Reducto: On-Camera Filtering for Resource-Efficient Real-Time Video Analytics

Yuanqi Li, Arthi Padmanabhan, Pengzhan Zhao, Yufei Wang, Harry Xu, Ravi Netravali

Presented by Hongpeng Guo

Slides are based on SIGCOMM20 presentation from Arthi Padmanabhan.
Video Analytics Trends

• More cameras and video data
• Greater ability to extract information from video
Video Analytics Pipelines

Goals:
- Accuracy target (e.g., 90%)
- Real-time (e.g., 30 fps)
Video Analytics Pipelines

Resource Intensive!

1 video at 1080p: 2 Mbps

Location of all cars?

Number of buses?

Query response

Frames

DNNs (e.g., Faster R-CNN)
Video Analytics Pipelines

Resource Intensive!

Faster RCNN: 6 sec to process 1 sec of video on $6000 GPU

Frames

DNNs (e.g., Faster RCNN)

Query response

Location of all cars?

Number of buses?
Frame Filtering

1) Approximate model (Focus, OSDI ‘18)

Approximate model (e.g., Tiny YOLO): low confidence?

Yes

DNNs

No

Query response
Frame Filtering

1) Approximate model

2) Binary classifier
   (NoScope, VLDB ‘17)

Binary classifier: frame contains car?

Query response

DNNs
Frame Filtering

1) Approximate model
2) Binary classifier
3) Pixel-level differences (Glimpse, SenSys ’15)

Pixel-level differences: frame change above threshold?

DNNs

Query response
Key Question

• Filtering benefits increase closer to the video source

Can we filter frames directly on the camera itself?

• What computational resources are available on existing cameras?
• How do existing approaches fare?
Camera Market Study

WyzeCam
Cores: 1
Speed: 1 GHz
Memory: 128 MB

Axis P33 series
Cores: 1
Speed: 1 GHz
Memory: 512 MB

Ambarella CV22
Cores: 4
Speed: 1.2 GHz
Memory: 4 GB
Accelerator: CVFlow

DNNCam
Cores: 2
Speed: 1.6 GHz
Memory: 8 GB
GPU: NVIDIA TX2

Existing deployments

$19.99

$2418.00
Existing Filtering Approaches

• Approximate models: too slow on camera (Tiny YOLO: 0.6 fps)
• Binary classification – misses 45% of filtering opportunities
Using Frame Differencing Effectively

- Dynamic threshold to deal with rapid changes
- Expand beyond pixel comparison

![Graph showing Best Threshold over time for Edge for Detection and Area for Counting.]

- Pixel: 0.016
  Area: 0.145

- Pixel: 0.003
  Area: 0.830
Reducto Overview

• Challenge #1: Which filtering threshold to use?
• Challenge #2: Which differencing feature to use?

Wimpy cameras can use cheap differencing techniques to filter frames effectively with guidance from a server
Challenge #1: Threshold

- Building table is expensive -> run on server
- Looking up table is cheap -> run on camera

<table>
<thead>
<tr>
<th>Diff</th>
<th>Threshold</th>
<th>% Filtered</th>
<th>Acc</th>
</tr>
</thead>
<tbody>
<tr>
<td>0.03</td>
<td>...</td>
<td>...</td>
<td>...</td>
</tr>
<tr>
<td>0.04</td>
<td>0.01</td>
<td>0.15</td>
<td>0.98</td>
</tr>
<tr>
<td></td>
<td>0.04</td>
<td>0.35</td>
<td>0.92</td>
</tr>
<tr>
<td></td>
<td>0.07</td>
<td>0.60</td>
<td>0.88</td>
</tr>
<tr>
<td></td>
<td>...</td>
<td>...</td>
<td>...</td>
</tr>
<tr>
<td>0.06</td>
<td>0.01</td>
<td>0.12</td>
<td>0.94</td>
</tr>
<tr>
<td></td>
<td>0.04</td>
<td>0.30</td>
<td>0.89</td>
</tr>
<tr>
<td></td>
<td>0.07</td>
<td>0.55</td>
<td>0.84</td>
</tr>
<tr>
<td></td>
<td>...</td>
<td>...</td>
<td>...</td>
</tr>
<tr>
<td>0.07</td>
<td>...</td>
<td>...</td>
<td>...</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Diff</th>
<th>Threshold</th>
</tr>
</thead>
<tbody>
<tr>
<td>0.03 – 0.05</td>
<td>0.04</td>
</tr>
<tr>
<td>0.05 – 0.07</td>
<td>0.01</td>
</tr>
<tr>
<td>...</td>
<td>...</td>
</tr>
</tbody>
</table>
Challenge #2: Differencing Feature

- Calculating best feature is expensive -> run on server

- Extract all differencing features

- % frames filtered while meeting accuracy

- Best feature = feature that filters most frames
Challenge #2: Differencing Feature

• Best feature changes between query types but not between videos
Putting It Together

Offline:

Best feature (e.g., Edge)

Profiler

Hash table generator

<table>
<thead>
<tr>
<th>Diff</th>
<th>Thresh</th>
</tr>
</thead>
<tbody>
<tr>
<td>…</td>
<td>…</td>
</tr>
</tbody>
</table>
Putting It Together

Online:

Extract diff → Calculate threshold → Diff > threshold? → Yes → ML pipeline → Query response
Evaluation: Methodology

• Three queries: detection, counting, tagging
• 8 traffic videos: 25 10-min clips each
• DNN on server: YOLOv3
• Camera: Raspberry Pi Zero or VM with matching resources

Sample screenshots
Evaluating Reducto

• Reducto vs. offline optimal filtering

• Speed on Camera

• Compute and bandwidth savings
Reducto vs. Offline Optimal Filtering

Reducto filters 36-51% of frames while meeting accuracy target
Speed on Camera

• 47.8 fps on Raspberry Pi Zero

Extract frame features
99.7 fps

Calculate frame difference
129.5 fps

Hash table lookups
318.6 fps
## Resource Savings

### Network

<table>
<thead>
<tr>
<th></th>
<th>Fraction Filtered (%)</th>
<th>Bandwidth Savings (%)</th>
</tr>
</thead>
<tbody>
<tr>
<td>Baseline</td>
<td>0.00</td>
<td>0.00</td>
</tr>
<tr>
<td>Reducto</td>
<td>53.42</td>
<td>22.30</td>
</tr>
<tr>
<td>Offline</td>
<td>72.80</td>
<td>39.33</td>
</tr>
</tbody>
</table>

- Reducto saves average of 22% bandwidth

### Compute

<table>
<thead>
<tr>
<th></th>
<th>Backend processing (fps)</th>
</tr>
</thead>
<tbody>
<tr>
<td>Baseline</td>
<td>41.13</td>
</tr>
<tr>
<td>Reducto</td>
<td>86.21</td>
</tr>
<tr>
<td>Offline</td>
<td>140.01</td>
</tr>
</tbody>
</table>

- Reducto doubles backend processing speed
Resource Savings

Network
• Reducto saves average of 22% bandwidth

Compute
• Reducto doubles backend processing speed

End-to-End Latency:
Reduces median response time by 22-26% (within 13% of offline optimal)
Comments

• Pros:
  • Insightful observation & significant performance.
  • Very good writings. Explain the design choices well.

• Cons:
  • This work is based on “frame filter” types of work. The idea itself is not very novel.

• Takeaway:
  • Use comprehensive data and survey to support motivation & observations.
  • Good Explanation for design choice & observations makes good paper.
Putting It Together

Online:

Extract diff → Calculate threshold → Diff > threshold?

Updated hash table

Hash table generator

ML pipeline → Query response

Yes
Putting It Together

Online:

Extract diff → Calculate threshold → Diff > threshold?

ML pipeline → Query response

Yes