Server-Driven Video Streaming for Deep Learning Inference

Kuntai Du, Ahsan Pervaiz, Xin Yuan, Aakanksha Chowdhery, Qizheng Zhang, Henry Hoffman, Junchen Jiang

THE UNIVERSITY OF CHICAGO

Google Brain

Overview

Video analytics is pervasive, but fortunately it allows for **aggressive** video compression

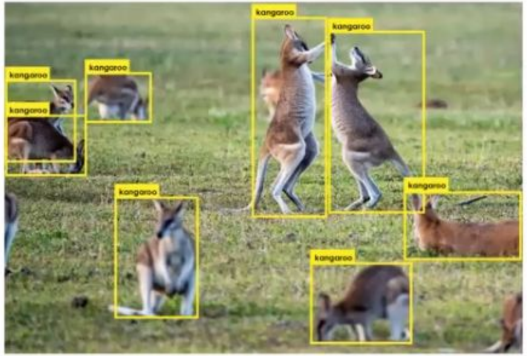
⇩

Only the **server-side DNN** has sufficient info to guide efficient video compression/streaming

⇩

Our idea: Drive video streaming by the real-time feedback from the server-side DNN
→DNN-Driven Streaming (DDS)

Video analytics become more and more pervasive



Wild-life camera
Learn about the habit of animals

Traffic camera
Monitor the traffic condition

Drone camera
Estimate the number of cars

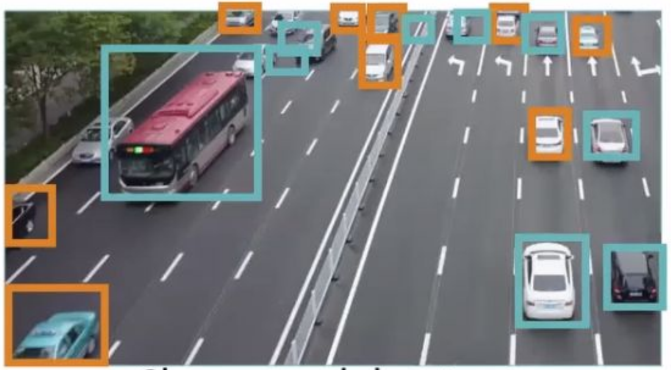
Our goal: Scale out video analytics.

Video needs to be streamed out for accurate analytics

missed objects

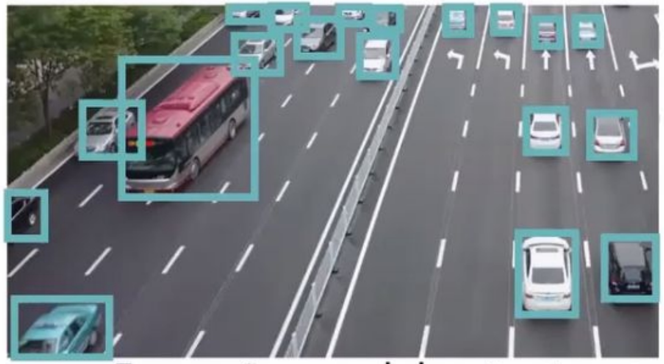
detected objects

Analyzed by **camera** locally



Cheap model output
(SSD-MobileNet-v2)

Analyzed by a GPU-equipped **server**

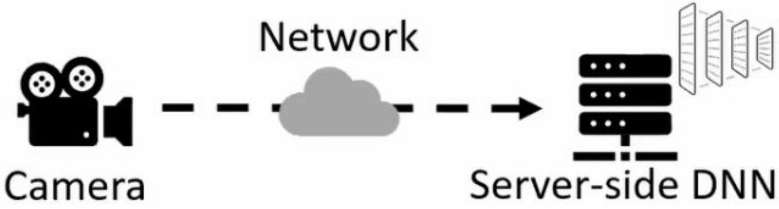


Expensive model output
(FasterRCNN-ResNet101)

Limited by camera-side compute power, the camera's local inference is inaccurate.

We need to stream the video to the server for accurate inference.

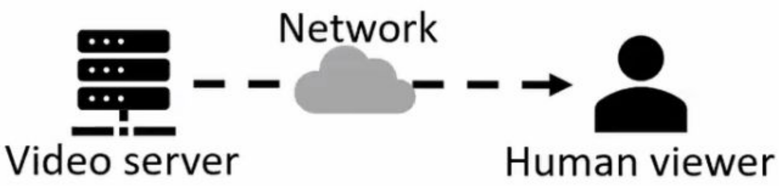
Design goals of video streaming protocol

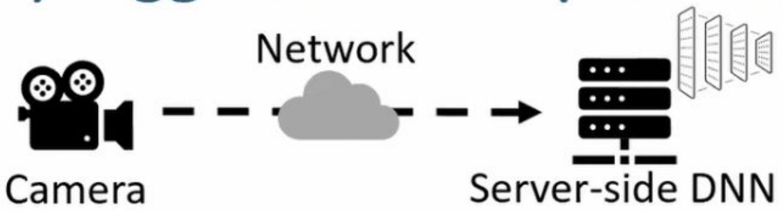


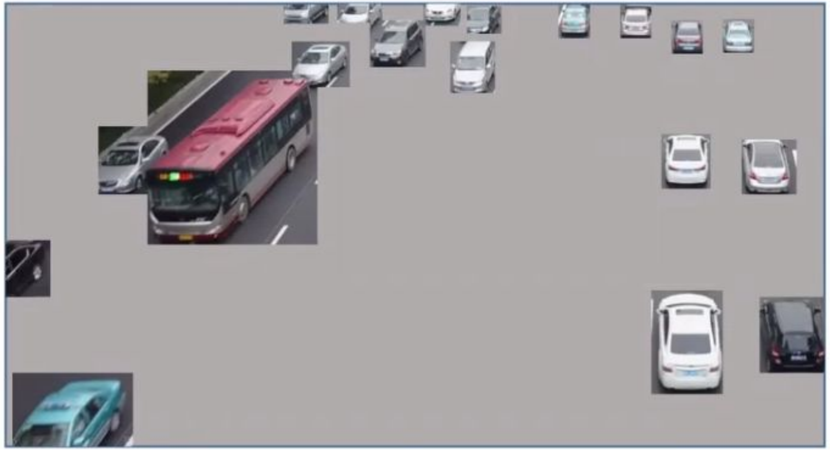
- **Preserver high accuracy**
  - As if the server-side DNN is directly inference on the raw video

- **Save bandwidth**
  - Bandwidth cost (e.g., cellular) ↓
  - Streaming delay per frame↓ ➔ Response delay ↓

Bandwidth saving opportunity: aggressive compression

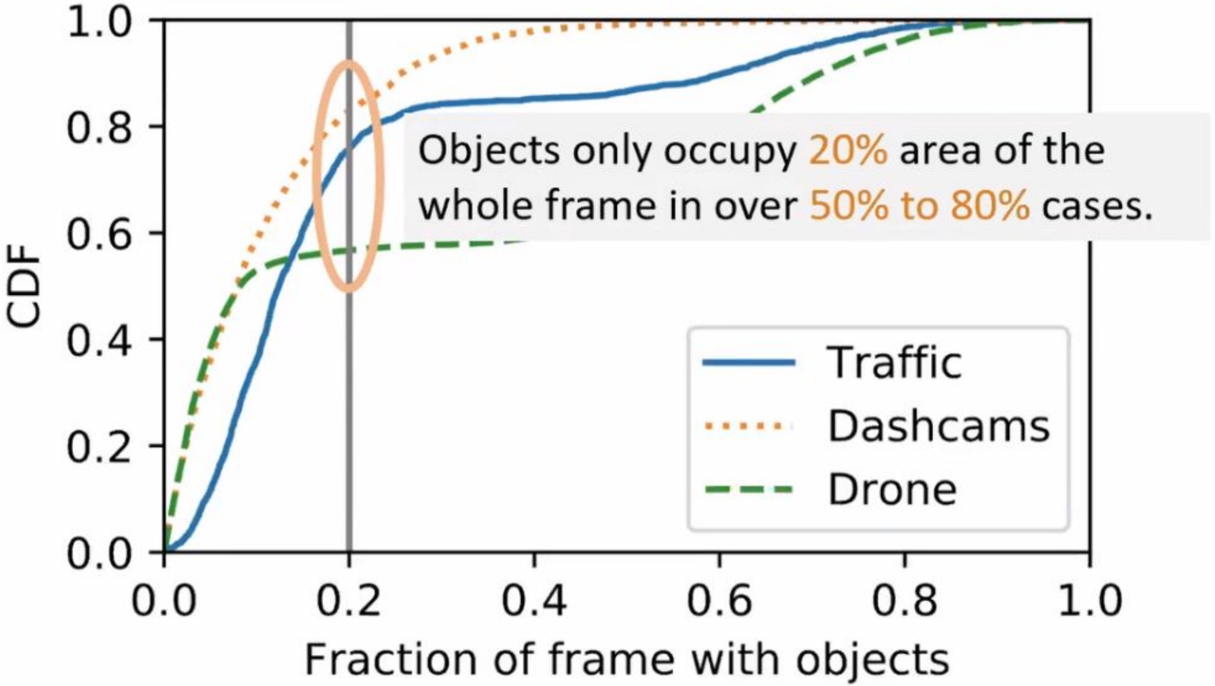






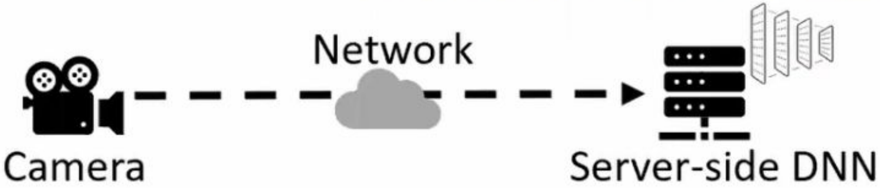
Video analytics enables aggressive compression on non-object pixels.

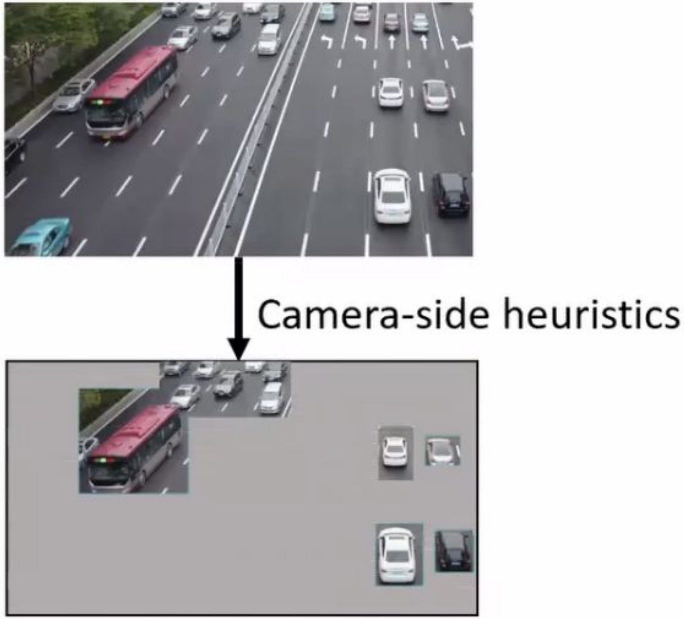
Potential bandwidth savings



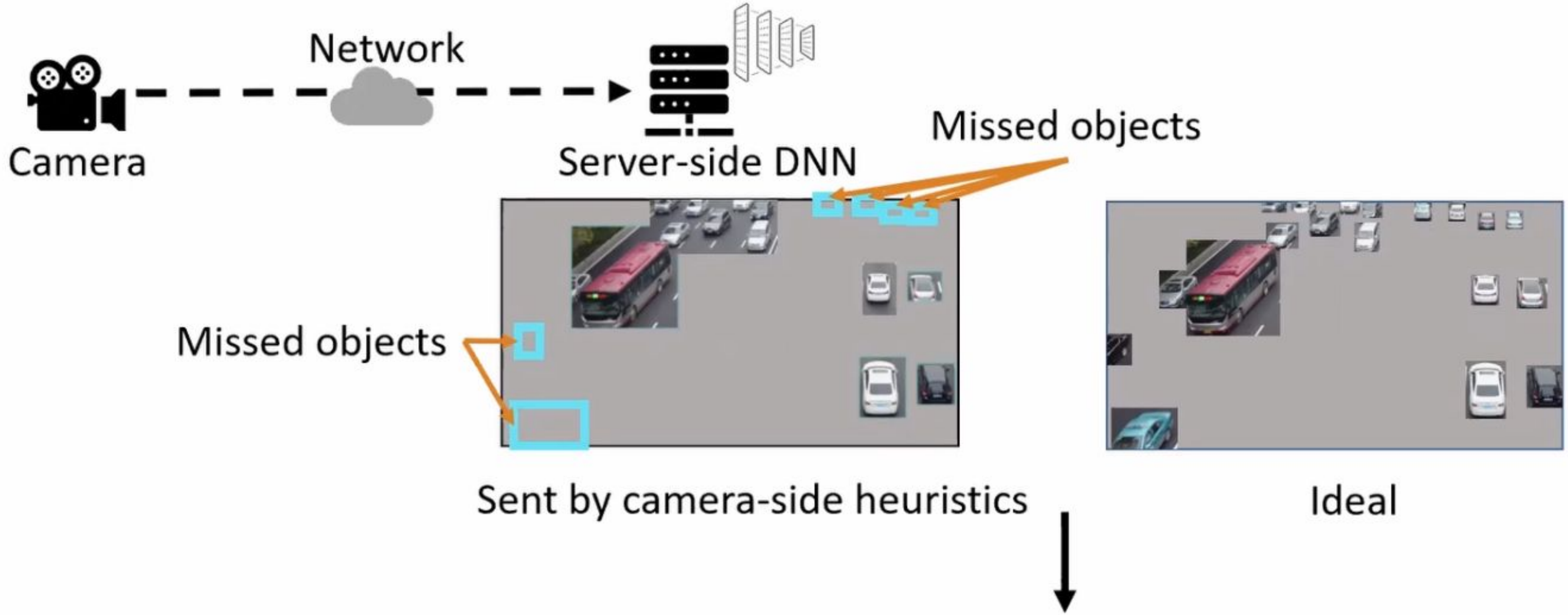
We can compress 80+% pixels without hurting accuracy!

Previous work type 1: Camera-side heuristics



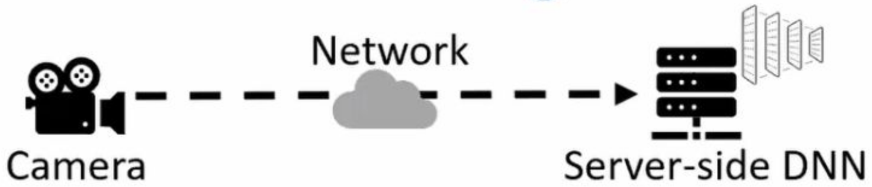


Previous work type 1: Camera-side heuristics



The camera-side heuristics may miss many objects which will never be recovered by the server DNN.

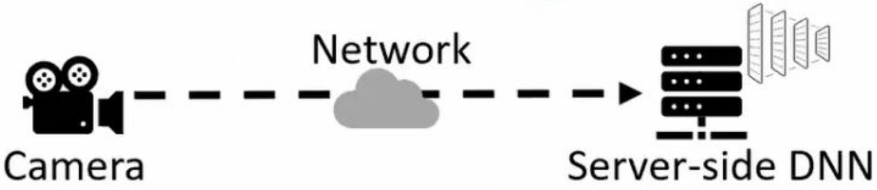
Previous work type 2: Video encoding informed by server DNN

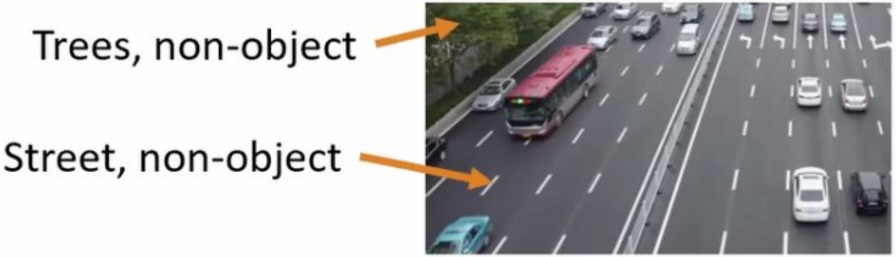


Video codec configuration

Video codec encode

Previous work type 2:
Video encoding informed by server DNN





Video sent by these methods

Ideal

These methods can stream all objects, but…

1. They pay too much bandwidth on non-object pixels

2. They need several minutes to adapt to new content
So they cannot react to real-time video content.

Why real-time content matters?

Why real-time content matters?

Several minutes later

Why real-time content matters?

Several minutes later

0 Accuracy!

Previous video streaming protocols

Type 1. Camera-side heuristics

The camera is lack of compute

↓

Miss objects

↓

Low accuracy

Type 2. Video encoding informed by server DNN

Tune current codec configs base on previous video

↓

1. Waste bandwidth on non-object pixels
2. Cannot react to real-time video content

↓

High bandwidth consumption

Sub-optimal bandwidth-accuracy trade-off!

Design choices

The video streaming protocol should
(1) be completely driven by the server-side DNN feedback and
(2) react to real-time video content

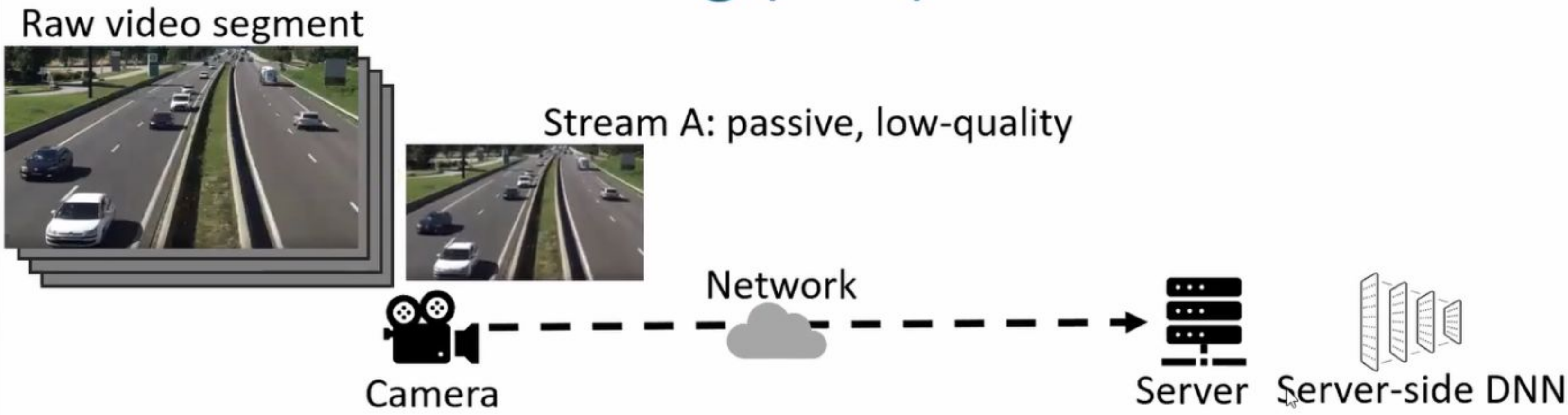
↓

Why this is difficult? A Chicken egg problem:
Camera needs the server-side feedback of **current video to encode current video**

↓

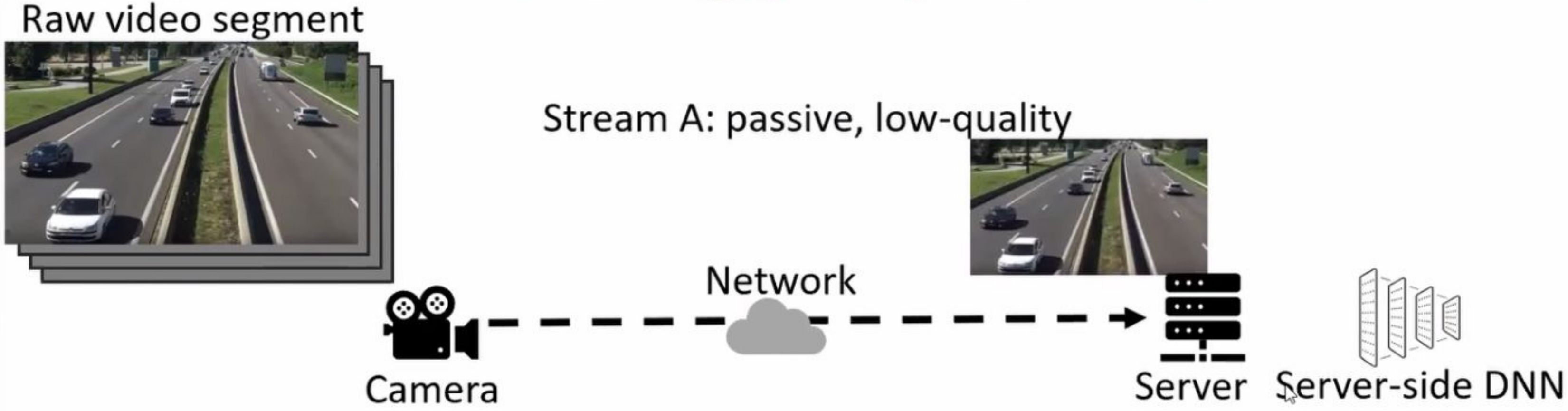
See the video first and iterate on how to encode the video

DNN-Driven Streaming (DDS): iterative workflow



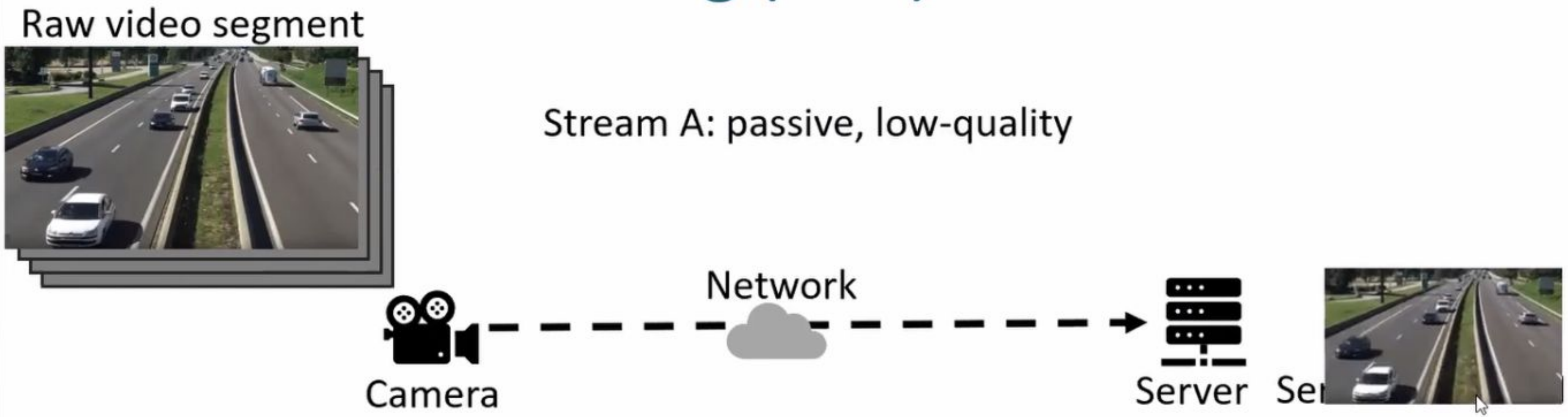
Stream B: high quality, driven by regions generated from server-side DNN

DNN-Driven Streaming (DDS): iterative workflow



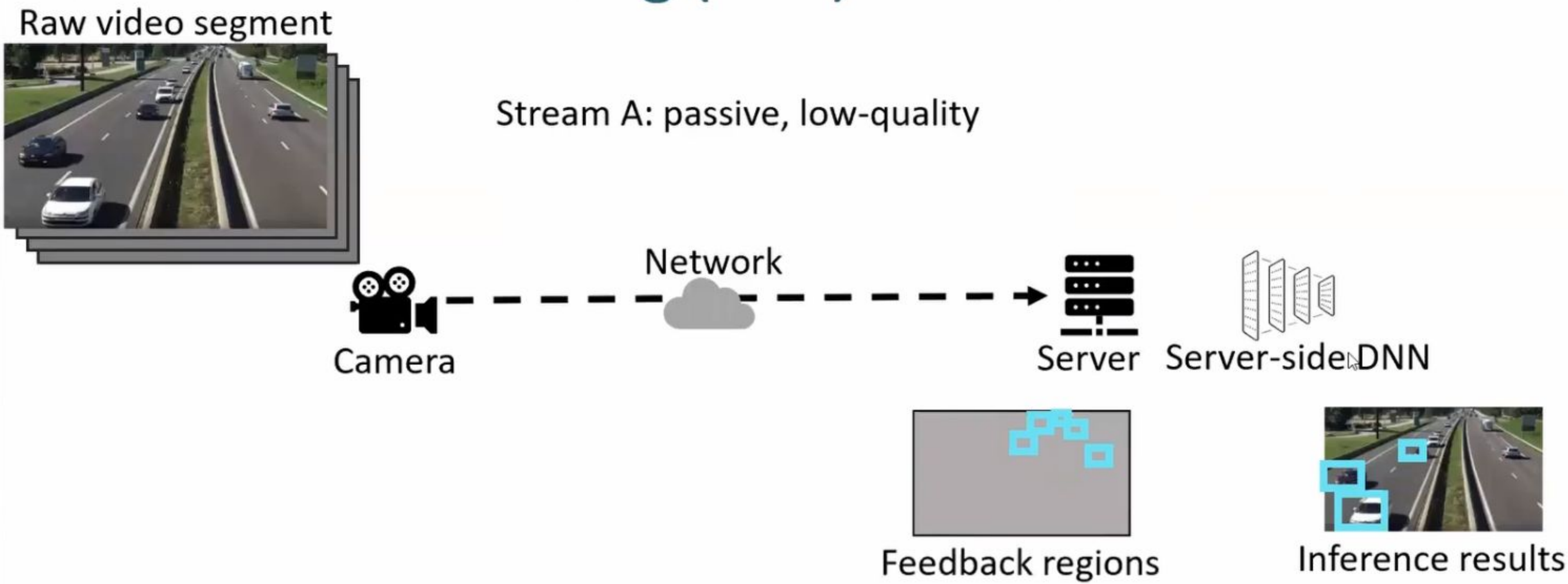
Stream B: high quality, driven by regions generated from server-side DNN

DNN-Driven Streaming (DDS): iterative workflow



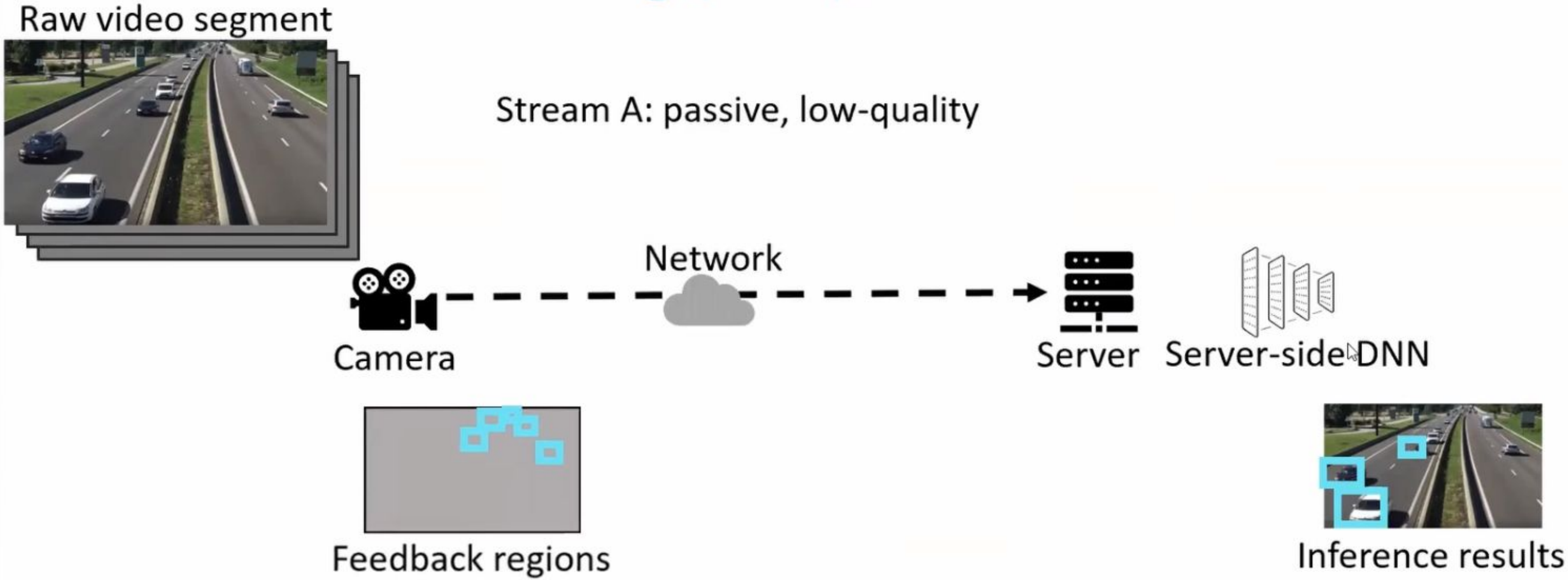
Stream B: high quality, driven by regions generated from server-side DNN

DNN-Driven Streaming (DDS): iterative workflow



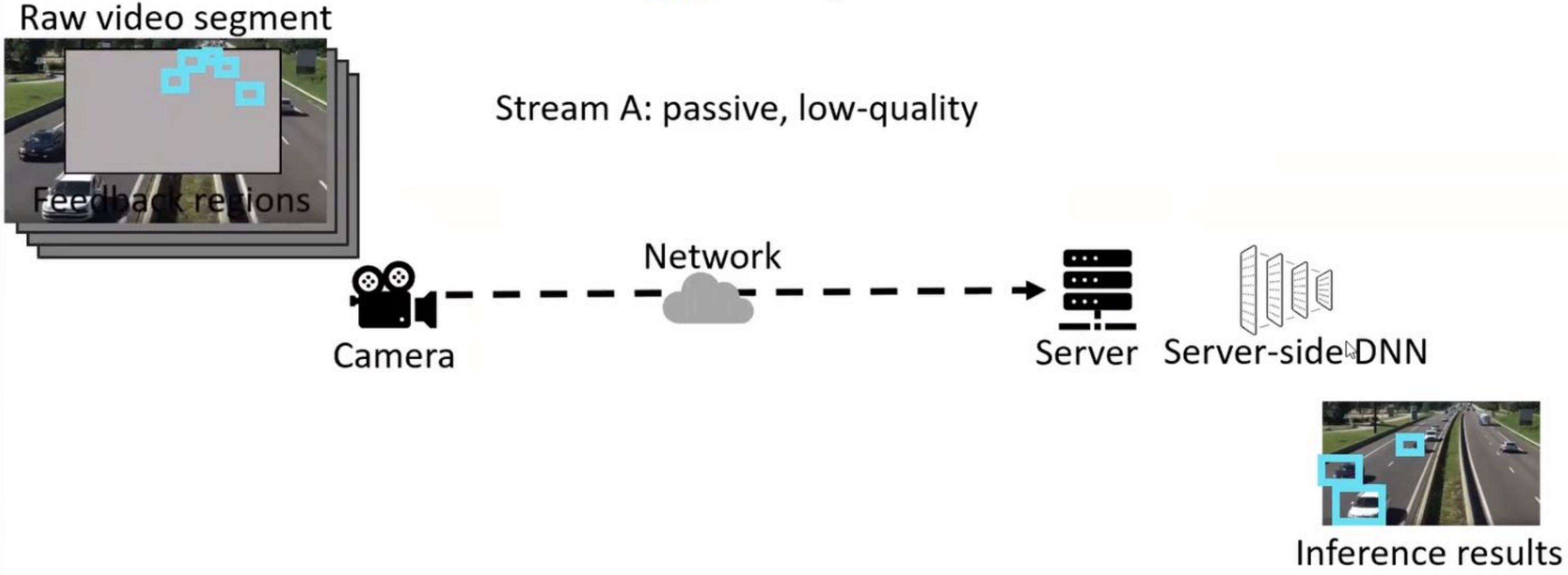
Stream B: high quality, driven by regions generated from server-side DNN

DNN-Driven Streaming (DDS): iterative workflow



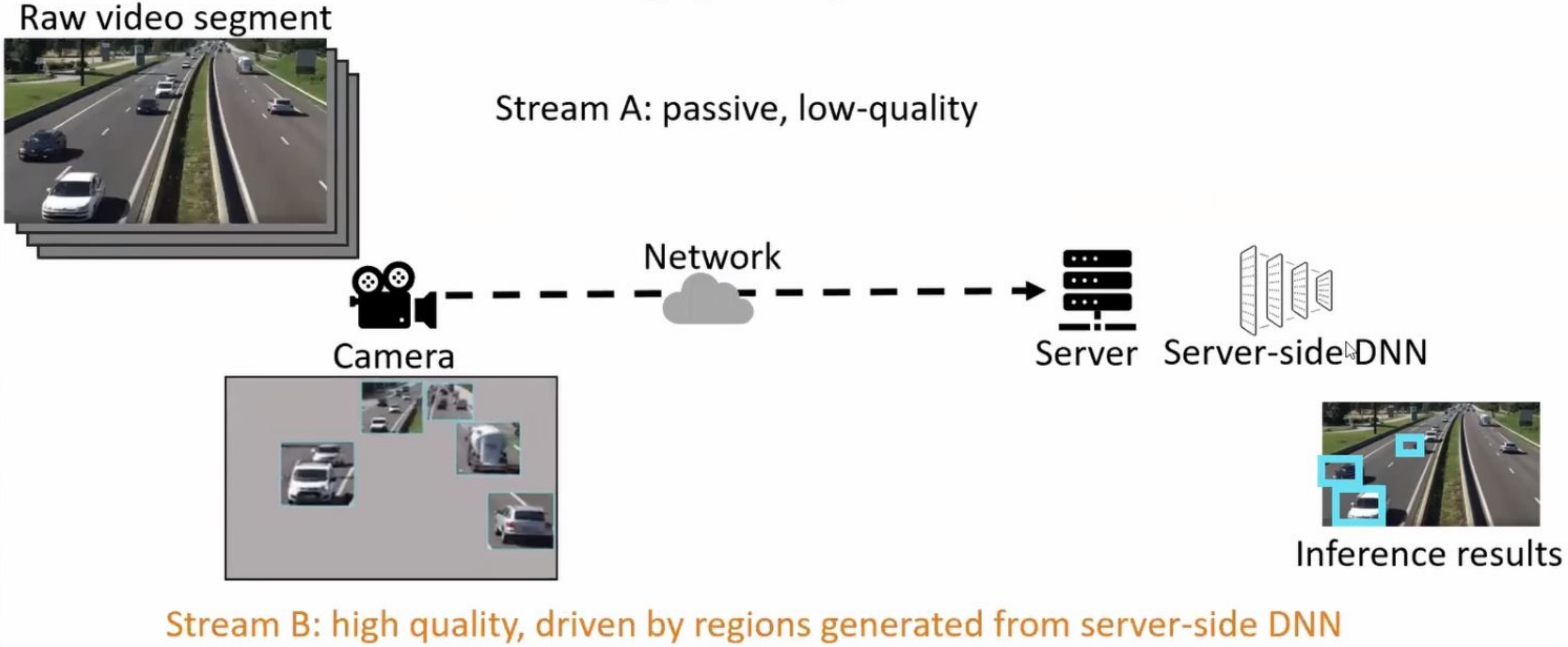
Stream B: high quality, driven by regions generated from server-side DNN

DNN-Driven Streaming (DDS): iterative workflow

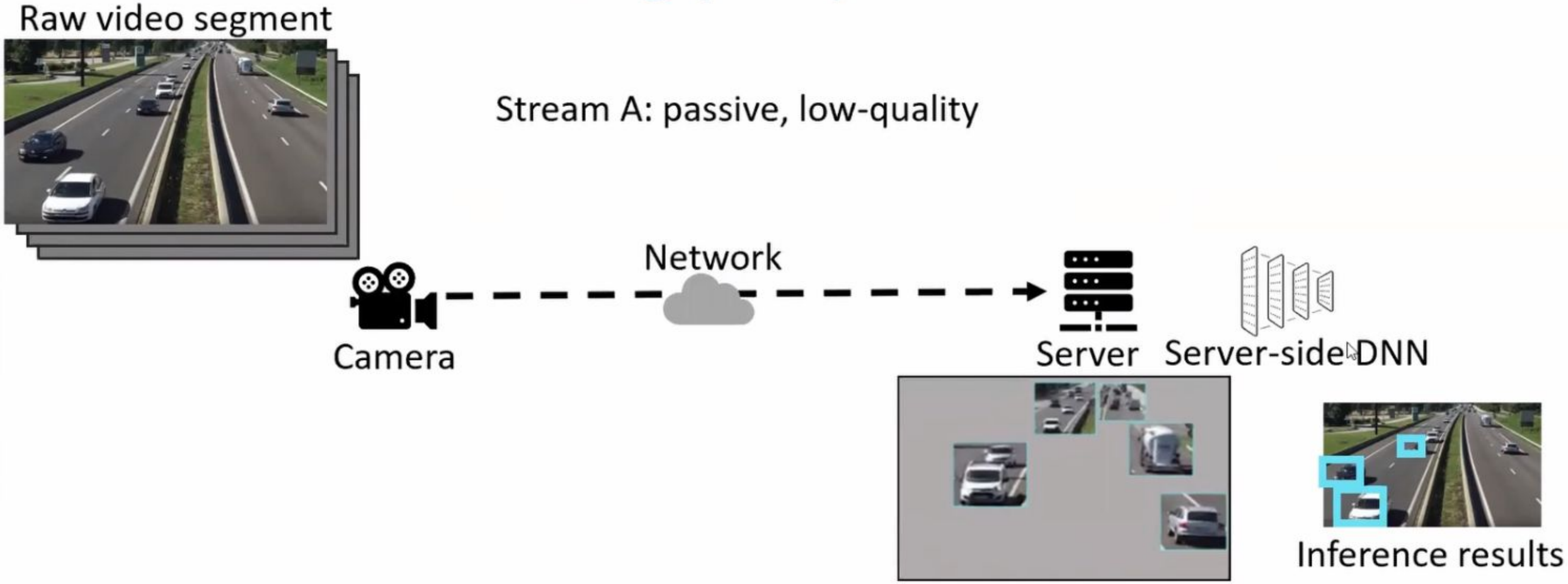


Stream B: high quality, driven by regions generated from server-side DNN

DNN-Driven Streaming (DDS): iterative workflow

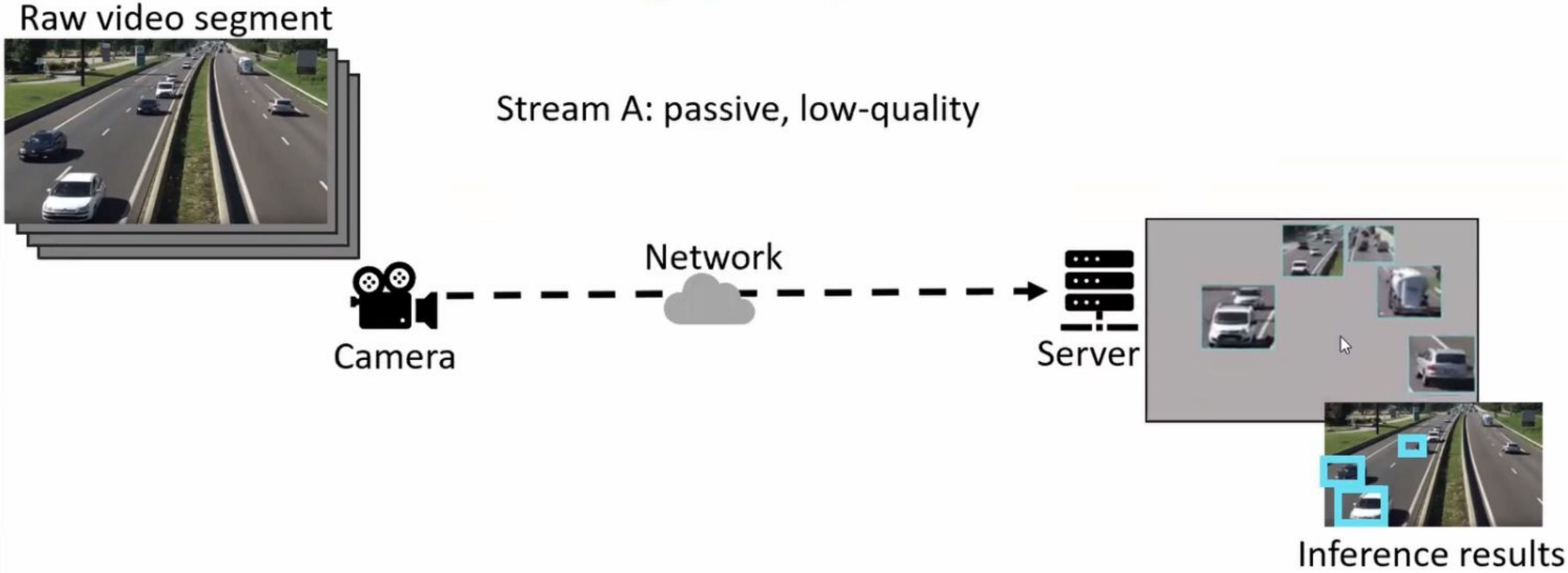


DNN-Driven Streaming (DDS): iterative workflow



Stream B: high quality, driven by regions generated from server-side DNN

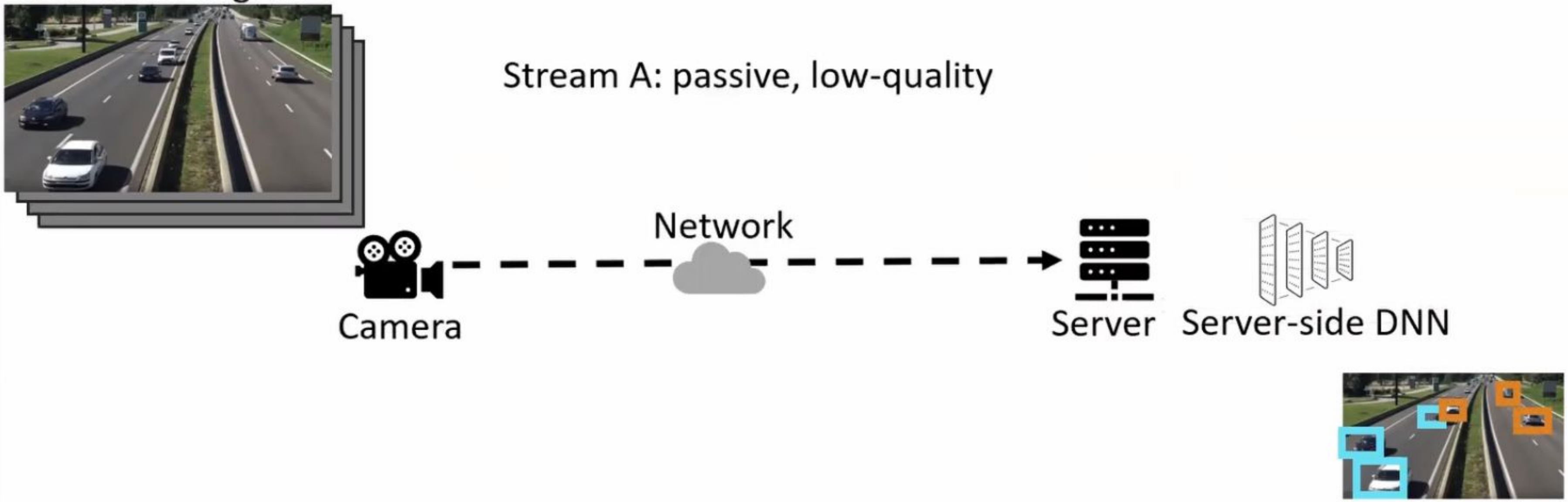
DNN-Driven Streaming (DDS): iterative workflow



Stream B: high quality, driven by regions generated from server-side DNN

DNN-Driven Streaming (DDS): iterative workflow

Raw video segment



Inference results

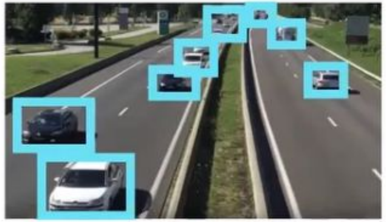
Stream B: high quality, driven by regions generated from server-side DNN

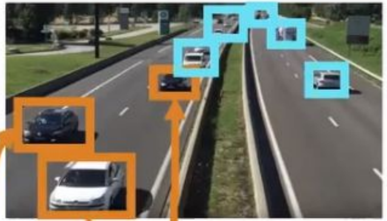
Example: object detection

Generate all regions that may contain objects

Eliminate regions that overlap with high-confidence inference results

Encode remaining regions in codec-friendly manner





Generated from the intermediate output of DNN

DNN is already confident

Faster-RCNN: region proposal network output with high confidence score

YoLo: detection bounding boxes with the sum of the score of all non-background classes over a threshold

Example: semantic segmentation

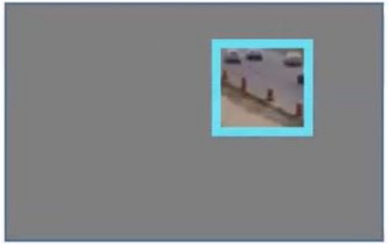
Generate all regions that may contain objects

Eliminate regions that overlap with high-confidence inference results

Encode remaining regions in codec-friendly manner

Label each pixel with the sum of the probability of all non-background classes. “Brighter” the pixel, higher the probability

DNN is already confident



Example: semantic segmentation

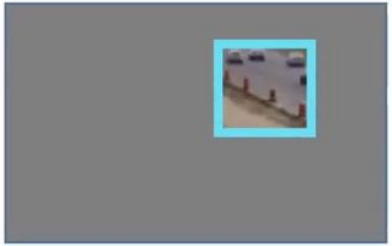
Generate all regions that may contain objects

Label each pixel with the sum of the probability of all non-background classes.

Eliminate regions that overlap with high-confidence inference results

DNN is already confident

Encode remaining regions in codec-friendly manner

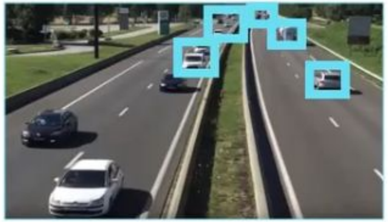


DDS find the regions where the DNN is indecisive.

Comparing DDS with other region-based streaming

High-quality regions sent to the server

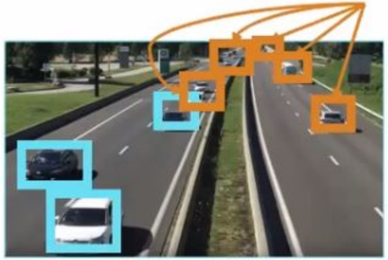
DDS



Segment by segment

(more temporal compression)

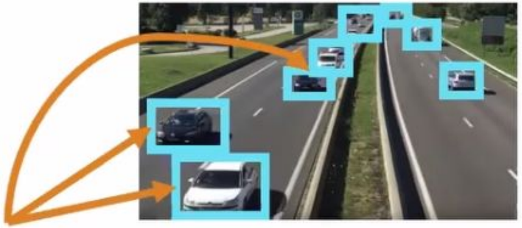
Vigil[1]



Miss small objects

Frame by frame

EAAR[2]

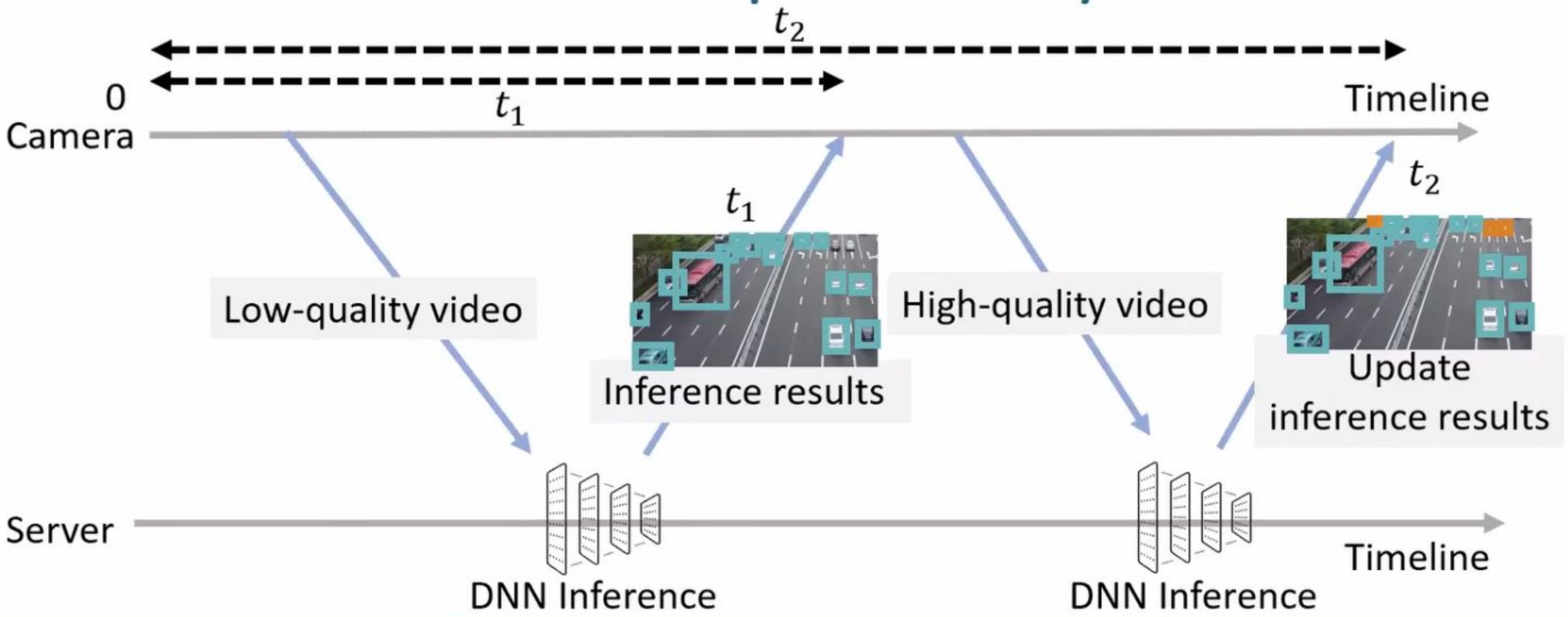


Can be detected in low quality

Frame by frame

[1]: The Design and Implementation of a Wireless Video Surveillance System, *Mobicom '15*
[2]: Edge Assisted Real-time Object Detection for Mobile Augmented Reality, *Mobicom '19*

Reduce the response delay of DDS



While we improve accuracy in the second inference, over 90% results are produced by the first inference and can be returned very quickly.

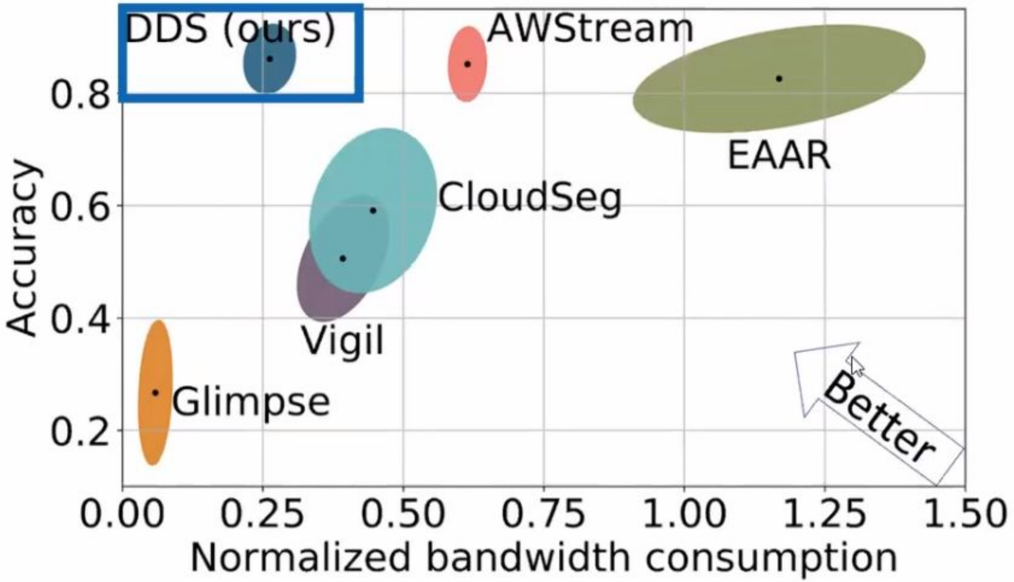
Evaluation setup

- Dataset
  - We evaluate DDS on 49 videos of various camera types and content genres.
  - Traffic camera videos, dacham videos (from YouTube), drone videos (from VisDrone[1]), face videos (from sitcom videos), total length: about 20,000 seconds

- Three vision tasks (and the servers-side DNN models)
  - Object detection: FasterRCNN-ResNet101
  - Semantic segmentation: FCN-ResNet101
  - Face recognition: InsightFace

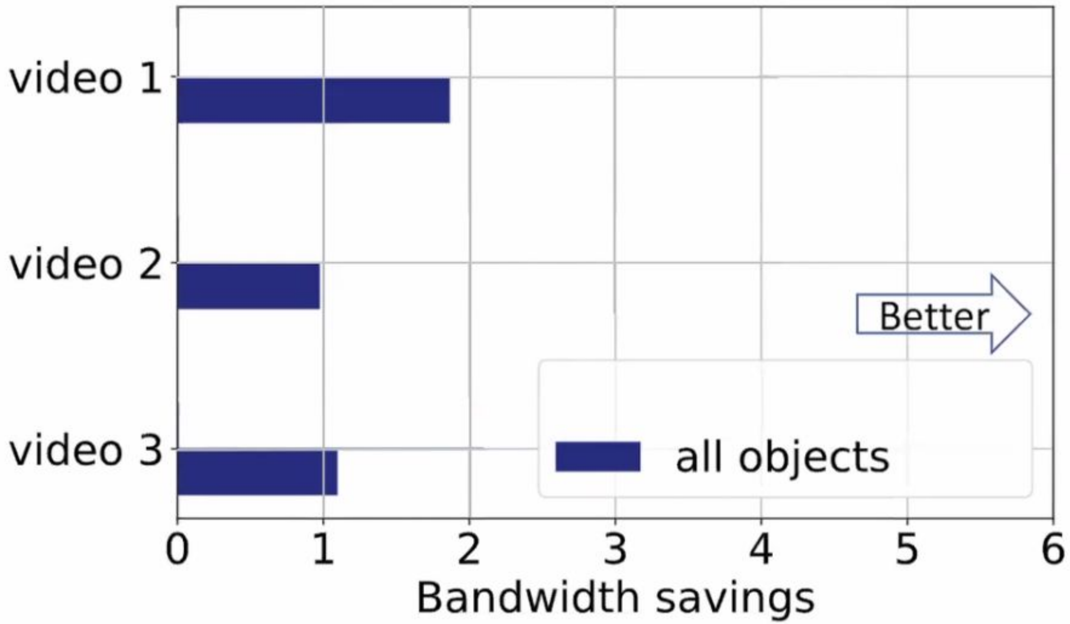
[1]: Vision Meets Drones: Past, Present and Future, *arXiv preprint*

Accuracy vs. bandwidth consumption



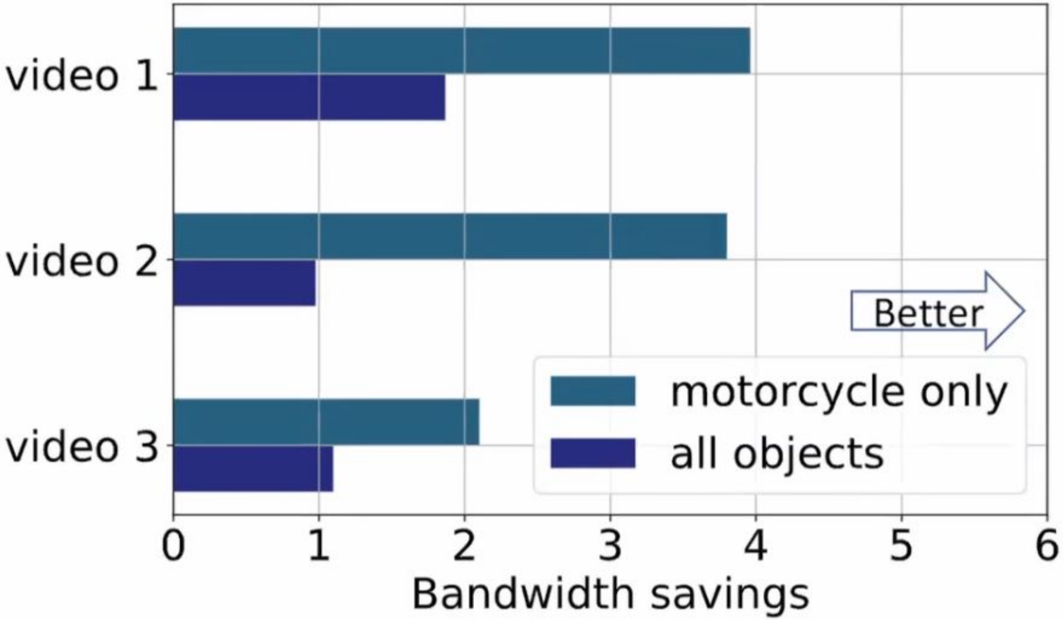
DDS can save upto 59% bandwidth and achieve higher accuracy.

Bandwidth savings vary with videos and queries



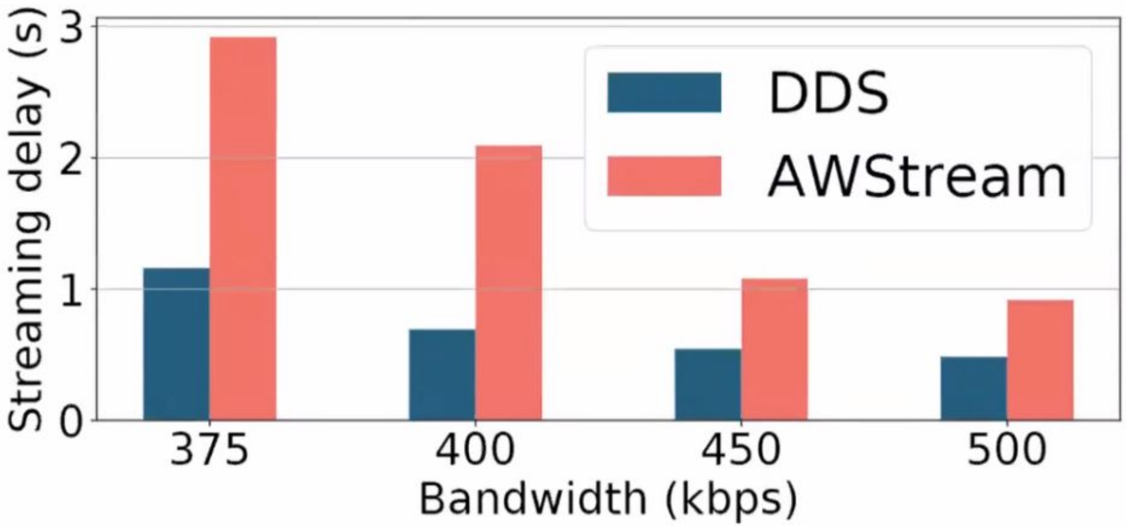
In our dataset, bandwidth savings varies from 1-4x in different videos.

Bandwidth savings vary with videos and queries



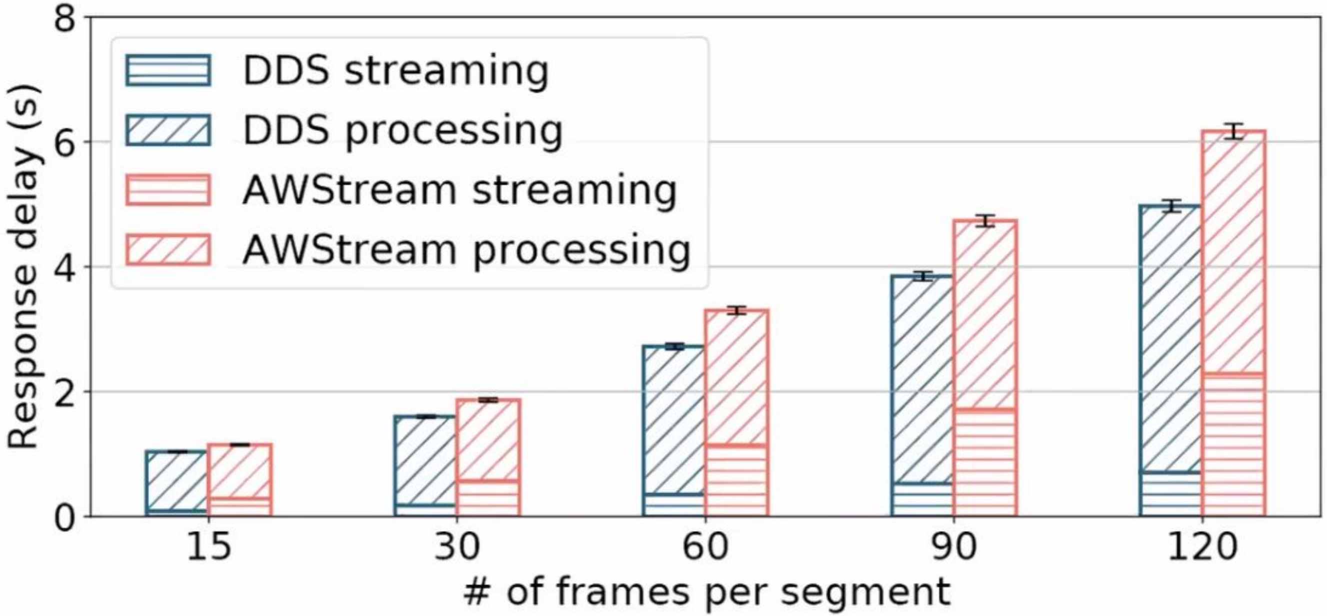
DDS achieves 2-4x more bandwidth savings on motorcycle only.

Streaming delay



DDS saves about 50% streaming under various bandwidth.

End-to-end delay

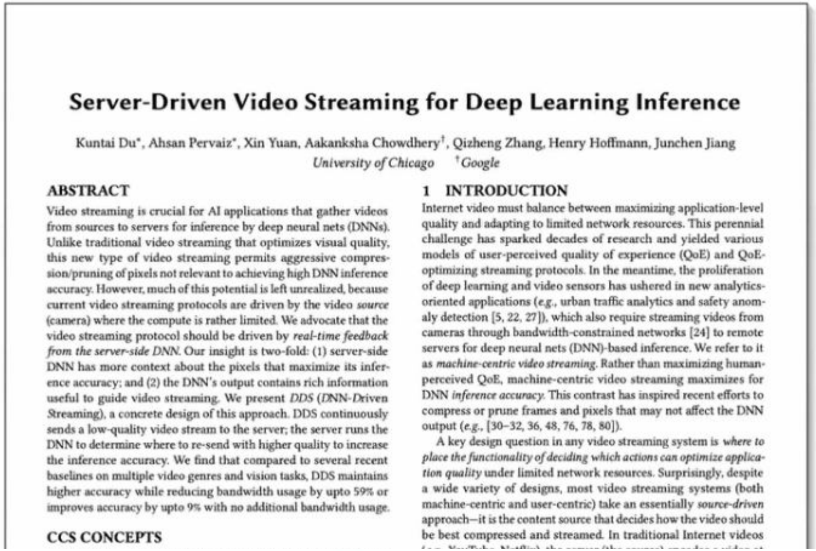


The end-to-end delay of DDS is consistently lower than AWStream.

More details in our paper:

- Cost-delay-accuracy analysis
- Bandwidth adaptation
- More design rationales
- End-to-end delay analysis
- DDS under various network conditions
- Parameter sensitivity analysis
- Fault tolerance

...



Server-Driven Video Streaming for Deep Learning Inference

Kuntai Du*, Ahsan Pervaiz*, Xin Yuan, Aakanksha Chowdhery†, Qizheng Zhang, Henry Hoffmann, Junchen Jiang
University of Chicago †*Google*

ABSTRACT

Video streaming is crucial for AI applications that gather videos from sources to servers for inference by deep neural nets (DNNs). Unlike traditional video streaming that optimizes visual quality, this new type of video streaming permits aggressive compression/pruning of pixels not relevant to achieving high DNN inference accuracy. However, much of this potential is left unrealized, because current video streaming protocols are driven by the video *source* (camera) where the compute is rather limited. We advocate that the video streaming protocol should be driven by *real-time feedback from the server-side DNN*. Our insight is two-fold: (1) server-side DNN has more context about the pixels that maximize its inference accuracy; and (2) the DNN's output contains rich information useful to guide video streaming. We present *DDS* (*DNN-Driven Streaming*), a concrete design of this approach. DDS continuously sends a low-quality video stream to the server; the server runs the DNN to determine where to re-send with higher quality to increase the inference accuracy. We find that compared to several recent baselines on multiple video genres and vision tasks, DDS maintains higher accuracy while reducing bandwidth usage by upto 59% or improves accuracy by upto 9% with no additional bandwidth usage.

CCS CONCEPTS

1 INTRODUCTION

Internet video must balance between maximizing application-level quality and adapting to limited network resources. This perennial challenge has sparked decades of research and yielded various models of user-perceived quality of experience (QoE) and QoE-optimizing streaming protocols. In the meantime, the proliferation of deep learning and video sensors has ushered in new analytics-oriented applications (*e.g.*, urban traffic analytics and safety anomaly detection [5, 22, 27]), which also require streaming videos from cameras through bandwidth-constrained networks [24] to remote servers for deep neural nets (DNN)-based inference. We refer to it as *machine-centric video streaming*. Rather than maximizing human-perceived QoE, machine-centric video streaming maximizes for DNN *inference accuracy*. This contrast has inspired recent efforts to compress or prune frames and pixels that may not affect the DNN output (*e.g.*, [30–32, 36, 48, 76, 78, 80]).

A key design question in any video streaming system is *where to place the functionality of deciding which actions can optimize application quality* under limited network resources. Surprisingly, despite a wide variety of designs, most video streaming systems (both machine-centric and user-centric) take an essentially *source-driven* approach—it is the content source that decides how the video should be best compressed and streamed. In traditional Internet videos

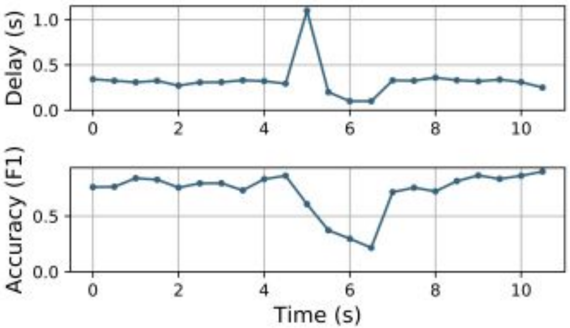


Figure 20: *DDS can handle server disconnection (or server failure) gracefully by falling back to client-side logic*

Conclusion

- Video streaming for analytics enables aggressive video compression
- Only the **server-side DNN** has sufficient info to accurately guide video streaming
- **Our contribution:** DDS: iterative workflow driven by DNN feedback on real-time content
- Results: better bandwidth-accuracy trade-off and lower end-to-end delay
- More resources: https://kuntaidu.github.io/aboutme