MobiCom '22

InFi : End-to-End Learnable Input Filter for Resource-Efficient Mobile-Centric Inference

Mu Yuan[1], Lan Zhang[1], Fengxiang He[2], Xueting Tong[1], Xiang-Yang Li[1]

[1]**University of Science and Technology of China**

[2]JD Explore Academy

中国科学技术大学 University of Science and Technology of China

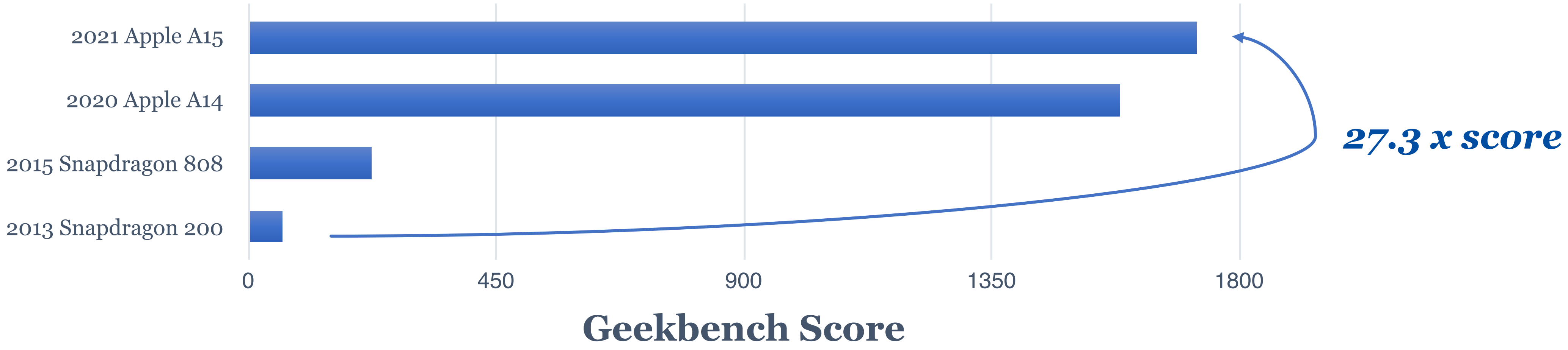
京东探索研究院 JD EXPLORE ACADEMY

Outline

1. **Introduction**
2. Framework and Analysis
3. Design and Implementation
4. Evaluation

1.1 Mobile-Centric Inference

- Mobile devices' increasing computing power



https://browser.geekbench.com/mobile-benchmarks

1.1 Mobile-Centric Inference

- Mobile devices' increasing computing power
- Growing demand for real-time sensor data analytics
  - Over **80%** of enterprise IoT projects will incorporate **AI** by 2022

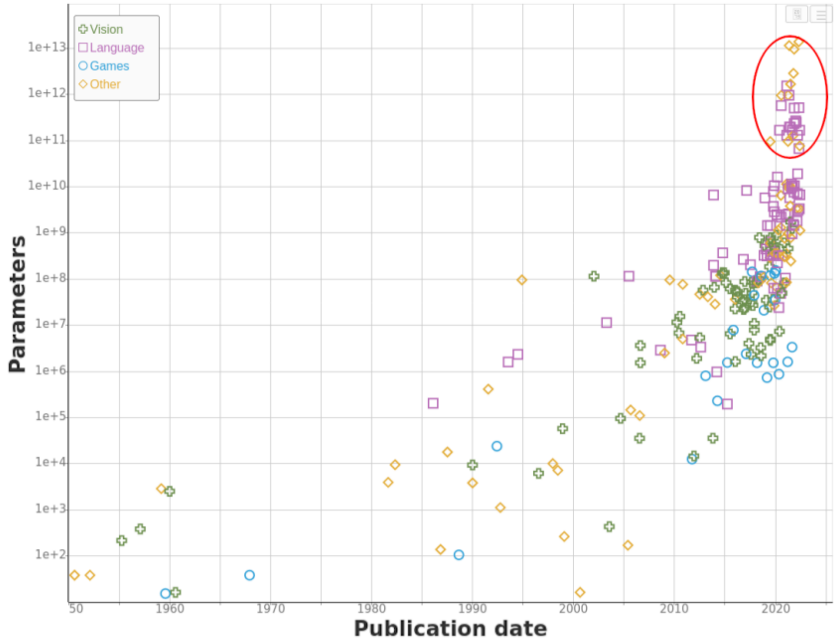
    https://www.visualcapitalist.com/aiot-when-ai-meets-iot-technology/

1.1 Mobile-Centric Inference

- Mobile devices' increasing computing power
- Growing demand for real-time sensor data analytics
  - Over **80%** of enterprise IoT projects will incorporate **AI** by 2022

    https://www.visualcapitalist.com/aiot-when-ai-meets-iot-technology/

- AI models with the state-of-the-art accuracy are too **computationally intensive**



Villalobos, Pablo, et al. "Machine Learning Model Sizes and the Parameter Gap." arXiv preprint arXiv:2207.02852 (2022).

1.1 Mobile-Centric Inference

- Mobile devices' increasing computing power
- Growing demand for real-time sensor data analytics
  - Over **80%** of enterprise IoT projects will incorporate **AI** by 2022

    https://www.visualcapitalist.com/aiot-when-ai-meets-iot-technology/

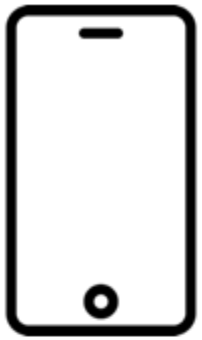
- AI models with the state-of-the-art accuracy are too **computationally intensive**
- **Resource-efficiency** is important for mobile-centric *model inference* workloads

Model Inference: the process of running pre-trained AI models on new inputs

1.2 Input Redundancy

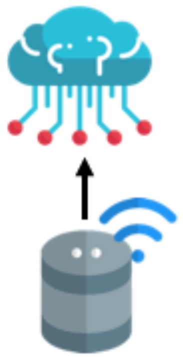
- Widespread redundancy in inputs
  - Type#1: **SKIP** the inputs that do not return valuable results

Photos



Face Detection on Mobile Phones

Audios



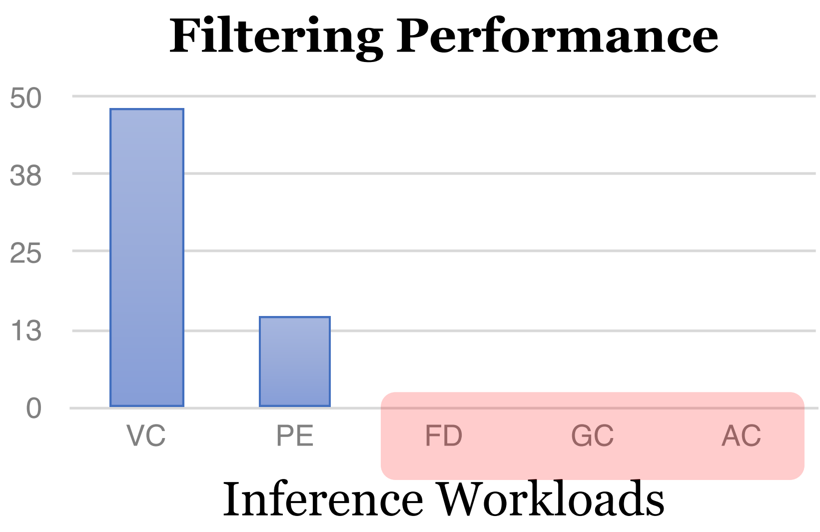
Speech Recognition offloaded on Cloud

1.2 Input Redundancy

- Widespread redundancy in inputs
  - Type#1: **SKIP** the inputs that do not return valuable results
  - Type#2: **REUSE** the results that previously computed

1.3 Key Goals

- Robust feature discriminability



SOTA method does not work

1.3 Key Goals

- Robust feature discriminability
- Theoretical filterability for application guidance
  - Tailored solutions bring the **cumbersome trial-and-error process**, due to the lack of theoretical analysis

Outline

1. Introduction
2. **Framework and Analysis**
3. Design and Implementation
4. Evaluation

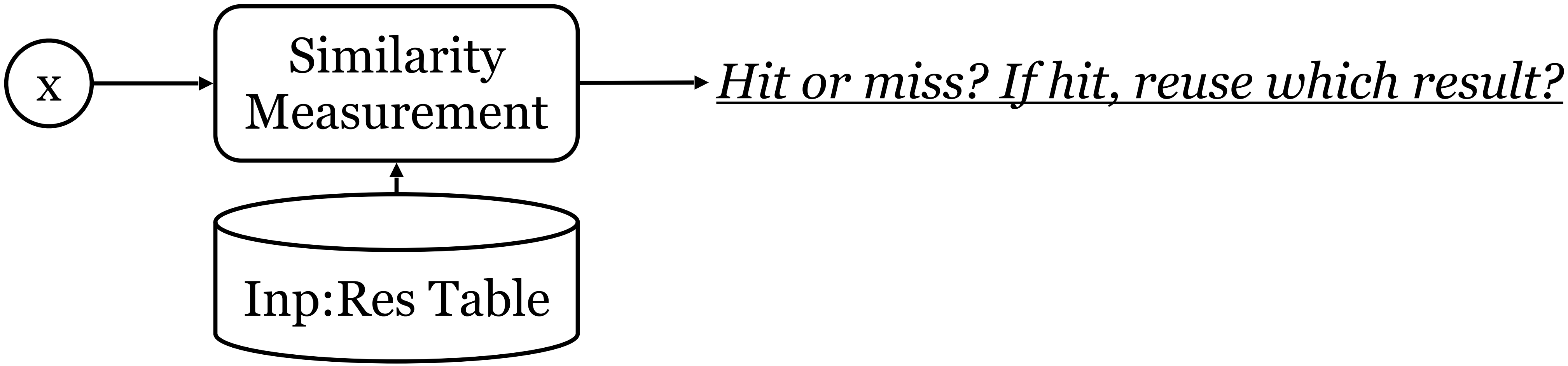
2.1 SKIP as REUSE

- SKIP workflow

2.1 SKIP as REUSE

- SKIP workflow

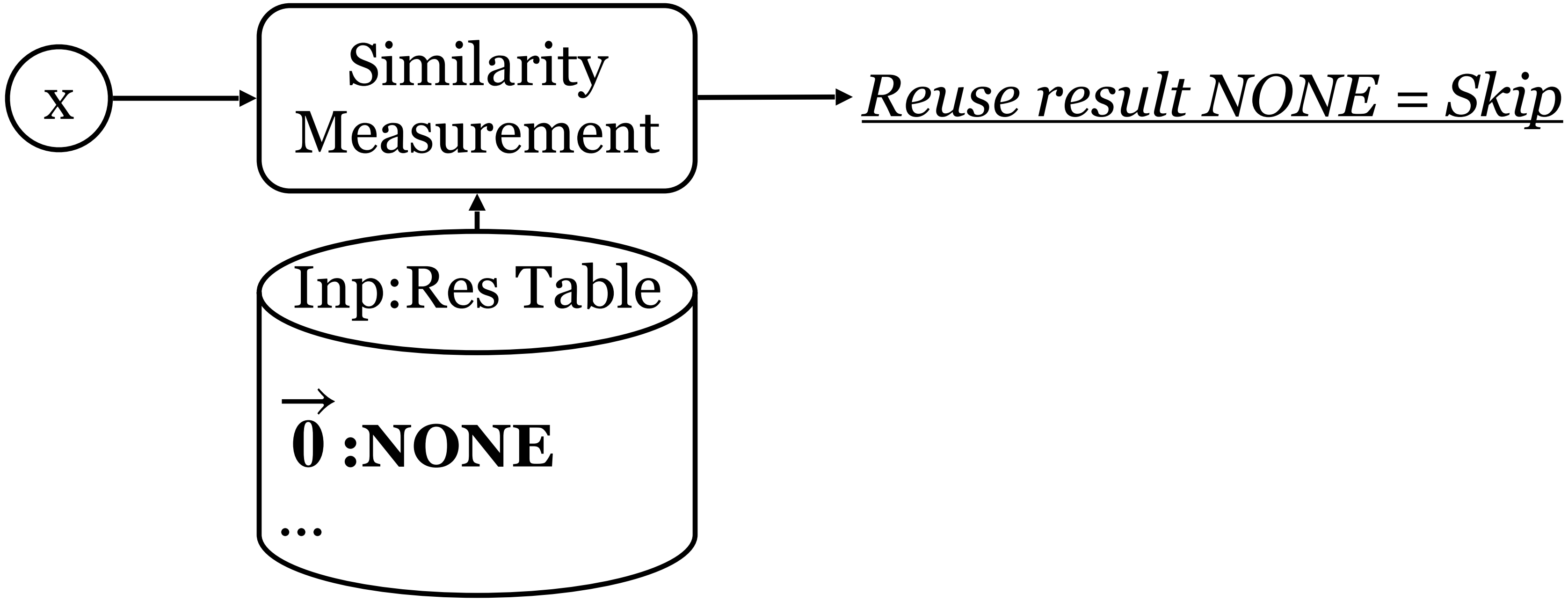
- REUSE workflow



2.1 SKIP as REUSE

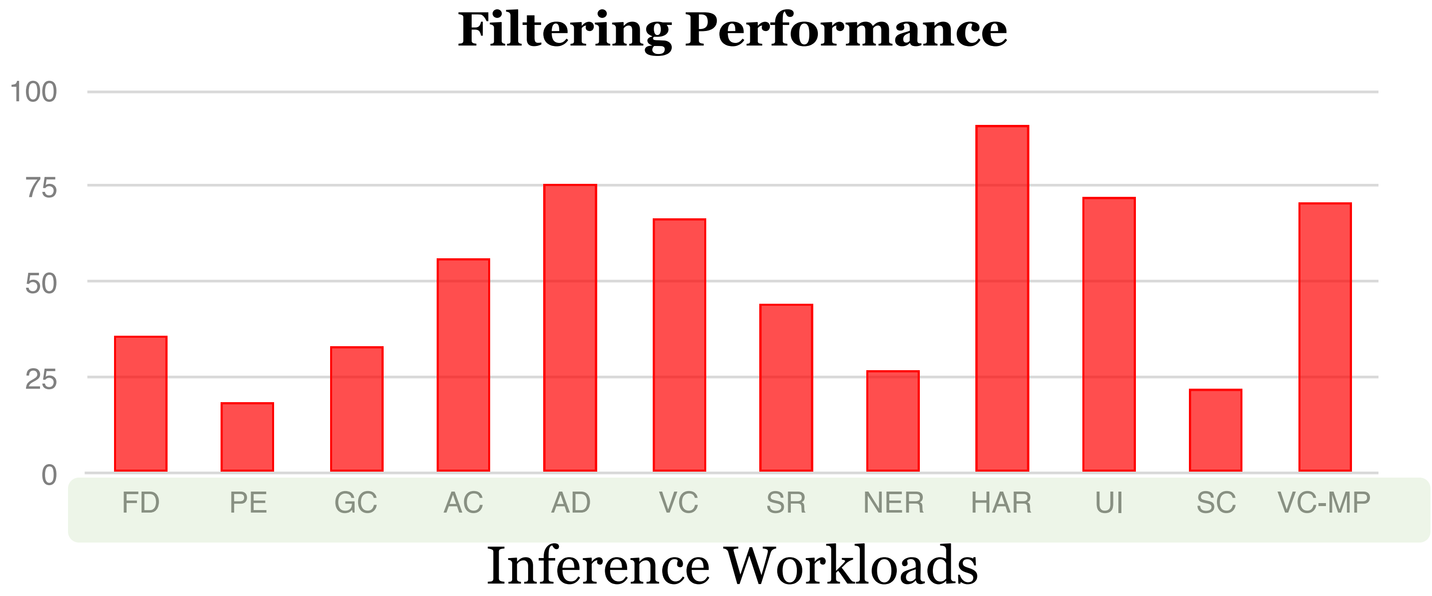
- Unify SKIP and REUSE approaches:

  SKIP equals to REUSE the NONE output of input $\vec{\mathbf{0}}$



2.2 End-to-End Learnability

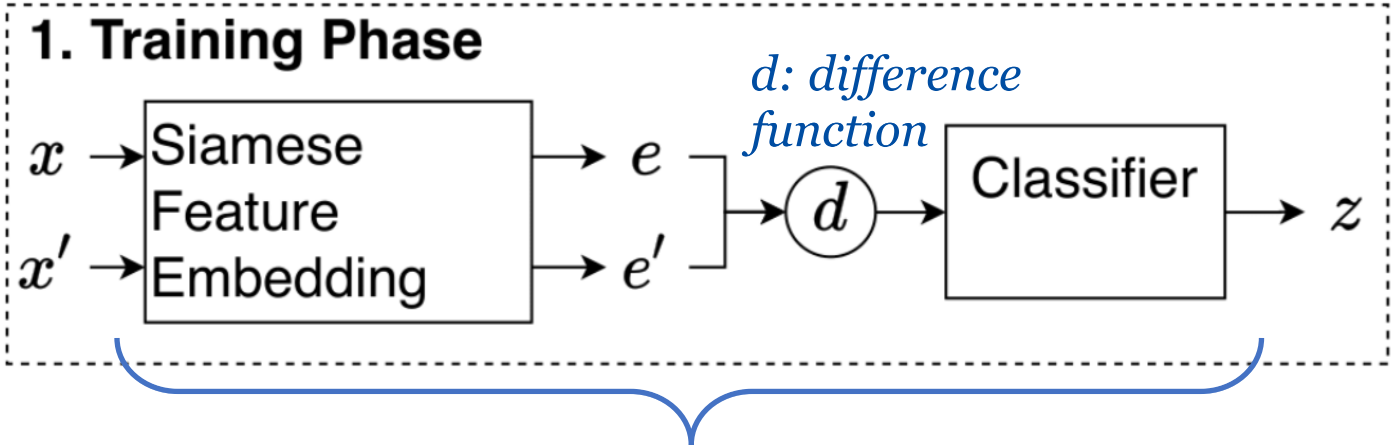
- End-to-end learning casts complex processing components into **coherent connections** in neural networks and optimizes itself by applying back-propagation all through the networks.
  - **Robust feature discriminability!**



Our method always works.

2.2 End-to-End Learnability

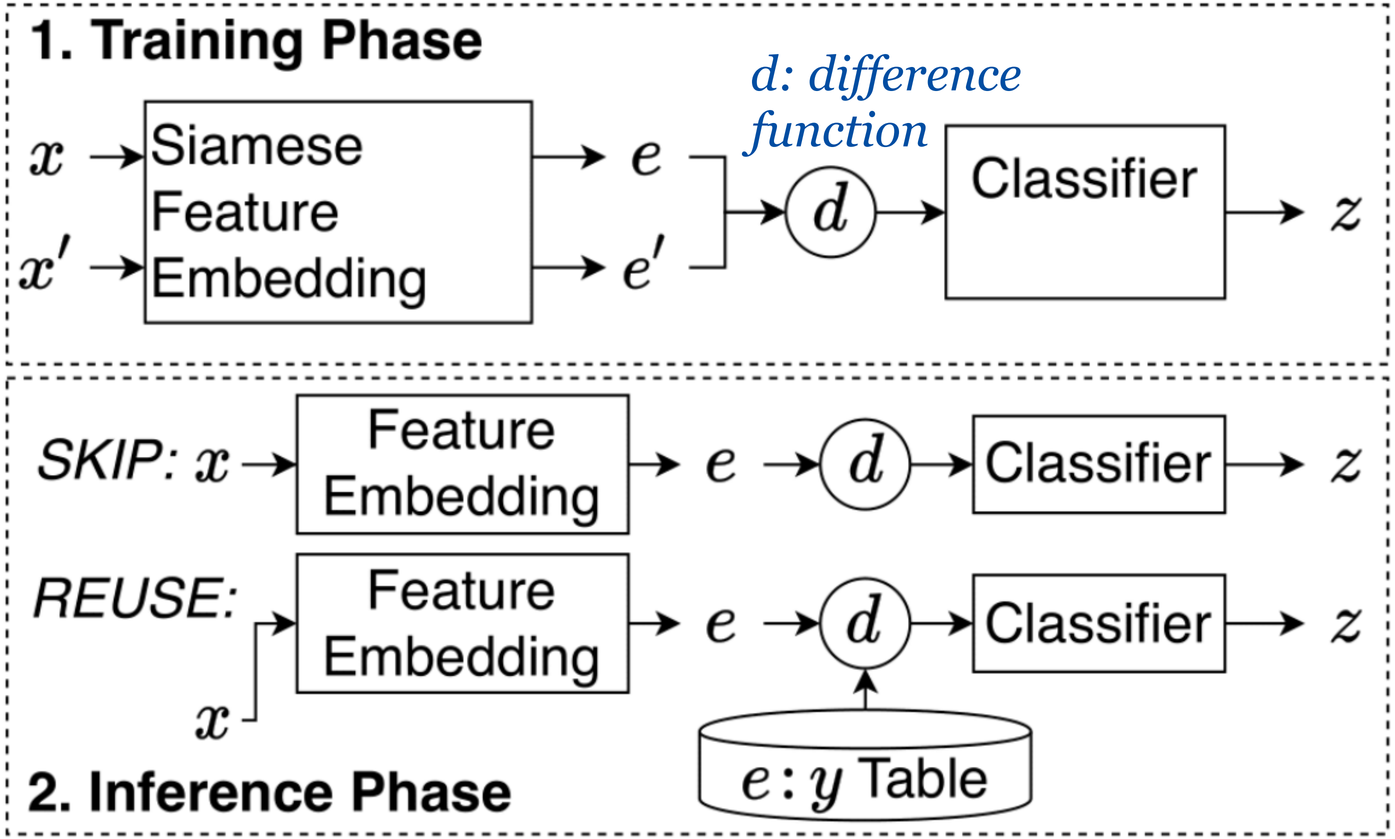
- Embedding-Difference-Classification Framework
  - Training Phase:



End-to-end learnable using metric learning paradigm

2.2 End-to-End Learnability

- Embedding-Difference-Classification Framework
  - Training Phase:
  - Inference Phase:



1.3 Key Goals

- **Robust feature discriminability ✅**
- Theoretical filterability for application guidance

2.3 Theoretical Analysis

- Learning problem formulation
- Filterability definition, based on hypothesis complexity comparison

> *Definition 2 (Filterability).* Let $Complex(\cdot)$ denote the complexity measurement of a hypothesis family. We say that the inference workload is filterable, if $Complex(\mathcal{G}) \leq Complex(\mathcal{H})$, where $h \in \mathcal{H}$ and $(f_h \circ h) \in \mathcal{G}$.

Intuition: If it is ***easier to learn the filter*** *than to learn the inference model, the workload is* ***FILTERABLE****.*

2.3 Theoretical Analysis

- Case1: low-confidence classification as redundancy

LEMMA 4. *Let $\mathcal{H}$ be a family of binary classifiers taking values in $\{-1, +1\}$. For $\mathcal{G} = \{sign(h(h+b))\}$ where $h \in \mathcal{H}, b \in \mathbb{R}$:*

Non-Filterable

See our paper for detailed **formulation, proof and analysis.**

$\mathcal{H} = \{\max(h_1, ..., h_l) : h_i \in \mathcal{H}_i, i = 1, ..., l\}$. *For* $\mathcal{G} = \{\max(h_i) : i \in J\}$, *where* $J \subseteq \{1, ..., l\}$:

$$\widehat{\mathfrak{R}}_S(\mathcal{G}) \leq \widehat{\mathfrak{R}}_S(\mathcal{H}). \quad (3)$$

Filterable

- Case3: regression bound as redundancy

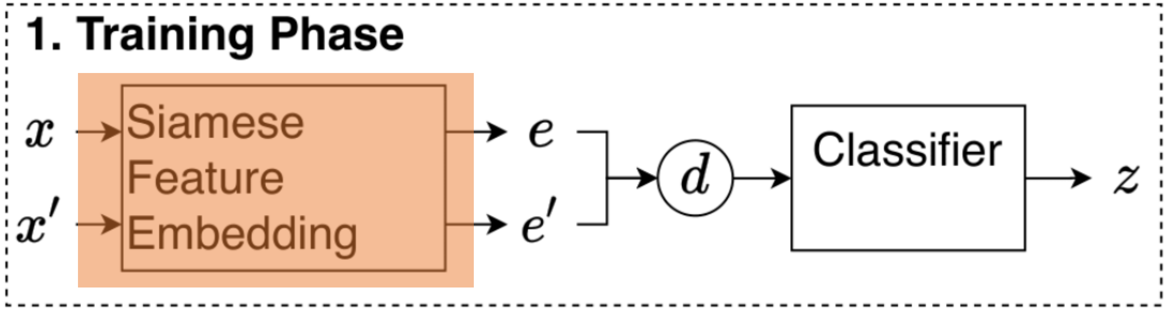
THEOREM 6. *Let $p \geq 1$ and $\mathcal{H} = \{x \mapsto |h(x) - c(x)|^p : h \in H\}$. Assume that $|h(x) - c(x)| \leq M$ for all $x \in X$ and $h \in H$. Then the following inequality holds:* $\widehat{\mathfrak{R}}_S(\mathcal{H}) \leq pM^{p-1}\widehat{\mathfrak{R}}_S(H)$.

Filterable

Outline

1. Introduction
2. Framework and Analysis
3. **Design and Implementation**
4. Evaluation

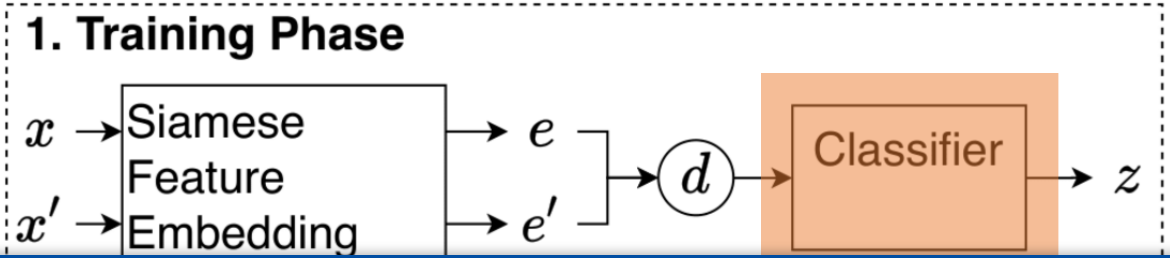
3.1 Architecture



Modality feature networks

- 5 modalities
    - *Text*
    - *Image*
    - *Video*
    - *Audio*
    - *Sensor signal and feature map (vector)*

3.1 Architecture



Modality feature networks

- 5 modalities

InFi: **INput FIltering**

  - *Audio*
  - *Sensor signal and feature map (vector)*

Task-agnostic classifier

- *Fully-connected neural networks*

3.2 Inference with InFi

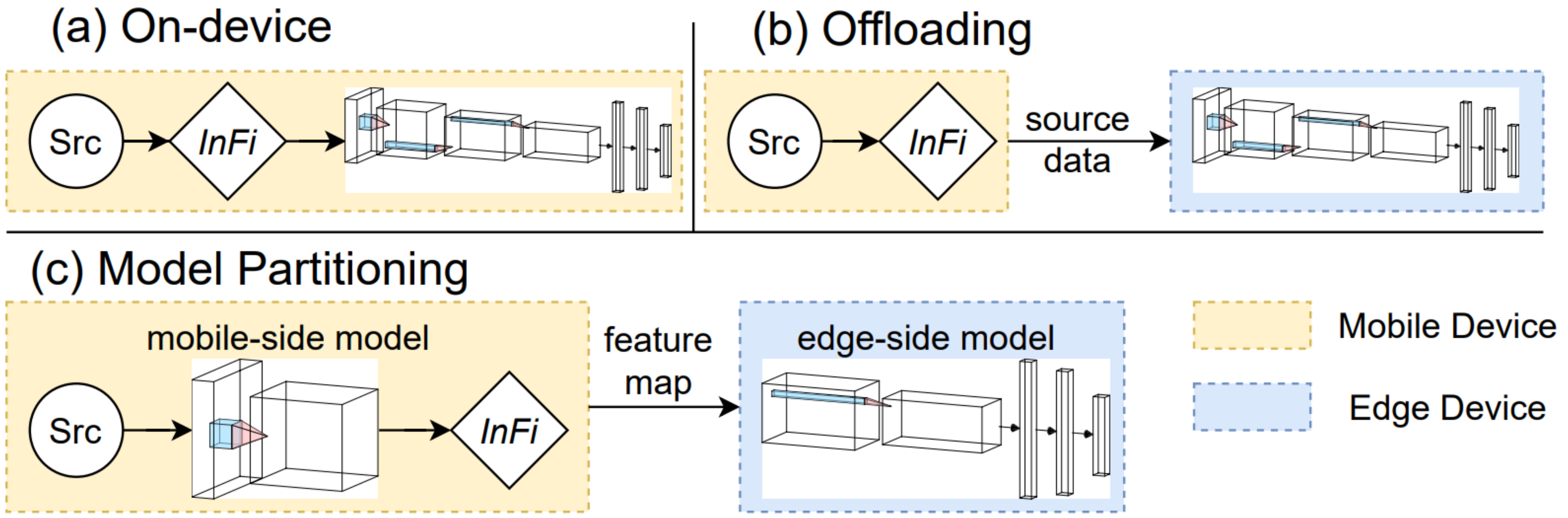
- **SKIP**
  - Confidence threshold
- **REUSE**
  - Cache entry: input embedding – inference result
  - Homogeneity score-based cache miss detection

    Guo, Peizhen, et al. "Foggycache: Cross-device approximate computation reuse." Proceedings of the 24th annual international conference on mobile computing and networking. 2018.

  - K-Nearest Neighbors algorithm for retrieval

3.3 Deployments

- Case#1: on-device
- Case#2: offloading
- Case#3: model partitioning



3.4 Implementation

- Feature networks and classifiers are built with TensorFlow 2.4
- TFLite are used to transform saved checkpoints into Java servable object for Android deployment

- Opensource: https://github.com/yuanmu97/infi

InFi

Outline

1. Introduction
2. Framework and Analysis
3. Design and Implementation
4. **Evaluation**

4.1 Setup

- 12 Workloads
  - 5 datasets
  - 6 modalities

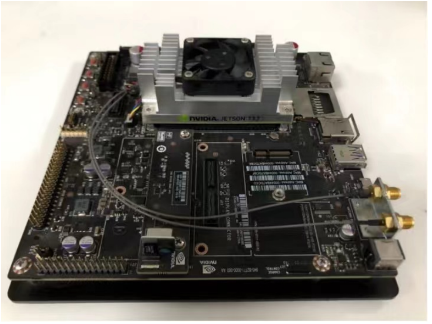
public

48x10 hours of videos (1FPS) collected from 10 cameras at road intersections

Datasets	Modality	Inference Task
Hollywood2	Video Clip	Action Classification (AC)
	Image	Face Detection (FD)
		Pose Estimation (PE)
		Gender Classification (GC)
	Audio	Speech Recognition (SR)
	Text	Named Entity Recognition (NER)
		Sentiment Classification (SC)
ESC-10	Audio	Anomaly Detection (AD)
UCI HAR	Motion Signal	Activity Recognition (HAR)
MoCap	Motion Signal	User Identification (UI)
City Traffic	Video Stream	Vehicle Counting (VC)
	Feature Map	Vehicle Counting (VC-MP)

4.1 Setup

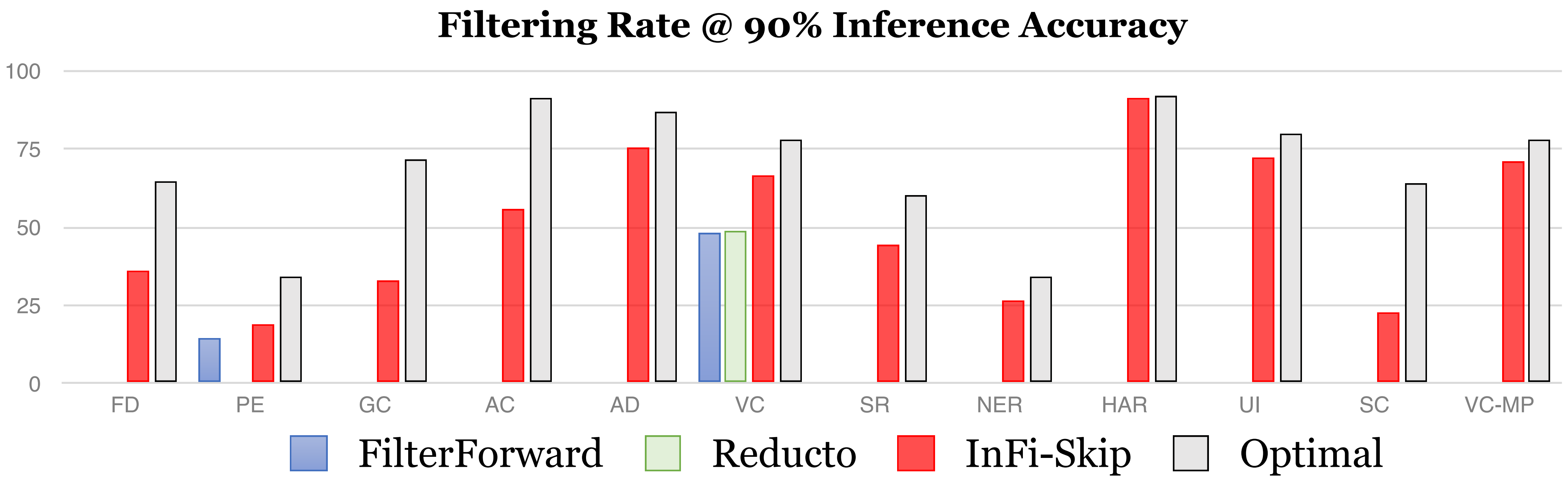
- 4 devices
  - GPU server (one NVIDIA 2080Ti)
  - Development board (NVIDIA JETSON TX2)
  - Mobile phone (XIAOMI Mi 5)
  - Smartwatch (HUAWEI WATCH)



- 3 baselines
  - FilterForward *MLSys '19*
  - Reducto *SIGCOMM '20*
  - FoggyCache *MobiCom '18*

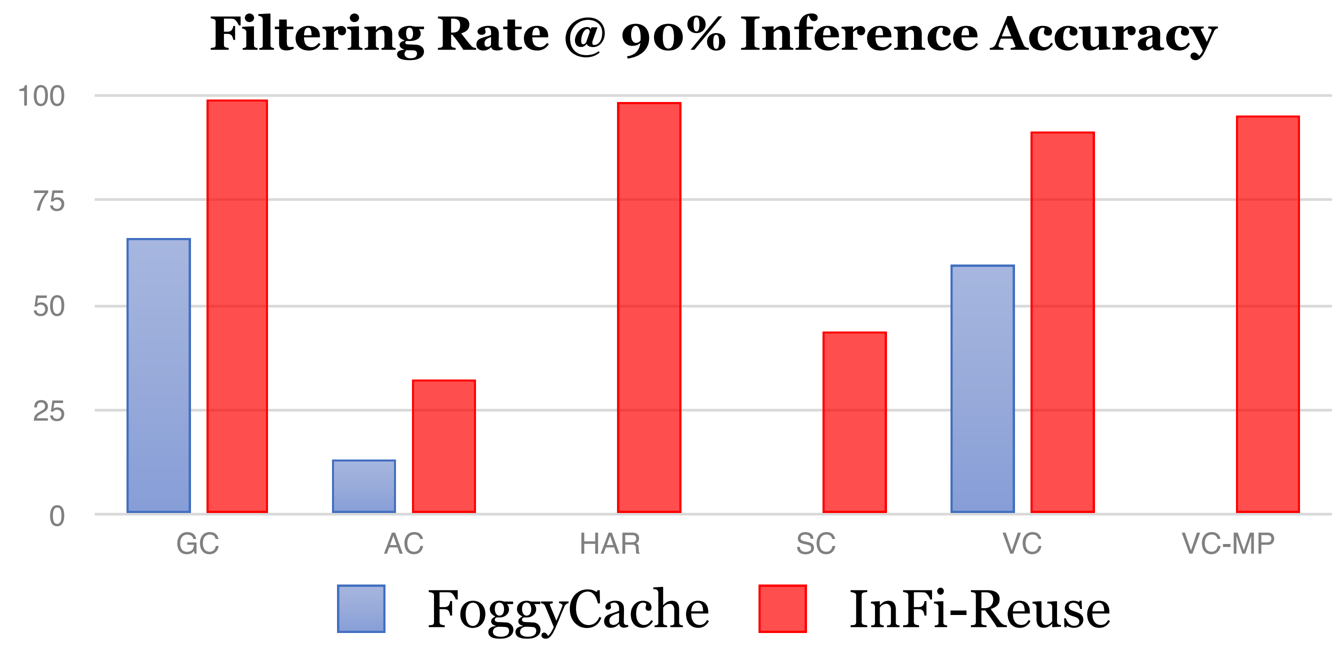
4.2 Filtering Performance

- SKIP



4.2 Filtering Performance

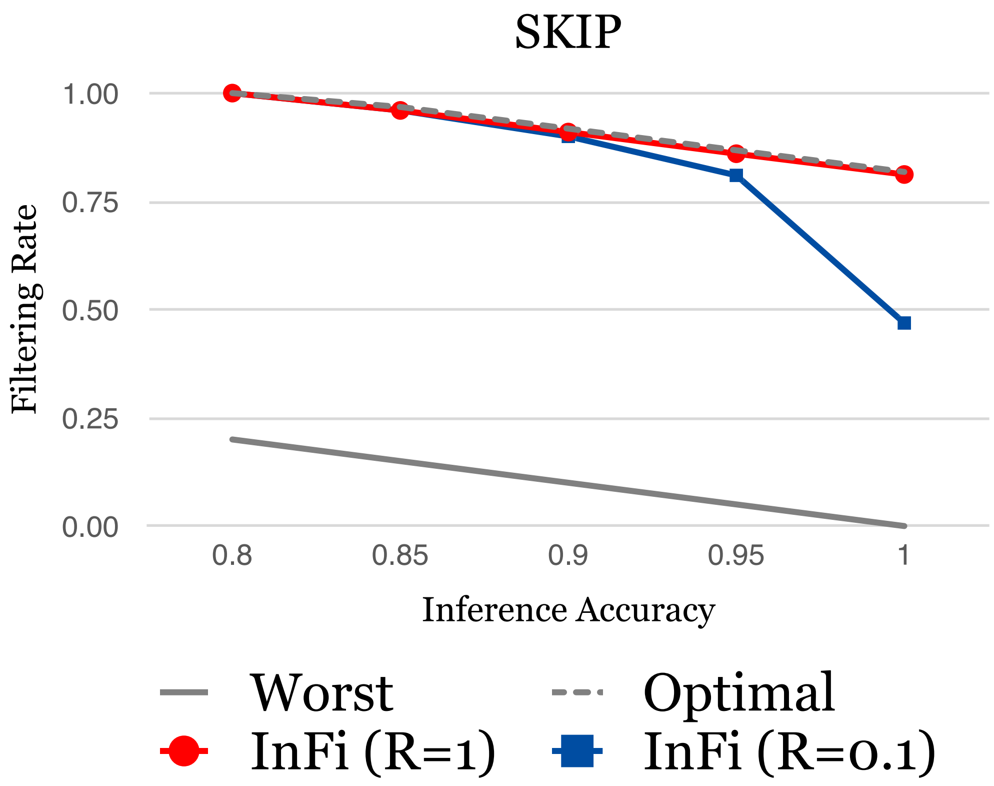
- REUSE

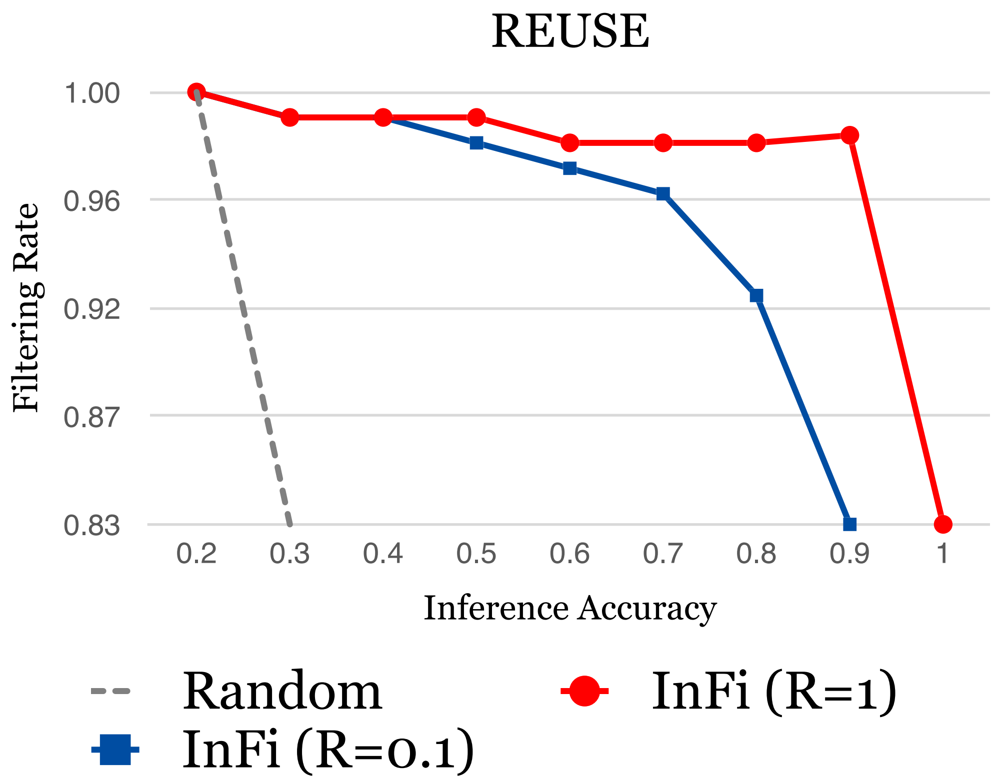


InFi is widely applicable to inference workloads.
InFi outperforms state-of-the-art approaches on ALL tasks.

4.2 Filtering Performance

- Sensitivity to training size





4.2 Filtering Performance

- Mobile-centric deployments
  - On-device, offloading, model partitioning
  - Vehicle counting workload

Throughput (FPS) / Bandwidth Saving (%)	YOLOv3	YOLOv3 + InFi-Skip	YOLOv3 + InFi-Reuse	YOLOv3-tiny
Inference Acc. (%)	100	90.3	**90.5**	**67.9**
On-device	3.2 / -	9.3 / -	27.2 / -	20.4 / -
Offloading	22.0 / -	55.2 / 66.5	77.2 / 91.1	225.3 / -
Model partitioning	24.5 / -	39.0 / 70.7	46.0 / **95.0**	230.4 / -

4.2 Filtering Performance

- Mobile-centric deployments
  - On-device, offloading, model partitioning
  - Pose estimation workload

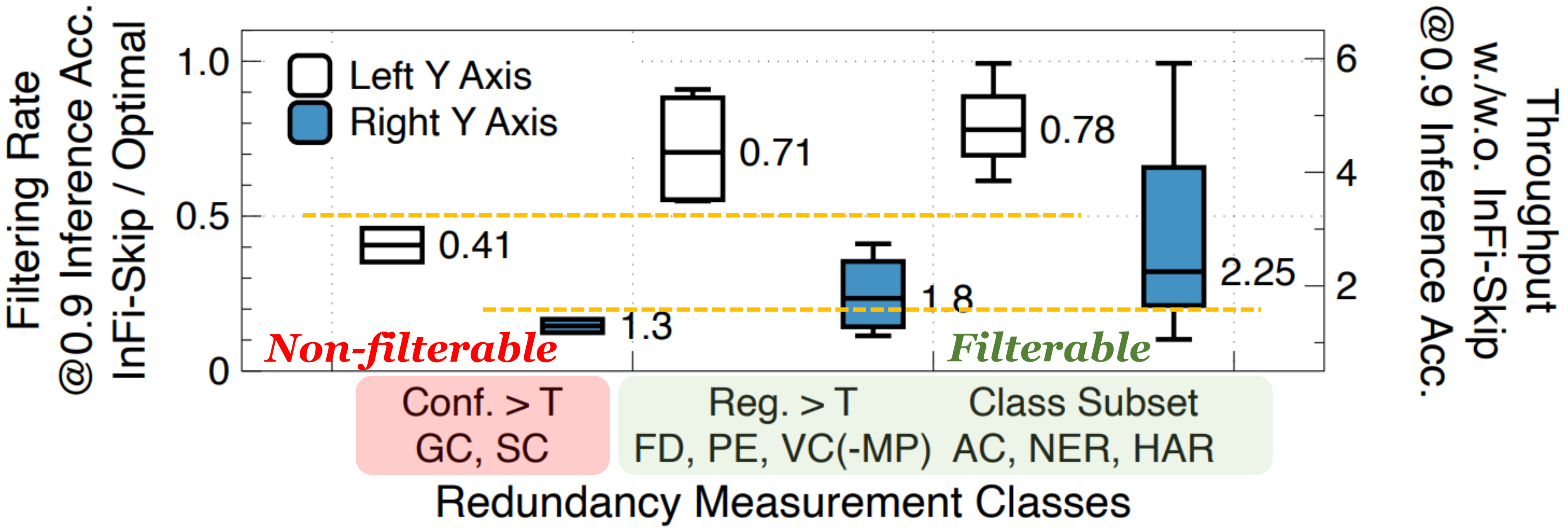
Throughput (FPS) / Bandwidth Saving (%)	OpenPose	OpenPose + InFi-Skip	OpenPose-light
Inference Acc. (%)	100	**90.1**	**76.5**
On-device	15.4 / -	18.0 / -	28.1 / -
Offloading	27.7 / -	31.5 / 18.9	98.5 / -
Model partitioning	29.2 / -	33.1 / **20.2**	102.4 / -

High throughput but low accuracy

Throughput boost, Bandwidth saved
Flexible resource-accuracy trade-off

4.3 Filterability

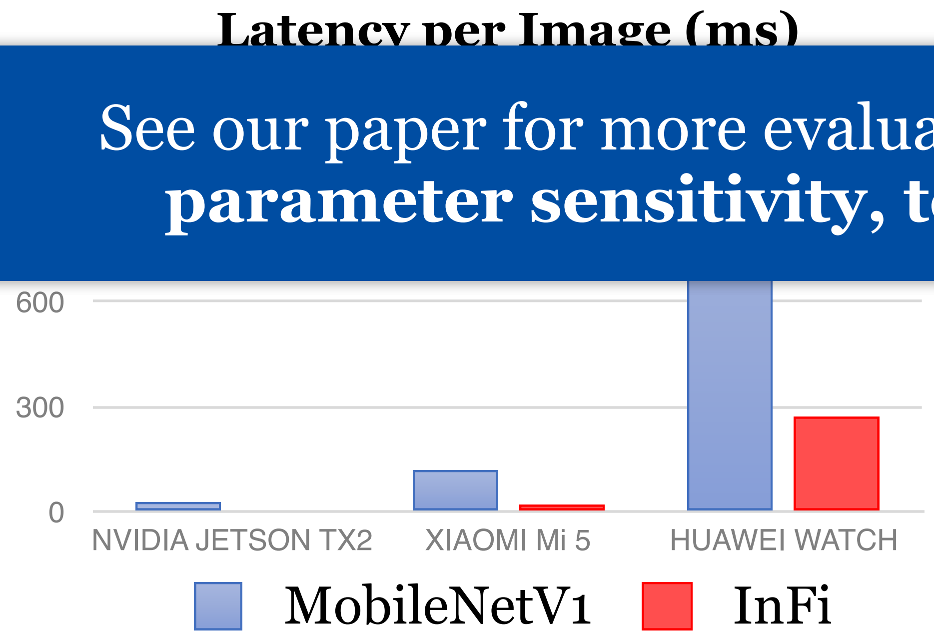
- Filterable vs. non-filterable

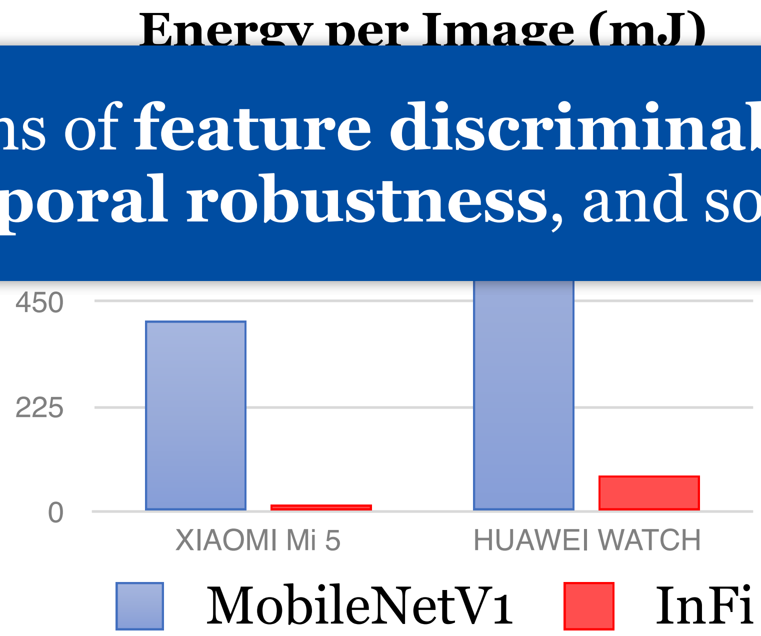


Filtering performance of FILTERABLE workloads is BETTER than NON-FILTERABLE ones.

4.4 Overhead

• Latency and energy costs on mobile platforms





See our paper for more evaluations of **feature discriminability, parameter sensitivity, temporal robustness**, and so on.


InFi Demo
MobiCom 2022

Take-Home Message

- Much **redundancy** exists in the input of inference workloads.
- We show that some workloads proved to be **filterable**, while some are non-filterable.
- Our InFi supports **almost all** mobile-centric inference workloads. Try it :)

Thank You!

Opensource: https://github.com/yuanmu97/infi

InFi : End-to-End Learnable Input Filter for Resource-Efficient Mobile-Centric Inference

Mu Yuan[1], Lan Zhang[1], Fengxiang He[2], Xueting Tong[1], Xiang-Yang Li[1]
[1]**University of Science and Technology of China** [2]JD Explore Academy

中国科学技术大学 University of Science and Technology of China
京东探索研究院 JD EXPLORE ACADEMY