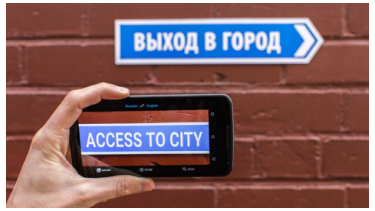
DeepDecision: A Mobile Deep Learning Framework for Edge Video Analytics

Xukan Ran*, Haoliang Chen*, Xiaodan Zhu^, Zhenming Liu^, Jiasi Chen*

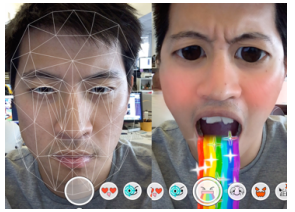
*University of California, Riverside ^College of William and Mary

Deep learning on mobile devices

- Augmented reality (AR) on mobile devices is gaining popularity



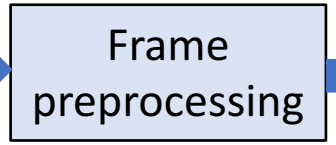
Google Translate
(text processing)

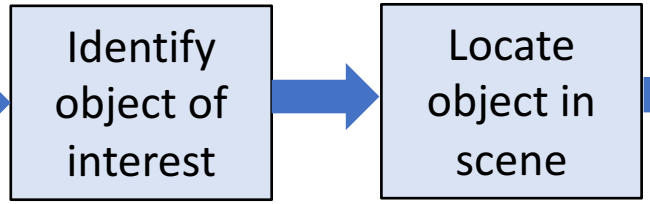


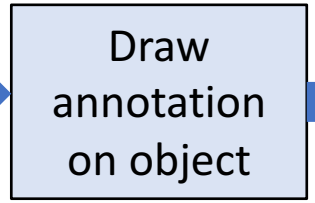
Snapchat filters
(face detection)

- Object detection + recognition are the bottlenecks in the AR processing pipeline

Input
(camera feed)



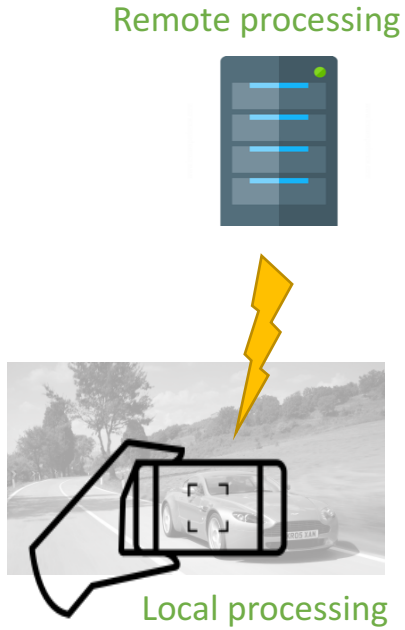




Output
(annotation)

- Deep learning is state-of-the-art in computer vision for object recognition

Challenges with current approaches



- Current approaches for machine learning on mobile devices
  - Local-only processing
    - Apple Photos, Google Translate
    - GPU speedup

    → Slow! (~600 ms/frame)[1]
  - Remote-only processing
    - Apple Siri, Amazon Alexa

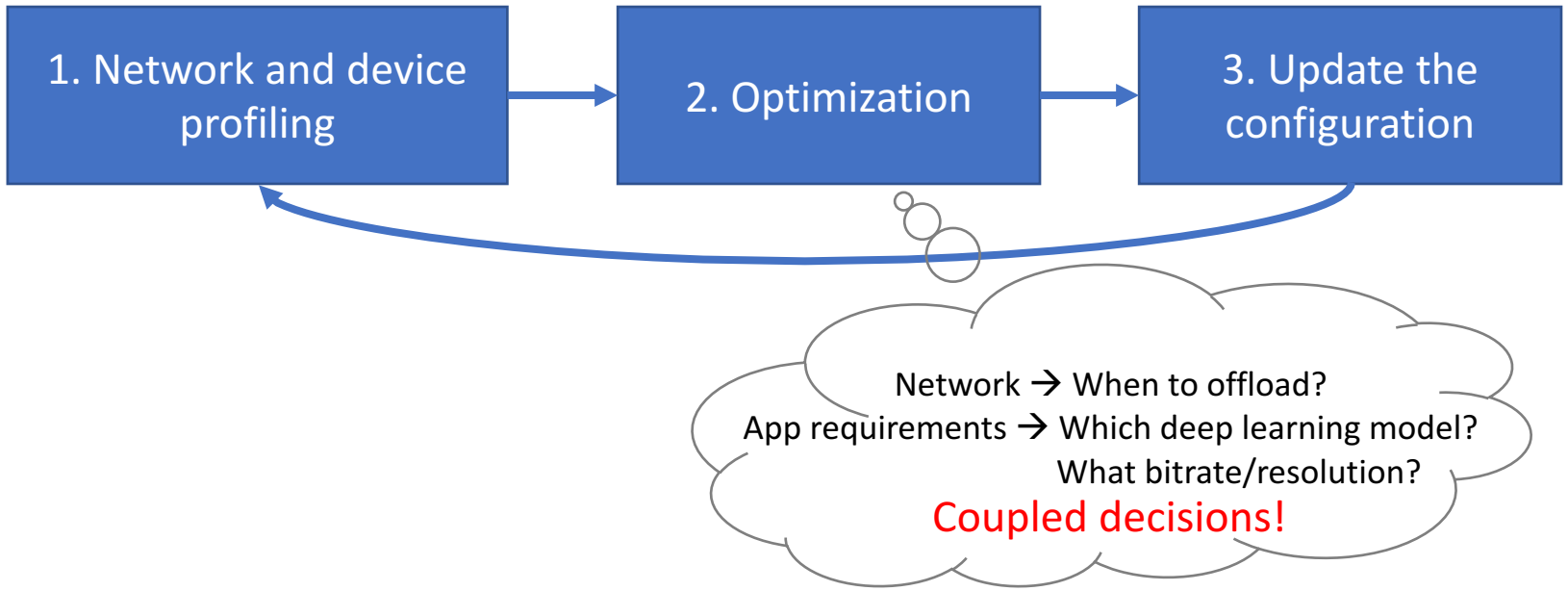
    → Doesn't work when network is bad
- Our observations
  - Different AR apps have different accuracy and latency requirements
  - Network latency is often higher than CPU/GPU processing time on the edge server
  - Video streams and deep learning models are "compressible"

- [1] L. Huynh, Y. Lee, R. Balan, "DeepMon: Mobile GPU-based Deep Learning Framework for Continuous Vision Applications", *ACM MobiSys*, 2017.

Problem Statement

- Problem: How should the mobile device be configured to meet the requirements of the AR app and the user?

- Solution: Periodically profile, optimize, and update the configuration



Related Works

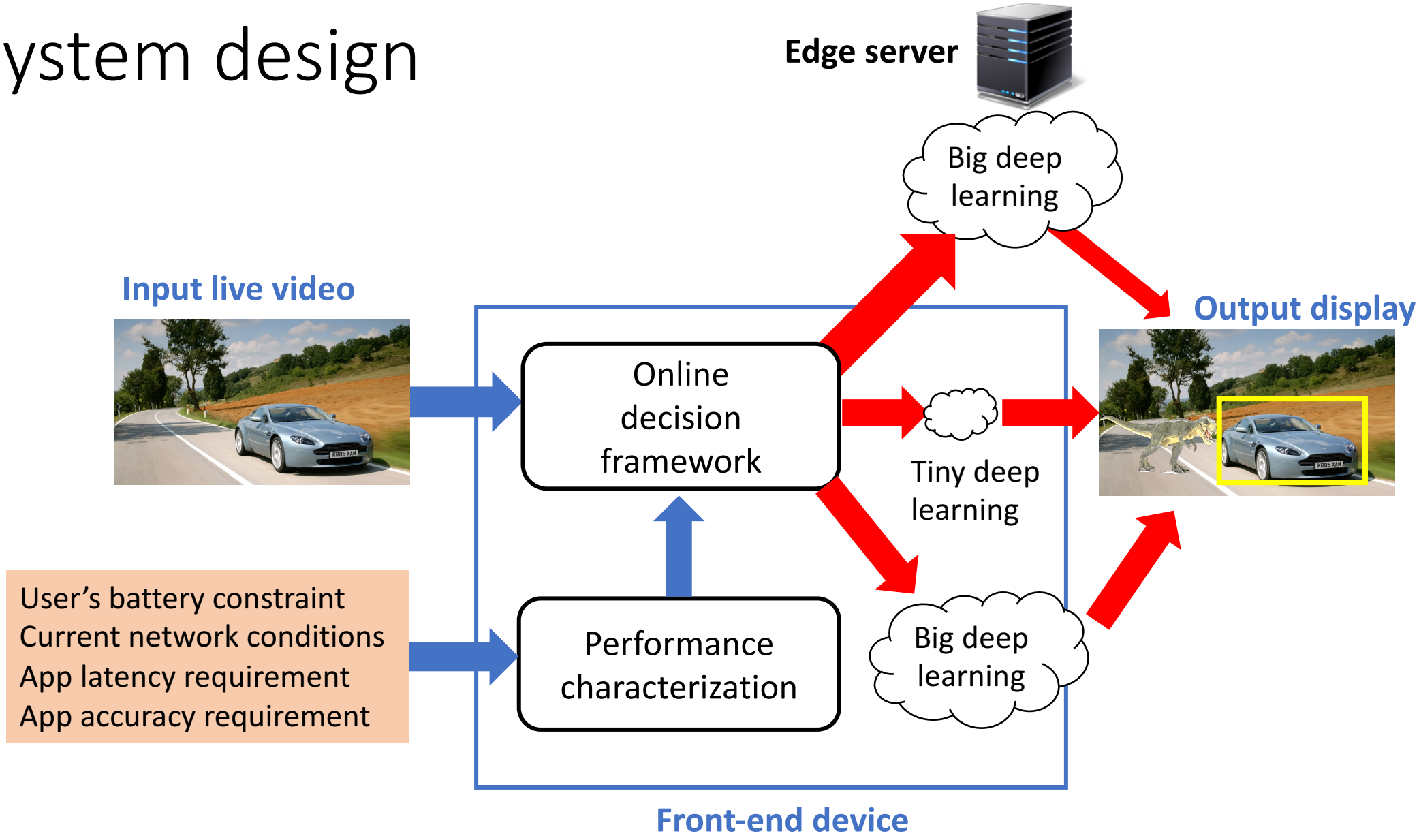
- Local on-device processing
  - Mobile GPUs to speed up deep learning, e.g. DeepMon[1]
  - Complementary to our work, we can profile and leverage this speedup

- Remote processing on server
  - Always run on nearby edge server, e.g. Glimpse[2], or [3]
  - Doesn't consider on-device machine learning with compressed models

- Latency-sensitive applications
  - Decision framework to minimize application latency, e.g., MCDNN[4]
  - We consider latency-accuracy tradeoff, network conditions, data compression

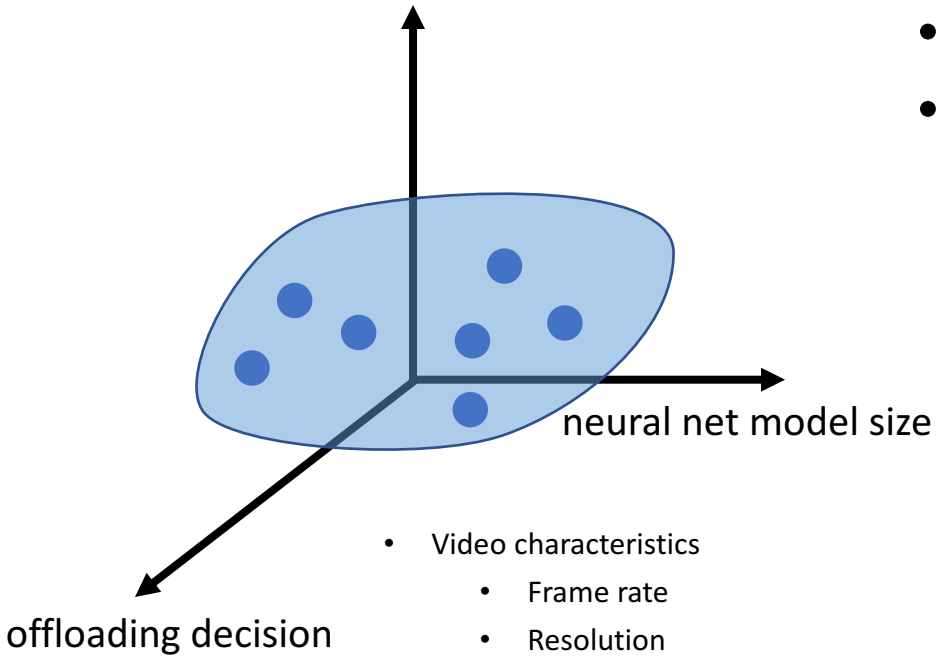
[1] L. Huynh, Y. Lee, R. Balan, "DeepMon: Mobile GPU-based Deep Learning Framework for Continuous Vision Applications", *ACM MobiSys*, 2017.
[2] T. Chen, L. Ravindranath, S. Deng, P. Bahl, and H. Balakrishnan. "Glimpse: Continuous, Real-Time Object Recognition on Mobile Devices", *ACM Sensys, 2015.*
[3] H. Zhang, G. Ananthanarayanan, P. Bodik, M. Philipose, P. Bahl, and M. Freedman. "Live video analytics at scale with approximation and delay-tolerance." *USENIX NSDI*, 2017.
[4] S. Han, H. Shen, M. Philipose, S. Agarwal, A. Wolman, A. Krishnamurthy, "MCDNN: An Approximation-Based Execution Framework for Deep Stream Processing Under Resource Constraints", *ACM Mobisys,* 2016.

System design



Online decision framework

Degrees of freedom:



- Video characteristics
  - Frame rate
  - Resolution
  - Bit rate
- Deep learning characteristics
  - Model size
  - Model latency / energy
  - Model accuracy

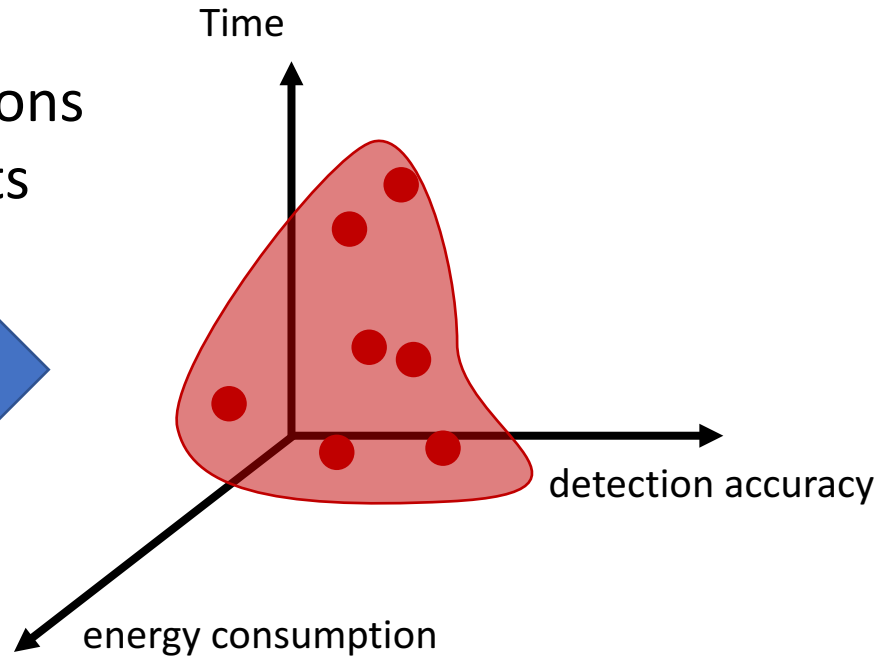
Constraints:

- Current network conditions
- Application requirements

Optimize decision

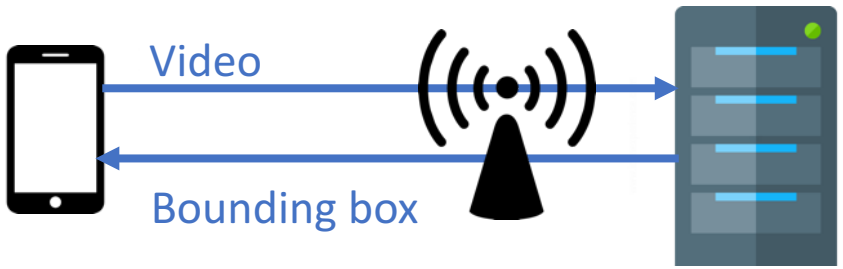
- Network condition
  - Bandwidth
  - Latency
- App requirements
  - Latency
  - Accuracy
  - Energy

Metrics:



Offline performance characterization

- Deep learning model: Yolo built on Tensorflow [1]
  - Tiny-DL: 9 convolutional layers (phone)
  - Big-DL: 22 convolutional layers (server and phone)
- Local processing: Samsung Galaxy S7 Android phone with 8-core CPU and 4 GB RAM
- Remote processing: Server with quad-core CPU, 8 GB RAM, NVIDIA GeForce GTX970 graphics card with 4GB of RAM
- Encoded 20 test videos [2] at different bit rate and resolutions as input



[1] Joseph Redmon, Ali Farhadi, “YOLO9000: Better, Faster, Stronger”, *CVPR*, 2017.
[2] xiph.org video test media. https://media.xiph.org/video/derf

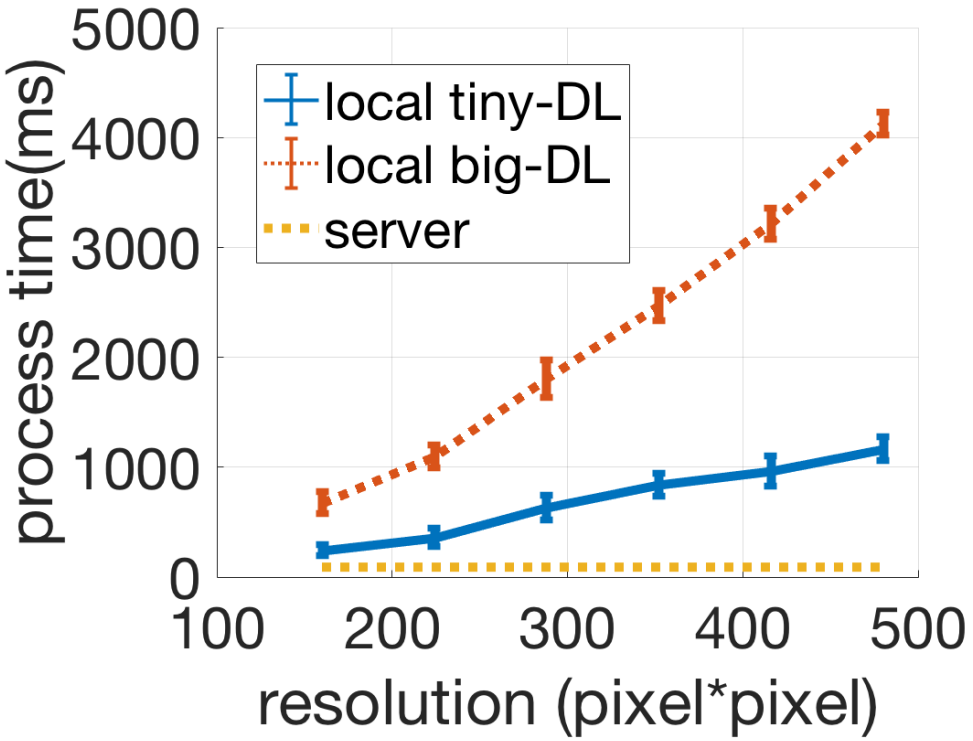
Metrics

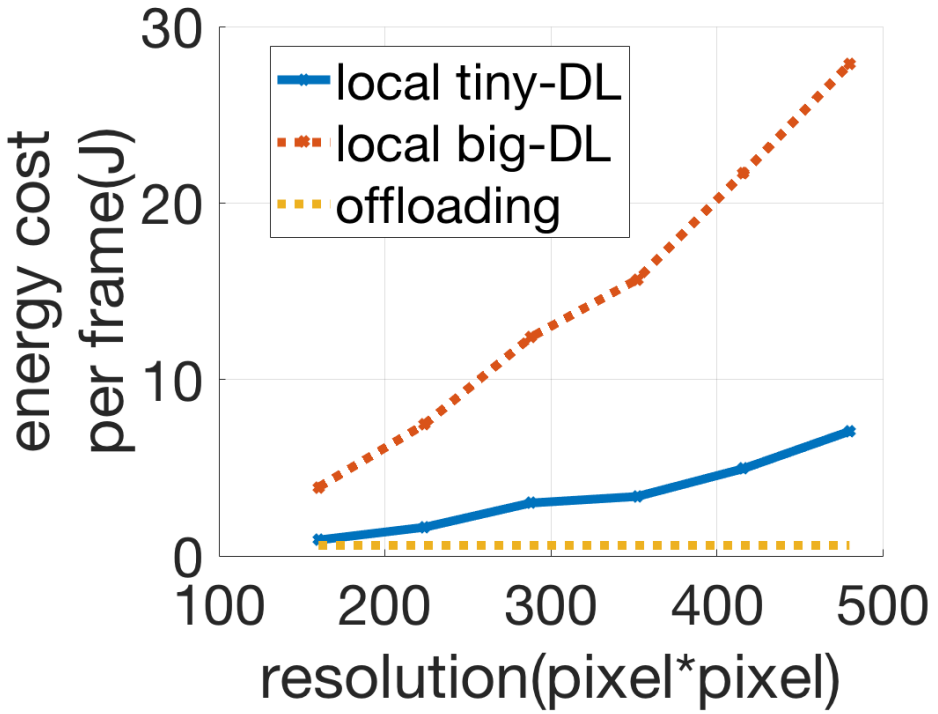
$$IoU = \frac{\text{Area of Overlap}}{\text{Area of Union}}$$

- Accuracy
  - Classification and location both important for AR
    - Intersection over union (IoU) metric
  - Ground truth: Big deep learning running on highest resolution

- Timing
  - Latency: time from when we sent the frame to getting the result
  - Frame rate: 1 / time between consecutive frames

Performance characterization:
How do latency and energy change with video resolution?



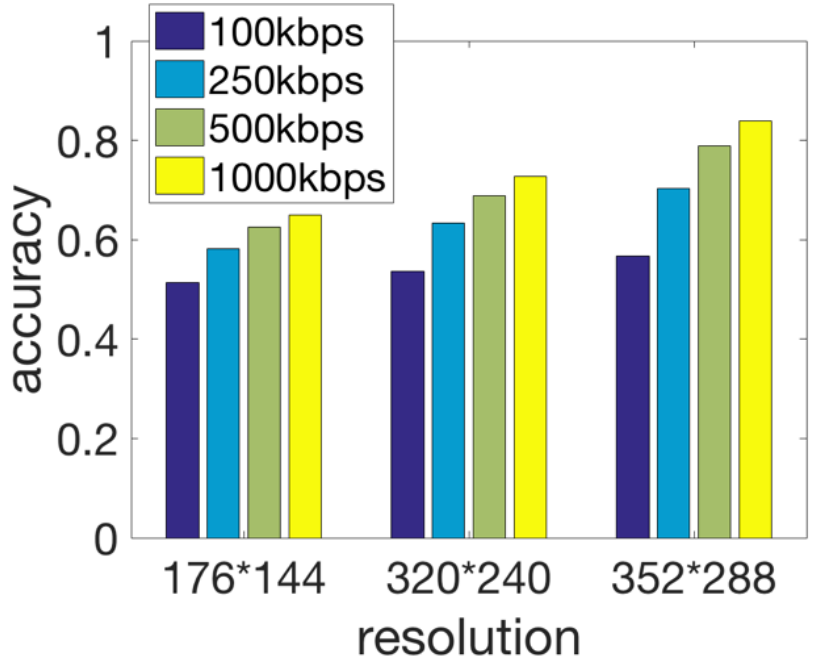


Energy and latency increase with pixels^2 for local processing

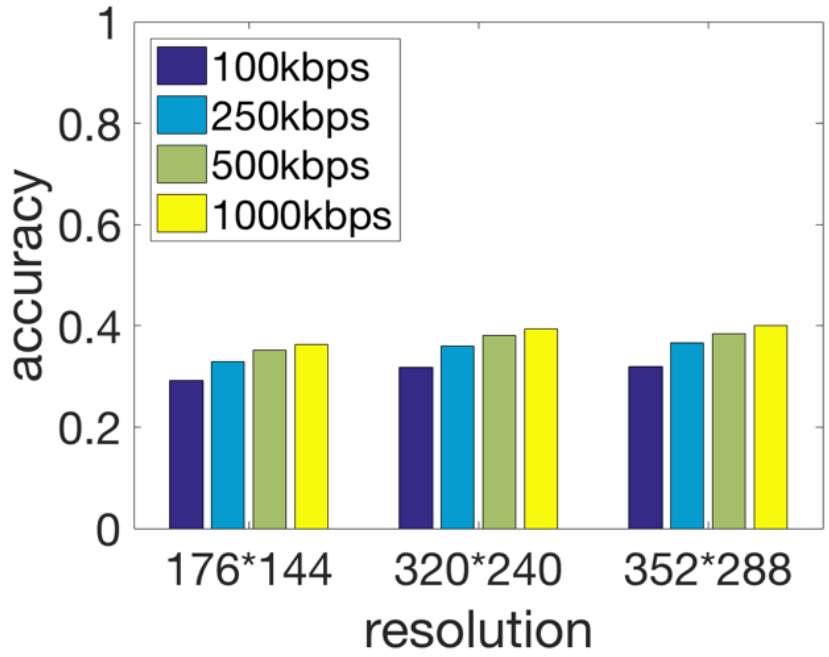
Performance characterization:
How does accuracy change with bit rate and resolution?

- Encoded videos at different bitrates and resolutions

Big deep learning:

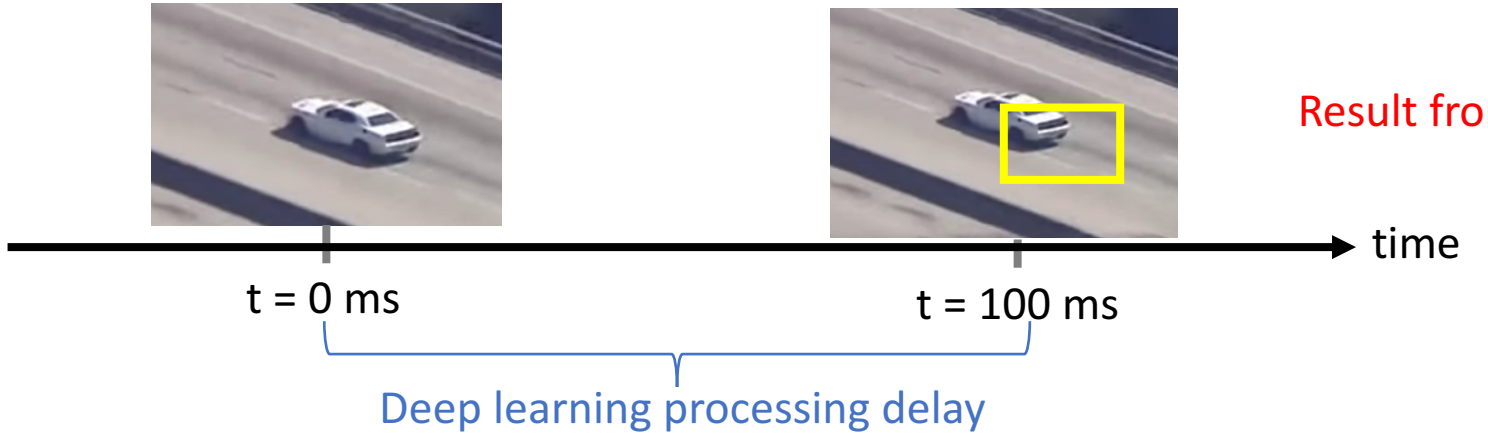


Tiny deep learning:



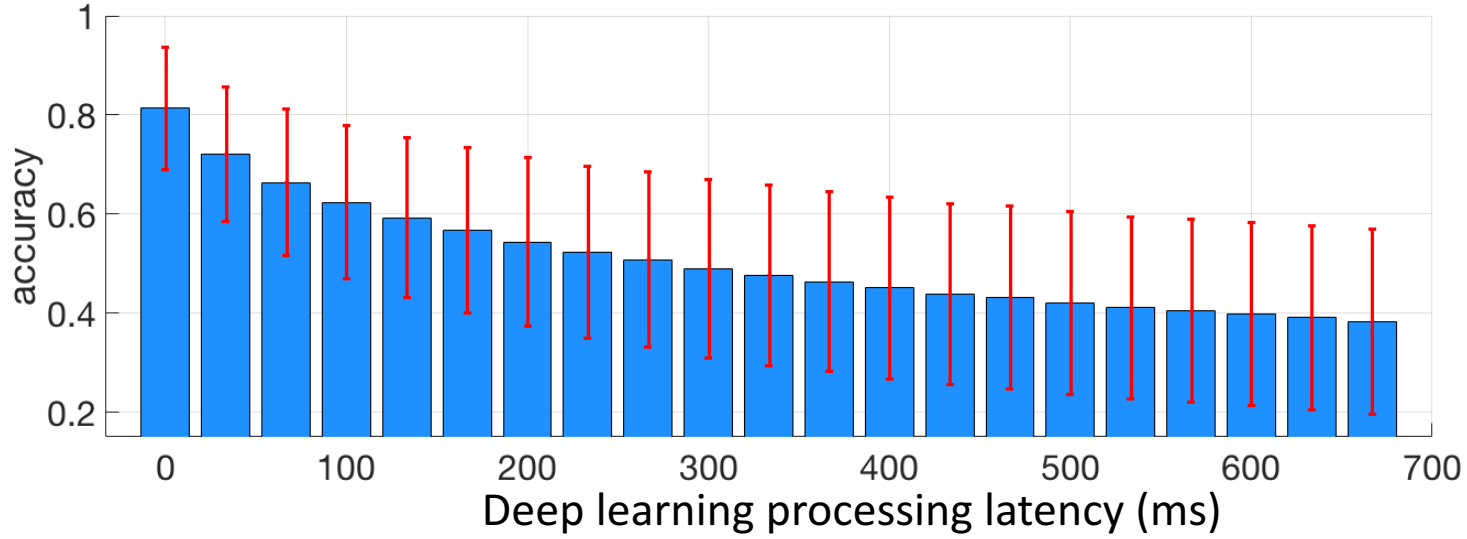
Accuracy increases more with resolution than bitrate, especially for big deep learning

Performance characterization:
How does accuracy change with latency?



Result from deep learning is stale!

- Measured accuracy as deep learning processing latency increased



Accuracy decreases as latency increases.

DeepDecision Optimization Problem

From offline performance characterization:

$a_i(p, r, li)$: accuracy function of model i
$l_i^{CNN}(p)$: latency function of model i
$b_i \ (p, r, f)$: battery function of model i

Frame rate — Accuracy

Maximize $f + \alpha\left(\sum_{i=0}^{N} a_i(p, r, l_i) \cdot y_i\right)$

Local processing time — Network transmission time

Subject to

$$l_i = \begin{cases} l_i^{CNN}(p) + \frac{r}{fB} + L & if\ i = 0 \\ l_i^{CNN}(p) & if\ \ i > 0 \end{cases}$$

Calculate end-to-end latency.

$\sum_{i=0}^{N} l_i^{CNN}(p) y_i \ \leq 1/f$ — Finish processing a frame before next frame arrives.

$\sum_{i=0}^{N} b_i(p, r, f) \cdot y_i \leq \mathcal{B}$ — Don't use more than *B* battery

$a_i(p, r, f) \geq A \cdot y_i\ , \forall\ i$: — Meet application accuracy requirement.

$f \geq F;$ — Meet application frame rate requirement.

$r \cdot y_0 \leq R$ — Don't use more than *R* bandwidth.

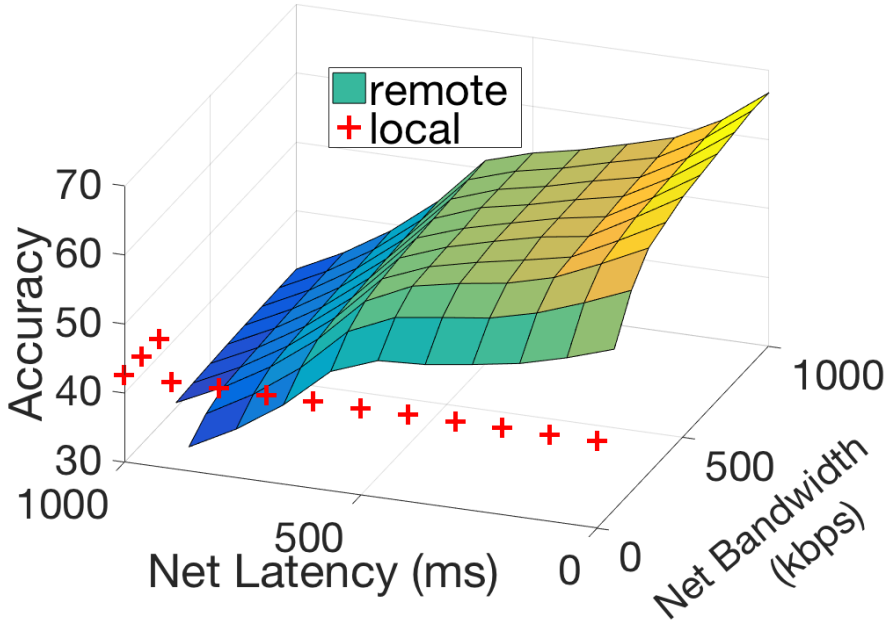
$\sum_{i=0}^{N} y_i = 1$

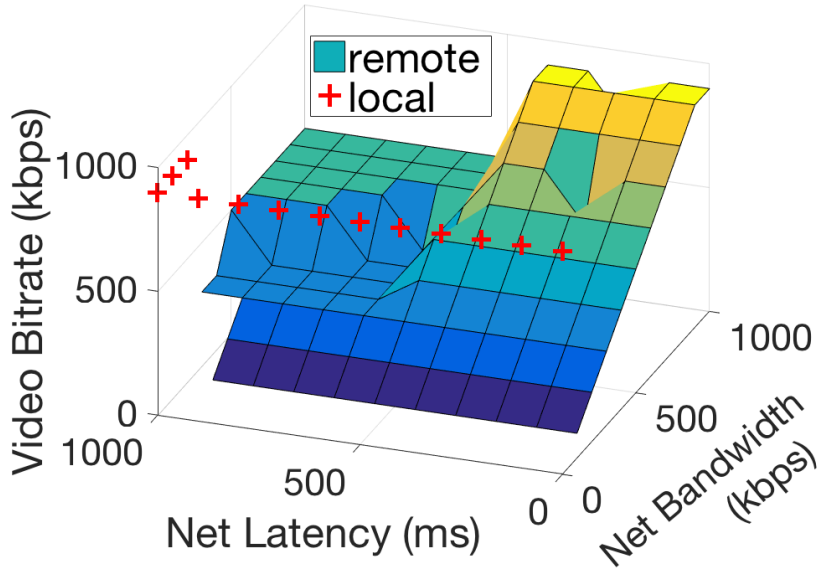
Variables

$p, r, f \ \geq 0;\ y_i \ \in \{0,1\};$

p: video resolution
r: video bitrate
f : frame rate
y_i : which deep learning model to run (local, remote)

How do network conditions impact the optimization solution?



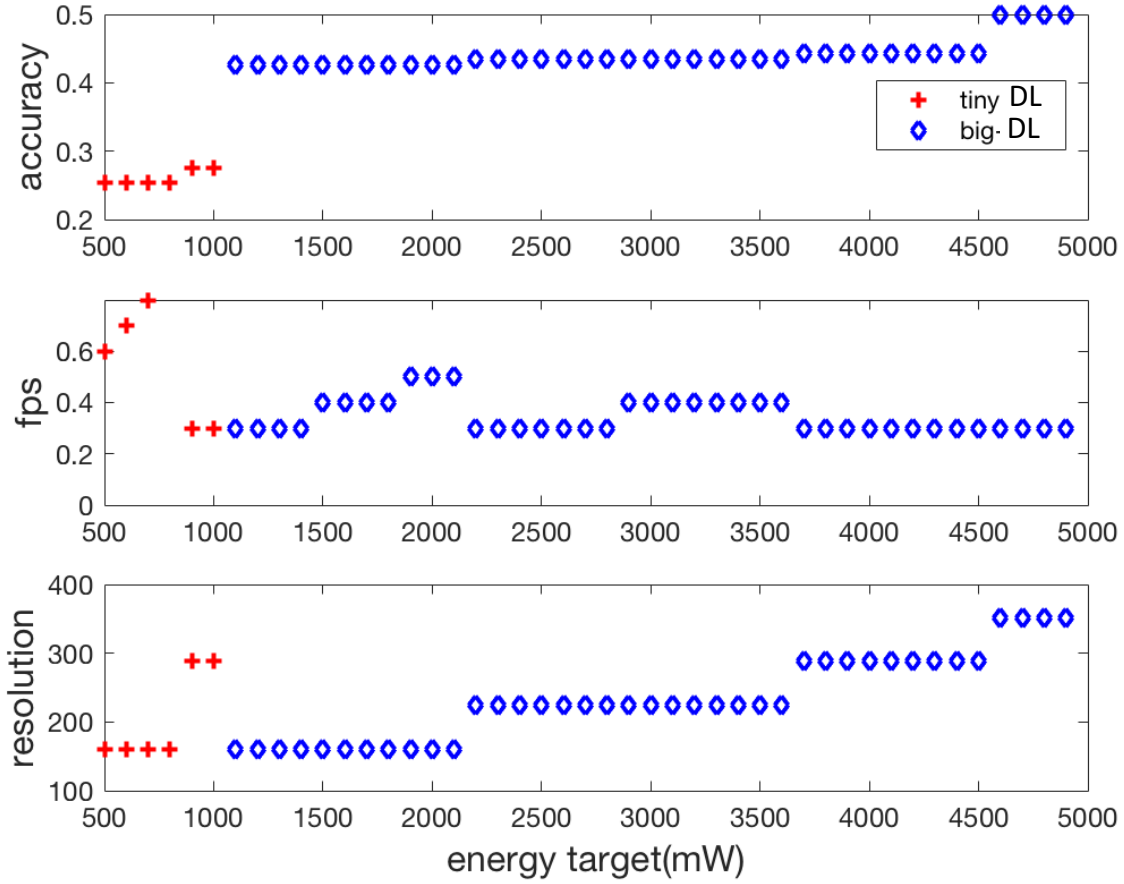


When bandwidth is bad or latency is too long, run local model.

Under the hood: Video bitrate increases to maximize accuracy

How does energy impact the optimization solution?

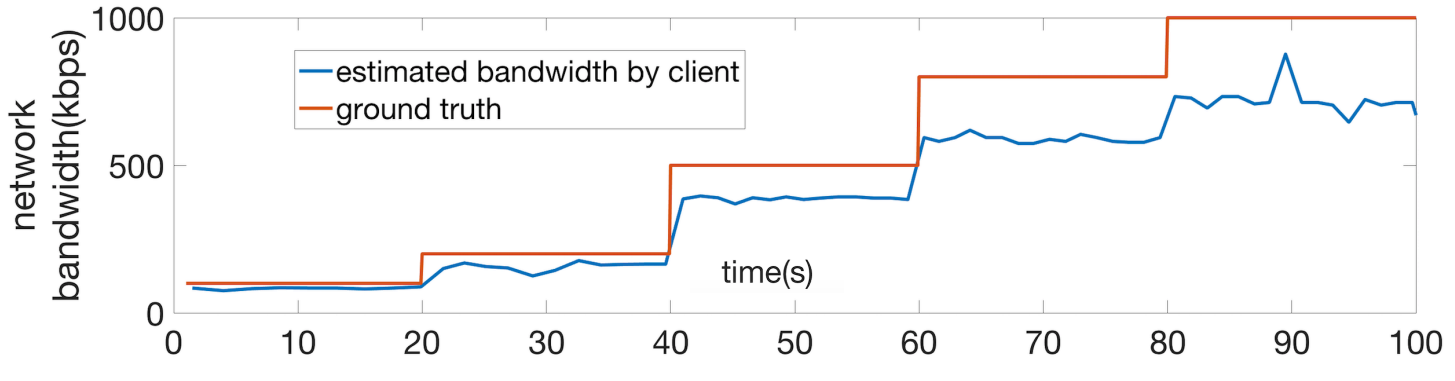
- Many possible scenarios, picked a particular instance
  - Poor network connectivity
- DeepDecision tries to maximize:
  - frame rate + $\alpha \cdot$ accuracy
- What happens as battery target increases?

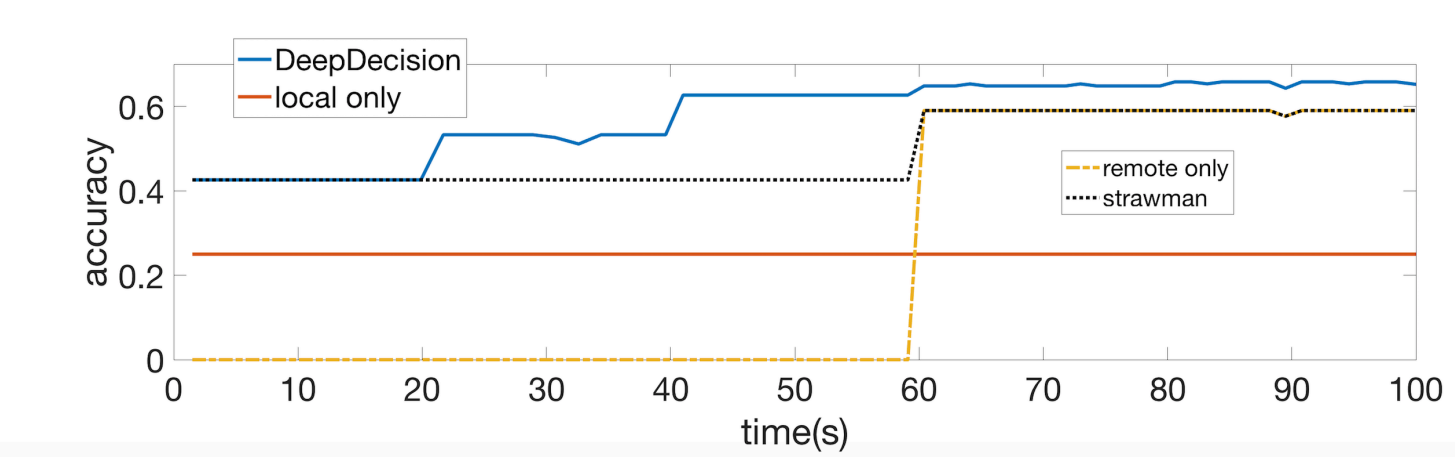


As DeepDecision uses more energy, objective function increase

End-to-end system evaluation

- Varied the network bandwidth over time
- Compared against 3 algorithms
  - *Local only*
  - *Remote only*
  - *Strawman*: Offload if bandwidth > 500 kbps, otherwise run locally
  - *DeepDecision:* our system





DeepDecision can adapt to different network condition by choosing the optimal configuration under the hood

Key Take-Aways

Real-time video analysis using local deep learning is slow (~600 ms/frame on current smartphones)

Relationship between degrees of freedom and metrics is complex, and requires profiling

DeepDecision adjusts to different network conditions by choosing the right video and model compression