smart Home


Autonomous Driving

Why TinyML is Challenging?

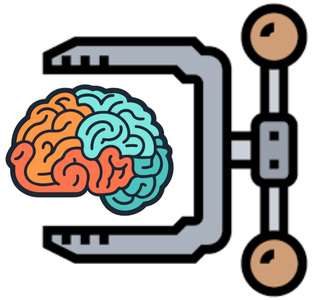
TinyML is Challenging

Memory size is too small to hold ML models



	Cloud AI	Mobile AI	Tiny AI
Memory (Activation)	80GB	~GB	**320KB**
Storage (Weights)	~TB/PB	~GB	**1MB**

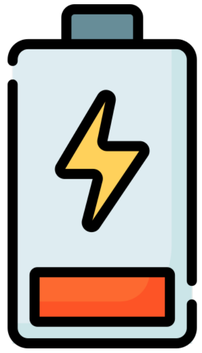
- $10^5 \times$ smaller than Cloud AI, $10^4 \times$ smaller than Mobile AI
- Reduce both **weights** and **activation** to fit DNNs for TinyML



TinyML is Challenging

Memory size is too small to hold ML models

Mobile AI

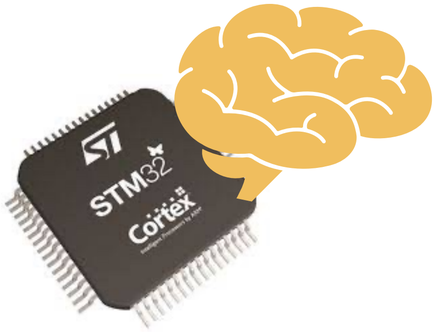


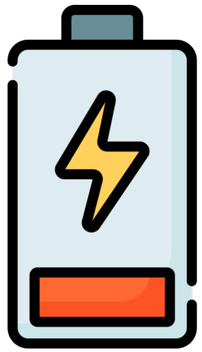
Latency constraint

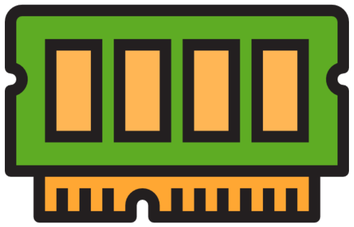
Energy constraint

Memory constraint

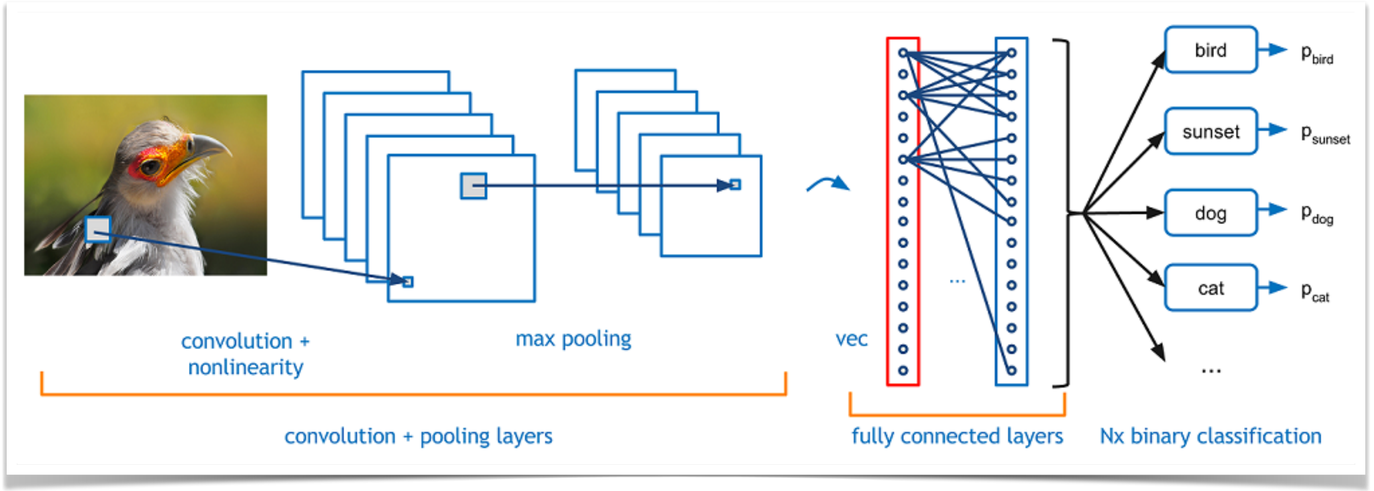
Tiny AI



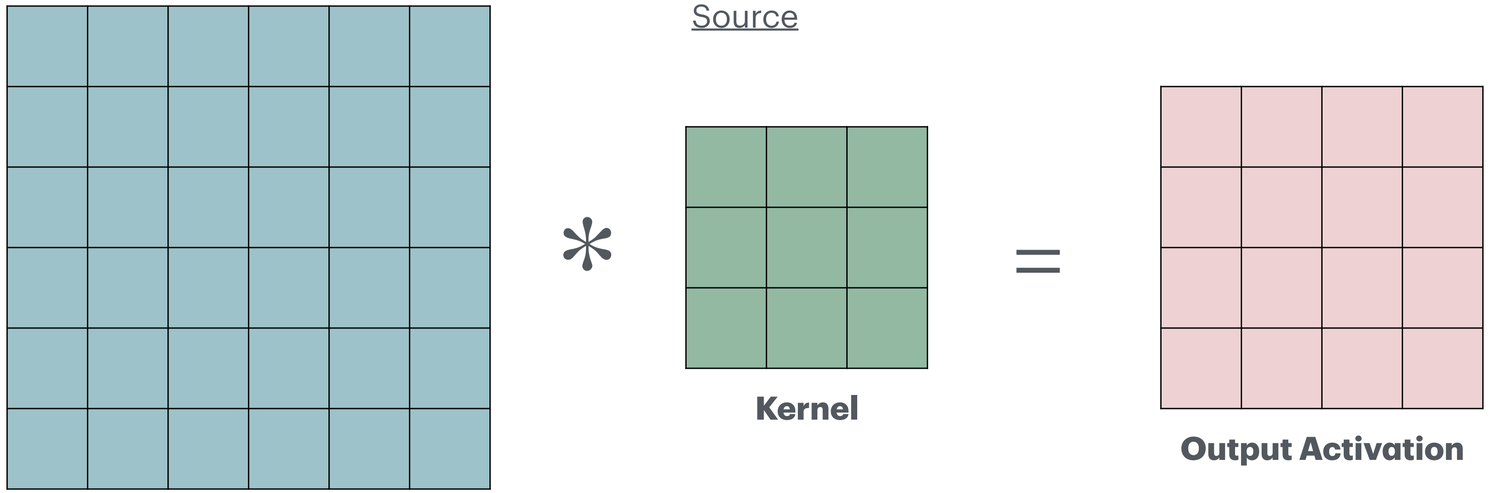




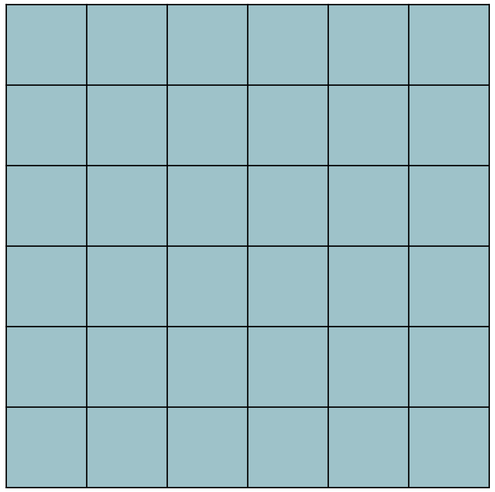
Running CNNs on Microcontrollers



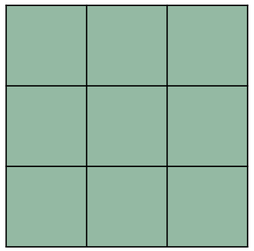
Source



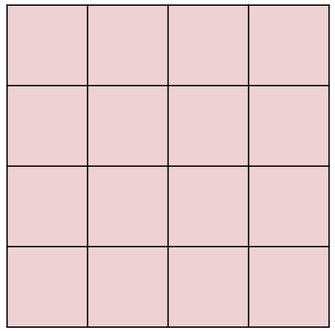
Running CNNs on Microcontrollers



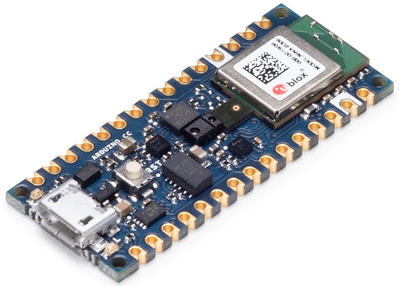
Input Activation



Kernel

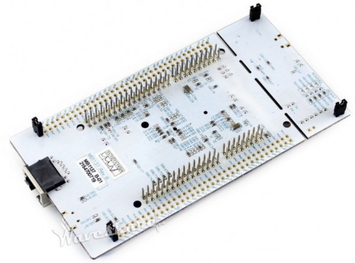


Output Activation



Arduino Nano 33 BLE Sense

- **SRAM: 256 KB**
- **Flash: 1 MB**



STM32 F746ZG

- **SRAM: 320 KB**
- **Flash: 1 MB**

SRAM

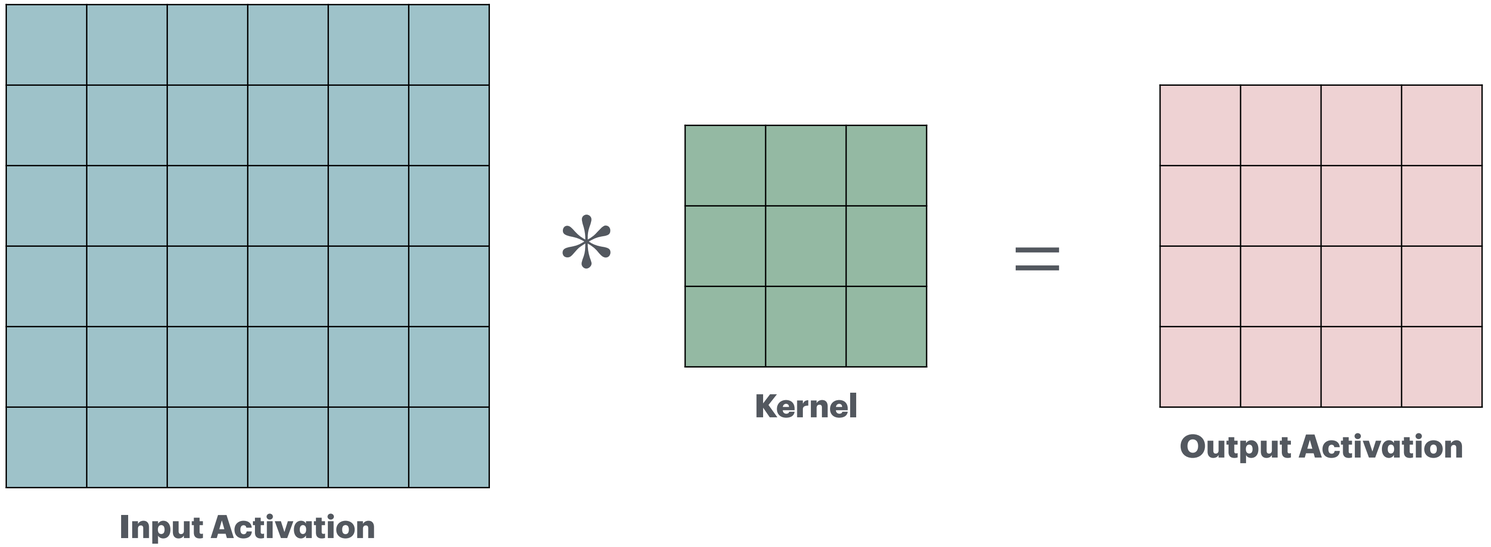
Memory

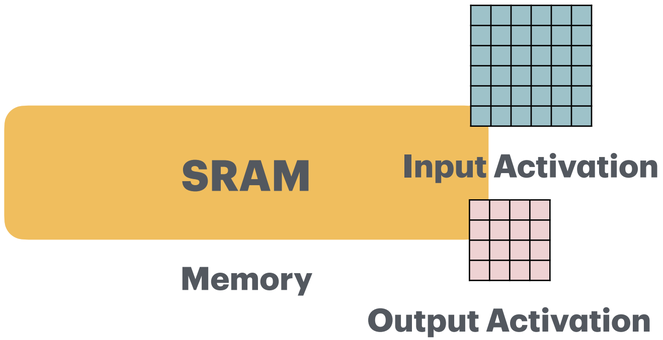
DRAM/Flash

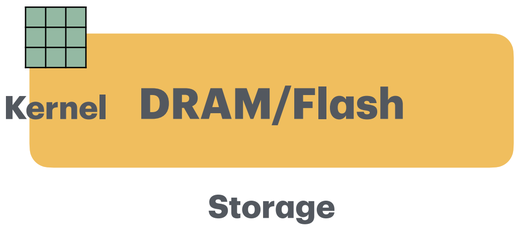
Storage

Running CNNs on Microcontrollers

A simplified method to estimate memory usage







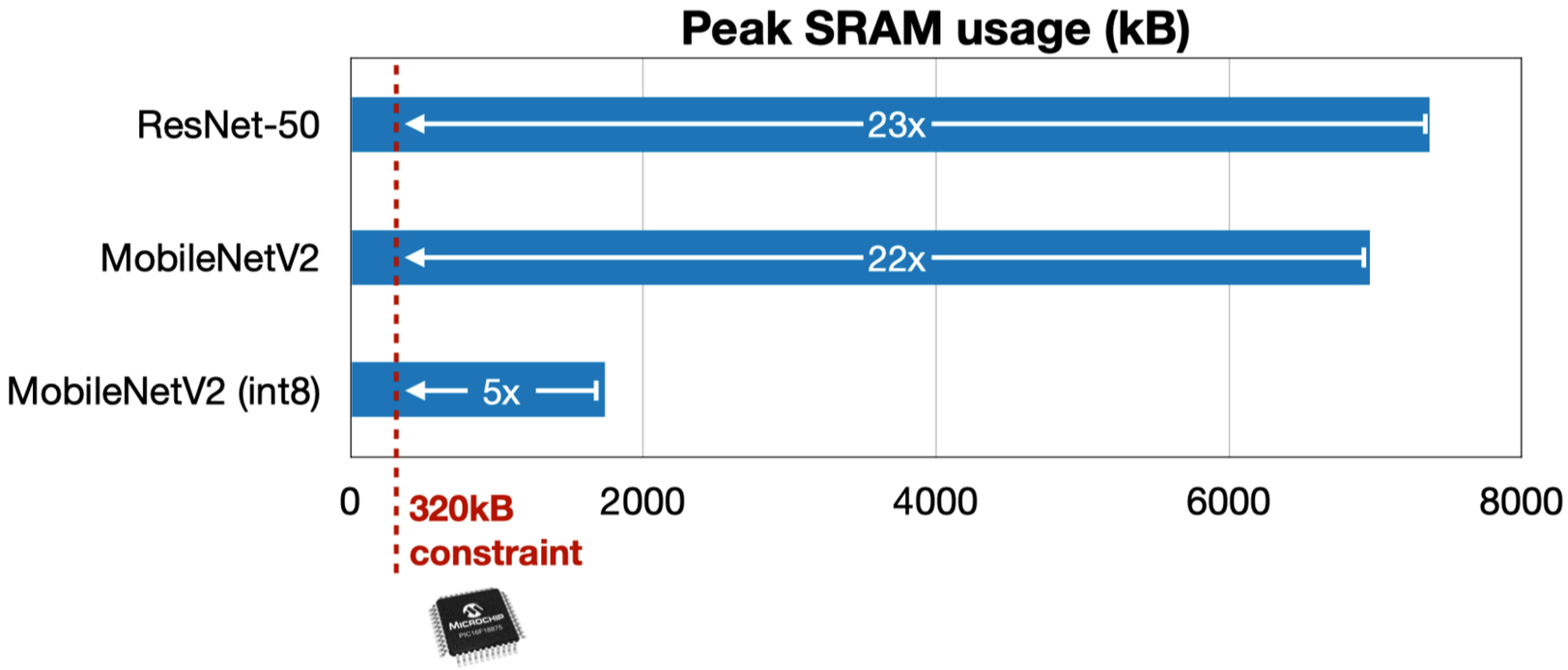
In the first order of approximation

- **Flash** usage
  - **= model size**
  - **Static**, needs to hold the entire model
- **SRAM** usage
  - **= Input activation + output activation**
  - **Dynamic**, different for each layer
  - **Peak** SRAM is important
  - (Weights are not counted since they can be partially fetched)

Simplified: does not consider temporary buffers for SRAM, binary code size, etc. for now.

Running CNNs on Microcontrollers

Memory too small to hold Tiny ML models



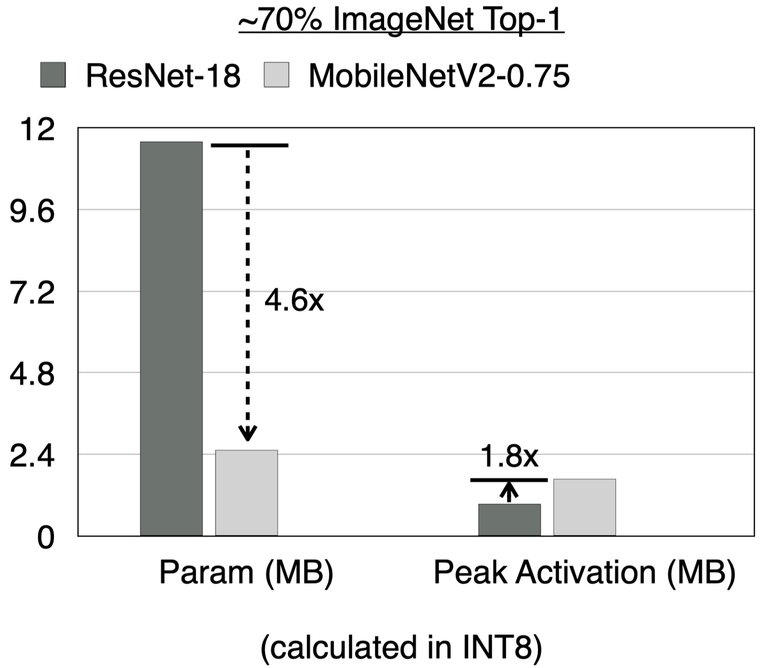
How to reduce the peak memory to fit in SRAM of microcontrollers?

Lin, J., Chen, W. M., Lin, Y., Gan, C., & Han, S. (2020). Mcunet: Tiny deep learning on iot devices. Advances in neural information processing systems, 33, 11711-11722.

Running CNNs on Microcontrollers

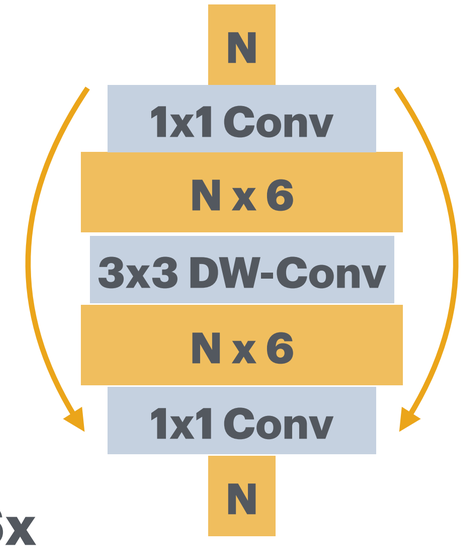
Reduce not only model size, but also activation size

MobileNetV2



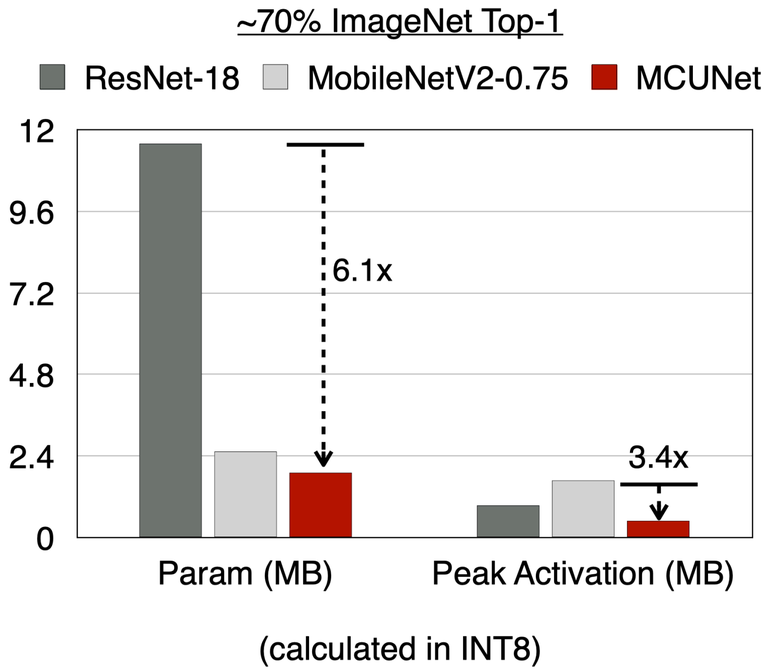
- From ResNet to MobileNet-v2, the model size was reduced, but not the activation size.
- 🤔 Why did the peak activation size increase from ResNet to MobileNetV2?
  - **Expansion ratio**. The middle layer has **6x** more channels than the input and output.

Feature Channels



Lin, J., Chen, W. M., Lin, Y., Gan, C., & Han, S. (2020). Mcunet: Tiny deep learning on iot devices. Advances in neural information processing systems, 33, 11711-11722.

MCUNet:
Tiny Deep Learning on IoT Devices



Lin, J., Chen, W. M., Lin, Y., Gan, C., & Han, S. (2020). Mcunet: Tiny deep learning on iot devices. Advances in neural information processing systems, 33, 11711-11722.

MCUNet: System-Algorithm Co-design

- Search the NN model on an existing library
  - e.g., ProxylessNAS, MnasNet
- Tune deep learning library given an NN model
  - e.g., TVM

Libraries
Such as PyTorch, Tensorflow

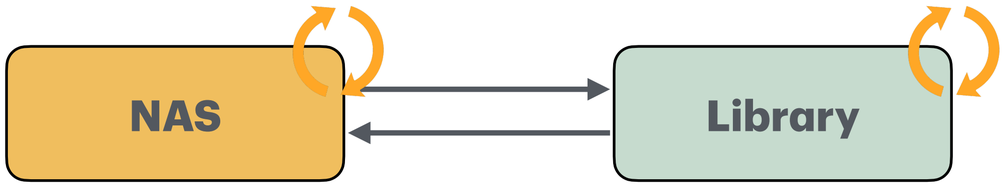
OR Inference libraries
Such as TensorRT, ONNX Runtime, TFLite, OpenVINO

Cai, H., Zhu, L., & Han, S. (2018). Proxylessnas: Direct neural architecture search on target task and hardware. arXiv preprint arXiv:1812.00332.

Tan, M., Chen, B., Pang, R., Vasudevan, V., Sandler, M., Howard, A., & Le, Q. V. (2019). Mnasnet: Platform-aware neural architecture search for mobile. In Proceedings of the IEEE/CVF conference on computer vision and pattern recognition (pp. 2820-2828).

Chen, T., Moreau, T., Jiang, Z., Zheng, L., Yan, E., Shen, H., ... & Krishnamurthy, A. (2018). {TVM}: An automated {End-to-End} optimizing compiler for deep learning. In 13th USENIX Symposium on Operating Systems Design and Implementation (OSDI 18) (pp. 578-594).

MCUNet: System-Algorithm Co-design



- Jointly design the neural architecture and the inference scheduling to fit the tight memory resource on microcontrollers

- **TinyEngine**: make full use of the limited resources on MCU, allowing a large design space for architecture search (Will introduce in the next lecture)

- **TinyNAS**: find high accurate model with a large degree of design freedom

Lin, J., Chen, W. M., Lin, Y., Gan, C., & Han, S. (2020). Mcunet: Tiny deep learning on iot devices. Advances in neural information processing systems, 33, 11711-11722.

From Manual Design to Automatic Design

Huge design space, manual design is unscalable

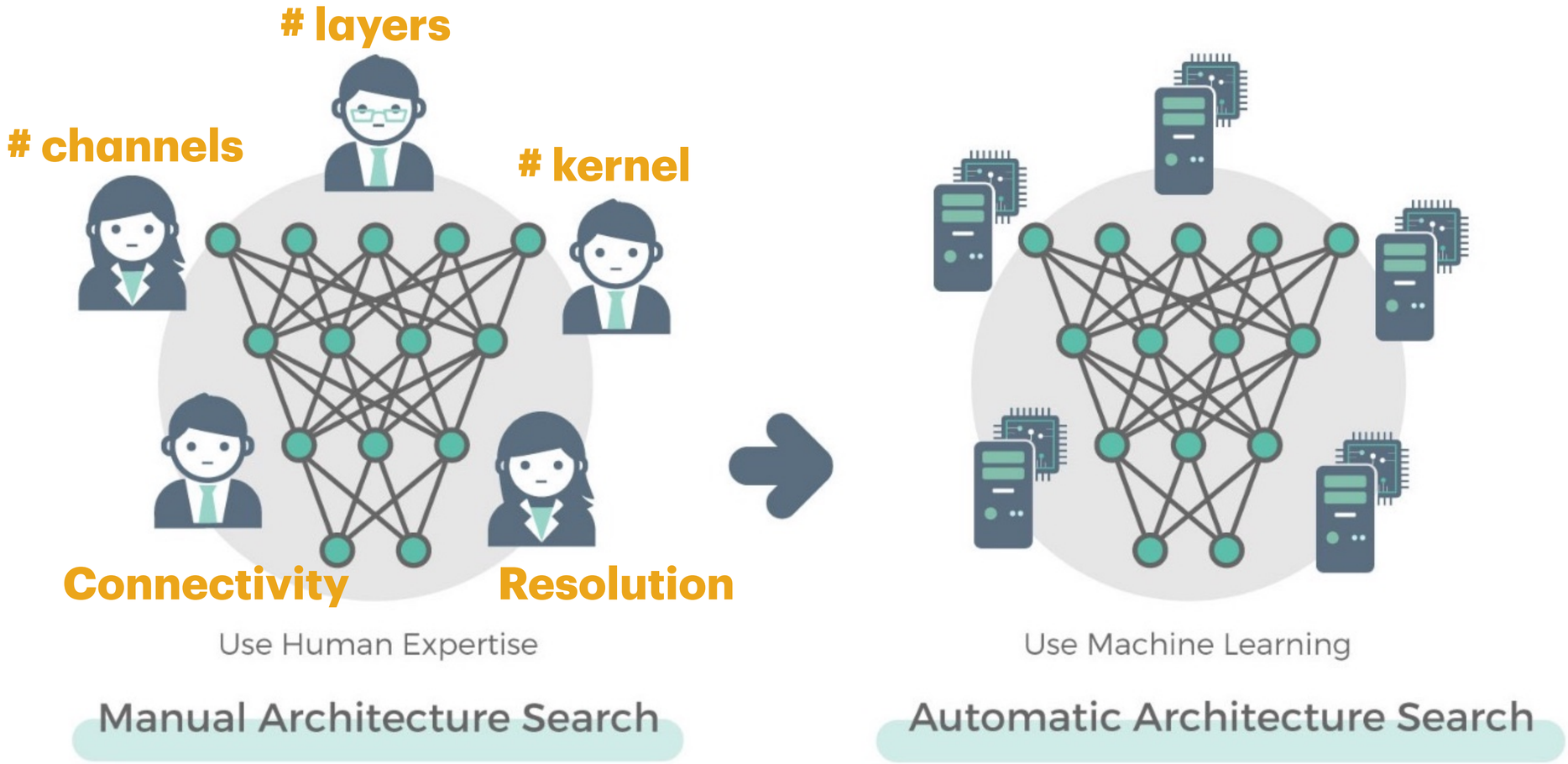


Figure source

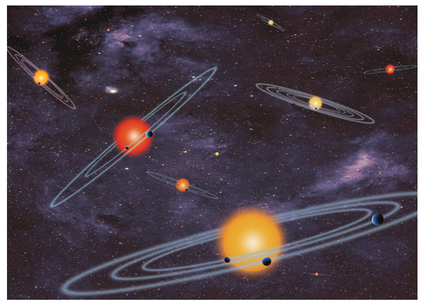
Search space design is crucial for NAS performance

TinyNAS: Two-Stage NAS for Tiny Memory Constraints

Problem: what is the right search space for TinyML?

- The **quality of search space** largely determines the performance of the searched model

Galaxy



Solar System

Subspace 1

Subspace 2

Subspace 3

...

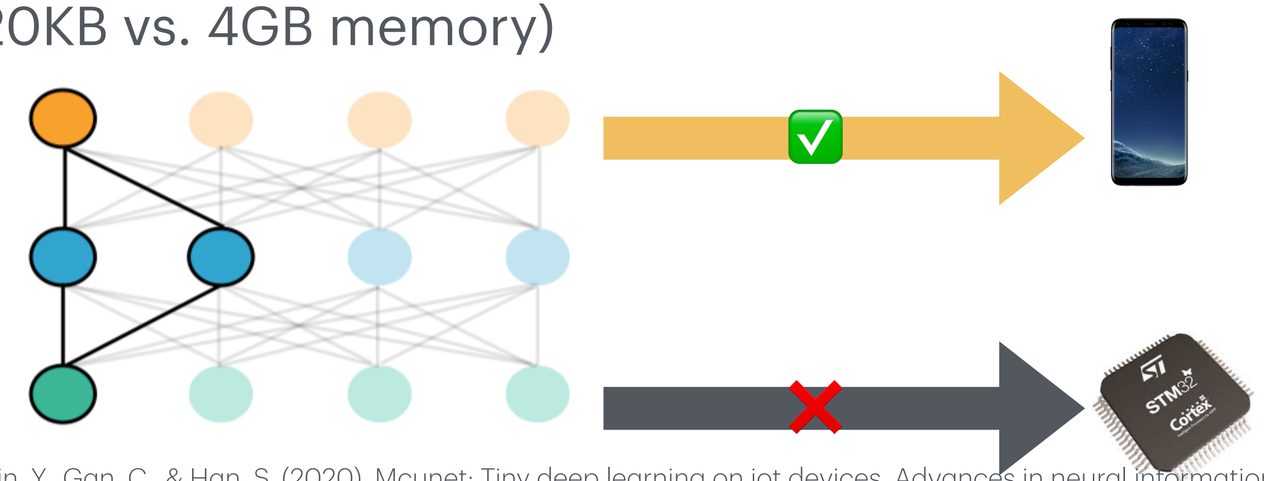
?

Lin, J., Chen, W. M., Lin, Y., Gan, C., & Han, S. (2020). Mcunet: Tiny deep learning on iot devices. Advances in neural information processing systems, 33, 11711-11722.

TinyNAS: Two-Stage NAS for Tiny Memory Constraints

Problem: what is the right search space for TinyML?

- The **quality of search space** largely determines the performance of the searched model
- There is no prior expertise in MCU model design.
- 🤔 Reuse the carefully designed mobile search space for TinyML?
  - 😔 The smallest sub-network in the search space **cannot** fit the hardware due to the huge gap (320KB vs. 4GB memory)



Lin, J., Chen, W. M., Lin, Y., Gan, C., & Han, S. (2020). Mcunet: Tiny deep learning on iot devices. Advances in neural information processing systems, 33, 11711-11722.

TinyNAS: Two-Stage NAS for Tiny Memory Constraints

Problem: what is the right search space for TinyML?

- 🤔 Smartly pick a search space for the IoT devices:
  - **Two-stage NAS: first design the design space, then search the sub-network.**

1. **Design the design space**: consider the memory and storage constraints for TinyML
2. **Search the sub-network**: as a regular NAS

- Recall: what dimensions did we consider when designing the design space in OFA? Which part(s) will affect SRAM and flash size?

Lin, J., Chen, W. M., Lin, Y., Gan, C., & Han, S. (2020). Mcunet: Tiny deep learning on iot devices. Advances in neural information processing systems, 33, 11711-11722.

TinyNAS: Two-Stage NAS for Tiny Memory Constraints

Design the design space

MobileNetV2

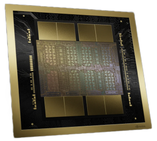
- Extended MBNet-alike search space to cover a wide range of hardware capacities:
  - ***S*** = *kernel size x expansion ratio x depth* x ***input resolution R x width multiplier W***

Regular NAS **Design/optimize the search space**

    - *expansion ratio: inverted bottleneck (3/4/6)
    - A specific R and W lead to a sub-search space
- Different R and W for different hardware capacity
  - The original search space (R=224, W=1.0) is good for smartphones
  - A scaled-up space is good for GPUs
  - 🤔 How to choose different R and W for different microcontrollers?

R=260, W=1.4

Found by OFA



R=224, W=1.0

R=?, W=?

F412/F743/F746/...
256KB/320KB/512KB/...

Lin, J., Chen, W. M., Lin, Y., Gan, C., & Han, S. (2020). Mcunet: Tiny deep learning on iot devices. Advances in neural information processing systems, 33, 11711-11722.

TinyNAS: Two-Stage NAS for Tiny Memory Constraints

Design the design space

- Extended MBNet-alike search space to cover a wide range of hardware capacities:
  - **Regular NAS** — **Design/optimize the search space**
  - ***S*** = *kernel size x expansion ratio x depth* x ***input resolution R x width multiplier W***
    - A specific R and W lead to a sub-search space

Memory/storage constraints ?

“Search space” ✅

Resolution
Width Multiplier

Kernel sizes
Expansion ratios
#blocks/ stage

Full Network Space → **Optimized Search Space** → **Neural Architecture Search**

Galaxy

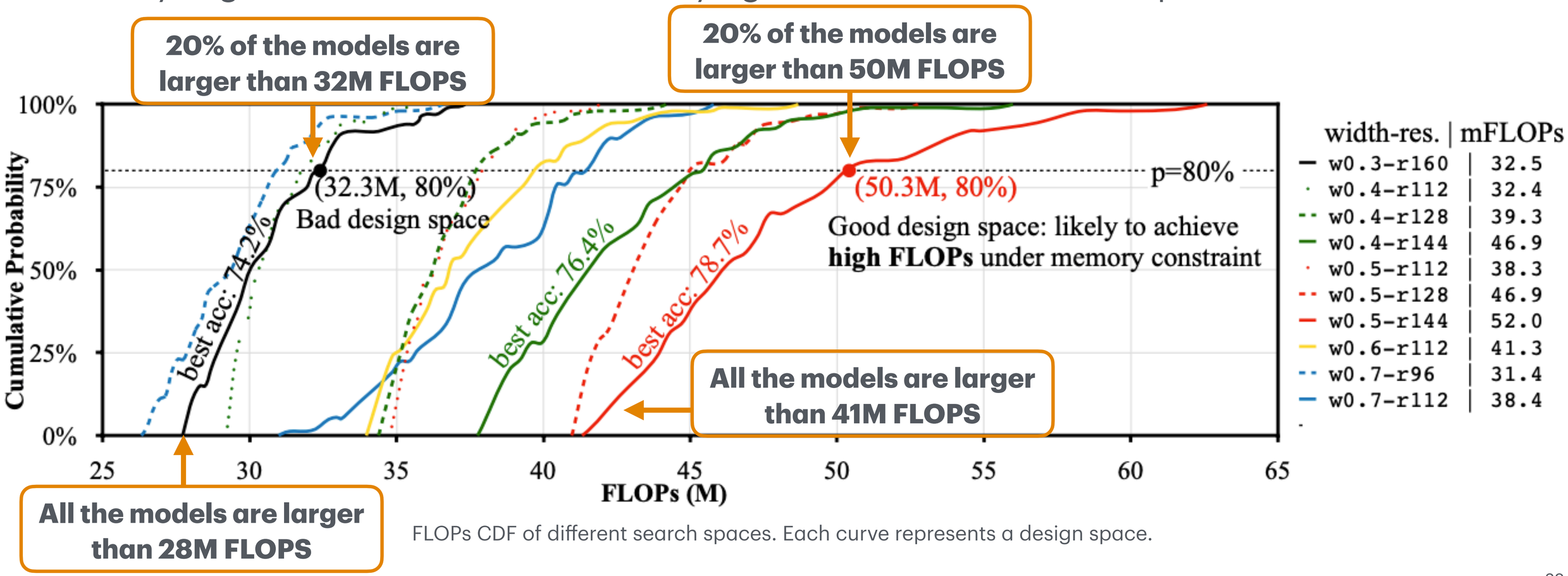
Solar System

Earth

TinyNAS: Automated Search Space Optimization

Larger FLOPs → larger model capacity → more likely to give higher accuracy

- Analyzing FLOPS **distribution** of satisfying models in each search space



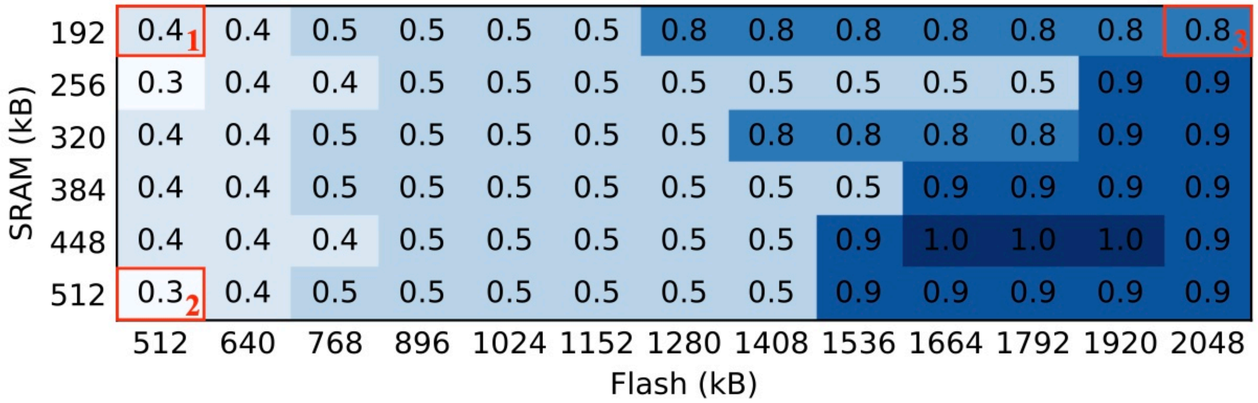
FLOPs CDF of different search spaces. Each curve represents a design space.

Lin, J., Chen, W. M., Lin, Y., Gan, C., & Han, S. (2020). Mcunet: Tiny deep learning on iot devices. Advances in neural information processing systems, 33, 11711-11722.

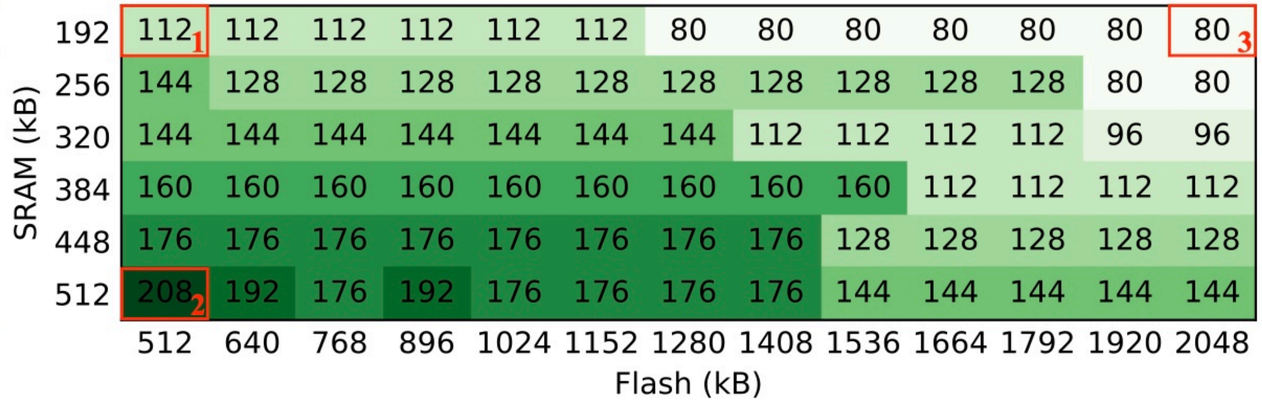
TinyNAS: Automated Search Space Optimization

Best search space configurations under different SRAM and Flash constraints

- **A larger SRAM → store a larger activation map → Use a higher input resolution**
- **A larger Flash → store a larger model**
- 🤔 From point 1 to 2, what is your observation? Can you explain it?
- 🤔 From point 1 to 3, what is your observation? Can you explain it?



Best width setting



Best resolution setting

Lin, J., Chen, W. M., Lin, Y., Gan, C., & Han, S. (2020). Mcunet: Tiny deep learning on iot devices. Advances in neural information processing systems, 33, 11711-11722.

TinyNAS: Two-Stage NAS for Tiny Memory Constraints

Design the design space

- Extended MBNet-alike search space to cover a wide range of hardware capacities:
  - **S** = *kernel size x expansion ratio x depth* x ***input resolution R x width multiplier W***

Regular NAS

Design/optimize the search space

    - A specific R and W lead to a sub-search space

Memory/storage constraints ✅

“Search space” 🔍

Resolution
Width Multiplier

Kernel sizes
Expansion ratios
#blocks/ stage

Full Network Space → **Optimized Search Space** → **Neural Architecture Search**

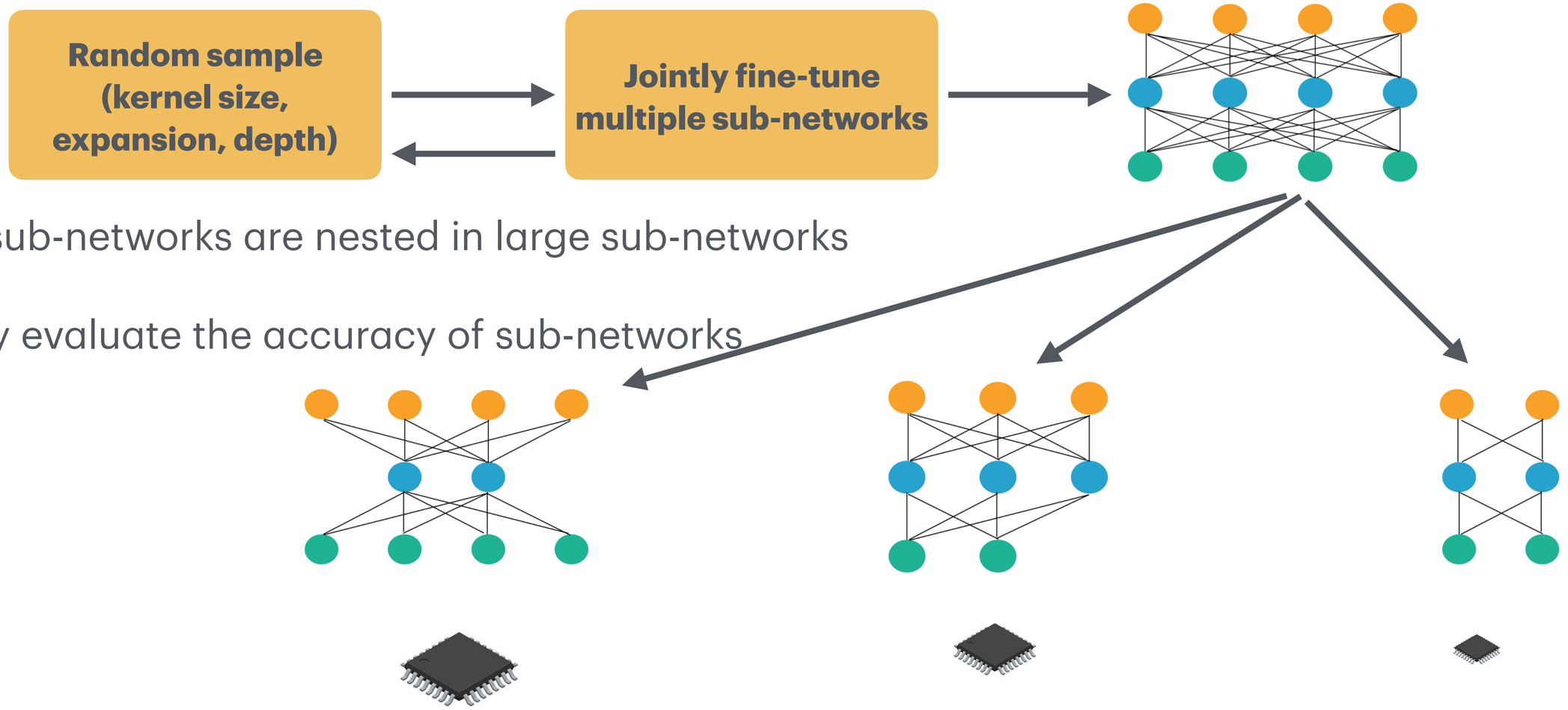
Galaxy

Solar System

Earth

TinyNAS: Resource-Constrained Model Specialization

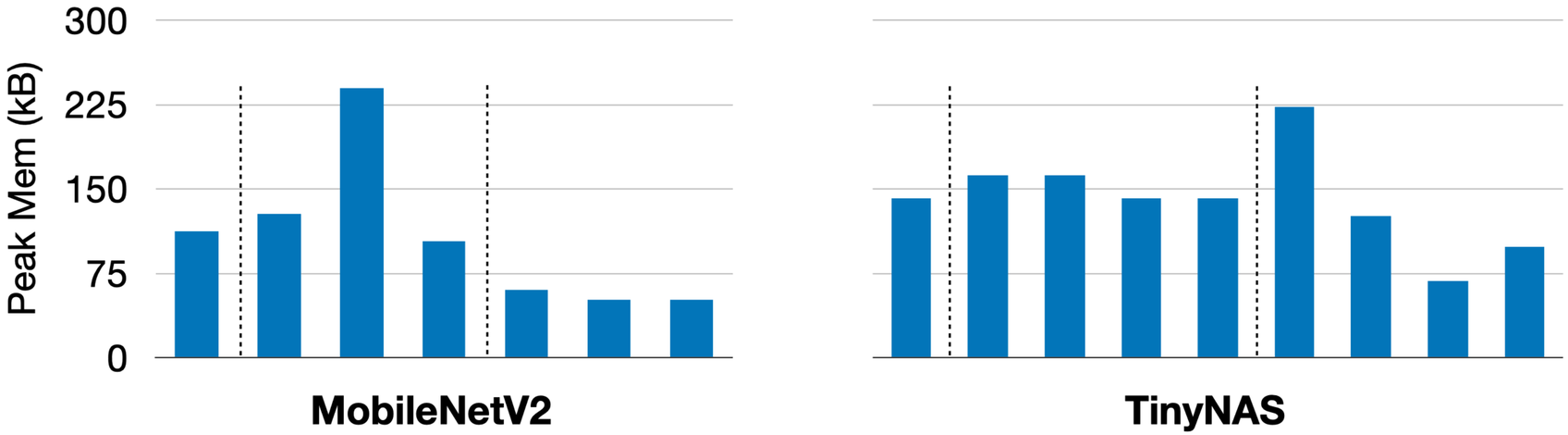
One-shot NAS through weight sharing



- Small sub-networks are nested in large sub-networks
- Directly evaluate the accuracy of sub-networks

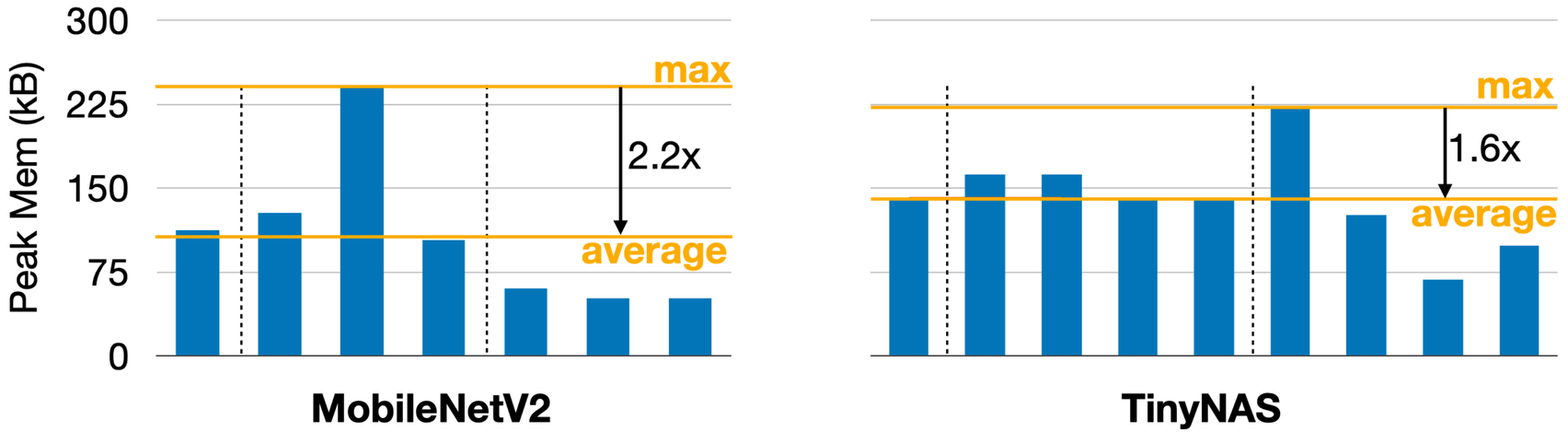
Lin, J., Chen, W. M., Lin, Y., Gan, C., & Han, S. (2020). Mcunet: Tiny deep learning on iot devices. Advances in neural information processing systems, 33, 11711-11722.

TinyNAS Better Utilize the Memory



Lin, J., Chen, W. M., Lin, Y., Gan, C., & Han, S. (2020). Mcunet: Tiny deep learning on iot devices. Advances in neural information processing systems, 33, 11711-11722.

TinyNAS Better Utilize the Memory

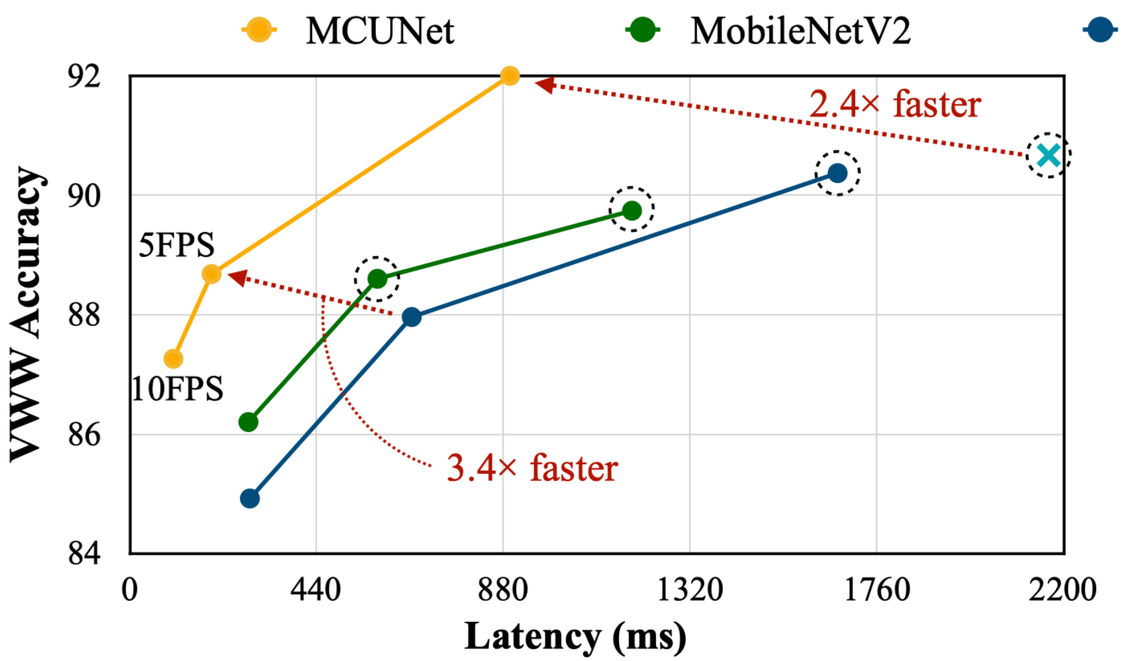


TinyNAS designs networks with more uniform peak memory for each block, allowing to fit a larger model at the same amount of memory.

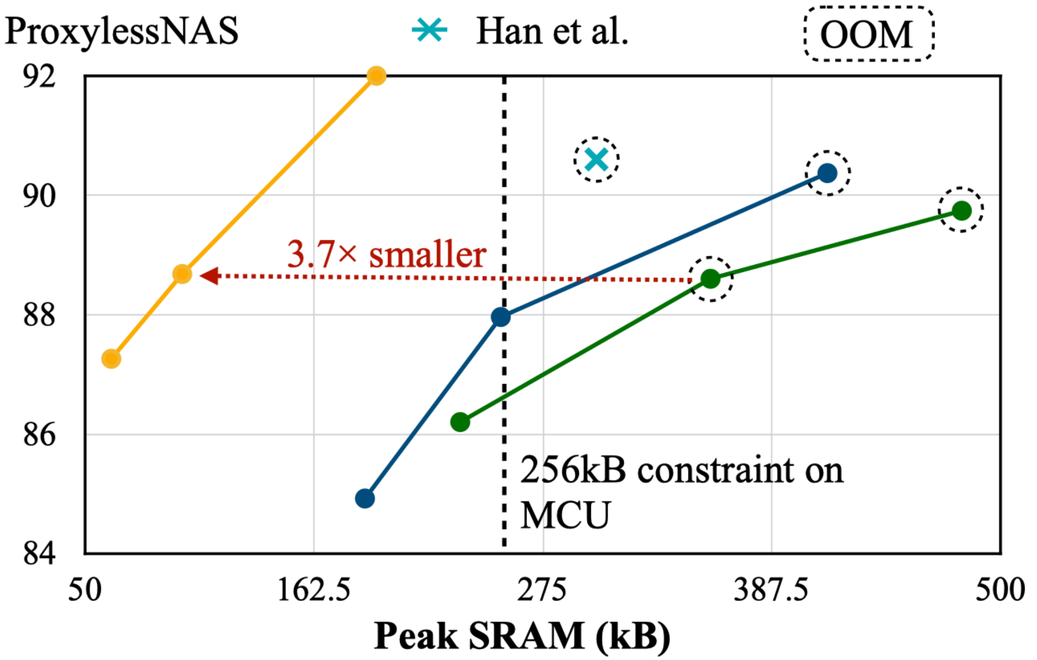
Lin, J., Chen, W. M., Lin, Y., Gan, C., & Han, S. (2020). Mcunet: Tiny deep learning on iot devices. Advances in neural information processing systems, 33, 11711-11722.

TinyNAS Outperforms Manual & NAS Models

- Higher accuracy at 3x faster inference speed, 4x smaller memory cost on the Visual Wake Words (WWW) dataset



(a) Trade-off: accuracy *vs.* measured latency

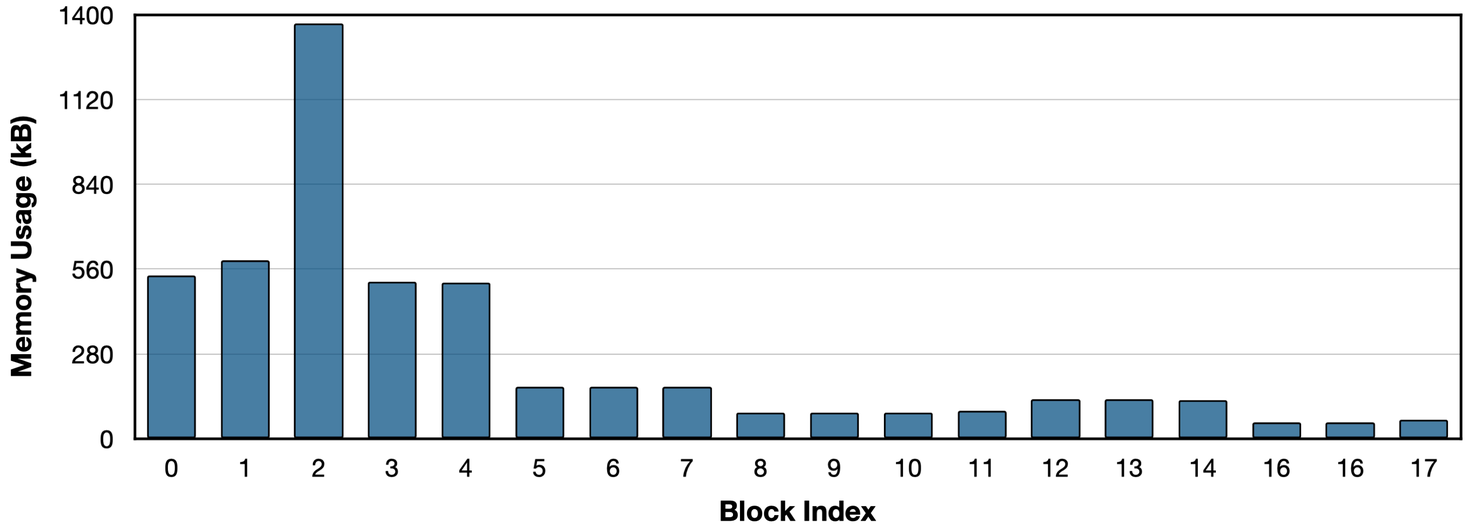


(b) Trade-off: accuracy *vs.* peak memory

Lin, J., Chen, W. M., Lin, Y., Gan, C., & Han, S. (2020). Mcunet: Tiny deep learning on iot devices. Advances in neural information processing systems, 33, 11711-11722.

Can we Make it Better?

MobileNetV2



SRAM usage of each block in MobileNetV2

Lin, J., Chen, W. M., Lin, Y., Gan, C., & Han, S. (2020). Mcunet: Tiny deep learning on iot devices. Advances in neural information processing systems, 33, 11711-11722.

Class Schedule

Lectures

Tuesday, Nov 5th (Today) - Thursday, Nov 21st

Guest Talk

Tuesday, Nov 26th

No class on Dec 3rd

Final Presentation

Thursday, Dec 5th (15mins per group)

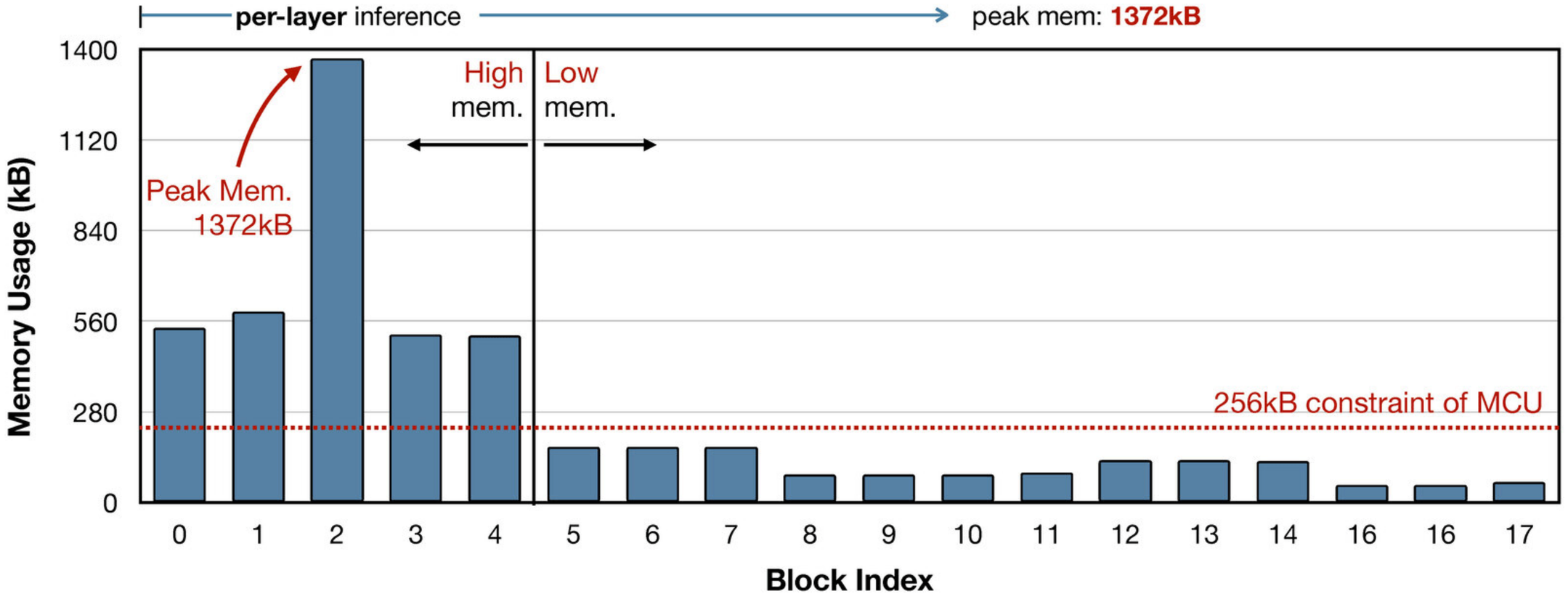
Meanwhile, you are working on your final project! 💪

MCUNet: Tiny Deep Learning on IoT Devices

MCUNetV2: Memory-Efficient Patch-based Inference for Tiny Deep Learning

Imbalanced Memory Distribution of CNNs

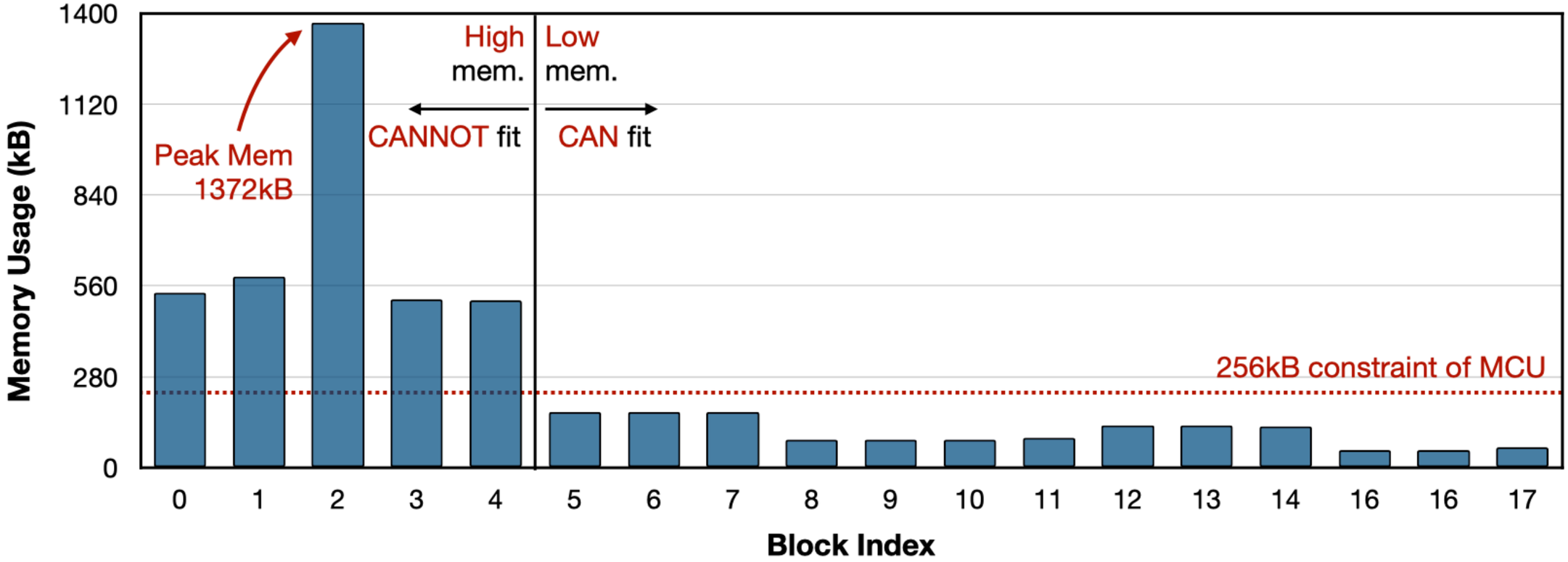
SRAM usage of each layer in MobileNetV2



Lin, J., Chen, W. M., Cai, H., Gan, C., & Han, S. (2021). Memory-efficient patch-based inference for tiny deep learning. Advances in Neural Information Processing Systems, 34, 2346-2358.

Imbalanced Memory Distribution of CNNs

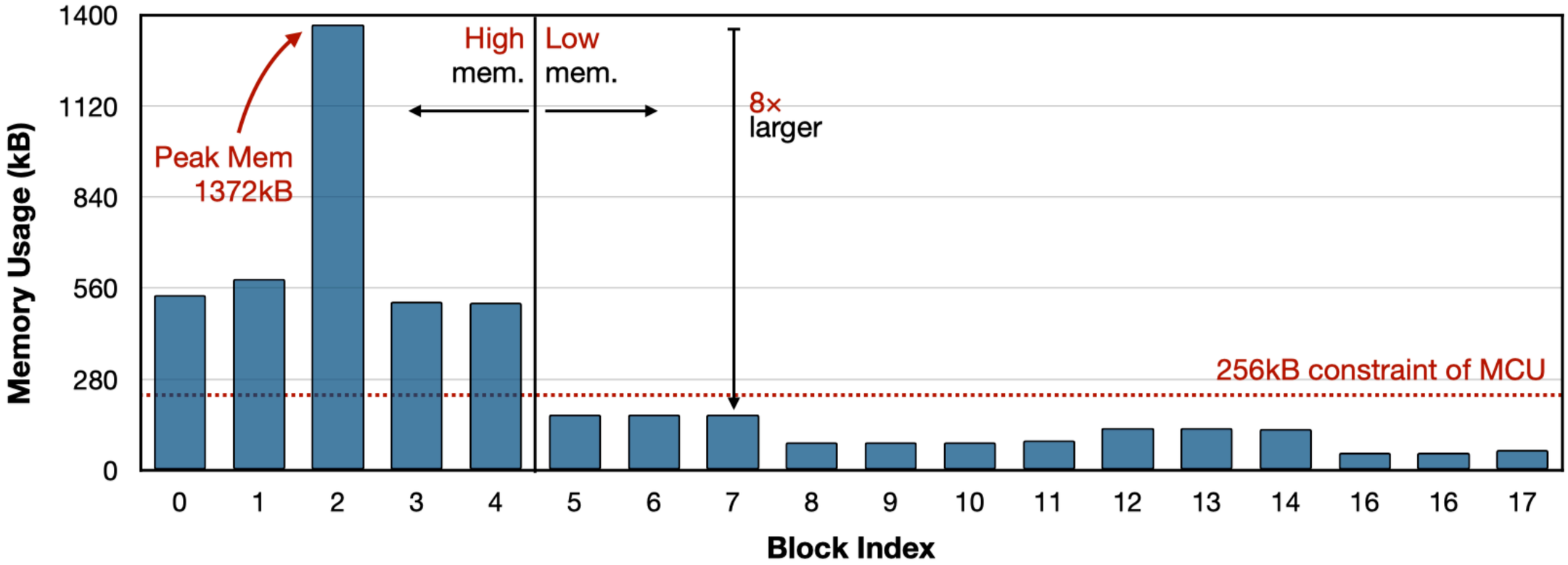
SRAM usage of each layer in MobileNetV2



Lin, J., Chen, W. M., Cai, H., Gan, C., & Han, S. (2021). Memory-efficient patch-based inference for tiny deep learning. Advances in Neural Information Processing Systems, 34, 2346-2358.

Imbalanced Memory Distribution of CNNs

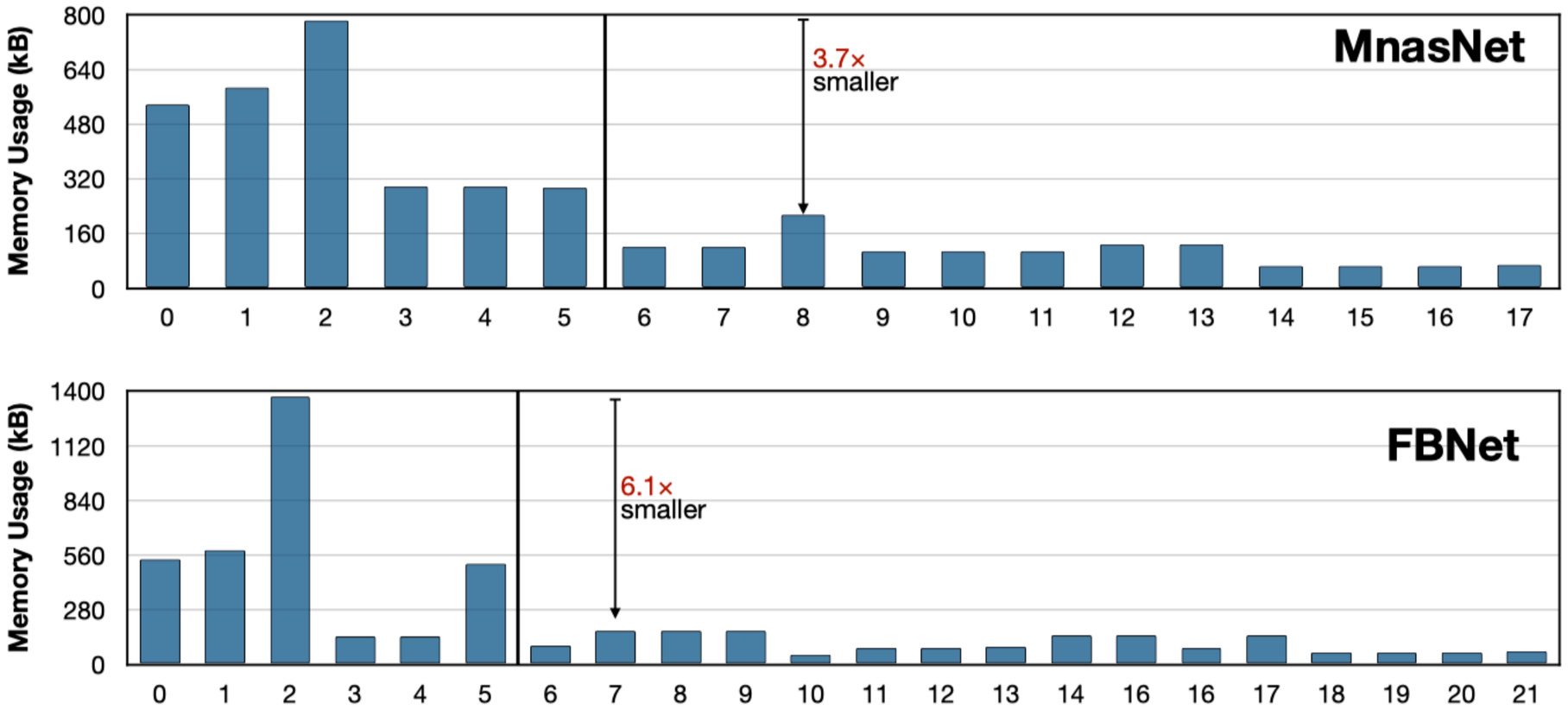
🤔 **Why do early layers have such a large peak number of activations?**



Lin, J., Chen, W. M., Cai, H., Gan, C., & Han, S. (2021). Memory-efficient patch-based inference for tiny deep learning. Advances in Neural Information Processing Systems, 34, 2346-2358.

Imbalanced Memory Distribution of CNNs

Common cases in efficient CNN design

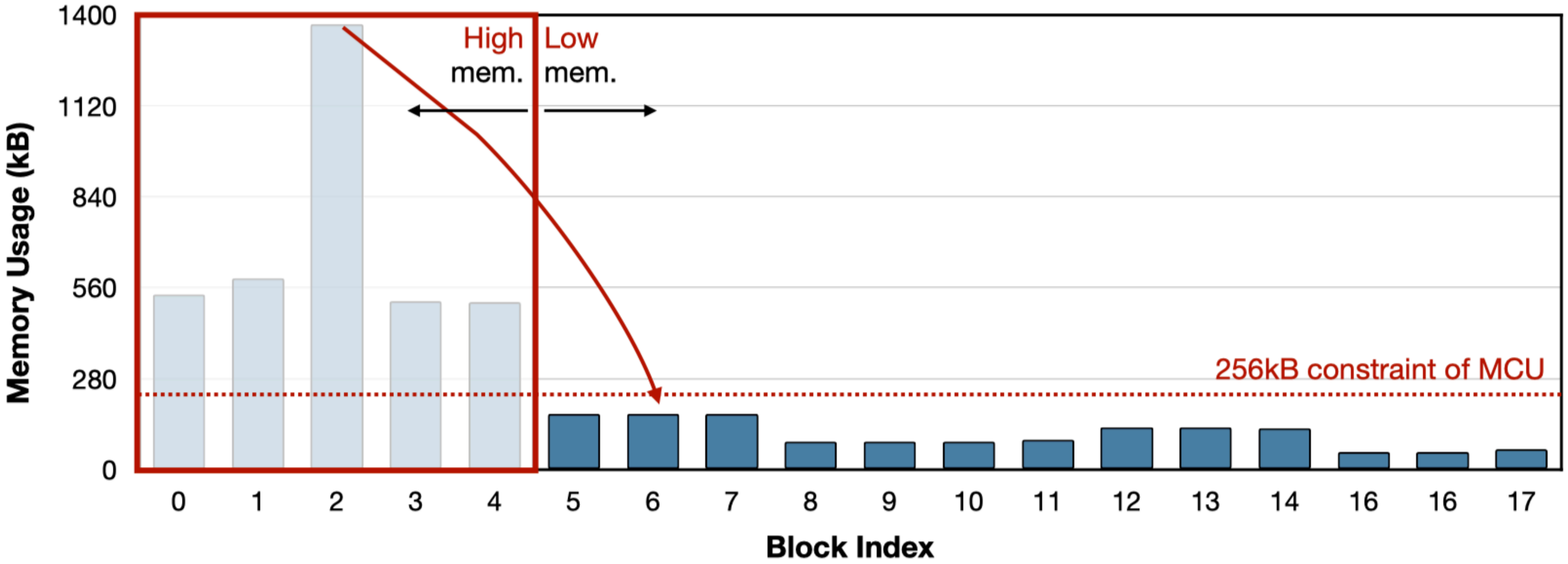


Lin, J., Chen, W. M., Cai, H., Gan, C., & Han, S. (2021). Memory-efficient patch-based inference for tiny deep learning. Advances in Neural Information Processing Systems, 34, 2346-2358.

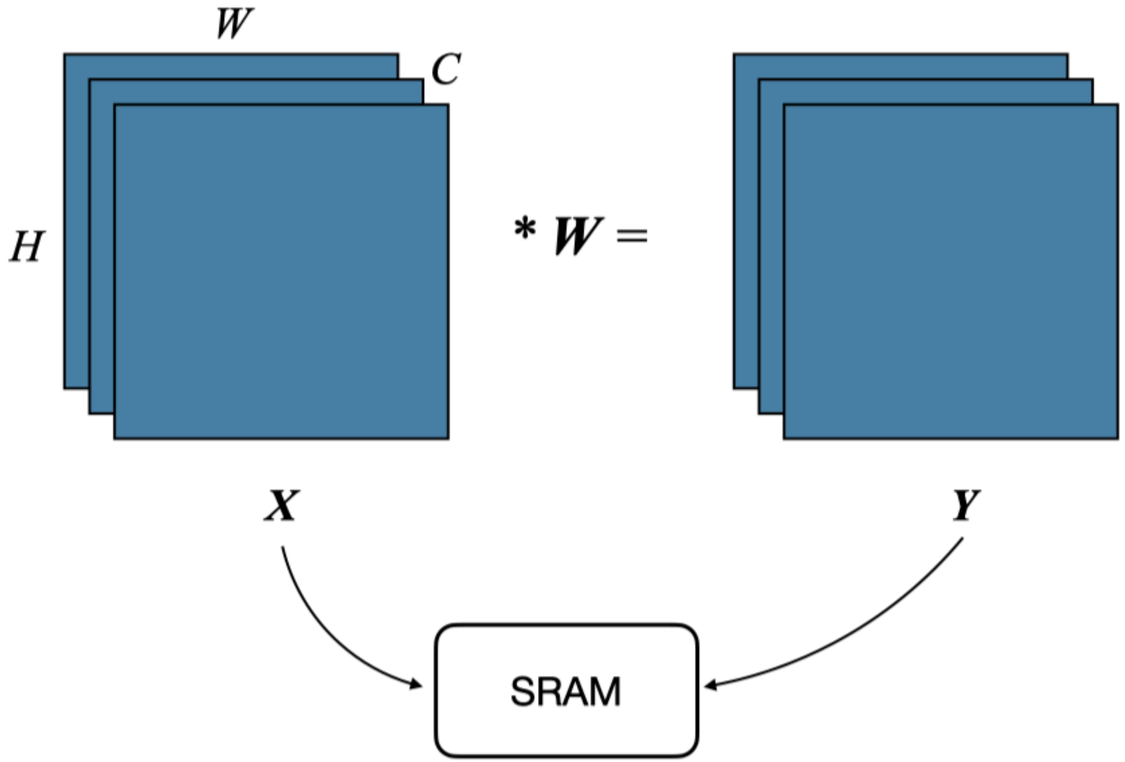
Imbalanced Memory Distribution of CNNs

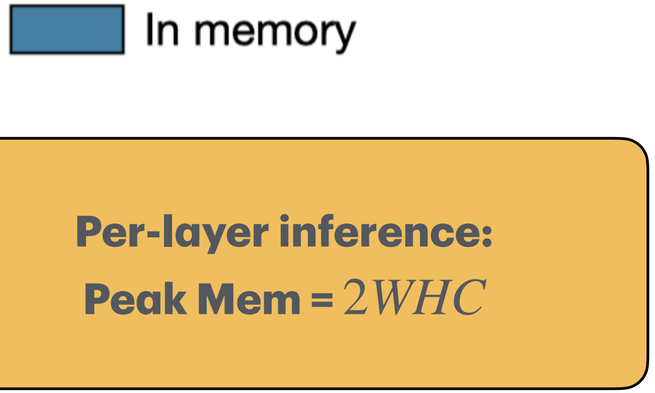
Reduce memory usage of the initial stage

→ Reduce the overall memory usage



MCUNetV2: Patch-based Inference



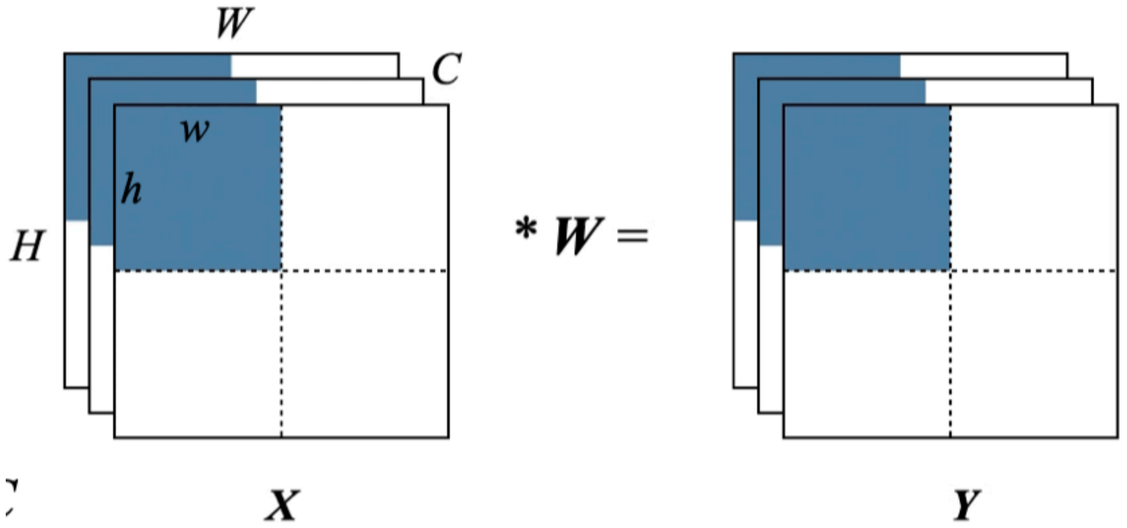


- Weights are usually partially fetched from Flash

Lin, J., Chen, W. M., Cai, H., Gan, C., & Han, S. (2021). Memory-efficient patch-based inference for tiny deep learning. Advances in Neural Information Processing Systems, 34, 2346-2358.

MCUNetV2: Patch-based Inference

Break the memory bottleneck with patch-based inference



In memory

Per-layer inference:

Peak Mem = $2WHC$

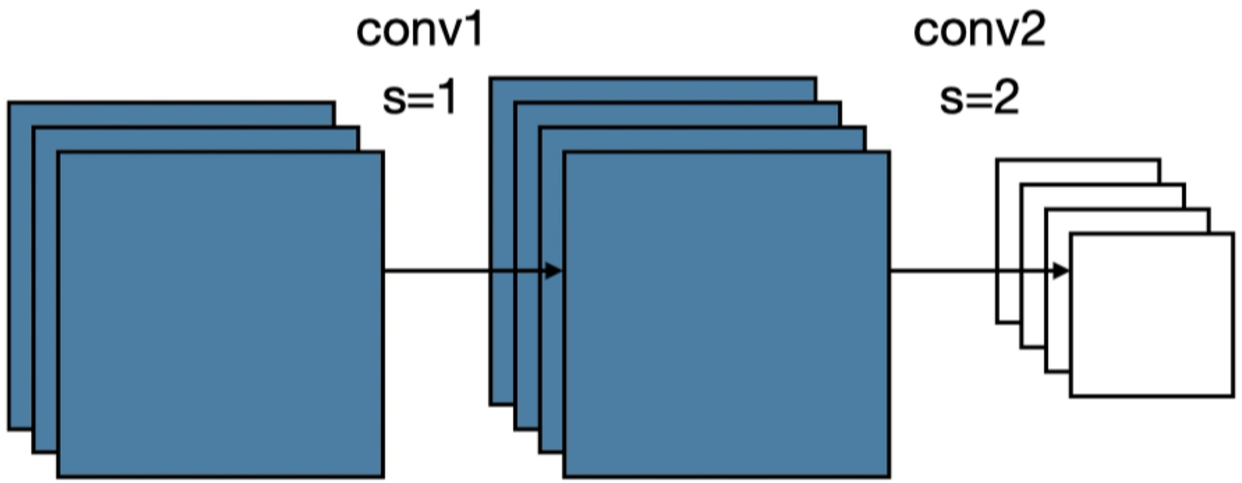
Per-patch inference:

Peak Mem = $2whc << 2WHC$

- Can use more than 2x2 patches

MCUNetV2: Patch-based Inference

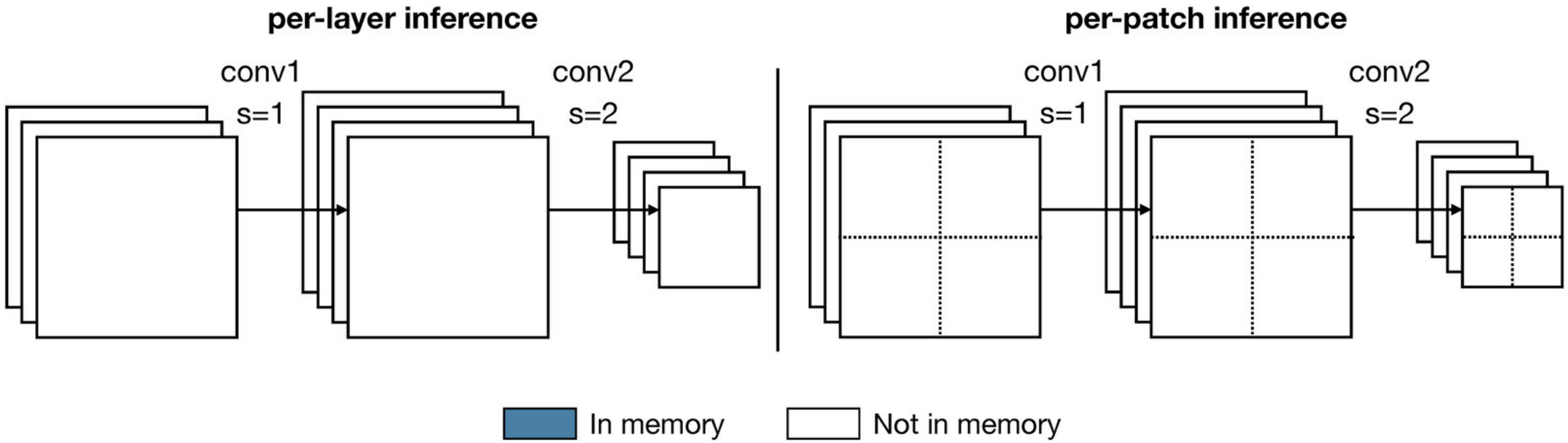
A practical 2-layer example



Lin, J., Chen, W. M., Cai, H., Gan, C., & Han, S. (2021). Memory-efficient patch-based inference for tiny deep learning. Advances in Neural Information Processing Systems, 34, 2346-2358.

MCUNetV2: Patch-based Inference

A practical 2-layer example

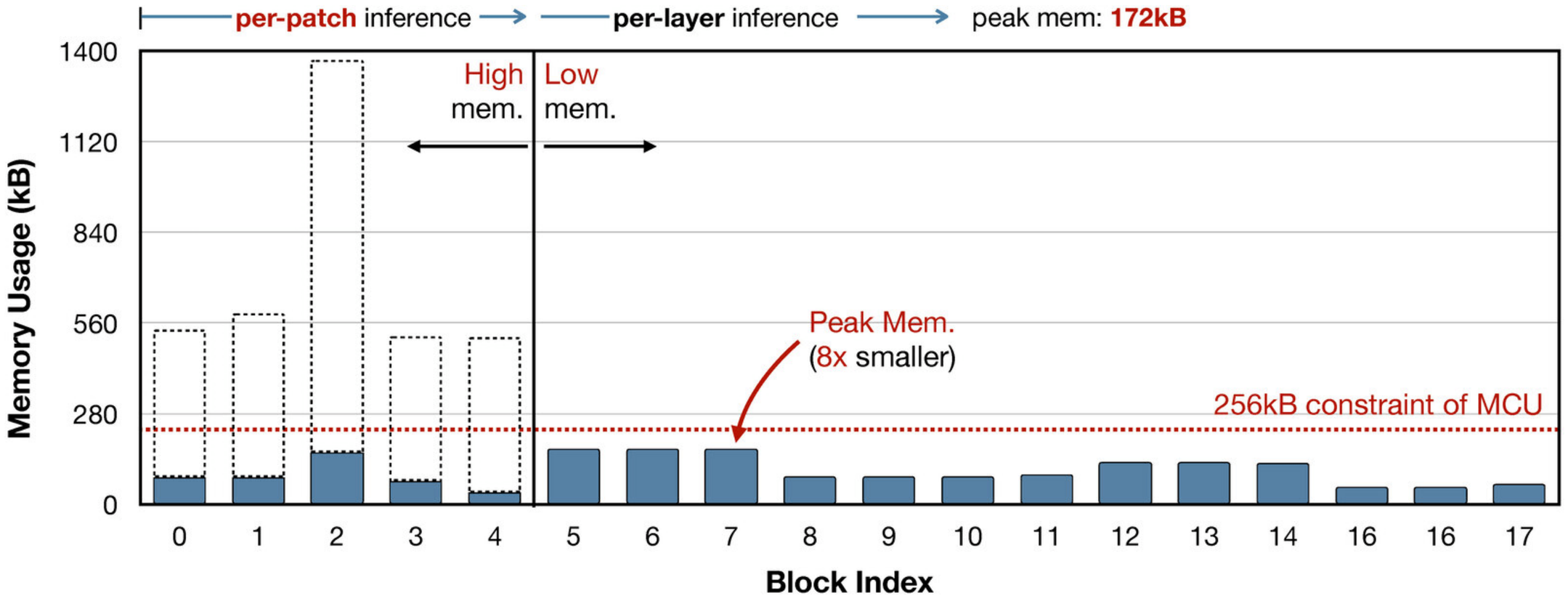


- Trade off the memory bottleneck with increased read-write operations between SRAM and Flash.

Lin, J., Chen, W. M., Cai, H., Gan, C., & Han, S. (2021). Memory-efficient patch-based inference for tiny deep learning. Advances in Neural Information Processing Systems, 34, 2346-2358.

Saving Memory with Patch-based Inference

SRAM usage of each layer in MobileNetV2 after patch-based inference

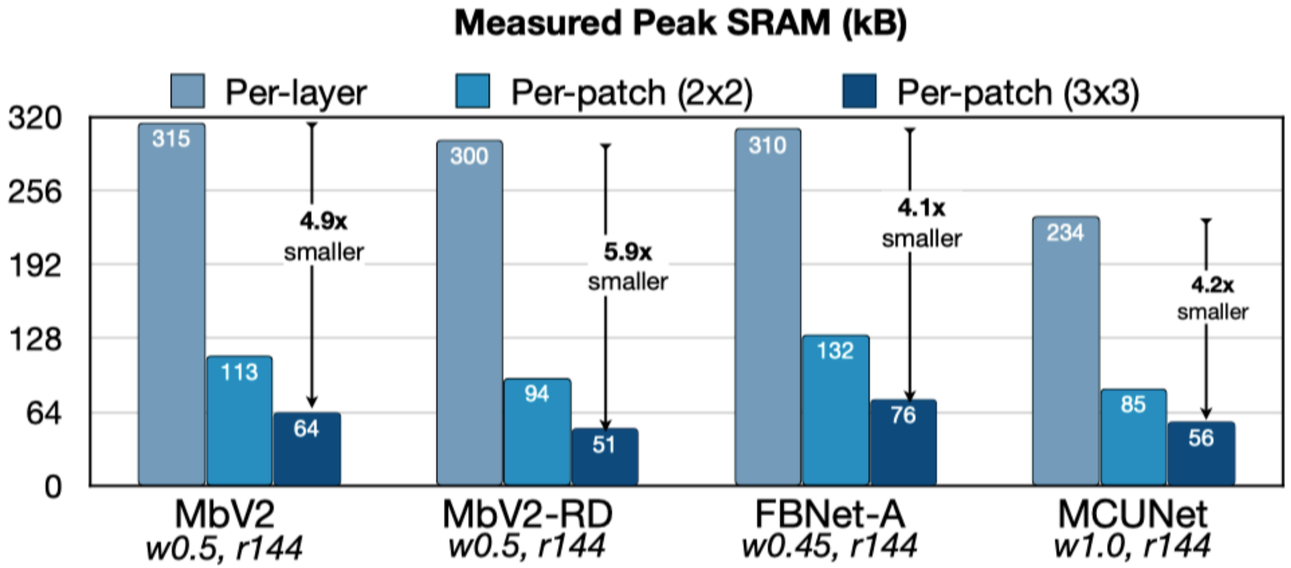


Lin, J., Chen, W. M., Cai, H., Gan, C., & Han, S. (2021). Memory-efficient patch-based inference for tiny deep learning. Advances in Neural Information Processing Systems, 34, 2346-2358.

Saving Memory with Patch-based Inference

Memory saving for other models

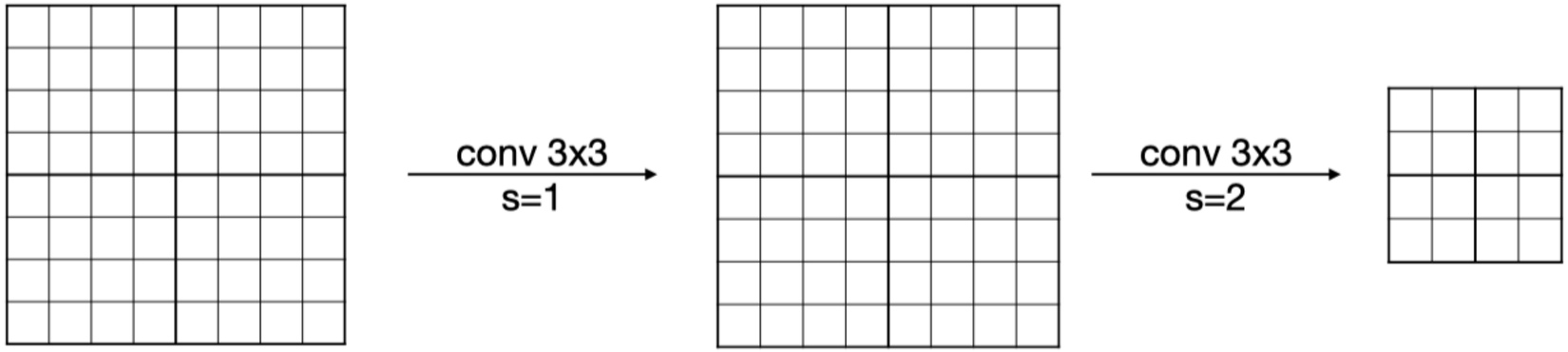
- Baseline: TinyEngine. Measured on STM32F746.



Lin, J., Chen, W. M., Cai, H., Gan, C., & Han, S. (2021). Memory-efficient patch-based inference for tiny deep learning. Advances in Neural Information Processing Systems, 34, 2346-2358.

MCUNetV2: Patch-based Inference

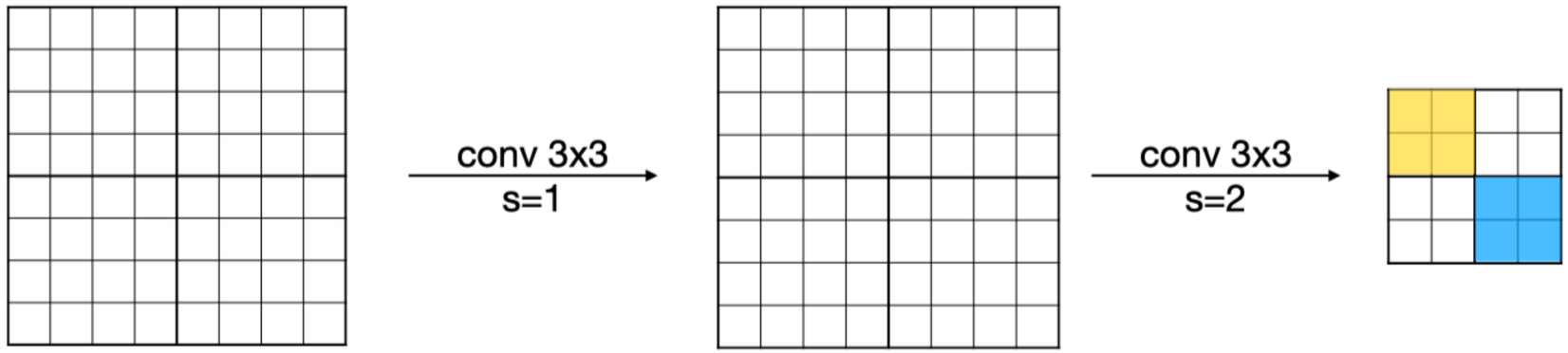
Let’s see the computation in 2x2 patches



Lin, J., Chen, W. M., Cai, H., Gan, C., & Han, S. (2021). Memory-efficient patch-based inference for tiny deep learning. Advances in Neural Information Processing Systems, 34, 2346-2358.

MCUNetV2: Patch-based Inference

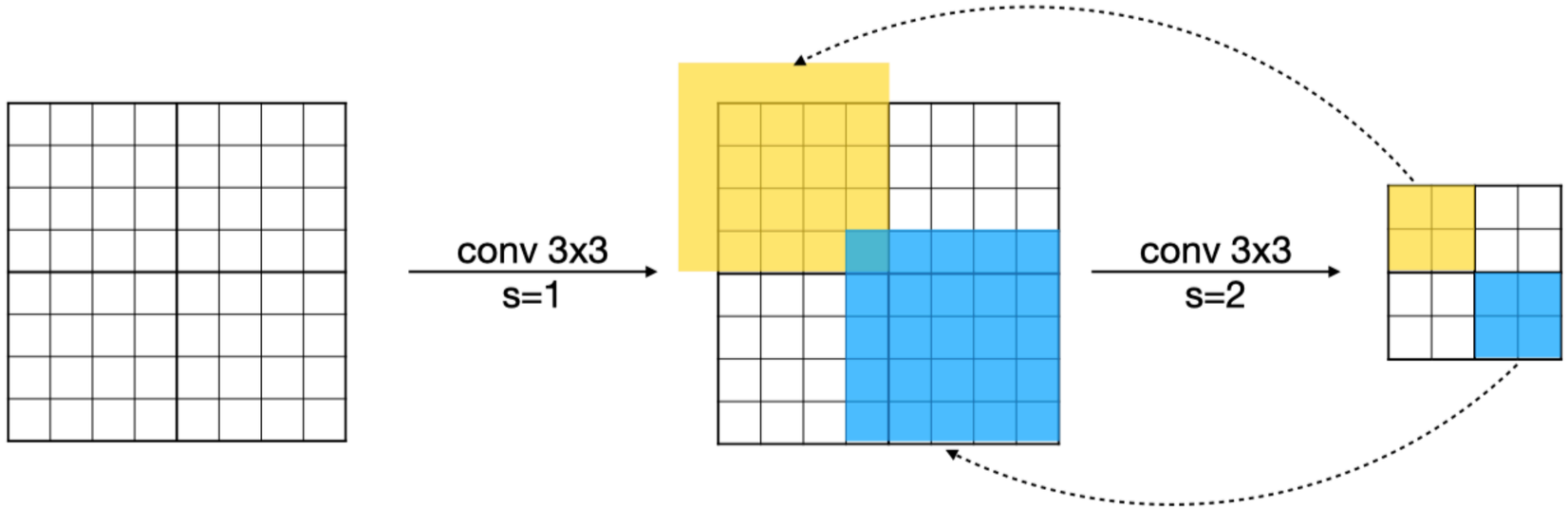
Let’s see the computation in 2x2 patches



Lin, J., Chen, W. M., Cai, H., Gan, C., & Han, S. (2021). Memory-efficient patch-based inference for tiny deep learning. Advances in Neural Information Processing Systems, 34, 2346-2358.

MCUNetV2: Patch-based Inference

Let's see the computation in 2x2 patches

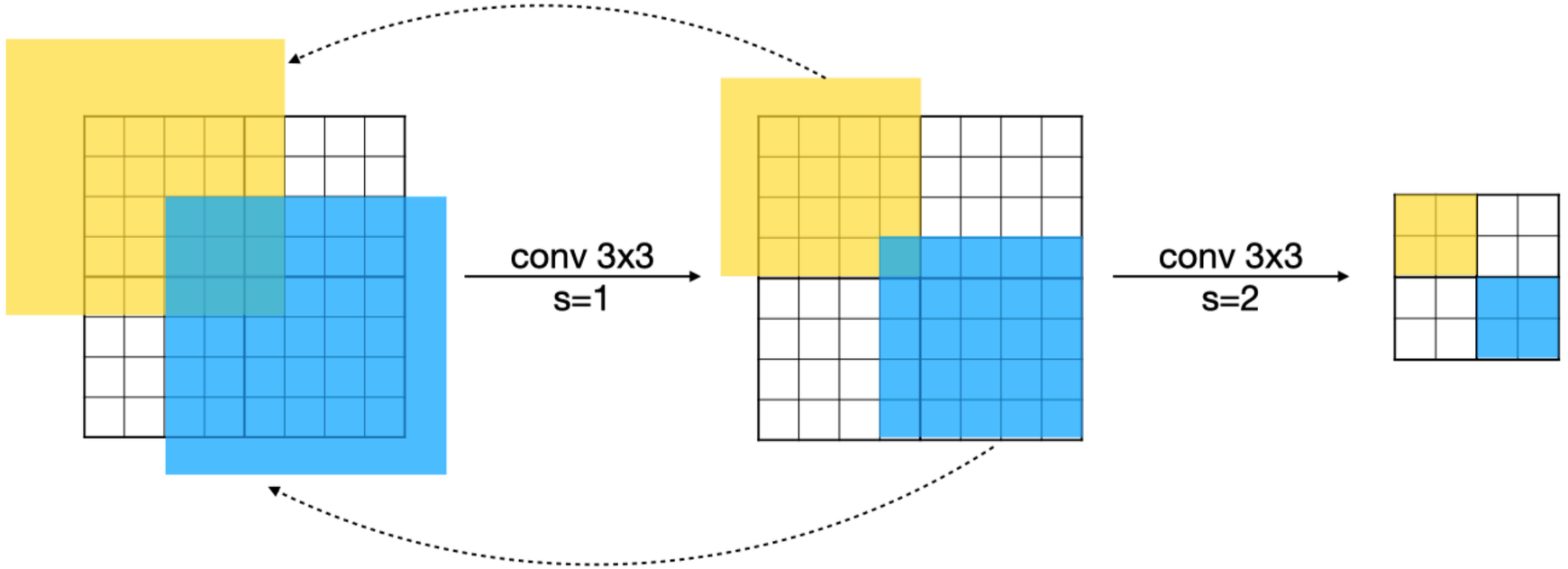


- Computation overhead from overlapping

Lin, J., Chen, W. M., Cai, H., Gan, C., & Han, S. (2021). Memory-efficient patch-based inference for tiny deep learning. Advances in Neural Information Processing Systems, 34, 2346-2358.

MCUNetV2: Patch-based Inference

Let's see the computation in 2x2 patches

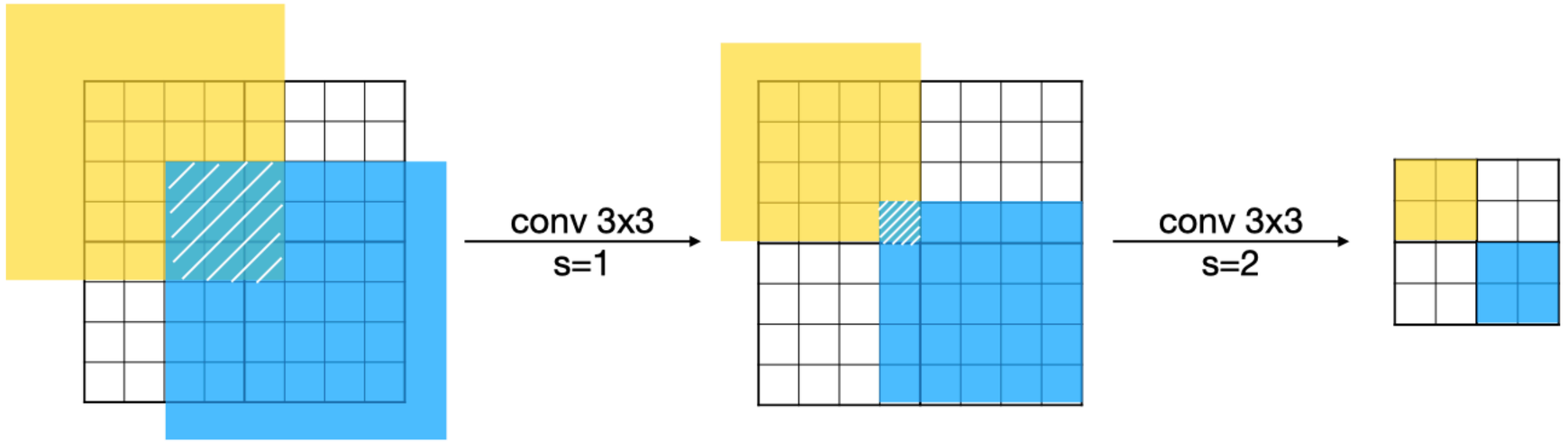


- Computation overhead from overlapping

Lin, J., Chen, W. M., Cai, H., Gan, C., & Han, S. (2021). Memory-efficient patch-based inference for tiny deep learning. Advances in Neural Information Processing Systems, 34, 2346-2358.

MCUNetV2: Patch-based Inference

Let's see the computation in 2x2 patches

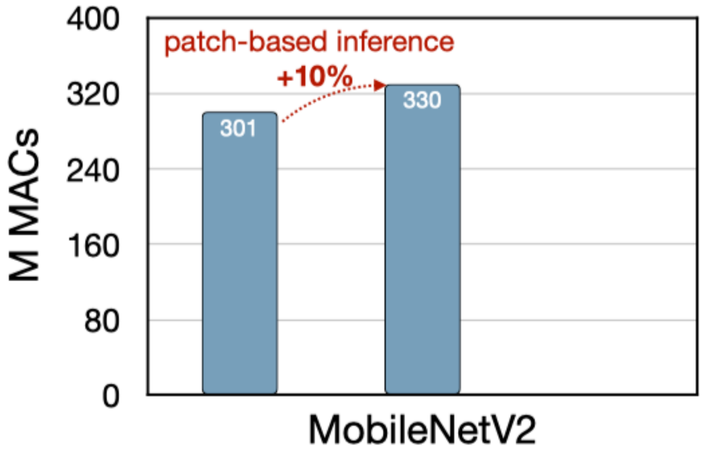


- Computation overhead from overlapping
- Spatial overlapping gets larger as the receptive field grows. Leading to larger computation overhead

Lin, J., Chen, W. M., Cai, H., Gan, C., & Han, S. (2021). Memory-efficient patch-based inference for tiny deep learning. Advances in Neural Information Processing Systems, 34, 2346-2358.

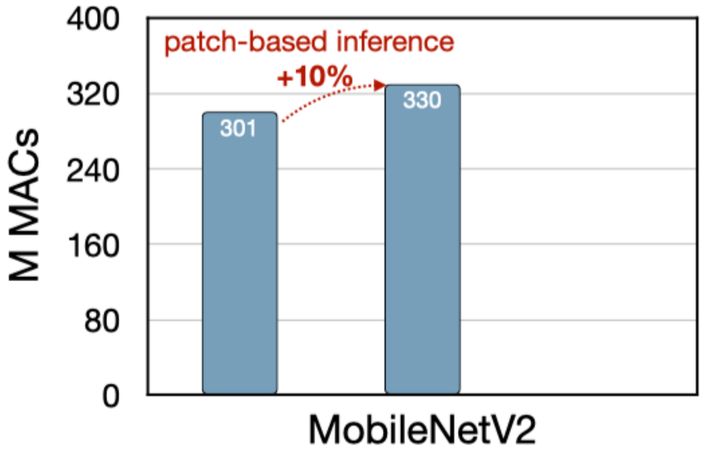
MCUNetV2: Patch-based Inference

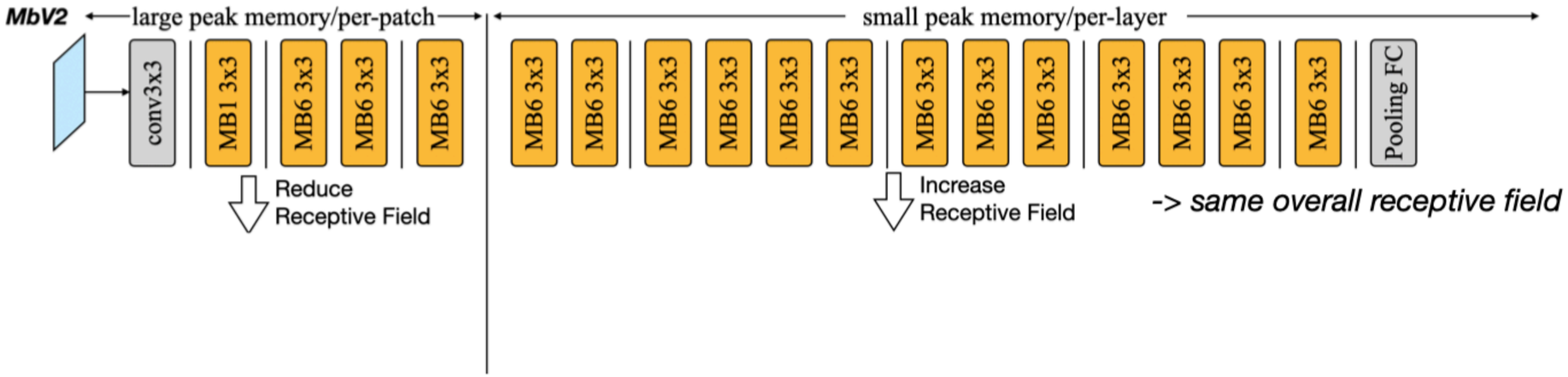
Problem: halo leads to repeated computation



Lin, J., Chen, W. M., Cai, H., Gan, C., & Han, S. (2021). Memory-efficient patch-based inference for tiny deep learning. Advances in Neural Information Processing Systems, 34, 2346-2358.

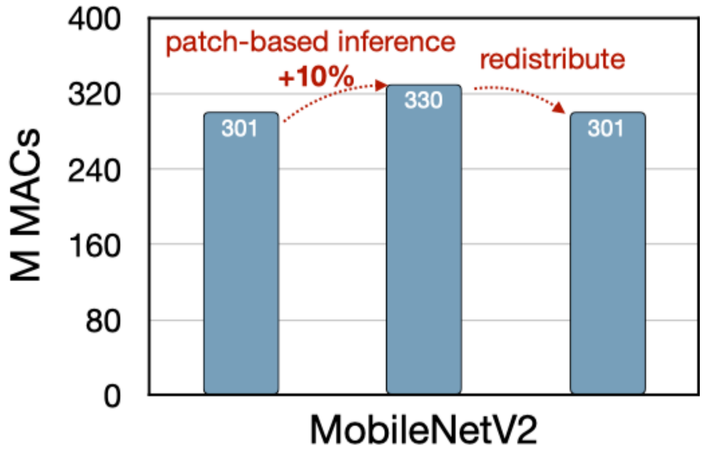
MCUNetV2: Network Redistribution to Reduce Overhead



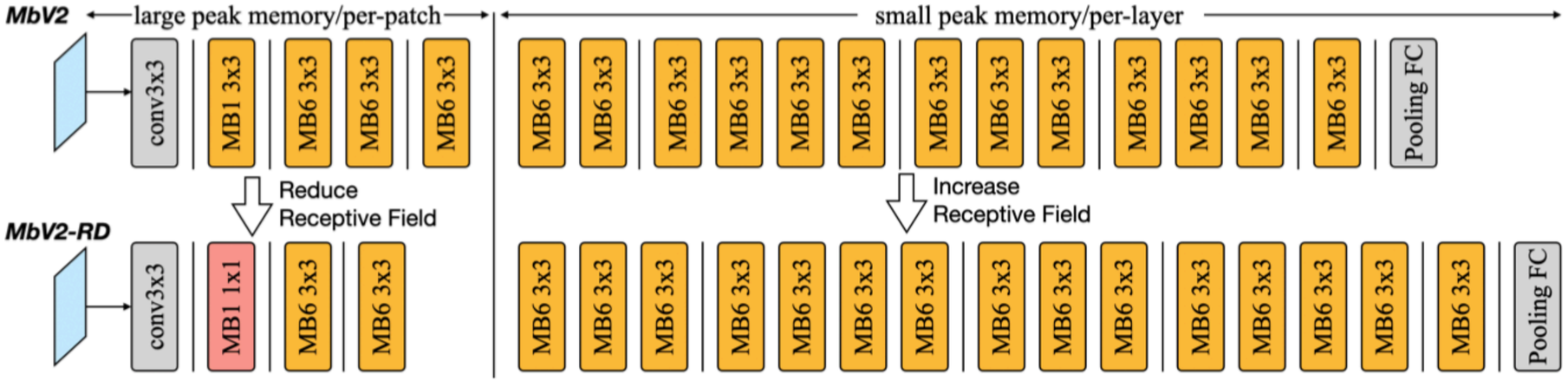


Lin, J., Chen, W. M., Cai, H., Gan, C., & Han, S. (2021). Memory-efficient patch-based inference for tiny deep learning. Advances in Neural Information Processing Systems, 34, 2346-2358.

MCUNetV2: Network Redistribution to Reduce Overhead

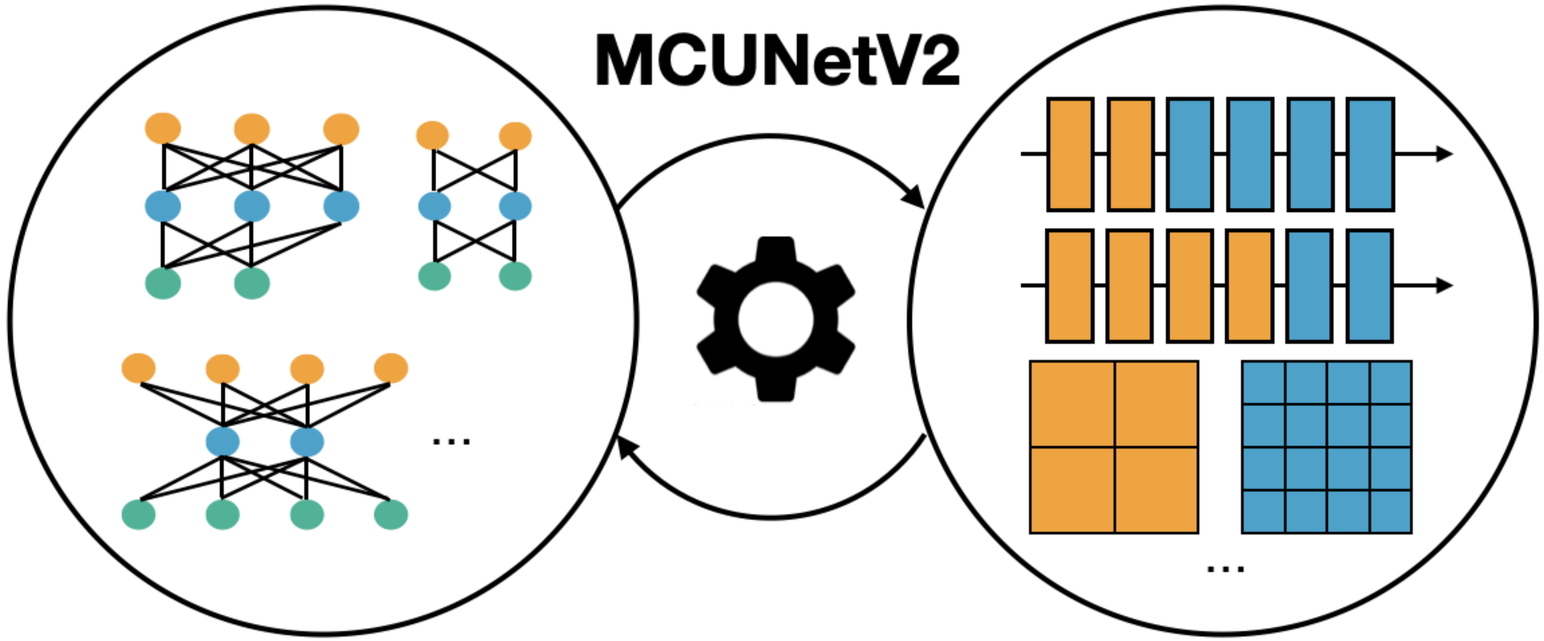


- Same performance on
  - Image classification
  - Object detection
  - ...



Lin, J., Chen, W. M., Cai, H., Gan, C., & Han, S. (2021). Memory-efficient patch-based inference for tiny deep learning. Advances in Neural Information Processing Systems, 34, 2346-2358.

MCUNetV2: Joint Automated Search for Optimization



Neural architecture
#layers
#channels
kernel size
...

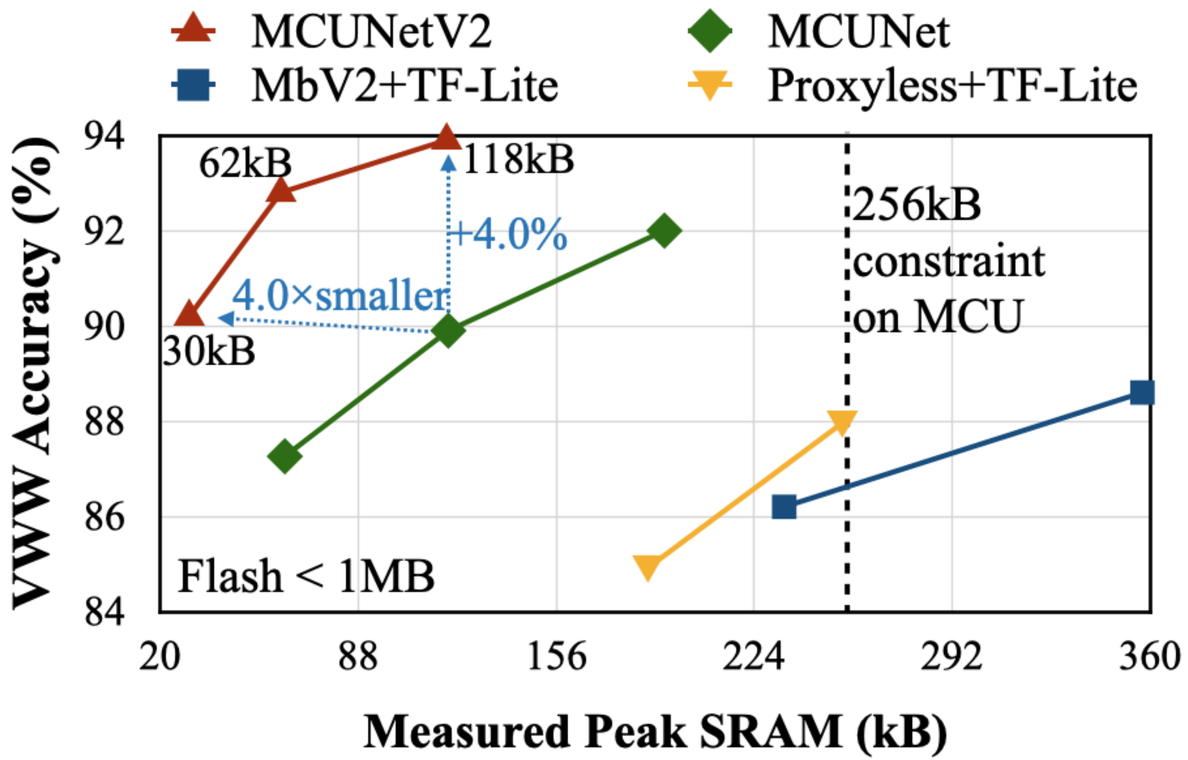
Inference scheduling
#patches
#layers for patch-based
other knobs from TinyEngine*
...

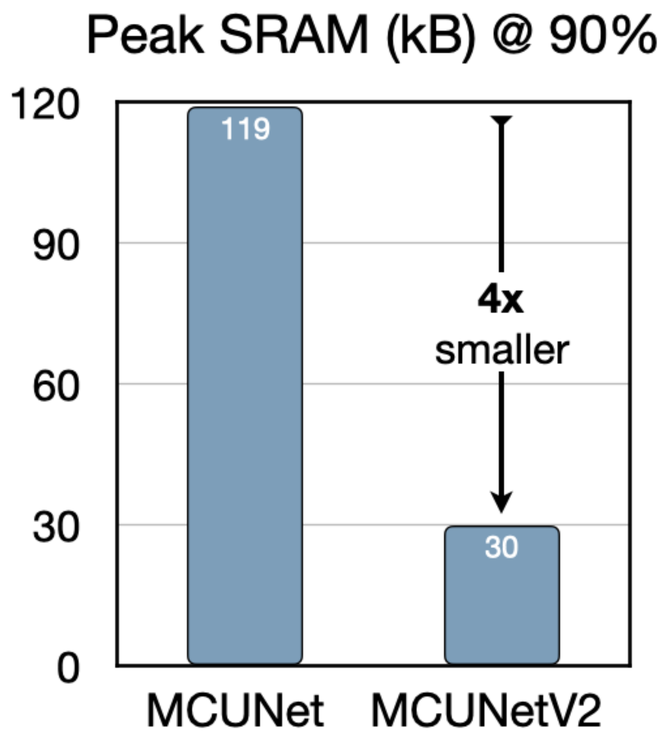
Lin, J., Chen, W. M., Cai, H., Gan, C., & Han, S. (2021). Memory-efficient patch-based inference for tiny deep learning. Advances in Neural Information Processing Systems, 34, 2346-2358.

MCUNetV2 for Tiny Image Classification

Visual Wake Words under 32KB memory

- Higher accuracy, 4x lower SRAM

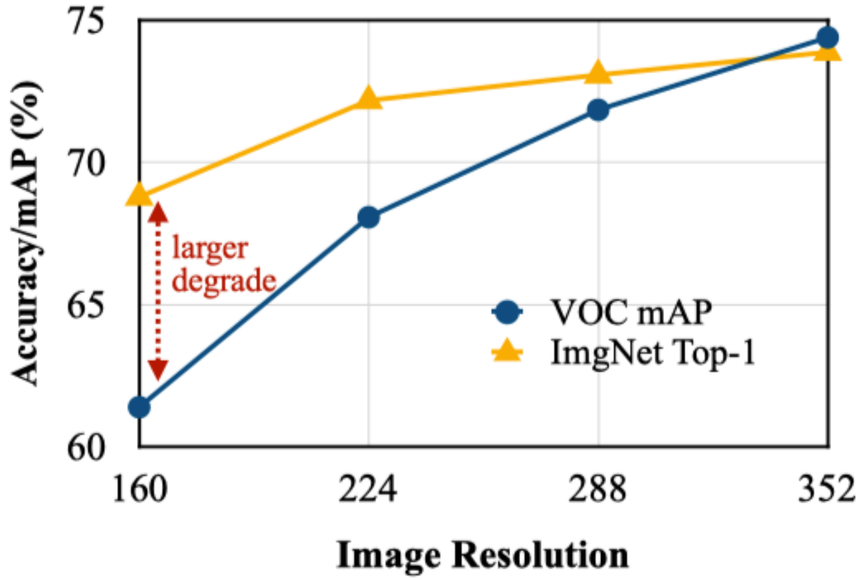


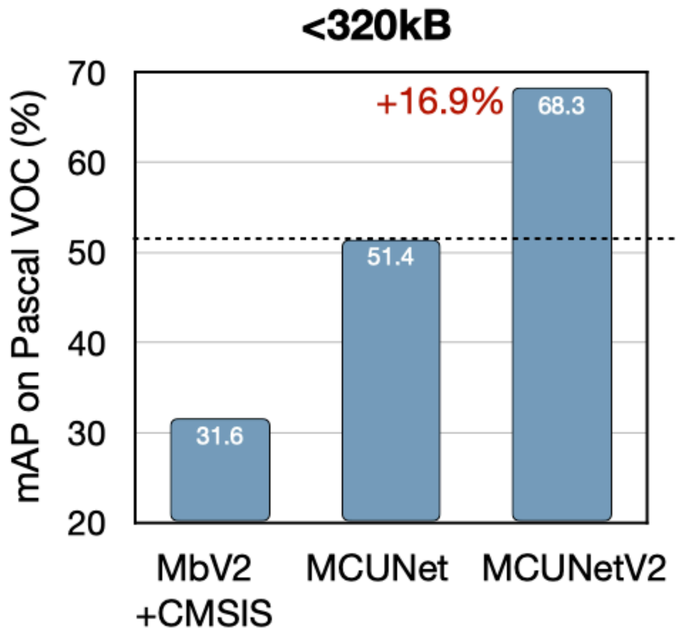


Lin, J., Chen, W. M., Cai, H., Gan, C., & Han, S. (2021). Memory-efficient patch-based inference for tiny deep learning. Advances in Neural Information Processing Systems, 34, 2346-2358.

MCUNetV2 for Tiny Object Detection

- Object detection is more sensitive to input resolution
- Patch-based inference allows for a larger resolution, improving detection performance





Lin, J., Chen, W. M., Cai, H., Gan, C., & Han, S. (2021). Memory-efficient patch-based inference for tiny deep learning. Advances in Neural Information Processing Systems, 34, 2346-2358.

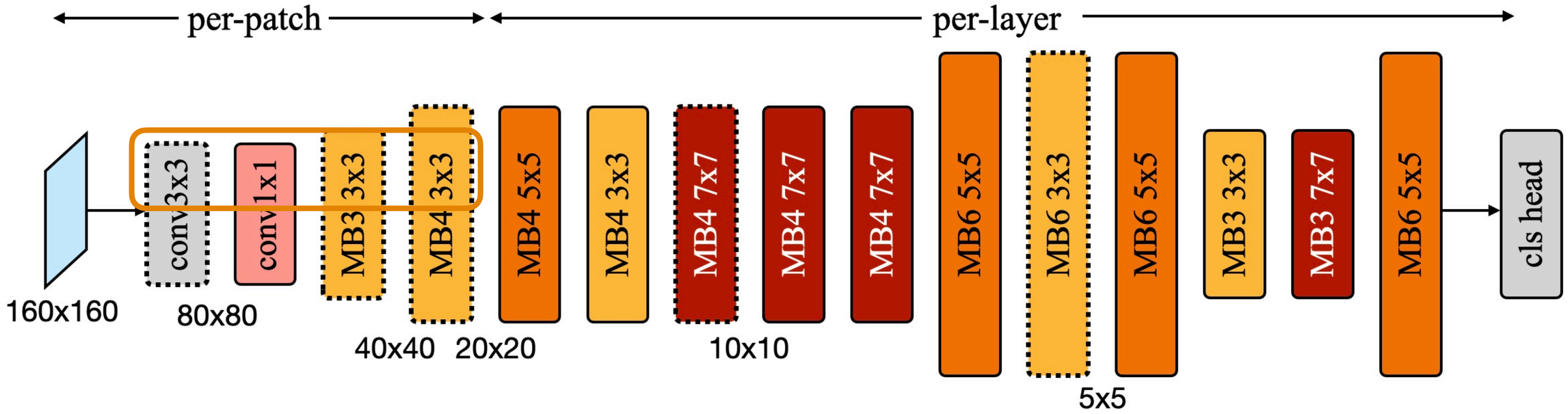
MCUNetV2 for Tiny Object Detection

Face detection on WIDER Face

- Better performance at the same memory budgets

Lin, J., Chen, W. M., Cai, H., Gan, C., & Han, S. (2021). Memory-efficient patch-based inference for tiny deep learning. Advances in Neural Information Processing Systems, 34, 2346-2358.

Dissecting MCUNetV2 Architecture

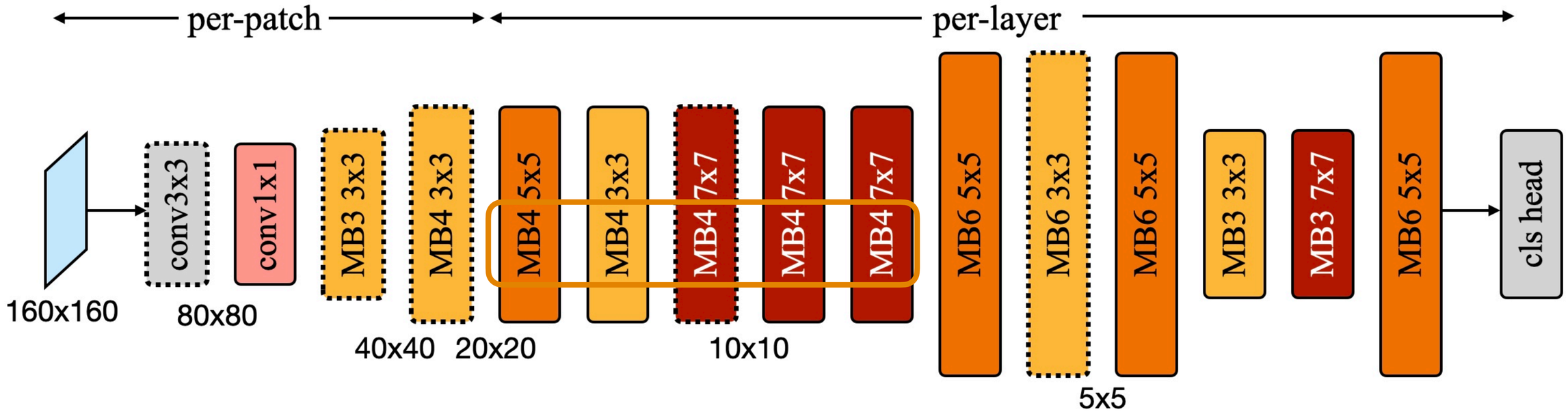


Sample architecture from VWW. Legend: MB{expansion}_{k_size}x{k_size}

- **Kernel size in the per-patch stage is small to reduce spatial overlapping**

Lin, J., Chen, W. M., Cai, H., Gan, C., & Han, S. (2021). Memory-efficient patch-based inference for tiny deep learning. Advances in Neural Information Processing Systems, 34, 2346-2358.

Dissecting MCUNetV2 Architecture

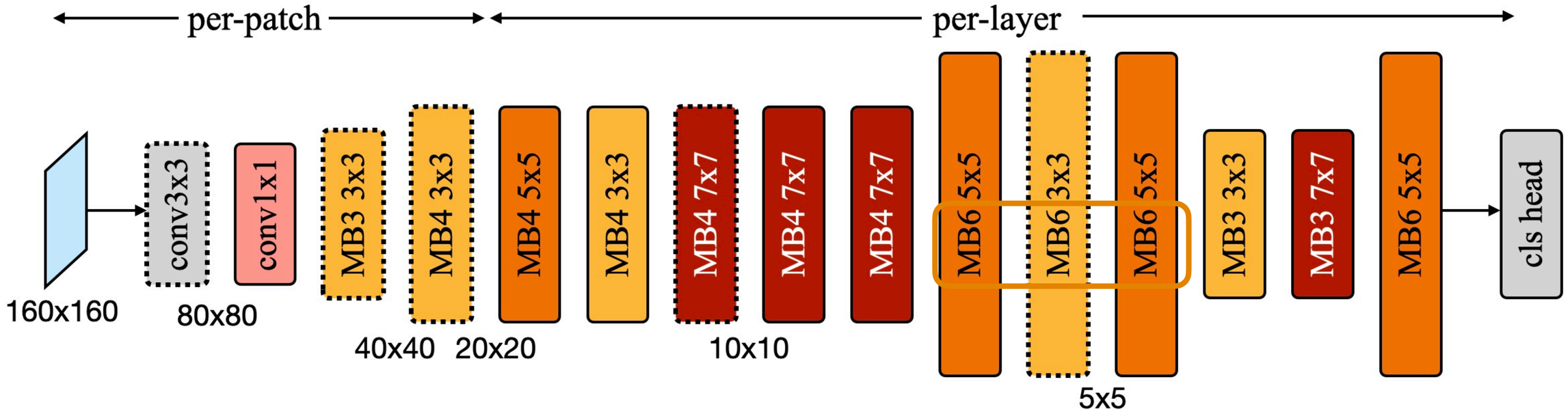


Sample architecture from VWW. Legend: MB{expansion}_{k_size}x{k_size}

- Kernel size in the per-patch stage is small to reduce spatial overlapping
- **The expansion ratio in the middle stage is small to reduce peak memory**

Lin, J., Chen, W. M., Cai, H., Gan, C., & Han, S. (2021). Memory-efficient patch-based inference for tiny deep learning. Advances in Neural Information Processing Systems, 34, 2346-2358.

Dissecting MCUNetV2 Architecture

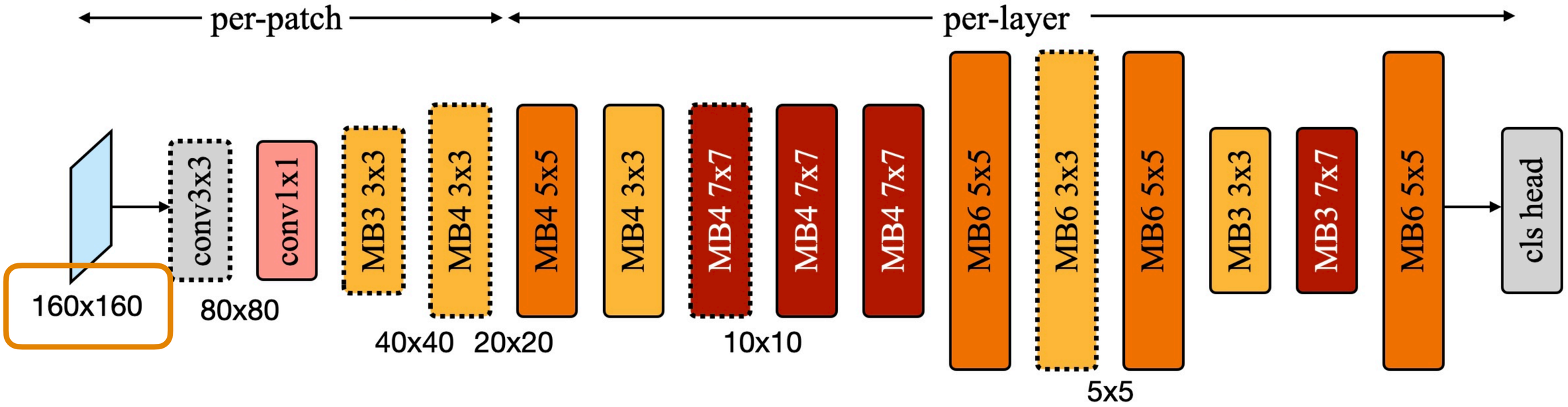


Sample architecture from VWW. Legend: MB{expansion}_{k_size}x{k_size}

- Kernel size in the per-patch stage is small to reduce spatial overlapping
- The expansion ratio in the middle stage is small to reduce peak memory; **and larger in the later stage to boost performance**

Lin, J., Chen, W. M., Cai, H., Gan, C., & Han, S. (2021). Memory-efficient patch-based inference for tiny deep learning. Advances in Neural Information Processing Systems, 34, 2346-2358.

Dissecting MCUNetV2 Architecture

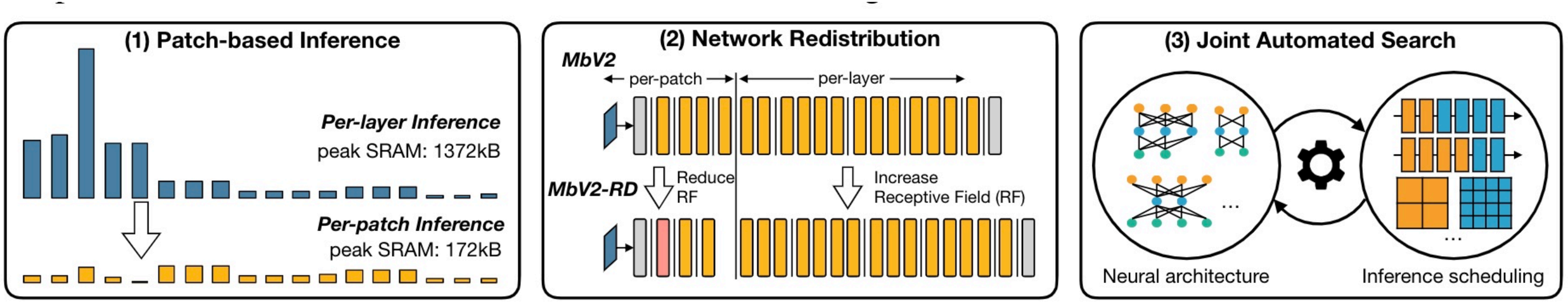


Sample architecture from VWW. Legend: MB{expansion}_{k_size}x{k_size}

- Kernel size in the per-patch stage is small to reduce spatial overlapping
- The expansion ratio in the middle stage is small to reduce peak memory; and larger in the later stage to boost performance
- **Larger input resolution for resolution-sensitive datasets like VWW (MCUNet: 128x128)**

Lin, J., Chen, W. M., Cai, H., Gan, C., & Han, S. (2021). Memory-efficient patch-based inference for tiny deep learning. Advances in Neural Information Processing Systems, 34, 2346-2358.

MCUNetV2

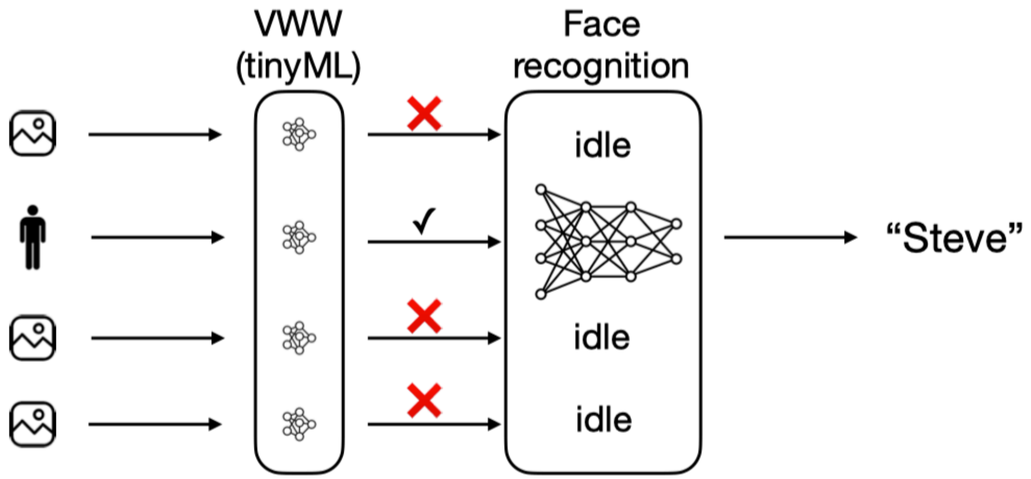


Lin, J., Chen, W. M., Cai, H., Gan, C., & Han, S. (2021). Memory-efficient patch-based inference for tiny deep learning. Advances in Neural Information Processing Systems, 34, 2346-2358.

Tiny Applications

Tiny Vision

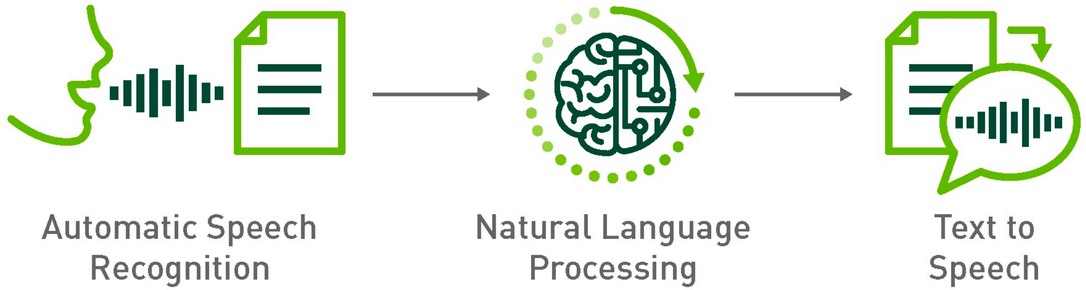
- Visual wake words can be used to activate the later ML pipeline
  - Vision counterpart of “Hey Siri”/“Ok Google”
  - For example, only enable face recognition when a person is in front of the camera, which saves a lot of energy. The face recognition model is much larger than the VMM model

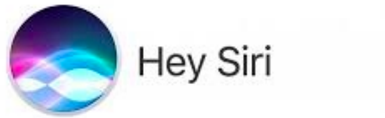


https://github.com/mit-han-lab/mcunet?tab=readme-ov-file

Tiny Audio

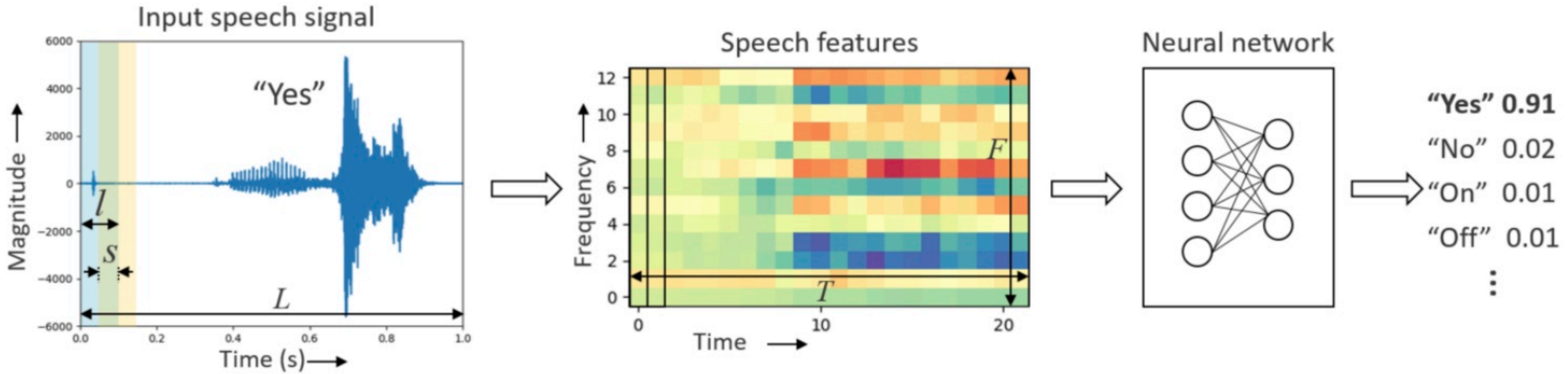
- Common audio applications in our daily life
  - Audio keywords/keyword spotting
  - Speech recognition
  - noise canceling
  - …





Tiny Audio

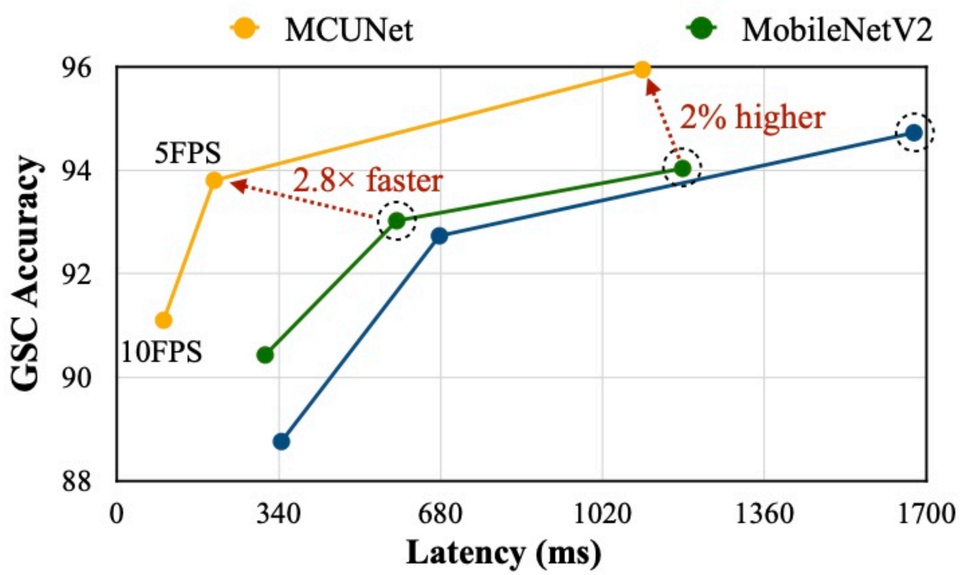
- The general pipeline of keyword spotting
  - Input: overlapping frames of length l with a stride s, giving a total of $T = (L - l)/s + 1$ frames.
  - Humen-engineered features: translating time-domain signal into a set of frequency-domain spectral coefficients
  - Neural network: generate the probabilities of the output classes
  - Spectral representations of speech have strong correlations in time and frequency



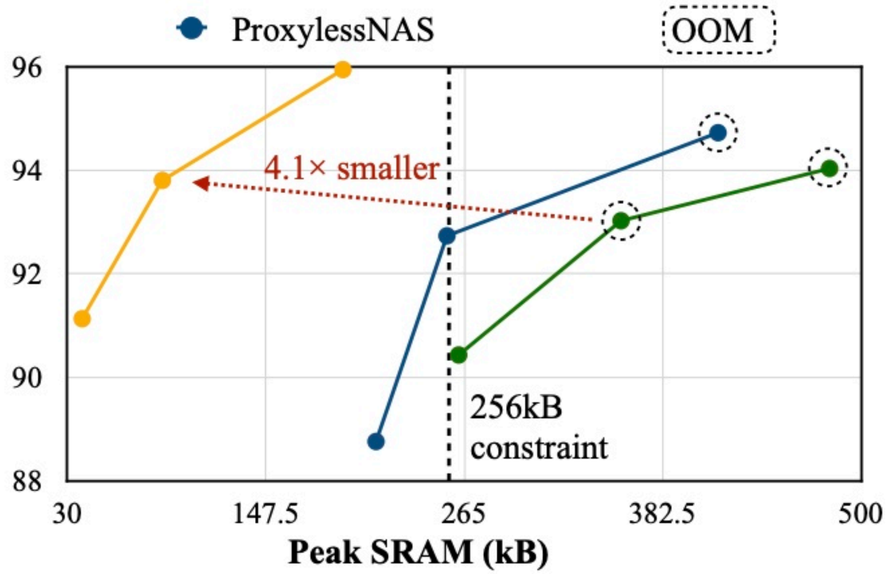
Zhang, Y., Suda, N., Lai, L., & Chandra, V. (2017). Hello edge: Keyword spotting on microcontrollers. arXiv preprint arXiv:1711.07128.

MCUNet on Speech Commands

Improves efficiency with software-hardware co-design



(a) Trade-off: accuracy *vs.* measured latency

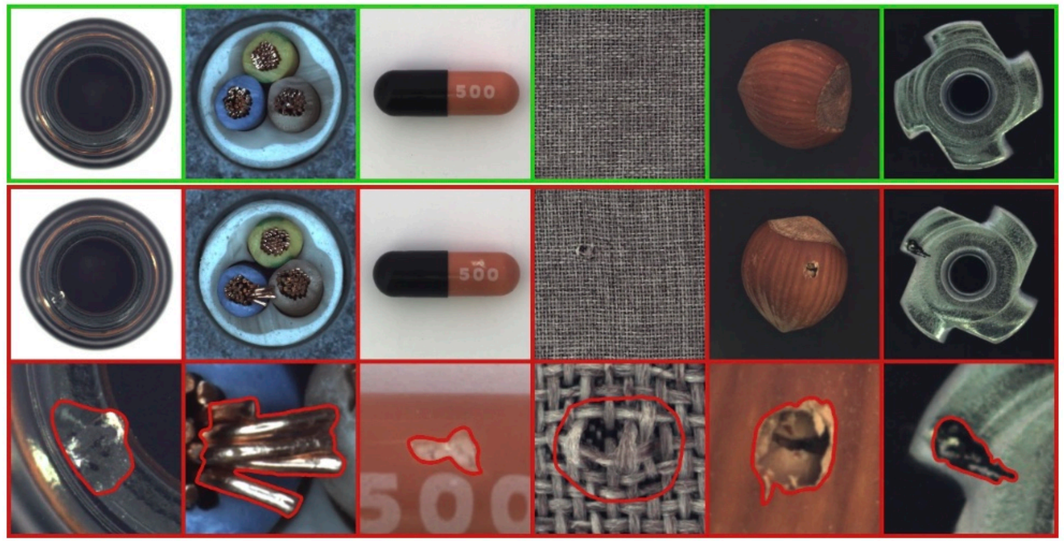


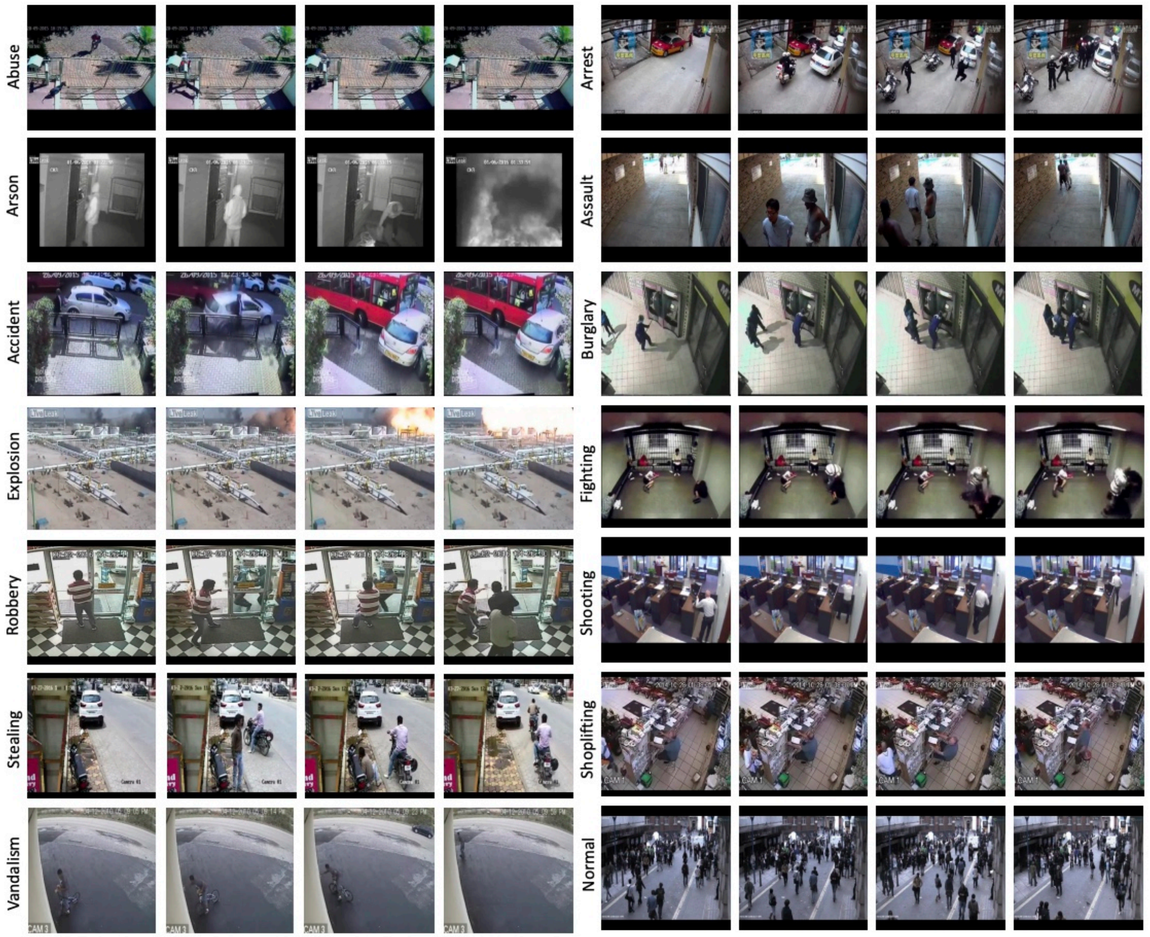
(b) Trade-off: accuracy *vs.* peak memory

Lin, J., Chen, W. M., Lin, Y., Gan, C., & Han, S. (2020). Mcunet: Tiny deep learning on iot devices. Advances in neural information processing systems, 33, 11711-11722.

Tiny Time Series/Anomaly Detection

- Anomaly detection applications
  - Video surveillance
  - Health care
  - Prevent fraud, adversary attacks

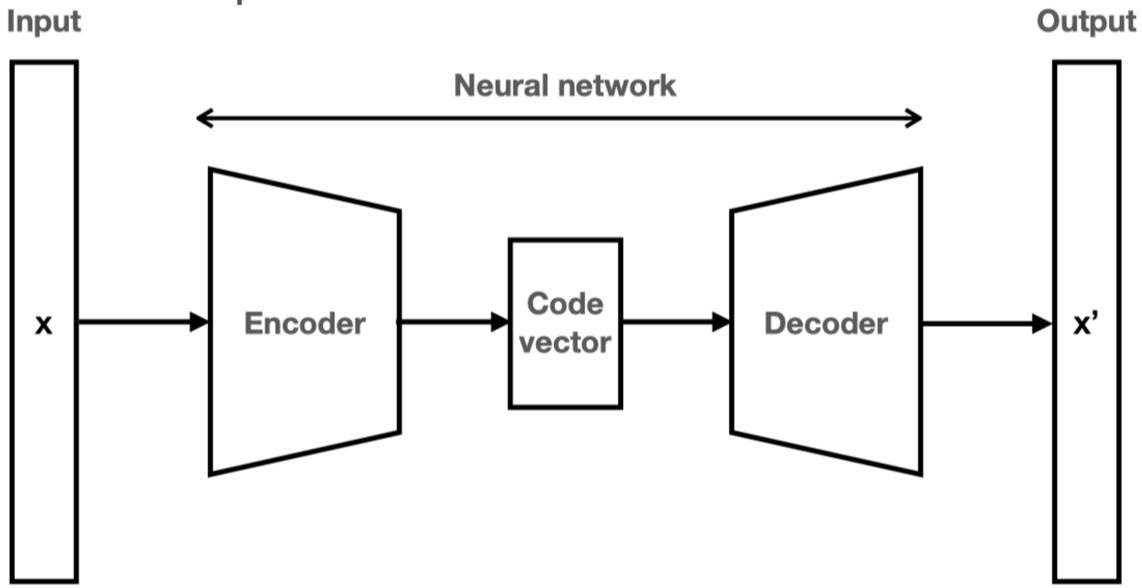




MVTec dataset: https://www.mvtec.com/company/research/datasets/mvtec-ad
Real-world Anomaly Detection in Surveillance Videos: https://arxiv.org/pdf/1801.04264.pdf

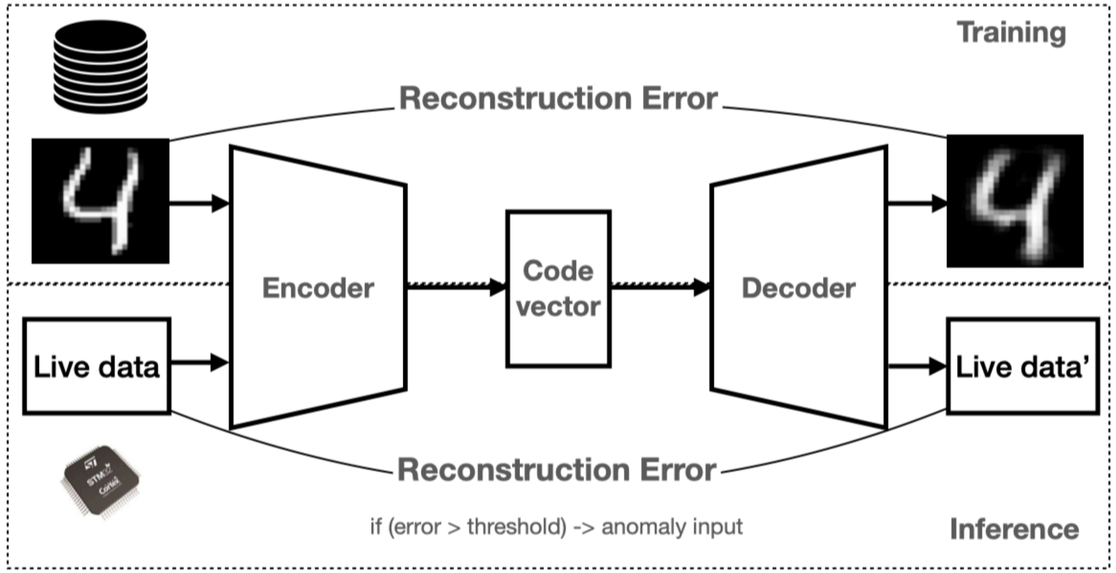
Detect Anomaly with Autoencoders

- An autoencoder is a neural network that predicts its input (ideally $x' = x$)
  - Encoder: compress the input into a lower-dimensional code vector
  - Code vector: abstraction of the input
  - Decoder: reconstruct the output from the code vector



Detect Anomaly with Autoencoders

- Training: minimizing the reconstruction error
- Inference: detect anomaly inputs with live data
- Properties of autoencoders
  - Unsupervised: we don’t need labels for training
  - Data-specific: they can only meaningfully compress data similar to the training dataset
  - Lossy: the output will not be the same as the input



Reference

- Lin, J., Chen, W. M., Lin, Y., Gan, C., & Han, S. (2020). Mcunet: Tiny deep learning on iot devices. Advances in neural information processing systems, 33, 11711-11722.

- Lin, J., Chen, W. M., Cai, H., Gan, C., & Han, S. (2021). Memory-efficient patch-based inference for tiny deep learning. Advances in Neural Information Processing Systems, 34, 2346-2358.

- Zhang, Y., Suda, N., Lai, L., & Chandra, V. (2017). Hello edge: Keyword spotting on microcontrollers. arXiv preprint arXiv:1711.07128.

- MCUNet and TinyML [MIT 6.5940]