TEXAS A&M UNIVERSITY

AMVP: Adaptive CNN-based Multitask Video Processing on Mobile Stream Processing Platforms

Mengyuan Chao, Radu Stoleru, Liuyi Jin, Shuochao Yao*, Maxwell Maurice†, Roger Blalock†

Department of Computer Science and Engineering, Texas A&M University

*Department of Computer Science, George Mason University

†National Institute of Standards and Technology (NIST)

The Fifth ACM/IEEE Symposium on Edge Computing (SEC'20)

Virtual, November 11-13, 2020

Agenda

1. Overview
    - **Background**
    - AMVP Framework
        a) Training
        b) Profiling
        c) Splitting
        d) Selection & Task Assignment (online)
    - Experimental Results
2. Positive Points
3. Negative Points
4. Questions/Discussion

Background – Effective Disaster Response

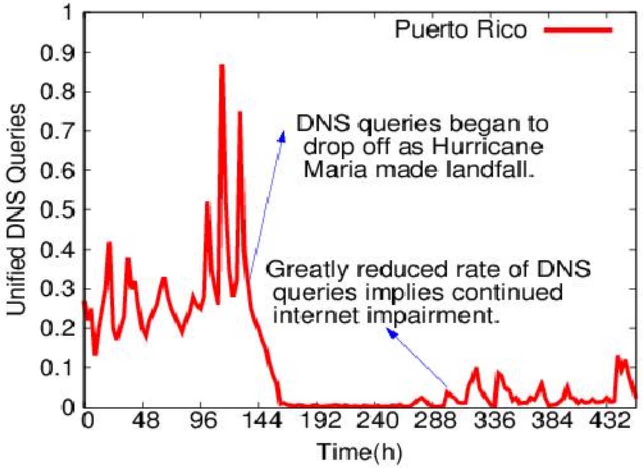
(200,000+, $15B, Indonesia, 2004)

(300,000+, $8.5B, Haiti, 2010)

(2,900+, $90B, Puerto Rico, 2017)

Effective
Disaster
Response
(EDR)

Challenges



No ISP Network

No 3G, 4G, LTE

No Social Networks

No Cloud Service

Edge
Computing
Communication
(ECC)

Pictures source: http://www.stayinbusiness.com/

Background – Techniques for ECC Enabling EDR

- **Current: System On Wheels (SOW)**

Incident Response Vehicle

AWS Public Safety & Disaster Response

Mobile Communication Vehicle

- **Limitations of SOW**
  - Need long time to arrive
  - Cannot go harshest area
  - Expensive to equip with

Picture source: https://www.fema.gov, Federal Emergency Management Agency

- **Next Generation: Edge Bubble (EB)**

The Next Generation First Responder

First responders work as a team

- **Advantages of EB**
  - Shorter time to arrive
  - Can reach harshest area
  - Cheaper to equip with
  - Closer to data source

Pictures source: https://www.fedscoop.com/dhs-wearables-first-responders/

Motivation – Multitask Video Processing at EB

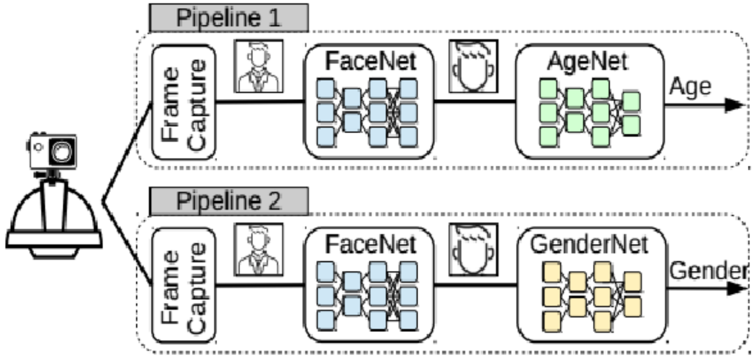
- **Scenario**: First responders need automatically record multiple information about survivors

Old or young?

Male or female?

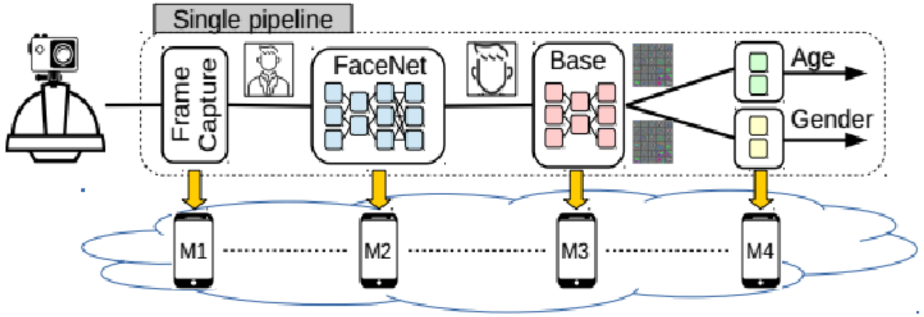
Looks OK or not?

Multiple similar CNN-based vision analysis pipelines



- **Challenges**
  - Limited computing resources on mobile devices
  - Computation intensive video stream analysis
  - Dynamic computing resources and networks
  - Diverse user performance requirements

- **Idea**: Combine multiple pipelines into a single one and adaptively share and offload CNN layers



Picture source: https://depositphotos.com/178854658/stock-photo-rescue-team-treating-injuring-field.html

Related Work – CNN-based Mobile Vision Processing

- **CNN Offloading**
  - **Cloud**: DDNN@ICDCS'17, Neurosurgeon@SIGARCH'17, JALAD@ICPADS'18, µLayer@EuroSys'19
  - **Cloudlet**: MODI@HotEdge'18, IONN@SoCC'18, DADS@Infocom'19, Couper@SEC'19
  - **IoT/Mobile**: Modnn@DATE'17, Mednn@ICCAD'17, MusicalChair@arXiv'18, DeepThings@TCAD'18
- **CNN Compression**
  - **One-fit-all**: XNOR-Net@ECCV'16, Thinet@ICCV'17, FactorizedCNN@ICCV'17, ShuffleNet@CVPR'18
  - **Adaptive**: DeepX@IPSN'16, AdaptDNN@LCTES'18, OnDemandDNN@MobiSys'18, ContextDNN@ICDCS'20
  - **Feature**: DeepFCPX@ICIP'18, ContextFCPX@CVPR'18, LosslessFCPX@MMSP'18, LossyFCPX@ICM'19
- **CNN Sharing**
  - **Inside a Model:** NestDNN@MobiCom'18, FoggyCache@MobiCom'18, DeepCache@MobiCom'18
  - **Among Models:** MCDNN@MobiSys'16, Mainstream@ATC'18

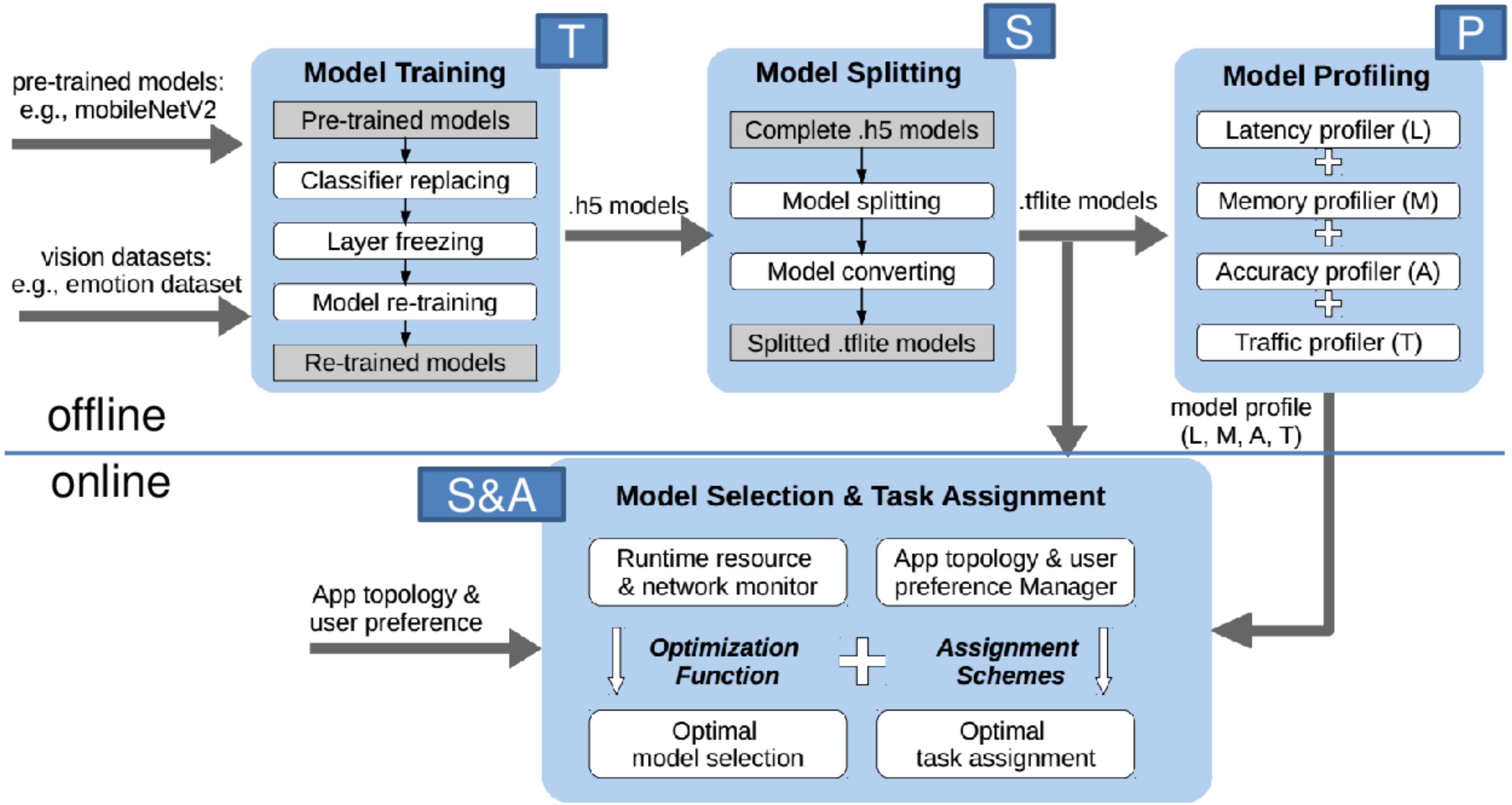
We study adaptive CNN-based multitask video processing on mobile devices with dynamic computing resources, network, and user goals.

Agenda

1. Overview
   - Background
   - **AMVP Framework**
     a) Training
     b) Profiling
     c) Splitting
     d) Selection & Task Assignment (online)
   - Experimental Results
2. Positive Points
3. Negative Points
4. Questions/Discussion

System Overview – Software Architecture

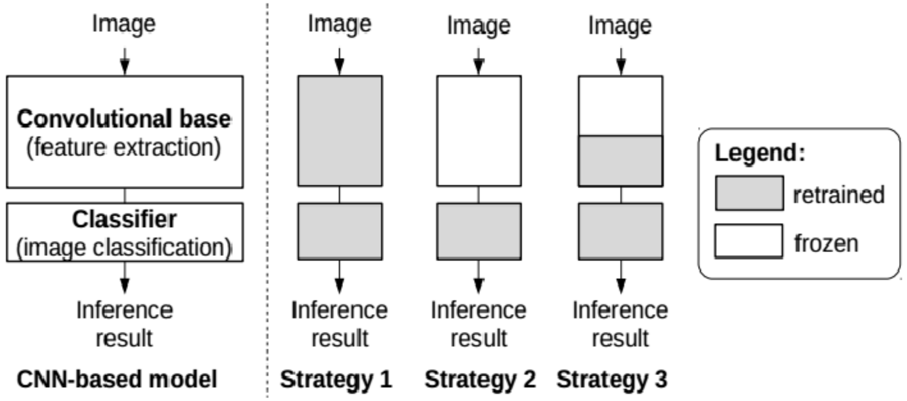
- **AMVP**: Adaptive CNN-based Multitask Video Processing Framework



Software Architecture of AMVP

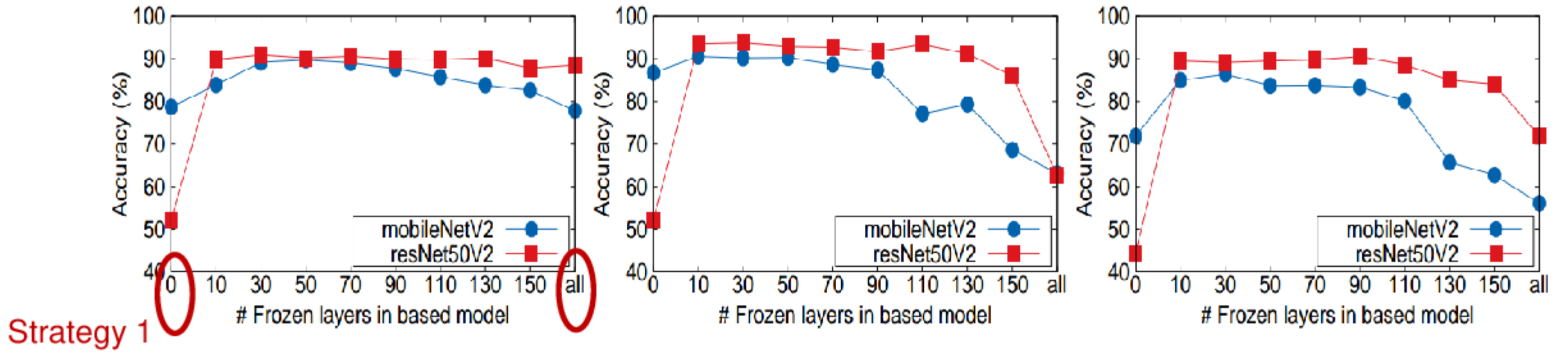
Adaptive Multitask Video Processing Framework

- ## Model Training via Transfer learning



- **Strategy 1**: retrains all weights, requires a large dataset and a lot of computation
- **Strategy 2**: freezes whole convolutional base, only trains classifier; suitable for training tasks similar to original task
- **Strategy 3**: freezes some layers in base, trains the rest; a small dataset, freezes more; a large dataset, retrains more

- ## Transfer learning results

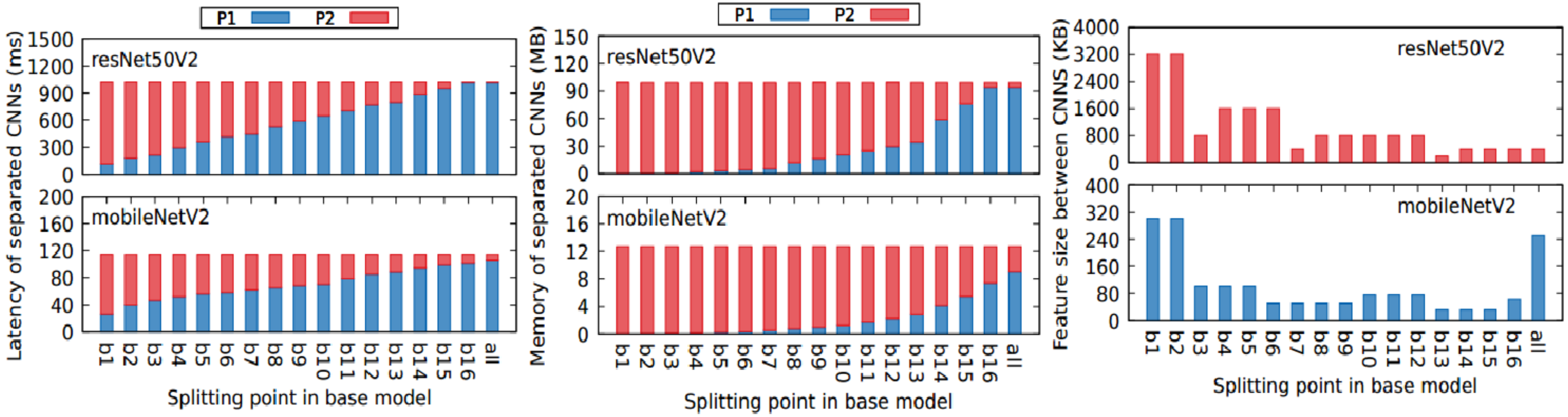


(a) GenderNet (b) EmotionNet (c) AgeNet

- **Observation 1**: resNet50V2-based are more accurate than mobileNetV2-based
- **Observation 2**: Strategy 3 achieves optimal performance for our tasks and datasets
- **Observation 3**: Simpler task is less affected by # frozen layers than complicated tasks

Adaptive Multitask Video Processing Framework

- **Model Profiling**



(a) Latency (b) Memory size (c) Feature size

➢ **Observation 1:** latency of P1 increases with block number

➢ **Observation 2:** memory is concentrated in the last few layers

➢ **Observation 3:** feature size gradually gets reduced along with inference process

T P S S&A

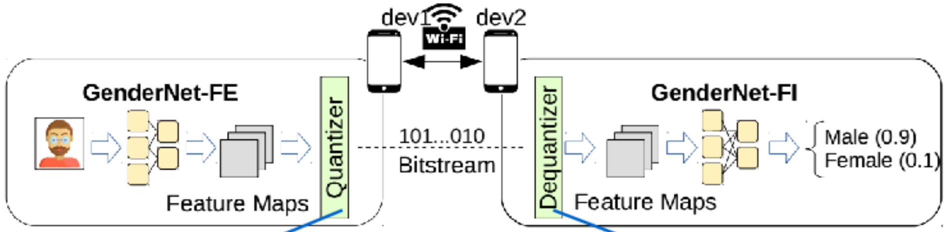
Adaptive Multitask Video Processing Framework

- **Model Splitting**

Split at earlier layer		Split at latter layer
➢ Higher inference accuracy	VS.	➢ Share more among CNNs
➢ Better computation balance		➢ Better memory balance
➢ Share less among CNNs		➢ Smaller feature traffic size
➢ Worse memory balance		➢ Lower inference accuracy
➢ Larger feature traffic size		➢ Worse computation balance

- Later layers are higher level
  - Split along frozen layers (102)
- Unlike cloud offloading (e.g. EdgeML), also need to consider “computation balance” (Chao et al., 2020, 100)

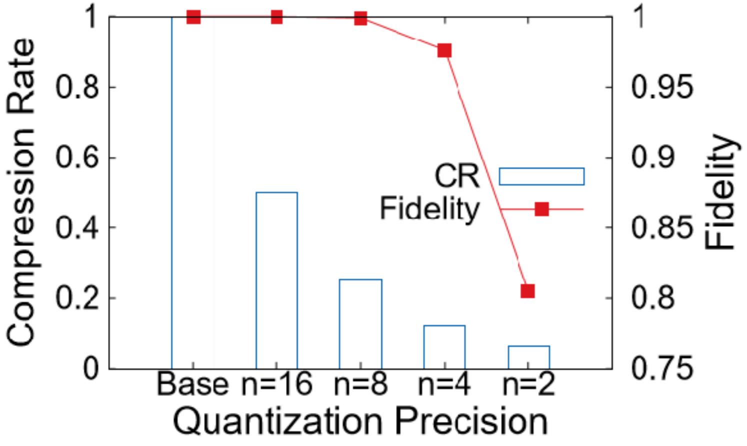
- **Quantization-based Feature Compression**



32 bits ==> n bits

$$\overline{\mathbf{F}} = \left\lfloor \frac{\mathbf{F} - \min(\mathbf{F})}{\max(\mathbf{F}) - \min(\mathbf{F})} \cdot (2^n - 1) \right\rceil$$

$$\hat{\mathbf{F}} = \frac{\max(\mathbf{F}) - \min(\mathbf{F})}{2^n - 1} \cdot \overline{\mathbf{F}} + \min(\mathbf{F})$$



Adaptive Multitask Video Processing Framework

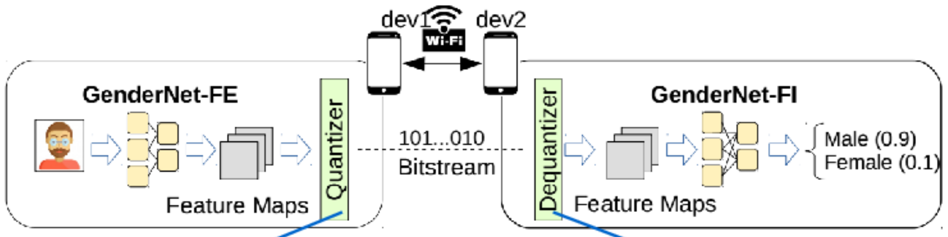
- **Model Splitting**

Split at earlier layer		Split at latter layer
➢ Higher inference accuracy	VS.	➢ Share more among CNNs
➢ Better computation balance		➢ Better memory balance
➢ Share less among CNNs		➢ Smaller feature traffic size
➢ Worse memory balance		➢ Lower inference accuracy
➢ Larger feature traffic size		➢ Worse computation balance

$$Fidelity = 1 - \frac{1}{2N}\sum_{i=1}^{N} Hamming(O_i^{og}, O_i^{cp}) \quad (4)$$

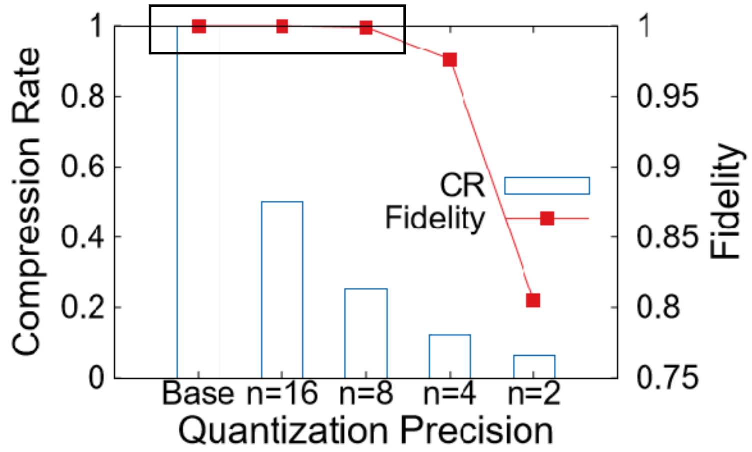
where O_i^{og} denotes the original onehot classification result of image sample i and O_i^{cp} denotes the onehot output inferred from the dequantized features. $Hamming(\cdot)$ is the hamming distance function and N denotes the total number of samples.

- **Quantization-based Feature Compression** (They compress to 8 bits without significant fidelity loss)



$$\overline{\mathbf{F}} = \left\lfloor \frac{\mathbf{F} - \min(\mathbf{F})}{\max(\mathbf{F}) - \min(\mathbf{F})} \cdot (2^n - 1) \right\rceil$$

$$\hat{\mathbf{F}} = \frac{\max(\mathbf{F}) - \min(\mathbf{F})}{2^n - 1} \cdot \overline{\mathbf{F}} + \min(\mathbf{F})$$



Adaptive Multitask Video Processing Framework

- **Model Selection and Task Assignment (MSTA)**
  - Cost function for each task

accuracy goal · accuracy · P1's delay · P2's delay · communication delay

$$C(\boldsymbol{m_s}, \boldsymbol{u_s^k}, s) = \alpha_s \cdot \frac{A_s - A(\boldsymbol{m_s})}{A_s} + \beta_s \cdot \frac{\max(0, L(\boldsymbol{m_s}, \boldsymbol{u_s^k}) - L_s)}{L(\boldsymbol{m_s}, \boldsymbol{u_s^k})} + \gamma_s \cdot \frac{\max(0, T_s - T(\boldsymbol{m_s}, \boldsymbol{u_s^k}))}{T_s}$$

$$\boldsymbol{m_s} = (m_s^1, m_s^2)$$

$$\boldsymbol{u_s^k} = (u_{m_s^1}^{k_1}, u_{m_s^2}^{k_2})$$

$$L(\boldsymbol{m_s}, \boldsymbol{u_s^k}) = \frac{l_{m_s^1}^{k_1}}{u_{m_s^1}^{k_1}} + \frac{l_{m_s^2}^{k_2}}{u_{m_s^2}^{k_2}} + D(f_{m_s^1 m_s^2}, b_{k_1 k_2})$$

delay goal

$$T(\boldsymbol{m_s}, \boldsymbol{u_s^k}) = \min\{u_{m_s^1}^{k_1} t_{m_s^1}^{k_1}, u_{m_s^2}^{k_2} t_{m_s^2}^{k_2}, \frac{1}{D(f_{m_s^1 m_s^2}, b_{k_1 k_2})}\}$$

throughput goal · P1's throughput · P2's throughput · communication throughput

  - Problem formulation

$$\begin{aligned} \underset{\boldsymbol{m_s}, \boldsymbol{u_s^k}}{\text{minimize}} \quad & C \\ \text{subject to:} \quad & \forall s : C(\boldsymbol{m_s}, \boldsymbol{u_s^k}, s) \le C, \\ & \forall k : \sum_{\{m_s^i\}} u_{m_s^i}^k \le U_k, \\ & \forall k : \sum_{\{m_s^i\}} r_{m_s^i}^k \le R_k \end{aligned}$$

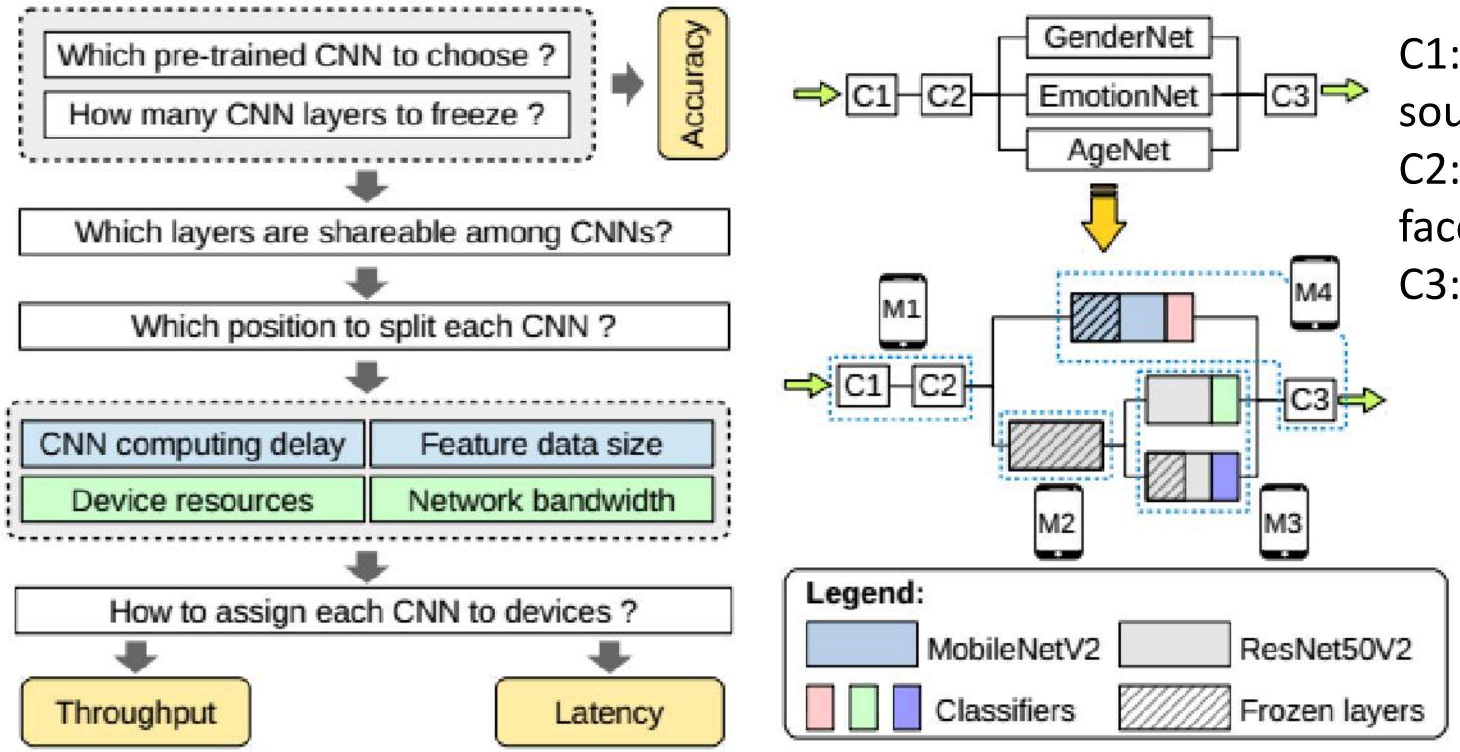
computing resource constraints

memory constraints

s: task
$\alpha_s, \beta_s, \gamma_s \in [0,1]$: user parameters, $\alpha_s + \beta_s + \gamma_s = 1$
m_s: candidate model (frozen layers and model type for task s)
(m_s^1, m_s^2): splitting m_s in two parts
u_s^k: device k utilization %
$\left(u_{\{m_s^1\}}^{\{k_1\}}, u_{\{m_s^2\}}^{\{k_2\}}\right)$: device utilization % of a model

Adaptive Multitask Video Processing Framework

- Greedy algorithm for MSTA (MinMaxCost: Minimizes max task's cost)



(a) Basic workflow of MSTA (b) An example of MSTA

C1: Get data from video source
C2: Filter out frames without faces
C3: Collect results

Agenda

1. Overview
   - Background
   - AMVP Framework
     a) Training
     b) Profiling
     c) Splitting
     d) Selection & Task Assignment (online)
   - **Experimental Results**
2. Positive Points
3. Negative Points
4. Questions/Discussion

Experiment Setup

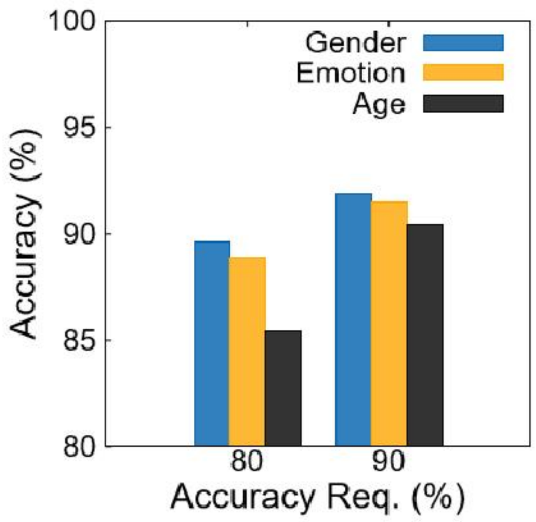
Hardware Implementation of Edge Bubble

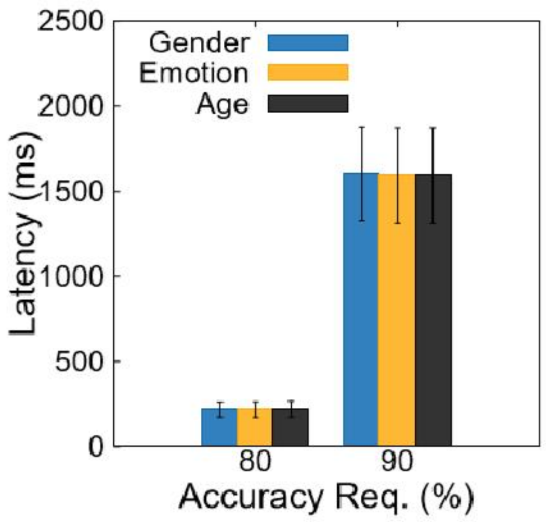
- Starting from mobileNetV2 and resNet50V2
  - Tasks: Gender, Emotion, Age
  - For each task, train group of CNNs with different starting point and frozen layers
  - For each CNN, split at different points
- Profile on Android phone
- Run on Helmet camera (Yi R), four Android phones (Essential) and a wireless manpack

- Vary values of $\alpha_s, \beta_s, \gamma_s$ (prioritizing accuracy, latency, throughput) for AVMP
- Compare to other strategies:
  - Pure Sharing Strategy (PSS): shared layers, single device
  - Pure Offloading Strategy (POS): no shared layers, multiple devices

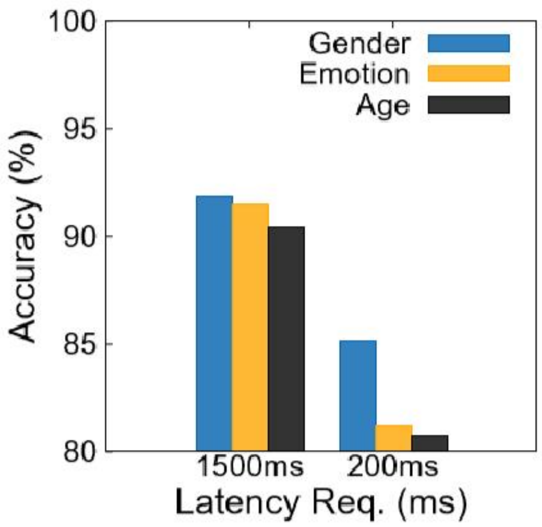
Experimental Results

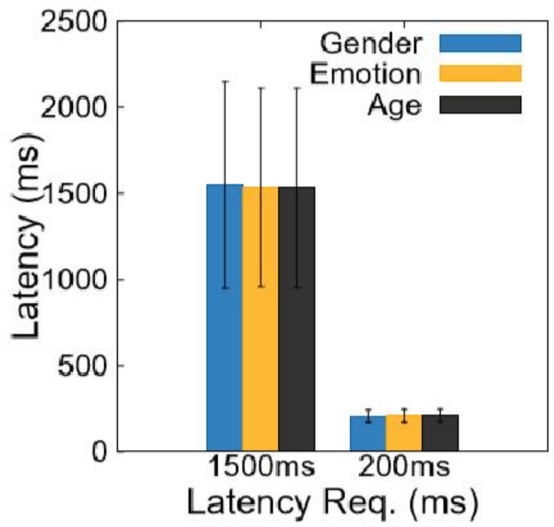
- **Adapt to different accuracy requirements**





- **Adapt to different latency requirements**

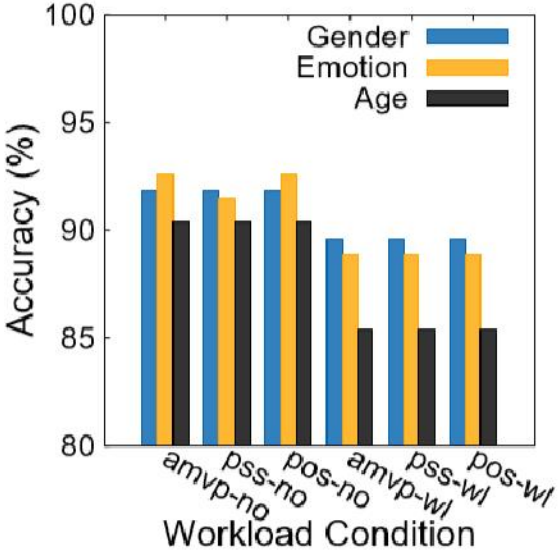


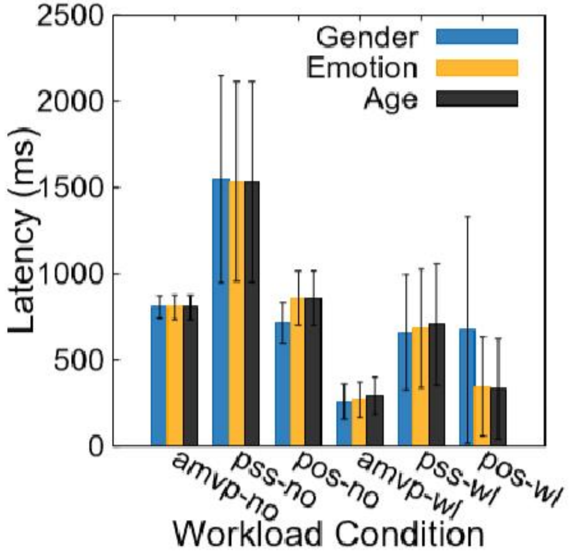


AMVP makes tradeoff between accuracy, latency, and throughput based on user preference at runtime

Experimental Results

- **Adapt to different computing resources**





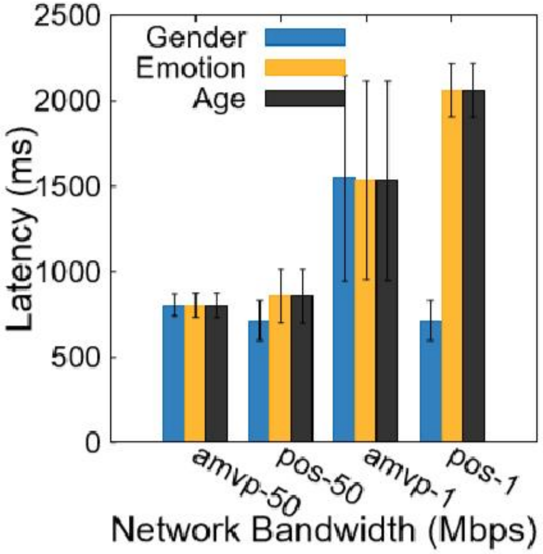
-no: Only AMVP running on device

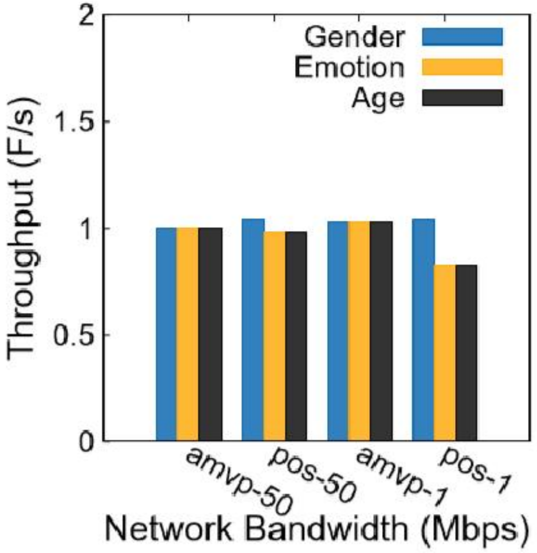
- 8% shorter latency than PSS
- around 6% shorter latency than POS

-wl: Other resource-intensive processes running on device

- 61% shorter latency than PSS and POS
- 10% higher throughput than PSS and POS

- **Adapt to different network conditions**





-1: bandwidth is 1Mbps

- Better latency for Emotion and Age

-50: bandwidth is 50Mbps

Conclusion and Future Work

- **Conclusion**: We propose AMVP, an Adaptive Multitask Video Processing Framework which supports dynamic CNN layer sharing among multiple CNN-based vision analysis tasks and adaptive CNN layer offloading from one mobile device to other devices at an Edge Bubble.

- **Future Work**
  - Apply accuracy metrics as in Chameleon [Sigcom'18], which includes precision, recall and F1 score.
  - Generalize AMVP to deploy on a heterogeneous stream processing platform including edge server
  - Support simultaneous processing of multiple videos
  - Support other AI applications such as voice recognition, speech recognition, NLP, etc.

Positive and Negative Points

- Positive:
  - Implemented and tested AMVP
  - Context and limitations of AVMP
  - Comparison against other strategies
- Negative:
  - No evaluation of energy consumption
    - Sending features might be more costly than running on one device

Questions

- Can the profiling step scale for many devices/models?
- Is this model generally applicable to many situations?
  - Limited set of models and devices
  - No connection to the cloud