

tinyML: neural networks design principles, scaling strategies and beyond

Francesco Paissan

Fondazione Bruno Kessler
fpaissan@fbk.eu

October 20, 2023

- **AI at the very edge:** artificial intelligence and advanced signal processing on resource-constrained wireless sensing platforms (MCU-based end-devices);
- **Energy efficient wireless embedded systems:** Wireless Sensor Networks, Wearable electronics, Internet of Things;
- Application in **HCI**, smart cities (always-on and event-based audio and vision sensing), **rehabilitation**, context-aware scenarios and **ambient intelligence**.



Presentation Overview

1 Introduction

What is tinyML?

Why tinyML?

Challenges of tinyML

2 Neural Network design

Rise and development of CNNs

tinyML-first CNNs

Hardware-Aware Scaling

3 Some applications...

YOLO-based

Zero-shot audio classification

References to audio-video processing at the edge

1 Introduction

What is tinyML?

Why tinyML?

Challenges of tinyML

2 Neural Network design

Rise and development of CNNs

tinyML-first CNNs

Hardware-Aware Scaling

3 Some applications...

YOLO-based

Zero-shot audio classification

References to audio-video processing at the edge

What is tinyML?

- a fast-growing subfield of machine learning targeting **on-device and near-sensor processing**;

What is tinyML?

- a fast-growing subfield of machine learning targeting **on-device and near-sensor processing**;
- includes hardware, algorithms and software capable of performing data analytics at extremely **low power consumption**;

What is tinyML?

- a fast-growing subfield of machine learning targeting **on-device and near-sensor processing**;
- includes hardware, algorithms and software capable of performing data analytics at extremely **low power consumption**;
- enables a variety of **always-on** use-cases and targets battery operated and energy neutral devices;

Why tinyML?

- **Bandwidth reduction** - reduction in amount of data that needs to be transmitted;

Why tinyML?

- **Bandwidth reduction** - reduction in amount of data that needs to be transmitted;
- **Low power consumption** - many low power platforms, can be powered by tiny solar panels;

Why tinyML?

- **Bandwidth reduction** - reduction in amount of data that needs to be transmitted;
- **Low power consumption** - many low power platforms, can be powered by tiny solar panels;
- **Infrastructure sustainability** - lower and cheaper maintenance;

Why tinyML?

- **Bandwidth reduction** - reduction in amount of data that needs to be transmitted;
- **Low power consumption** - many low power platforms, can be powered by tiny solar panels;
- **Infrastructure sustainability** - lower and cheaper maintenance;
- **Privacy by Design** - personal data never leaves the device and is not stored;

Why tinyML?

- **Bandwidth reduction** - reduction in amount of data that needs to be transmitted;
- **Low power consumption** - many low power platforms, can be powered by tiny solar panels;
- **Infrastructure sustainability** - lower and cheaper maintenance;
- **Privacy by Design** - personal data never leaves the device and is not stored;
- Robust, cheap, scalable, ...

Challenges of tinyML?



WORKSTATION

RAM: 10-100 GB

Storage: 10s of TB

Speed: 100 Billions of ops/s



PC/SBC

RAM: 1-10 GB

Storage: 10-100 GB

Speed: 1-10 Billions of ops/s



MCU

RAM: 10s - 100s of KBs

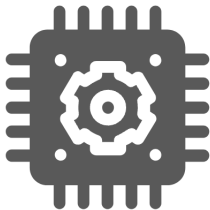
Storage: KBs - MBs

Speed: Millions of ops/s

$\div 10$

$\div 10\ 000$

Target platforms

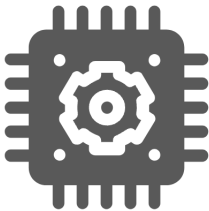


microcontrollers, SBC,
neuromorphic processors, ...

Target platforms

small memory availability

(kB - MB)



microcontrollers, SBC,
neuromorphic processors, ...

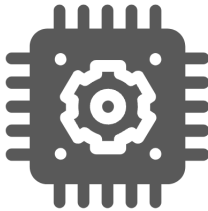
Target platforms

small memory availability

(kB - MB)

low operations per second

(million ops/s)



microcontrollers, SBC,
neuromorphic processors, ...

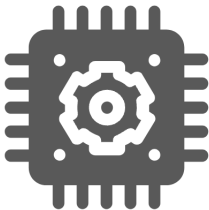
Target platforms

small memory availability

(kB - MB)

low operations per second

(million ops/s)



microcontrollers, SBC,

neuromorphic processors, ...

low working memory

(kB - MB)

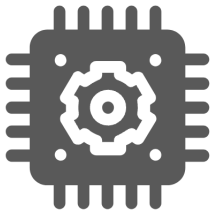
Target platforms

small memory availability

(kB - MB)

low operations per second

(million ops/s)



microcontrollers, SBC,

neuromorphic processors, ...

low working memory

(kB - MB)

limited operations support
(generally optimized for CNNs)

1 Introduction

What is tinyML?

Why tinyML?

Challenges of tinyML

2 Neural Network design

Rise and development of CNNs

tinyML-first CNNs

Hardware-Aware Scaling

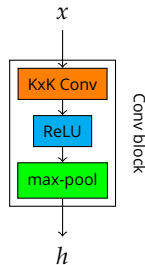
3 Some applications...

YOLO-based

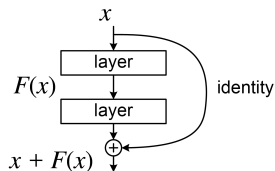
Zero-shot audio classification

References to audio-video processing at the edge

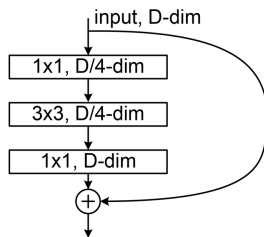
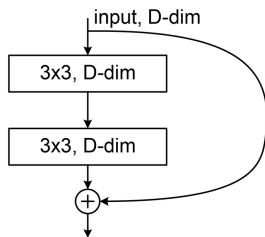
- ground-breaking CNN from 2012 [Krizhevsky et al., 2012] was the first one to get good results on ImageNet;
- composed by a **sequence of convolutional blocks**, with varying configurations;



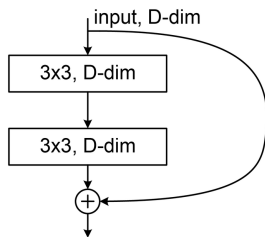
- improves the performance by enabling deeper networks via **skip connections** [Wightman et al., 2021];
- again, is composed by a **sequence of convolutional blocks**, called residual blocks;
- residual blocks follow a wide/narrow/wide structure in the number of channels;



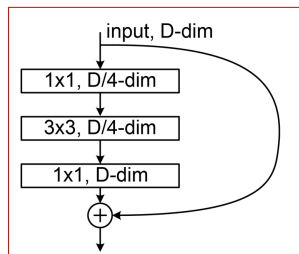
ResBlock variants



ResBlock variants



Wide-narrow-wide channel structure

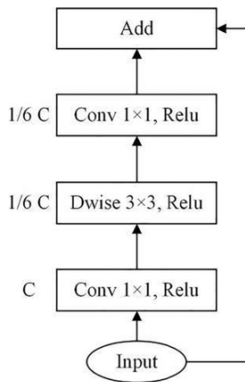


- tries to improve CNN efficiency by proposing the **inverted residual block**[Howard et al., 2017];

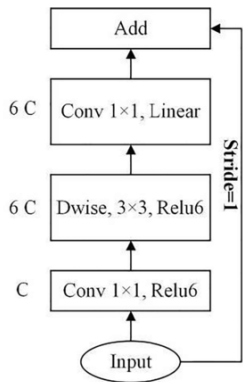
- tries to improve CNN efficiency by proposing the **inverted residual block**[Howard et al., 2017];
- differently from a ResBlock, this uses a narrow/wide/narrow structure in the number of channels;

- tries to improve CNN efficiency by proposing the **inverted residual block**[Howard et al., 2017];
- differently from a ResBlock, this uses a narrow/wide/narrow structure in the number of channels;
- additionally, groups are used inside the convolutions to reduce the computational complexity (depthwise convolutions);

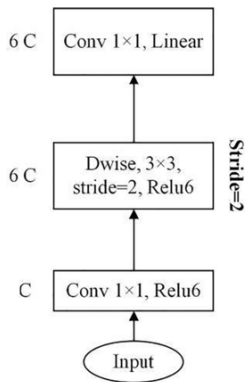
Inverted Convolutional Block



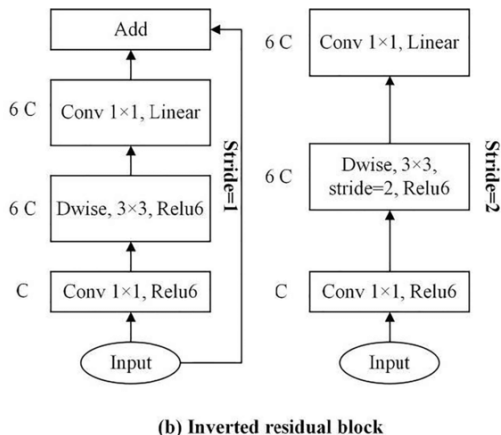
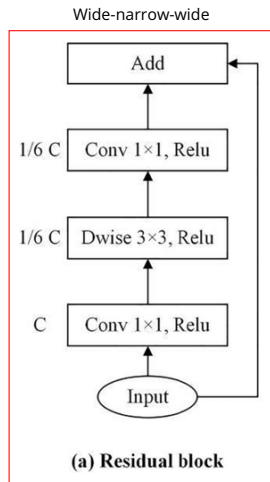
(a) Residual block



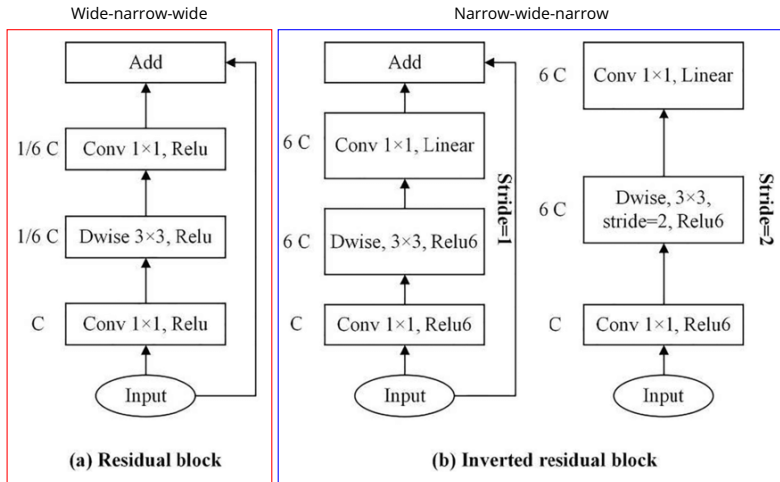
(b) Inverted residual block



Inverted Convolutional Block

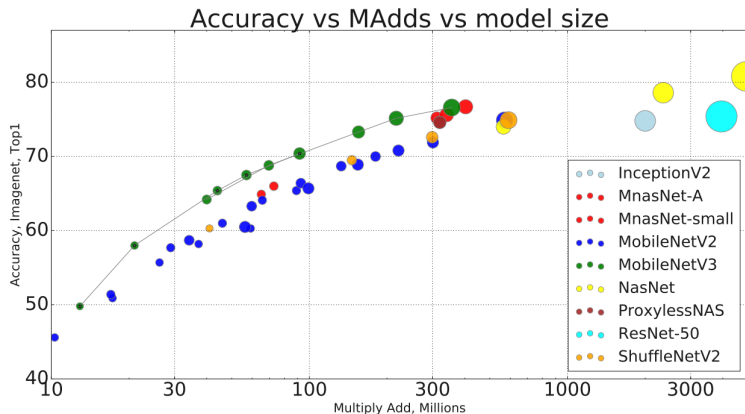


Inverted Convolutional Block

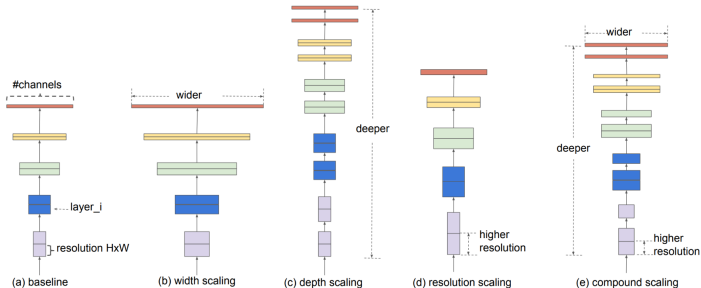


Just for comparison...

As of MobilNetv3 (Nov. 2019)...



- focuses on how we 'should' be scaling CNNs to obtain optimal performance[Tan and Le, 2019];
- introduces the concept of compound scaling (i.e. scaling all dimensions is better than one dimension at a time);



Shortcomings of mainstream CNNs

- these neural networks are **too demanding** to run on edge devices and/or compromise performance too much trying to fit;

Shortcomings of mainstream CNNs

- these neural networks are **too demanding** to run on edge devices and/or compromise performance too much trying to fit;
- the convolutional block is not **designed ad-hoc** to exploit all the capabilities of edge devices;

Shortcomings of mainstream CNNs

- these neural networks are **too demanding** to run on edge devices and/or compromise performance too much trying to fit;
- the convolutional block is not **designed ad-hoc** to exploit all the capabilities of edge devices;
- compound scaling changes all the computational complexities in a **coupled** way;

- a neural network that can **scale to low computational complexity** (≤ 1 MB of FLASH, ≤ 1 MB of RAM);
- a convolutional block that is designed to **exploit the available resources** maximally;
- a scaling strategy that allows fitting neural networks on **different edge platforms** based on the applications scenarios;

- based on **inverted residual blocks**, modified to decouple the computational axes;

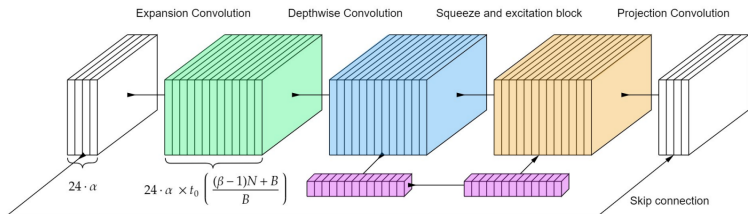
- based on **inverted residual blocks**, modified to decouple the computational axes;
- designed and optimized for **multimedia analytics** at the edge (audio-video);

- based on **inverted residual blocks**, modified to decouple the computational axes;
- designed and optimized for **multimedia analytics** at the edge (audio-video);
- has three main hyperparameters (α, β, t_0);
- controls RAM (t_0), FLASH (α) and operations (β);

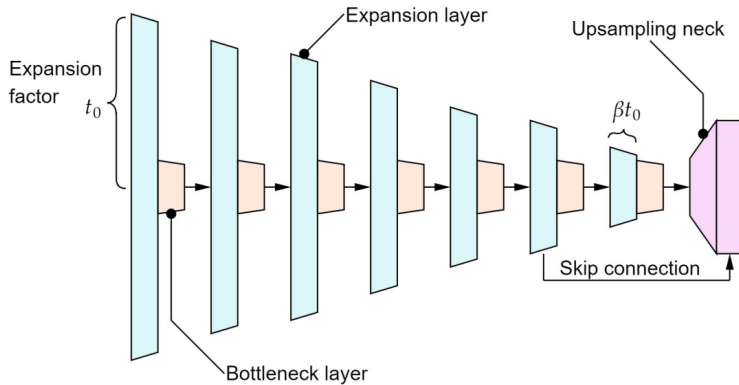
[Paissan et al., 2021]

PhiNets convolutional block

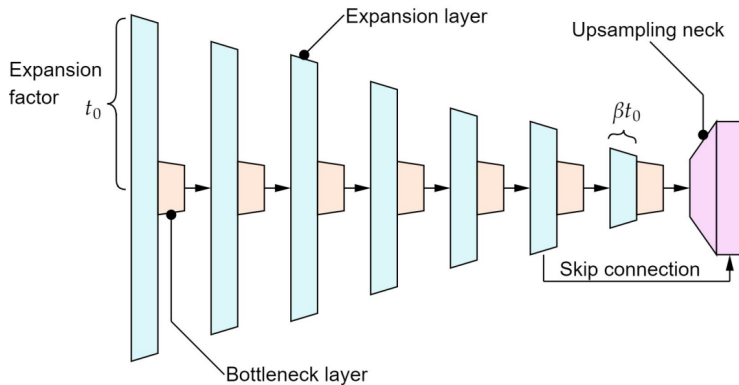
Narrow-wide-narrow structure for the number of channels...



The sequence of PhiNets conv blocks



The sequence of PhiNets conv blocks



Note: The expansion factor decreases with the increasing block number.

Designing an optimized convolutional block

- PhiNets are designed based on **indirect efficiency metrics**, thus could be an ideal version of edge CNNs;

Designing an optimized convolutional block

- PhiNets are designed based on **indirect efficiency metrics**, thus could be an ideal version of edge CNNs;
- what happens if we try to break free of the common standards for convolutional block design and investigate from first principles?

Designing an optimized convolutional block

- PhiNets are designed based on **indirect efficiency metrics**, thus could be an ideal version of edge CNNs;
- what happens if we try to break free of the common standards for convolutional block design and investigate from first principles?

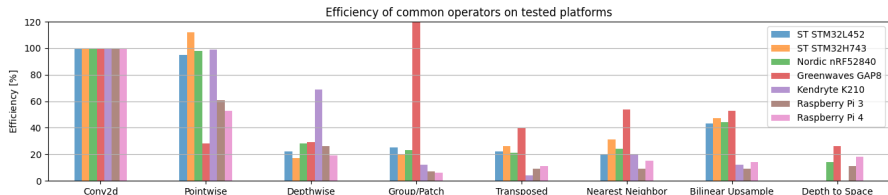
Let's see...

Definition 2.1

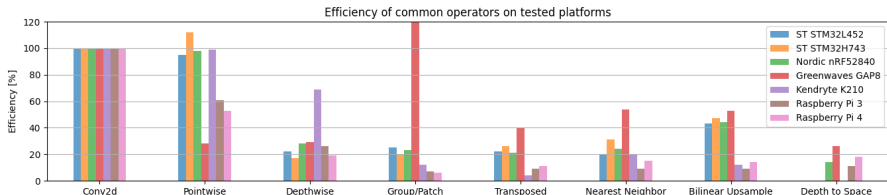
We assessed the actual efficiency of each operator (η_{op}) by calculating the ratio between the energy needed for a standard convolution (E_S) and the energy of the chosen operator (E_{op}) to perform an equivalent number of MACs.

$$\eta_{op} = \frac{E_S}{E_{op}}$$

Empirical evaluation of CNN operators...

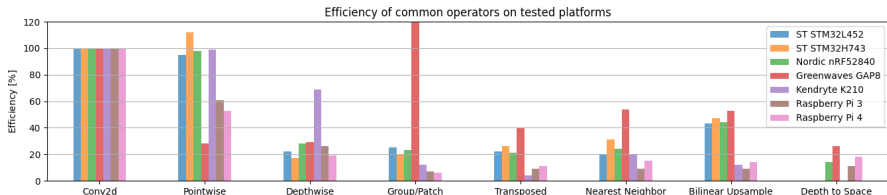


Empirical evaluation of CNN operators...



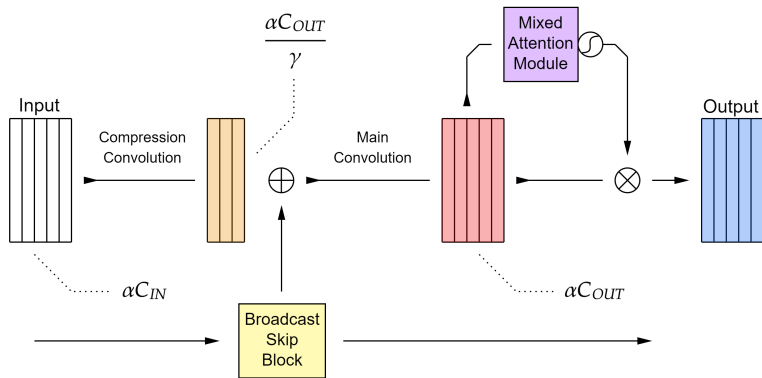
- this suggests that standard convolutions (AlexNet-style) are, on average, more efficient than other variants;

Empirical evaluation of CNN operators...

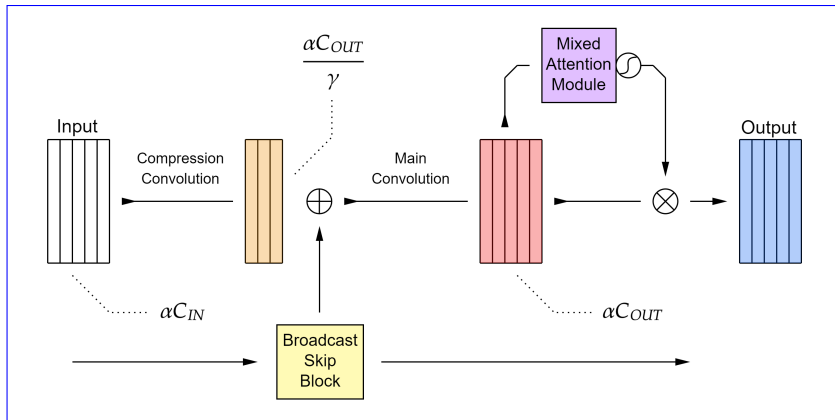


- this suggests that standard convolutions (AlexNet-style) are, on average, more efficient than other variants;
- but how do we exploit them with low parameter memory?

XiNet convolutional block

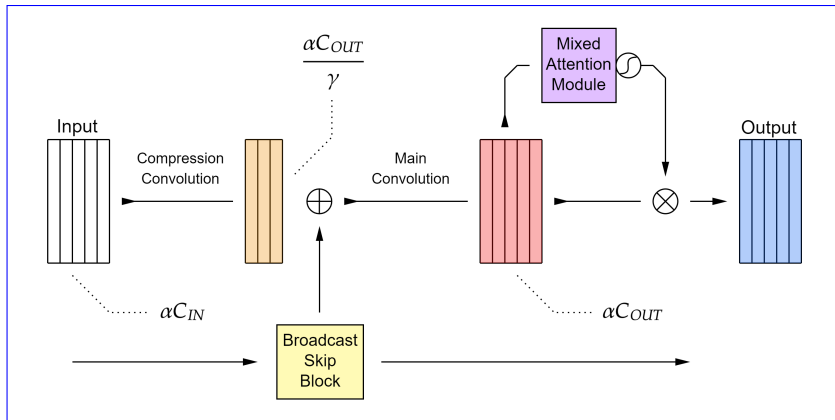


XiNet convolutional block

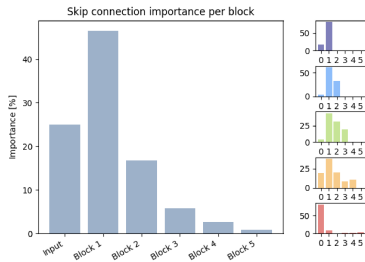


XiNet convolutional block

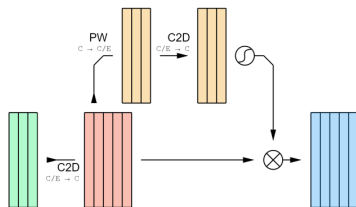
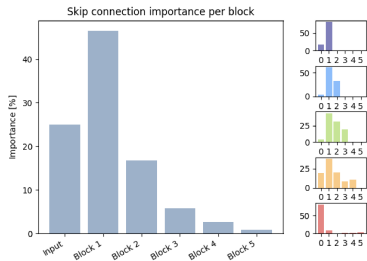
Wide-narrow-wide structure for channels, and much more...



Skip connections and attention block



Skip connections and attention block



- composed by a sequence of these convolutional blocks;

- composed by a sequence of these convolutional blocks;
- similarly to PhiNets, its computational complexity is controlled using **three hyperparameters** (α, γ, β) ;

- composed by a sequence of these convolutional blocks;
- similarly to PhiNets, its computational complexity is controlled using **three hyperparameters** (α, γ, β);
- designed based on the **empirical benchmark** of the different operators to be very efficient;

[Ancilotto et al., 2023c]

Hardware-aware scaling

- **scaling strategy** that exploits the advanced PhiNets and XiNet architectures;
- helps deploy CNNs on a wide variety of edge platforms via its one-shot network optimization procedure;
- **inverts the mapping between computational complexity and hyperparameters** so that it can be solved with a mathematical programming toolkit for specific computational requirements;

1 Introduction

What is tinyML?

Why tinyML?

Challenges of tinyML

2 Neural Network design

Rise and development of CNNs

tinyML-first CNNs

Hardware-Aware Scaling

3 Some applications...

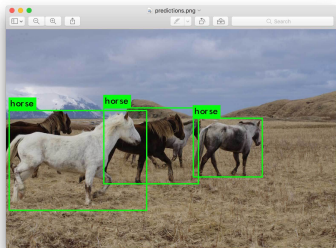
YOLO-based

Zero-shot audio classification

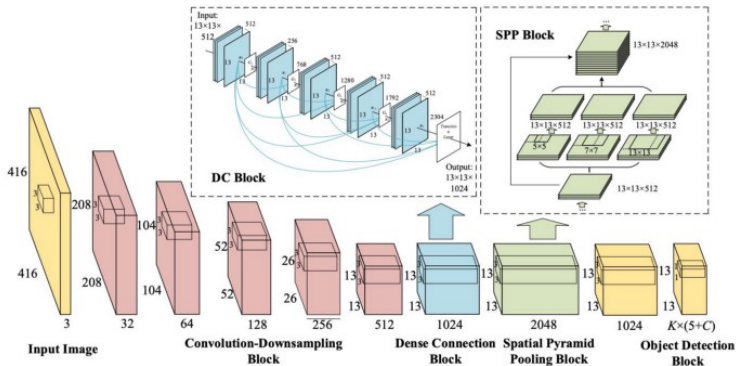
References to audio-video processing at the edge

You Only Look Once (YOLO)

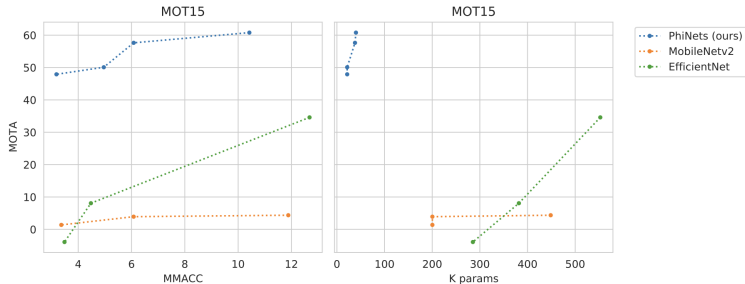
- originally proposed as an object detection pipeline;
- well known for its **good performance/complexity tradeoff**;
- mainly related to its ability to detect objects using **only one inference step** (no region proposal networks, etc...);
- recently extended to support image segmentation, keypoint detection/pose estimation;



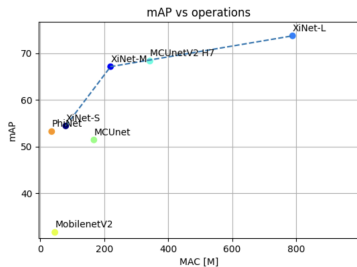
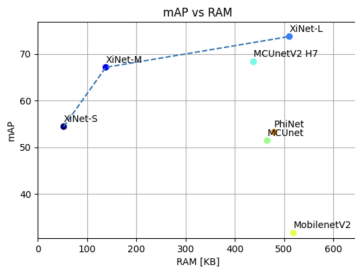
YOLO Architecture



YOLOPhinet



Deployed on an Arm-Cortex M7 MCU with 2 MB of internal Flash and 1 MB of RAM; achieves power requirements in the order of 10 mW.



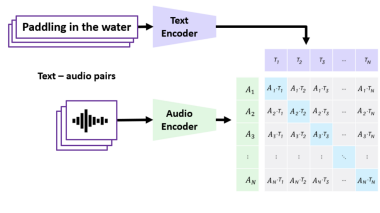
Deployed on an Arm-Cortex M7 MCU with 2 MB of internal Flash and 1 MB of RAM;
Achieves a reduction in the number of operations of $2\times$ and a reduction in RAM usage of $9\times$ with respect to MCUNet, with the same performance.

Contrastive Language-Audio pretraining

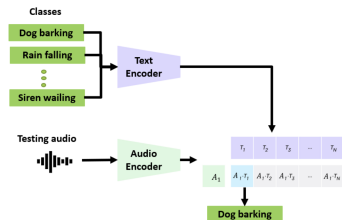
- learns a **similarity score** between two modalities (audio and text);
- can be exploited for **zero-shot** classification;
- makes the network very **flexible** wrt the applications scenario they can be deployed to;

Zero-shot classification

1. Contrastive Pretraining



2. Use pretrained encoders for zero-shot prediction in a new dataset or task

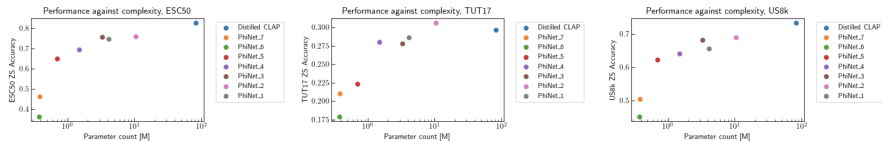


- exploits the learned similarity score to learn a **more efficient audio network** (via a distillation process);

- exploits the learned similarity score to learn a **more efficient audio network** (via a distillation process);
- assumes the pre-trained **text encoder** does **not** need to be **deployed**;

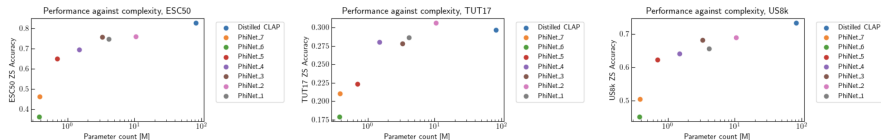
- exploits the learned similarity score to learn a **more efficient audio network** (via a distillation process);
- assumes the pre-trained **text encoder** does **not** need to be **deployed**;
- achieves good performance-complexity tradeoff for ZS classification, and state-of-the-art for a benchmark;

tinyCLAP: performance



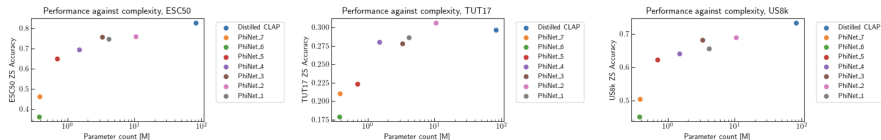
- follows a common power-law scaling behaviour;

tinyCLAP: performance



- follows a common power-law scaling behaviour;
- was not yet deployed on edge platforms (WIP);

tinyCLAP: performance



- follows a common power-law scaling behaviour;
- was not yet deployed on edge platforms (WIP);
- uses only 6% of original CLAP parameters, with a minor ZS accuracy drop of only 4% averaged on all benchmarks;

Additional references to our works

Following is a list of references to works related to the topics discussed in the presentation:

- Video processing:
[Ancilotto et al., 2022, Paissan et al., 2021, Ancilotto et al., 2023c]
- Generative modeling: [Ancilotto et al., 2023a, Ancilotto et al., 2023b]
- Audio processing:
[Paissan et al., 2022, Brutti et al., 2022, Ali et al., 2023, Paissan et al., 2023]
- Multimodal processing: tinyCLAP (under review)

The End

Questions? Comments?

References I






Ali, M. N., Paissan, F., Falavigna, D., and Brutti, A. (2023).
Scaling strategies for on-device low-complexity source
separation with conv-tasnet.
ArXiv, [abs/2303.03005](https://arxiv.org/abs/2303.03005).



Ancilotto, A., Paissan, F., and Farella, E. (2022).
On the role of smart vision sensors in energy-efficient
computer vision at the edge.
*2022 IEEE International Conference on Pervasive Computing and
Communications Workshops and other Affiliated Events (PerCom
Workshops)*, pages 497–502.

References II

-  Ancilotto, A., Paissan, F., and Farella, E. (2023a). Phinet-gan: Bringing real-time face swapping to embedded devices. *2023 IEEE International Conference on Pervasive Computing and Communications Workshops and other Affiliated Events (PerCom Workshops)*, pages 677–682.
-  Ancilotto, A., Paissan, F., and Farella, E. (2023b). Ximswap: many-to-many face swapping for tinymml. *ACM Transactions on Embedded Computing Systems*.
-  Ancilotto, A., Paissan, F., and Farella, E. (2023c). Xinet: Efficient neural networks for tinymml. *ICCV2023*.



References III

-  Brutti, A., Paissan, F., Ancilotto, A., and Farella, E. (2022). Optimizing phinet architectures for the detection of urban sounds on low-end devices. *2022 30th European Signal Processing Conference (EUSIPCO)*, pages 1121–1125.
-  Howard, A. G., Zhu, M., Chen, B., Kalenichenko, D., Wang, W., Weyand, T., Andreetto, M., and Adam, H. (2017). Mobilenets: Efficient convolutional neural networks for mobile vision applications. *ArXiv*, abs/1704.04861.
-  Krizhevsky, A., Sutskever, I., and Hinton, G. E. (2012). Imagenet classification with deep convolutional neural networks. *Communications of the ACM*, 60:84 – 90.

References IV

-  Paissan, F., Ancilotto, A., Brutti, A., and Farella, E. (2022). Scalable neural architectures for end-to-end environmental sound classification. *ICASSP 2022 - 2022 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, pages 641–645.
-  Paissan, F., Ancilotto, A., and Farella, E. (2021). Phinets: A scalable backbone for low-power ai at the edge. *ACM Transactions on Embedded Computing Systems*, 21:1 – 18.
-  Paissan, F., Sahabdeen, A. M., Ancilotto, A., and Farella, E. (2023). Improving latency performance trade-off in keyword spotting applications at the edge. *2023 9th International Workshop on Advances in Sensors and Interfaces (IWASI)*, pages 299–304.

References V

-  Tan, M. and Le, Q. V. (2019).
Efficientnet: Rethinking model scaling for convolutional neural networks.
ArXiv, abs/1905.11946.
-  Wightman, R., Touvron, H., and J'egou, H. (2021).
Resnet strikes back: An improved training procedure in timm.
ArXiv, abs/2110.00476.