Real-time Neural Network Inference on Extremely Weak Devices: Agile Offloading with Explainable AI

MobiCom '22, October 17–21, 2022, Sydney, NSW, Australia

Visit https://snspace.top/2023/11/01/AgileNN/

Reporter : Sun Hao

2023.11.9

Salute to Authors

Kai Huang

The Department of Electrical and Computer Engineering, University of Pittsburgh

A PhD candidate, supervised by Dr. Wei Gao

Interests: Efficient AI, Large Language Models, AI for Systems, Internet of Things, Mobile and Edge Computing

[MobiSys'23] ElasticTrainer: Speeding Up On-Device Training with Runtime Elastic Tensor Selection
[MobiCom'22] Real-time Neural Network Inference on Extremely Weak Devices: Agile Offloading with Explainable AI
[SenSys'22] AiFi: AI-Enabled WiFi Interference Cancellation with Commodity PHY-Layer Information
[MobiSys'20] MagHacker: eavesdropping on stylus pen writing via magnetic sensing from commodity mobile devices

---

Wei Gao

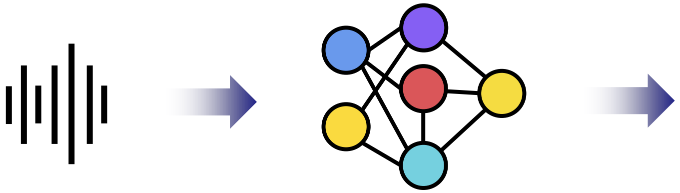
The Department of Electrical and Computer Engineering, University of Pittsburgh

An Associate Professor, direct the Pitt Intelligent System Laboratory (ISL)

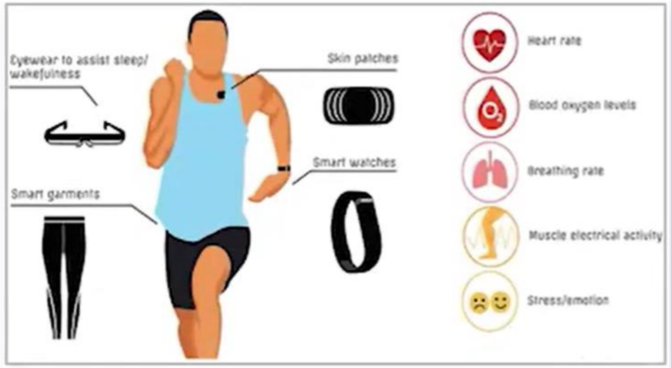
Interests: On-device AI, Mobile and embedded computing systems, Mobile and connected health, Cyber-physical systems and Internet of Things

Co-chair of the 2023 ACM Conference on Embedded Networked and Sensor Systems (SenSys)
Co-Chair of the 2023 IEEE/ACM Int'l Conference on Connected Health: Applications, Systems and Engineering Technologies (CHASE)
Co-chairing the 7th Mobile App Competition in conjunction with ACM MobiCom'22
Co-Chair of the 2022 EAI Int'l Conference on Mobile and Ubiquitous Systems: Computing, Networking and Services (MobiQuitous)

Bring Real-time AI to Weak Embedded Devices



Prediction & Decision making



Wearables for
Health Monitoring

Small Robots for
Autonomous Navigation

Sensors & Actuators
for Smart Home

Resource Limits on Weak Embedded Devices

Weak Embedded Devices

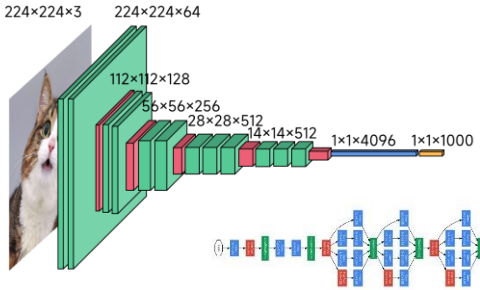
STM32

MSP430

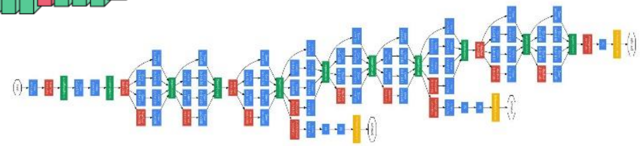
< 1 MB memory and storage

16~216 MHz CPU

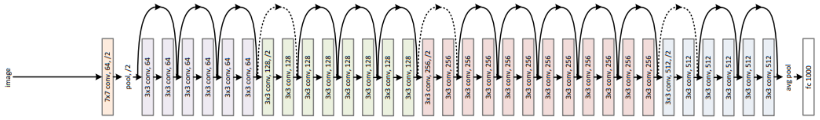
Large Neural Networks



VGG



GoogLeNet



ResNet

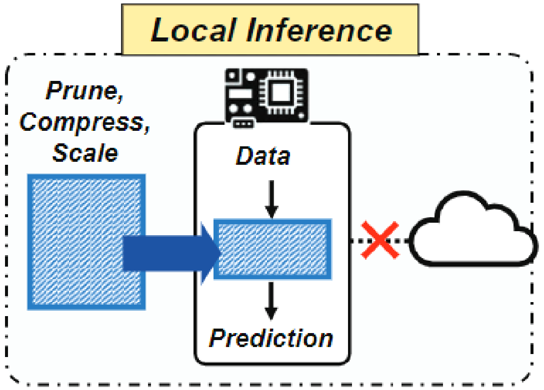
Require > 100 MB of memory space

> 2 GHz CPU for 60 ms latency

Existing Solutions

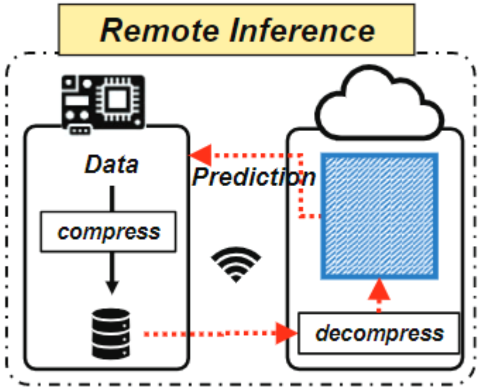
Local Inference

- Pruning, Compression, NAS
- Leads to **oversimplified NN structures**
- >10% accuracy loss



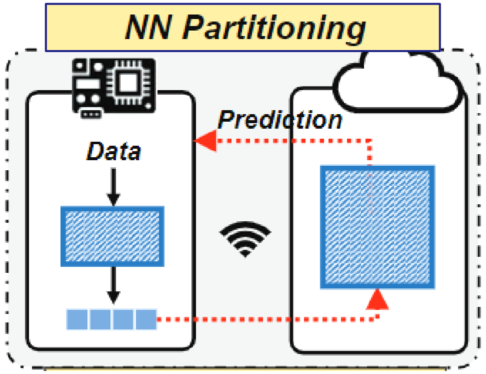
Remote Inference

- Compress raw data before transmission
- **Limited data compressibility** when the accuracy loss is minimum

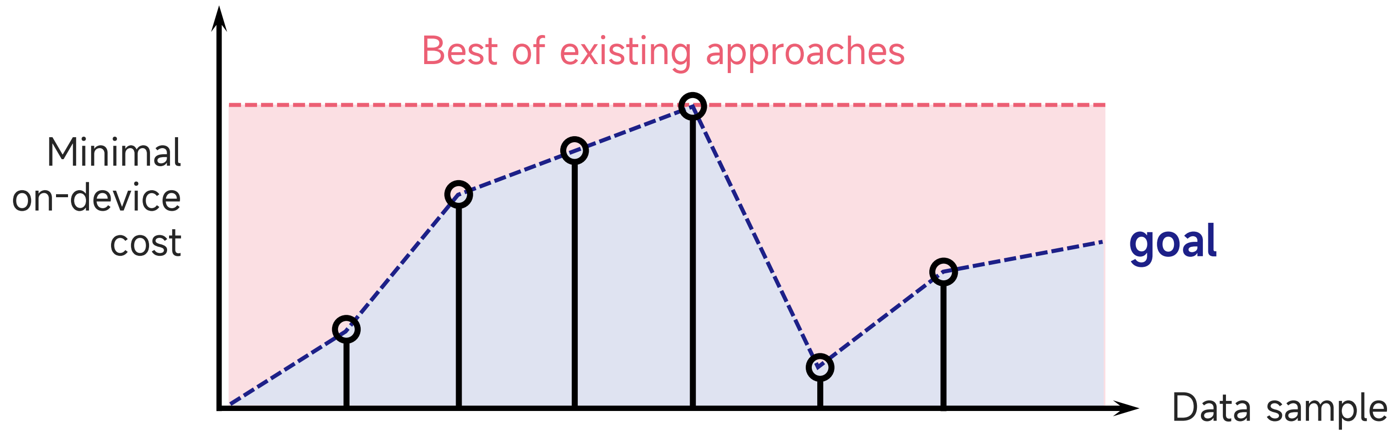


NN Partitioning

- Use a local NN to sparsify & compress data
- Higher compressibility but **expensive local NNs**



Existing NN Partitioning Solutions



Fixed learning schemes

↓

Data-centric
Consider data samples' heterogeneity
— feature importance

Cumbered by the worst case

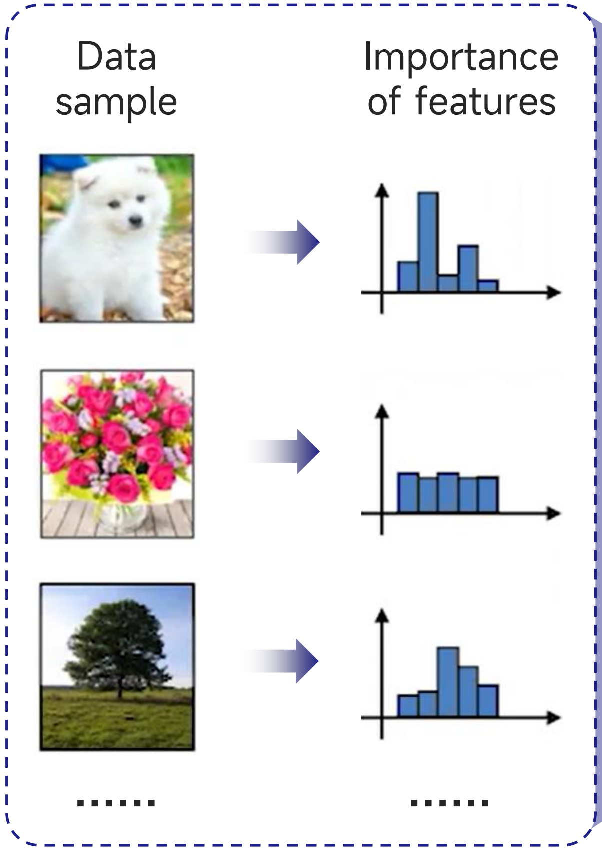
↓

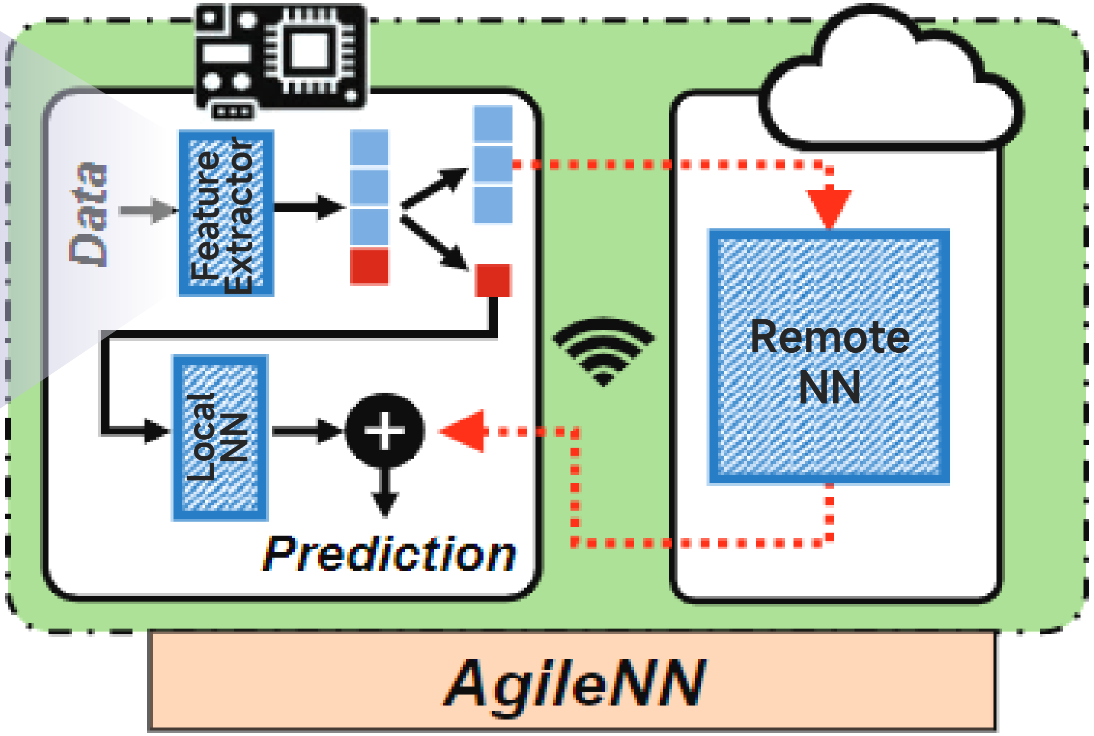
Agile offloading
Adaptive partitioning to minimize the offloading cost

AgileNN: From Fixed to Data-centric & Agile

Important features: clearer to perceive → Handled locally

Less important features: more compressible → Offload

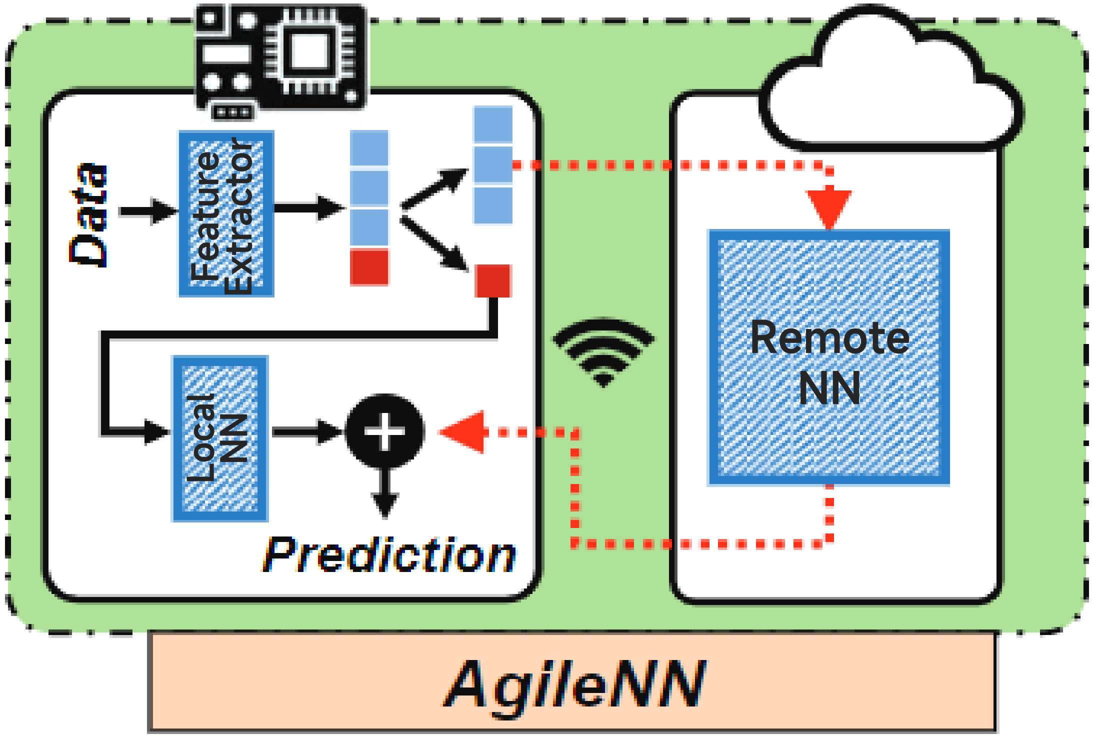




AgileNN: From Fixed to Data-centric & Agile

Important features: clearer to perceive → Handled locally

Less important features: more compressible → Offload

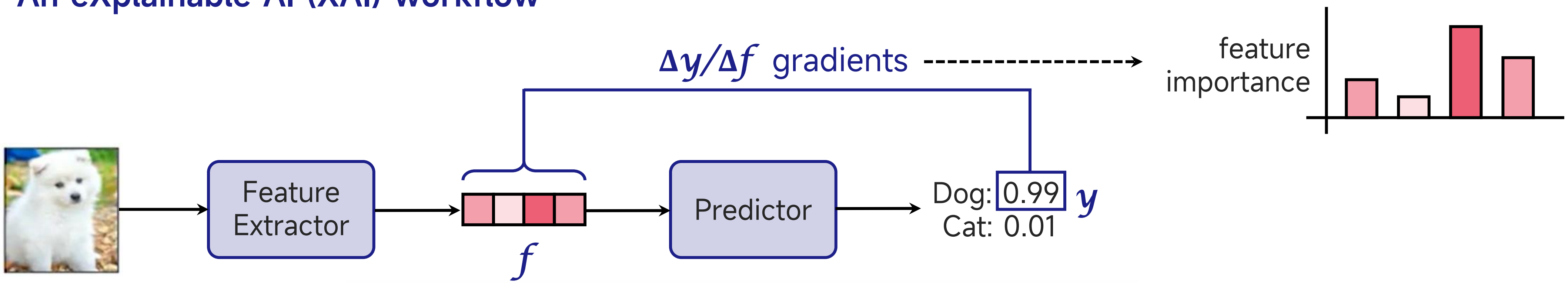


Challenge 1: How to correctly evaluate feature importance?

Challenge 2: How to maximize the compressibility of less important features?

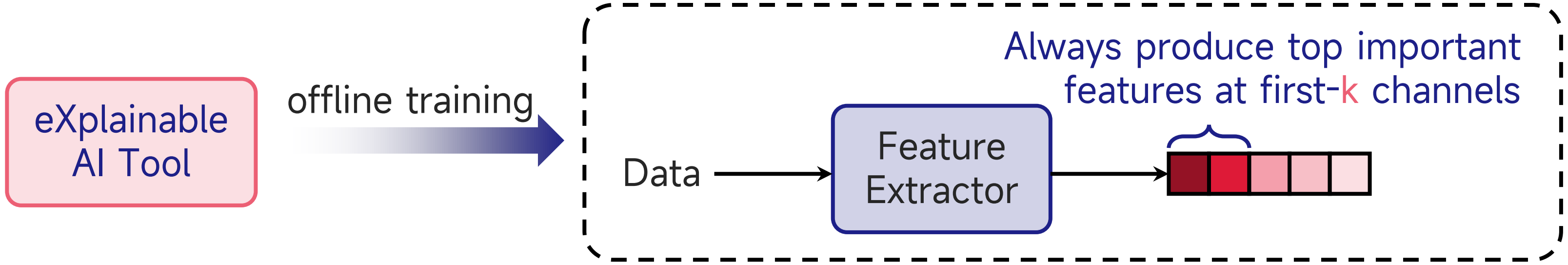
Solving Challenge 1: Evaluating Feature Important via eXplainable AI

- An eXplainable AI (XAI) workflow



But XAI tools are computationally expensive

- Training an XAI-enabled feature extractor offline

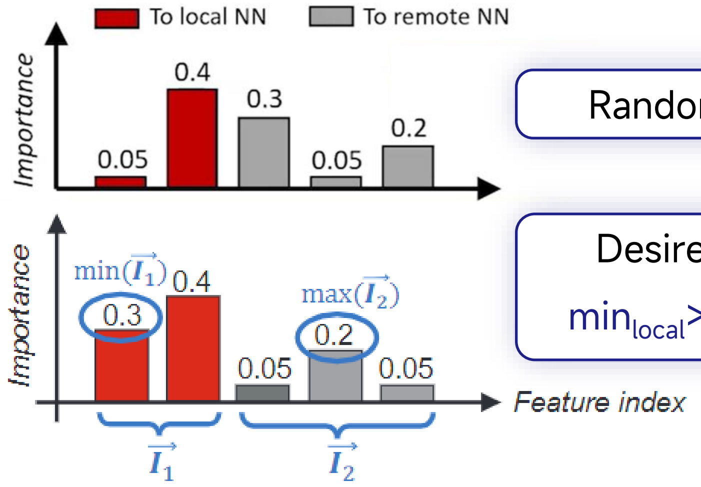


Solving Challenge 2: Enforcing Skewed Distribution of Feature Importance via XAI Loss Function

Disorder Loss

Ensure topmost important features are extracted into the first-k channels

→ Avoid online importance evaluation



Random order

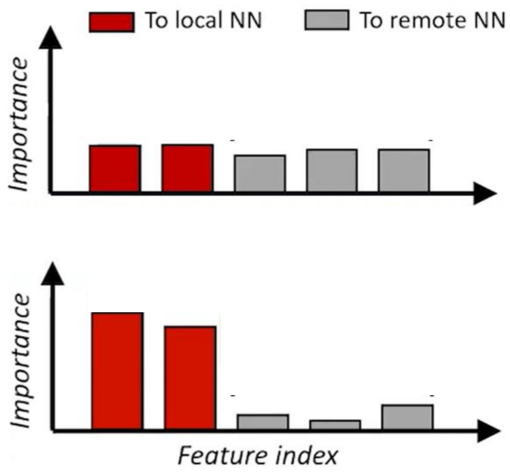
Desired order

$\min_{local} > \max_{remote}$

$$L_{disorder} = max\left(0, max(\vec{I_2}) - min(\vec{I_1})\right)$$

Skewness Loss

Enhance the importance of top-k features to ensure compressibility of the others



Not enough compressibility

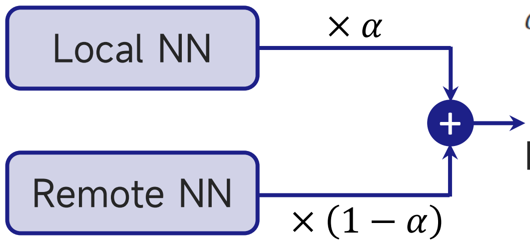
Higher compressibility

$$L_{skewness} = max\left(0, \rho - |\vec{I_1}|\right)$$

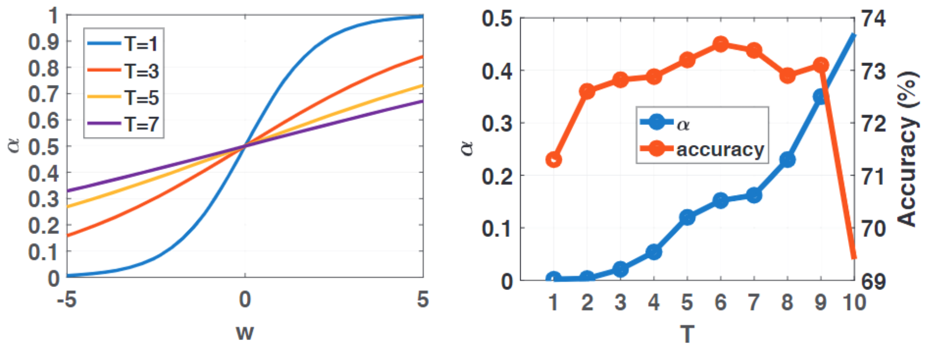
Combining Predictions & Preprocessing

Combining local & remote predictions

Ensure predictions to be in the same scale

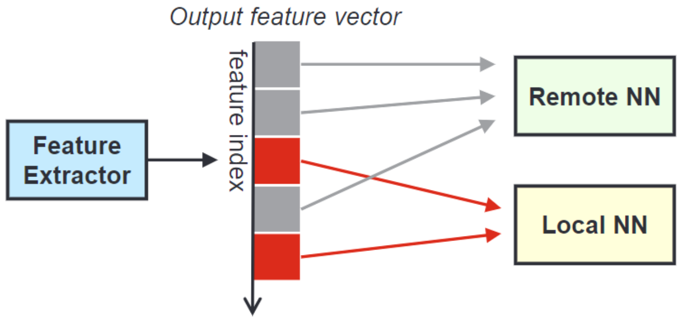


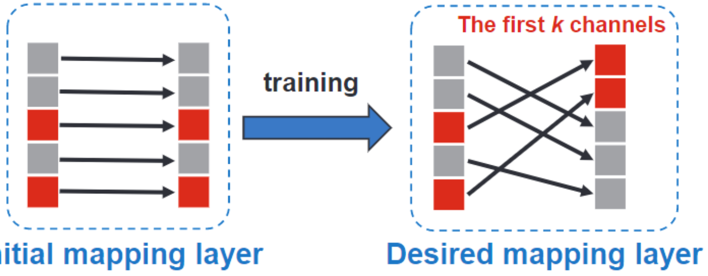
$$\alpha(w;T) = \frac{1}{1+e^{-w/T}}$$



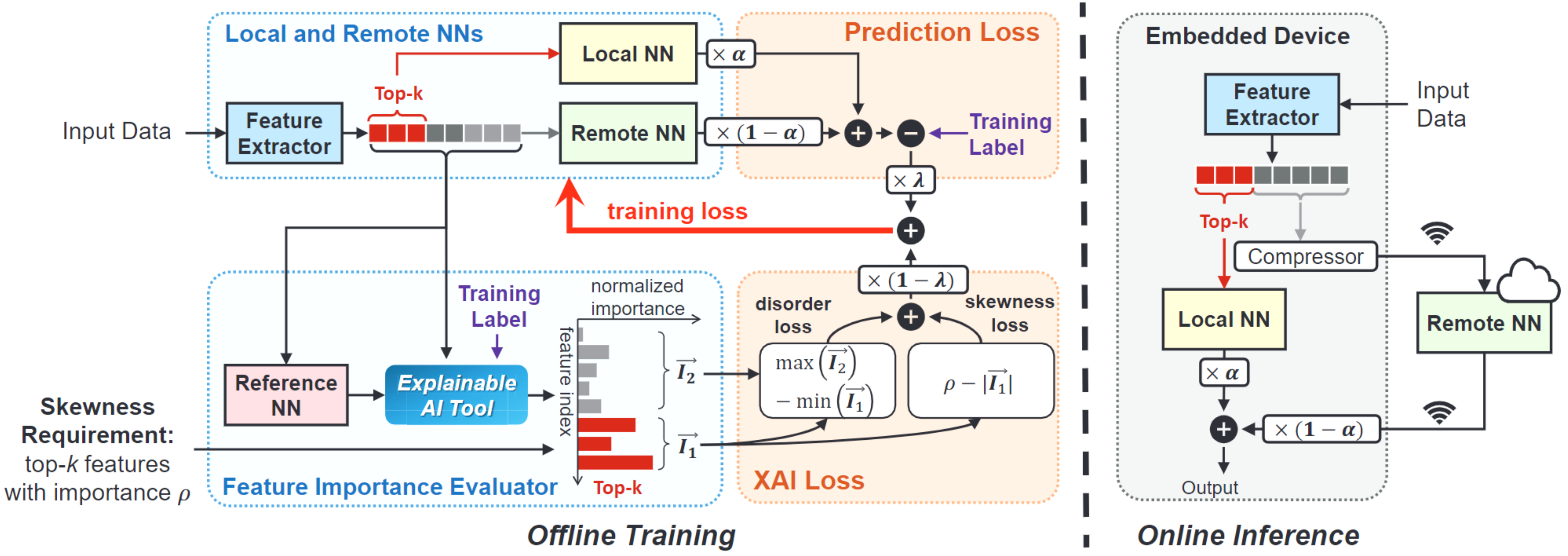
Pre-processing the feature extractor

Select k initial channels where the top-k features with high importance are most likely to be located





AgileNN's Offline Training & Online Inference Framework



Implementation & Evaluation Setup

- **Local device**

STM32F746NG MCU board, 216MHz, 320kB SRAM, 1MB FRAM

ESP-WROOM-02D WiFi module @6Mbps

- **Remote device**

Dell Precision 7820 workstation

- A 3.6GHz 8-core Intel Xeon CPU and a 48GB Nvidia RTX A6000 GPU

- **Baselines**

MCUNet [1] — NAS to find the best local NN

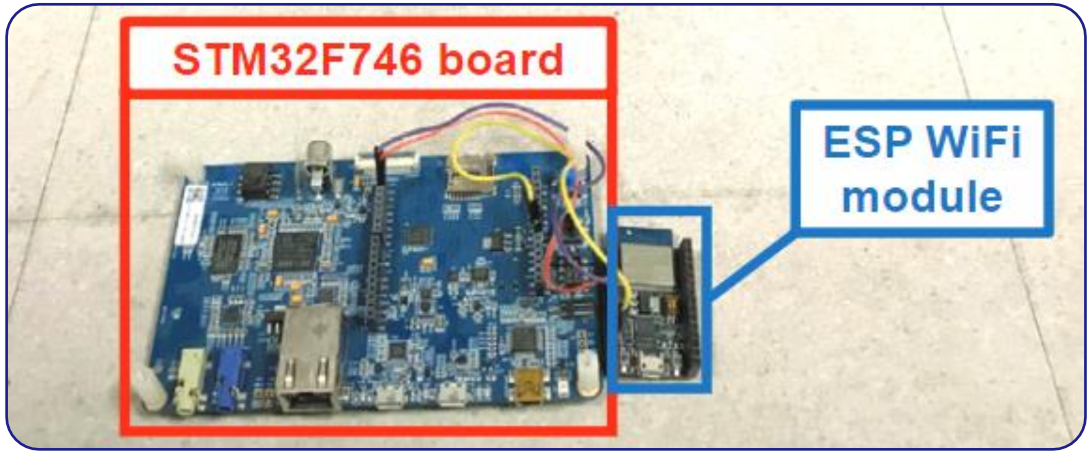
Edge-only — compress and offload raw data

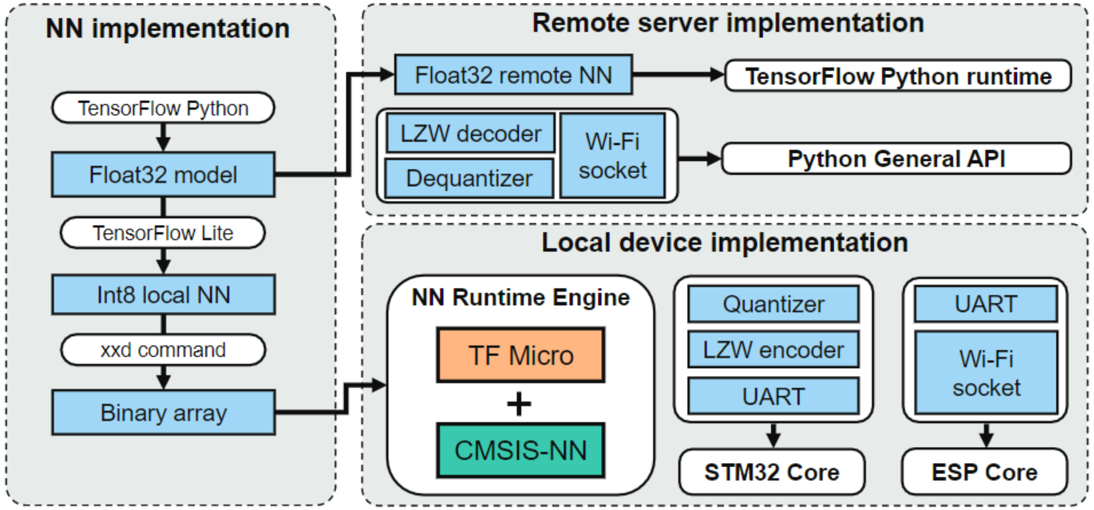
DeepCOD [2] — use a NN-based encoder

SPINN [3] — early-exit inference

- **Datasets**

CIFAR-10/100, SVHN, Tiny ImageNet (200 classes)

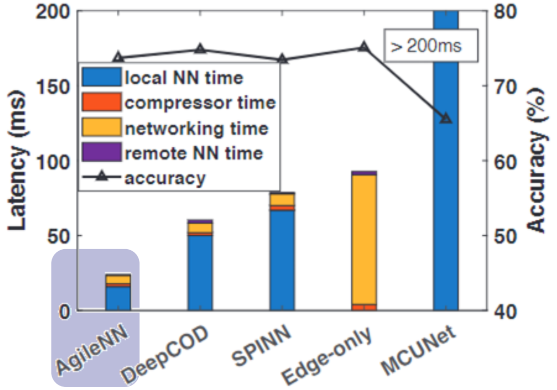




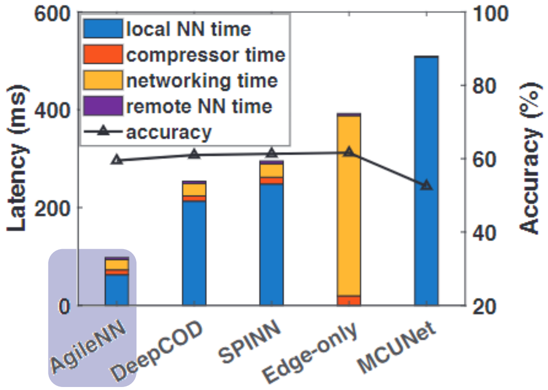
[1] MCUNet: Tiny deep learning on IoT devices, NIPS 2020.
[2] Deep compressive offloading: Speeding up neural network inference by trading edge computing for network latency, Sensys 2020.
[3] SPINN: synergistic progressive inference of neural network over device and cloud, Mobicom 2020.

Overall Performance

AgileNN reduces end-to-end latency by 2×-2.5×

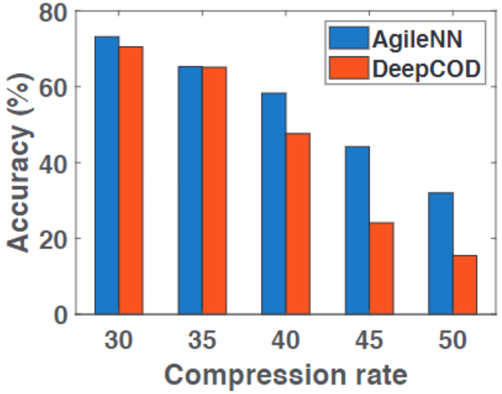


(b) CIFAR-100



(d) ImageNet-200

Accuracy & Compression rate



(a) CIFAR-100

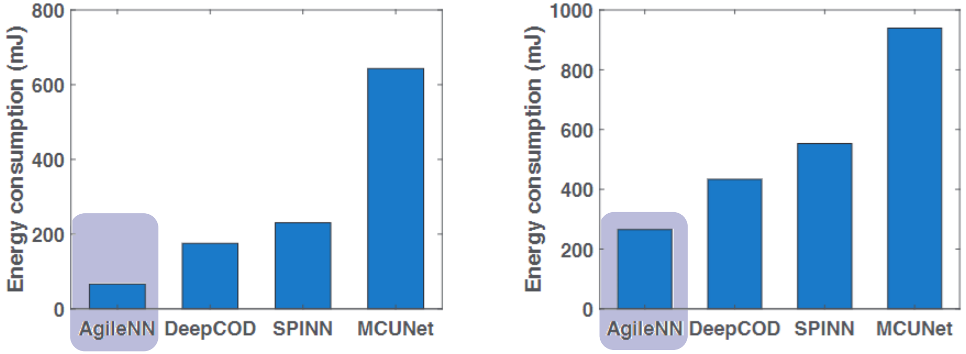
Reduction of transmitted data size compared to DeepCOD

Dataset	CIFAR-10	CIFAR-100	SVHN	ImageNet
Reduction	43.7%	15.8%	72.3%	20.8%

Local Resource Saving

Local Energy Consumption

- 1.6×-2.5× more efficient than DeepCOD
- 8× more efficient than MCUNet

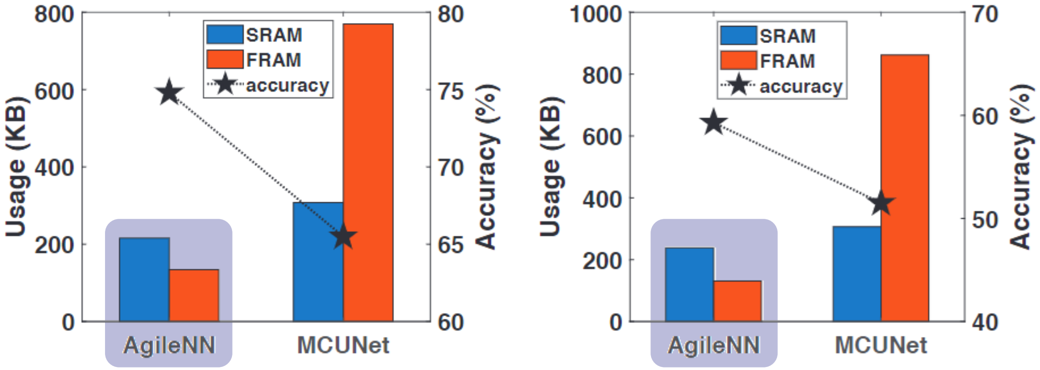


(a) CIFAR-100 (b) ImageNet-200

Local Memory & Storage

Memory — SRAM, storage — FRAM

- Local NN saves 40%-50% memory and >50% storage
- 10% higher accuracy

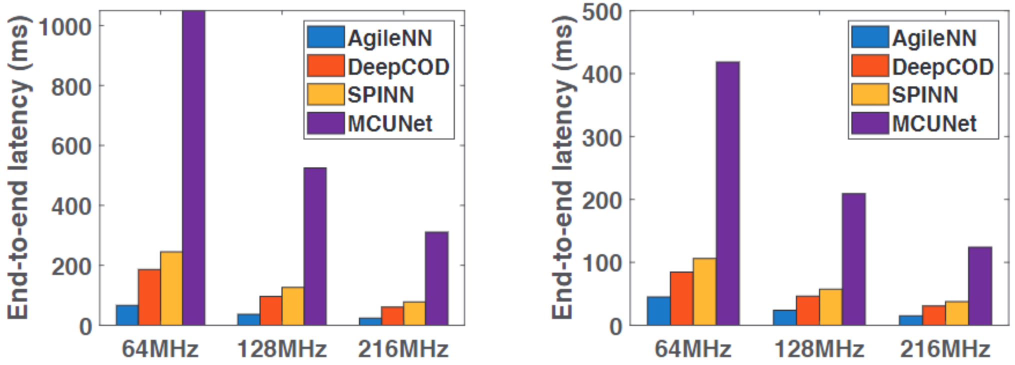


(a) CIFAR-100 (b) ImageNet-200

Different System Settings

Impact of Local CPU Frequency

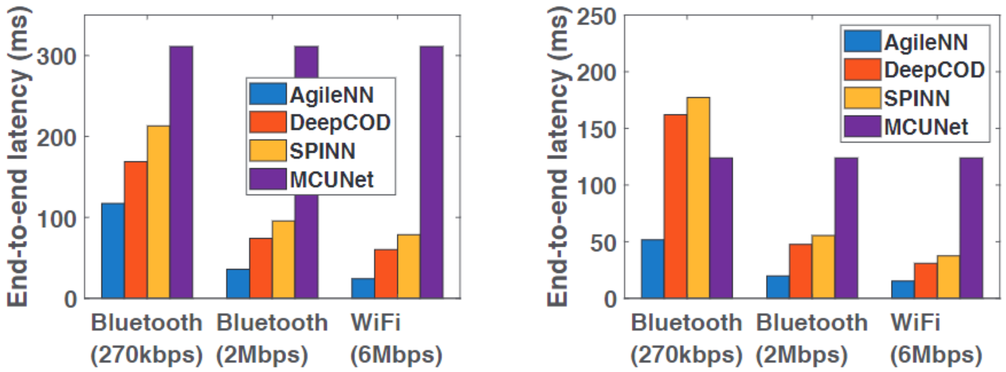
- 64MHz – 216MHz
- Reduce latency by 2.1×-2.5×



(a) CIFAR-100 (b) SVHN

Impact of Network Bandwidth

- Bluetooth (270kbps, 2Mbps), WiFi (6Mbps)
- Keeps outperforming baselines



(a) CIFAR-100 (b) SVHN

Summary

Agile Offloading for Neural Network Inference

- AgileNN: shifts the rationale of offloading from fixed to data-centric & agile
- Leveraging XAI to achieve such agility
- >6× lower latency and >8× resource consumption for extremely weak devices

Explainable AI for Systems

- Integrate XAI techniques into NN offloading systems
- Migrating XAI computation from device to offline training

Summary

Topic

Deploying NNs on Extremely Weak Devices

Contradiction

Latency ⟷ Accuracy

weak capabilities

Key Idea

Important features: Handled locally
Less important features: Offload

Techniques

- Skewness Manipulation
- Pre-processing the Feature Extractor
- Combining Local and Remote Predictions

Thanks for your attention
Q & A

Real-time Neural Network Inference on Extremely Weak Devices: Agile Offloading with Explainable AI

Reporter : Sun Hao

2023.11.9