

CS 8803-MDS

# Human-in-the-loop Data Analytics

Lecture 6

09/11/23

# Logistics

Project group signup due today!

Course staff will follow up with everyone who doesn't have a group

Project proposal assignment (due 09/25)

Extra credit for discussing your outline during OH

# This Class

[Blazelt: Optimizing Declarative Aggregation and Limit Queries for Neural Network-Based Video Analytics](#)

- Authors: Maansi, Meghan
- Archaeologist: David
- Practitioner: Chitti



[General Information](#) ▾

[Conference Program](#) ▾

[Call for Contributions](#) ▾

[Dates & Guidelines](#) ▾

[Sponsorship](#) ▾

## 45<sup>th</sup> International Conference on Very Large Data Bases

Los Angeles, California - August 26-30, 2019

A photograph of the Hollywood sign, a landmark in Los Angeles, California. The sign is made of large white letters and is set against a clear blue sky. The sign is located on a hillside with sparse vegetation. The letters are arranged in a slightly curved line across the hillside. The background shows a clear blue sky and some utility poles on the right side of the hill.

# **Blazelt: Optimizing Declarative Aggregation and Limit Queries for Neural Network-Based Video Analytics**

Original Authors: Daniel Kang, Peter Bailis, Matei Zaharia

Presented By: MAANSI JAIN and MEGHAN KULKARNI

# Video Analytics

# Video Analytics

1. Cameras are cheap and widely deployed

# Video Analytics

1. Cameras are cheap and widely deployed
2. Deep neural networks (DNNs) can now automatically produce video annotations



# Video Analytics

1. Cameras are cheap and widely deployed
2. Deep neural networks (DNNs) can now automatically produce video annotations
  - a. **Challenges:**
    - i. require complex, imperative programming across many low-level libraries
    - ii. performing object detection on every frame of video is cost prohibitive at scale

# Video Analytics

1. Cameras are cheap and widely deployed
2. Deep neural networks (DNNs) can now automatically produce video annotations
  - a. **Challenges:**
    - i. require complex, imperative programming across many low-level libraries
    - ii. performing object detection on every frame of video is cost prohibitive at scale

# Current Optimizations

- Focused on filtering via approximate predicates
  - ◆ NOSCOPE and TAHOMA

# Current Optimizations

- Focused on filtering via approximate predicates
  - ◆ NOSCOPE and TAHOMA
  
- Do not handle **aggregate** and **limit** queries.

# Current Optimizations

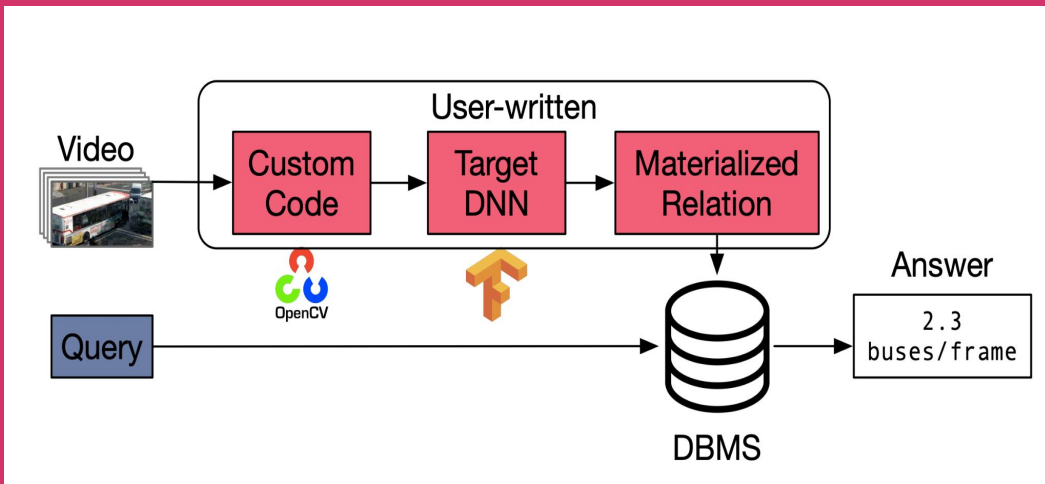
- Focused on filtering via approximate predicates
  - ◆ NOSCOPE and TAHOMA
- Do not handle **aggregate** and **limit** queries.
- Still require non-expert users to write complex code to deploy.

# Blazelt

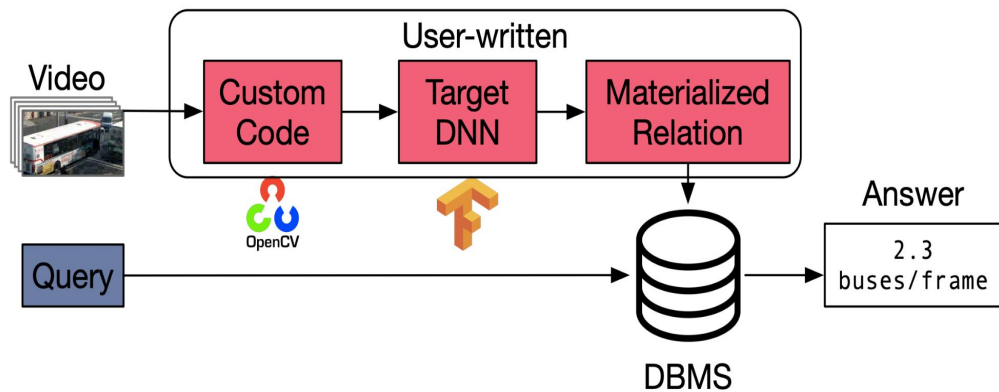
**A video analytics system with a declarative query language and two novel optimizations for aggregation and limit queries.**

---

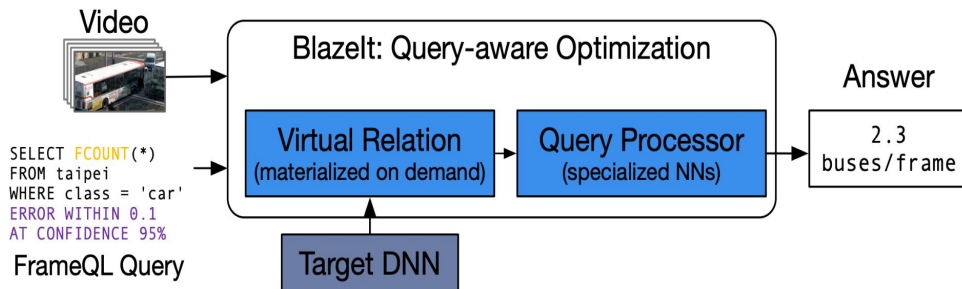
# Naive



# Naive



# Optimized





# System Overview

Configuration, TMAS, Proxy Model and specialized NNs

# Configuration

- Target object detection method       $OD(\text{frame}) \rightarrow \text{Set}\langle \text{Tuple}\langle \text{class}, \text{box} \rangle \rangle$
- Entity resolution
  - ◆ Takes neary frames and boxes, returns true if referring to the same object



# Configuration

- Target object detection method       $OD(\text{frame}) \rightarrow \text{Set}\langle \text{Tuple}\langle \text{class}, \text{box} \rangle \rangle$
- Entity resolution
  - ◆ Takes nearby frames and boxes, returns true if referring to the same object
- Both can be changed and personalized to the query at hand



# Configuration

- Target object detection method       $OD(\text{frame}) \rightarrow \text{Set}\langle \text{Tuple}\langle \text{class}, \text{box} \rangle \rangle$
- Entity resolution
  - ◆ Takes nearby frames and boxes, returns true if referring to the same object
- Both can be changed and personalized to the query at hand
- UDFs
  - ◆ Functions that accept a timestamp, mask, and rectangular set of pixels
  - ◆ Used to answer more complex queries
    - determine color, object size/location, etc.



# Target-model annotated set (TMAS)

- At ingestion time, using NN, object detection is performed on a small sample of frames
  - ◆ metadata is stored as FrameQL tuples (**TMAS**)



# Target-model annotated set (TMAS)

- At ingestion time, using NN, object detection is performed on a small sample of frames
  - ◆ metadata is stored as FrameQL tuples (TMAS)
- Object detection is performed only once at data ingestion



# Target-model annotated set (TMAS)

- At ingestion time, using NN, object detection is performed on a small sample of frames
  - ◆ metadata is stored as FrameQL tuples (TMAS)
- Object detection is performed only once at data ingestion
- TMAS is split between training data and held out data



# Proxy models and specialized NNs

- Proxy models are used to speed up query execution while providing a guaranteed accuracy
  - ◆ Blazelt can infer proxy models/filters from query predicates (need to be trained from data)





# Proxy models and specialized NNs

- Proxy models are used to speed up query execution while providing a guaranteed accuracy
  - ◆ Blazelt can infer proxy models/filters from query predicates (need to be trained from data)
- Use specialized NNs (mini ResNet) as proxy models
  - ◆ Specialized NNs run faster than their counterpart NNs



# Proxy models and specialized NNs

- Proxy models are used to speed up query execution while providing a guaranteed accuracy
  - ◆ Blazelt can infer proxy models/filters from query predicates (need to be trained from data)
- Use specialized NNs (mini ResNet) as proxy models
  - ◆ Specialized NNs run faster than their counterpart NNs
- Blazelt figures out when specialized NN needs to be used



# Limitations

## TMAS

BLAZEIT requires the object detection method (a bottleneck) to be run over a portion of the data for training specialized NNs and filters as a preprocessing step.

# Limitations

## TMAS

BLAZEIT requires the object detection method (a bottleneck) to be run over a portion of the data for training specialized NNs and filters as a preprocessing step.

## Model Drift

When the data distribution may change, BLAZEIT will still provide accuracy guarantees but performance may be reduced.

Mitigate with labels on new data, might require continuous retraining

# Limitations

## TMAS

BLAZEIT requires the object detection method (a bottleneck) to be run over a portion of the data for training specialized NNs and filters as a preprocessing step.

## Model Drift

When the data distribution may change, BLAZEIT will still provide accuracy guarantees but performance may be reduced.

Mitigate with labels on new data, might require continuous retraining

## Object Detection

BLAZEIT depends on the target object detection method and does not support object classes beyond what the method returns.

Mitigate with UDFs

# FRAMEQL: EXPRESSING COMPLEX SPATIOTEMPORAL VISUAL QUERIES

FRAMEQL's syntax, data model, query format, and examples  
Comparison to prior languages

# FrameQL Syntax

```
SELECT * | expression [, ...]
  FROM table_name
  [ WHERE condition ]
  [ GROUP BY expression [, ...] ]
  [ HAVING condition [, ...] ]
  [ LIMIT count ]
  [ GAP count ]
  [ ERROR WITHIN tol AT CONFIDENCE conf ]
```

Syntactic element	Description
FCOUNT	Frame-averaged count (equivalent to time-averaged count), i.e., $\text{COUNT}(\ast) / \text{MAX}(\text{timestamp})$
ERROR WITHIN	Absolute error tolerance
FPR WITHIN	Allowed false positive rate
FNR WITHIN	Allowed false negative rate
CONFIDENCE	Confidence level
GAP	Minimum distance between returned frames



# FrameQL Data Model

- Uses videos as virtual relations
- Each tuple = one obj in a frame

Field	Type	Description
timestamp	float	Time stamp
class	string	Object class (e.g., bus, car)
mask	(float, float)*	Polygon containing the object of interest, typically a rectangle
trackid	int	Unique identifier for a continuous time segment when the object is visible
content	float*	Content of pixels in mask
features	float*	The feature vector output by the object detection method.





# FrameQL Data Model

- Uses videos as virtual relations
- Each tuple = one obj in a frame
- Automatically populates

- ◆ `mask`, `class`, and `features` from object detection method
- ◆ `trackid` from entity resolution method
- ◆ `timestamp` and `content` from video's metadata

Field	Type	Description
<code>timestamp</code>	<code>float</code>	Time stamp
<code>class</code>	<code>string</code>	Object class (e.g., bus, car)
<code>mask</code>	<code>(float, float)*</code>	Polygon containing the object of interest, typically a rectangle
<code>trackid</code>	<code>int</code>	Unique identifier for a continuous time segment when the object is visible
<code>content</code>	<code>float*</code>	Content of pixels in mask
<code>features</code>	<code>float*</code>	The feature vector output by the object detection method.



# FrameQL Data Model

→ Uses videos as virtual relations

→ Each tuple = one obj in a frame

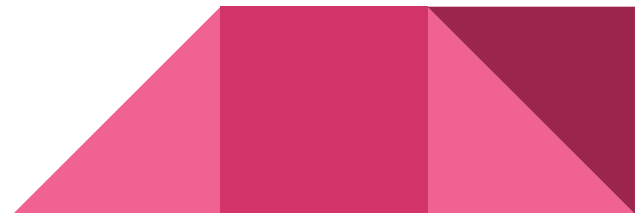
→ Automatically populates

- ◆ `mask`, `class`, and `features` from object detection method
- ◆ `trackid` from entity resolution method
- ◆ `timestamp` and `content` from video's metadata

→ Ability to overwrite

- ◆ Object detection methods
- ◆ Entity resolution methods

Field	Type	Description
<code>timestamp</code>	<code>float</code>	Time stamp
<code>class</code>	<code>string</code>	Object class (e.g., bus, car)
<code>mask</code>	<code>(float, float)*</code>	Polygon containing the object of interest, typically a rectangle
<code>trackid</code>	<code>int</code>	Unique identifier for a continuous time segment when the object is visible
<code>content</code>	<code>float*</code>	Content of pixels in mask
<code>features</code>	<code>float*</code>	The feature vector output by the object detection method.



# FrameQL Query Format

→ GAP

- ◆ Set GAP to an integer
- ◆ Ensures the returned frames are at least GAP frames apart when user selects timestamps

Syntactic element	Description
FCOUNT	Frame-averaged count (equivalent to time-averaged count), i.e., $\text{COUNT}(\ast) / \text{MAX}(\text{timestamp})$
ERROR WITHIN	Absolute error tolerance
FPR WITHIN	Allowed false positive rate
FNR WITHIN	Allowed false negative rate
CONFIDENCE	Confidence level
GAP	Minimum distance between returned frames



# FrameQL Query Format

## → GAP

- ◆ Set `GAP` to an integer
- ◆ Ensures the returned frames are at least `GAP` frames apart when user selects timestamps

## → ERROR WITHIN

- ◆ Specify error bounds (maximum absolute error, false positive error, and false negative error) and confidence levels
- ◆ Supplies fast responses to exploratory queries and may tolerate some error

Syntactic element	Description
FCOUNT	Frame-averaged count (equivalent to time-averaged count), i.e., $\text{COUNT}(\ast) / \text{MAX}(\text{timestamp})$
ERROR WITHIN	Absolute error tolerance
FPR WITHIN	Allowed false positive rate
FNR WITHIN	Allowed false negative rate
CONFIDENCE	Confidence level
GAP	Minimum distance between returned frames

# FrameQL Query Format

## → GAP

- ◆ Set `GAP` to an integer
- ◆ Ensures the returned frames are at least `GAP` frames apart when user selects timestamps

## → ERROR WITHIN

- ◆ Specify error bounds (maximum absolute error, false positive error, and false negative error) and confidence levels
- ◆ Supplies fast responses to exploratory queries and may tolerate some error

## → FCOUNT

- ◆ Shortcut to return a frame averaged count
- ◆ Normalized way to compute errors

Syntactic element	Description
FCOUNT	Frame-averaged count (equivalent to time-averaged count), i.e., $\text{COUNT}(\ast) / \text{MAX}(\text{timestamp})$
ERROR WITHIN	Absolute error tolerance
FPR WITHIN	Allowed false positive rate
FNR WITHIN	Allowed false negative rate
CONFIDENCE	Confidence level
GAP	Minimum distance between returned frames

# Comparison to prior languages

- Proposed schemas and assume relation is already populated
  - ◆ Data created by humans



# Comparison to prior languages

- Proposed schemas and assume relation is already populated
  - ◆ Data created by humans
- Blazelt automatically populates FrameQL's relation
- FrameQL's schema is virtual
  - ◆ Rows only populated when needed for specific queries
  - ◆ Allows for variety of optimizations



# Query Optimization

Optimizing Aggregates, Optimizing Limit Queries



# Optimizing Aggregate Queries

**Data:** TMAS, unseen video,  
 $uerr \leftarrow$  user's requested error rate,  
 $conf \leftarrow$  user's confidence level  
**Result:** Estimate of requested quantity  
**if** *training data has instances of object* **then**  
    train specialized NN on TMAS;  
     $err \leftarrow$  specialized NN error rate;  
     $\tau \leftarrow$  average of specialized NN over unseen video;  
    **if**  $P(err < uerr) < conf$  **then**  
        return  $\tau$ ;  
    **else**  
         $\hat{m} \leftarrow \hat{m} = m + c \cdot (a - \alpha)$   
        return  $\hat{m}$ ;  
    **end**  
**else**  
    Return result of random sampling.;  
**end**

# Optimizing Aggregate Queries

1. Process TMAS training data

# Optimizing Aggregate Queries

1. Process TMAS training data
2. Is there less than 1% of training data that has instances of the object

# Optimizing Aggregate Queries

1. Process TMAS training data
2. Is there less than 1% of training data has instances of the object
  - a. Not enough data (i.e.  $\leq 1\%$ )?
    - i. Default to random sampling (an adaptive sampling algorithm incorporating info from query and TMAS)

# Optimizing Aggregate Queries

1. Process TMAS training data
2. Is there less than 1% of training data has instances of the object
  - a. Not enough data (i.e.  $\leq 1\%$ )?
    - i. Default to random sampling (an adaptive sampling algorithm incorporating info from query and TMAS)
  - b. Enough data (i.e.  $>1\%$ )?
    - i. Train the specialized NN by selecting the number of classes equal to the highest count that is at least 1% of the video plus one
    - ii. Estimate the error rate on TMAS held out data

# Optimizing Aggregate Queries

1. Process TMAS training data
2. Is there less than 1% of training data has instances of the object
  - a. Not enough data (i.e.  $\leq 1\%$ )?
    - i. Default to random sampling (an adaptive sampling algorithm incorporating info from query and TMAS)
  - b. Enough data (i.e.  $>1\%$ )?
    - i. Train the specialized NN by selecting the number of classes equal to the highest count that is at least 1% of the video plus one
    - ii. Estimate the error rate on TMAS held out data
      1. Error smaller than specified error at confidence level?
        - a. Trained specialized NN is accurate enough and can be executed on unseen data and returns an answer

# Optimizing Aggregate Queries

1. Process TMAS training data
2. Is there less than 1% of training data has instances of the object
  - a. Not enough data (i.e.  $\leq 1\%$ )?
    - i. Default to random sampling (an adaptive sampling algorithm incorporating info from query and TMAS)
  - b. Enough data (i.e.  $>1\%$ )?
    - i. Train the specialized NN by selecting the number of classes equal to the highest count that is at least 1% of the video plus one
    - ii. Estimate the error rate on TMAS held out data
      1. Error smaller than specified error at confidence level?
        - a. Trained specialized NN is accurate enough and can be executed on unseen data and returns an answer
      2. Error larger?
        - a. Trained specialized NN is not accurate enough and instead used as a control variate to approximate the statistic

# Optimizing Aggregate Queries

- If aggregation query has predicates (ex: count num of large red buses)
  - ◆ Similar to original algorithm
  - ◆ First applies predicates to training data
    - Not enough data?
      - Instead generate a specialized NN to count the most selective set of predicates that contains enough data (ex: num of large buses OR num or red buses)
      - No training data still?
        - ◆ Standard sampling



# Optimizing Limit Queries

→ The user is interested in finding a limited number of events, typically for manual inspection.




# Optimizing Limit Queries


- The user is interested in finding a limited number of events, typically for manual inspection.
- Supports limit queries searching for at least N of an object class (e.g., at least one bus and at least five cars).




# Optimizing Limit Queries

- The user is interested in finding a limited number of events, typically for manual inspection.
  - Supports limit queries searching for at least N of an object class (e.g., at least one bus and at least five cars).
  - We use specialized NNs to bias which frames to sample:
    - ◆ If there are no instances of the query in the training set, BLAZEIT will default to performing the object detection method over every frame and applying applicable filters as in prior work (random sampling is also possible).
    - ◆ If there are examples, BLAZEIT will train a specialized NN to recognize frames that satisfy the query.
    - ◆ BLAZEIT rank orders the unseen data by the confidence from the specialized NN.
    - ◆ BLAZEIT will perform object detection in the rank order until the requested number of events is found.
- 


# Optimizing Limit Queries

- The user is interested in finding a limited number of events, typically for manual inspection.
  - Supports limit queries searching for at least N of an object class (e.g., at least one bus and at least five cars).
  - We use specialized NNs to bias which frames to sample:
    - ◆ If there are no instances of the query in the training set, BLAZEIT will default to performing the object detection method over every frame and applying applicable filters as in prior work (random sampling is also possible).
    - ◆ **If there are examples, BLAZEIT will train a specialized NN to recognize frames that satisfy the query.**
    - ◆ BLAZEIT rank orders the unseen data by the confidence from the specialized NN.
    - ◆ BLAZEIT will perform object detection in the rank order until the requested number of events is found.
- 

# Optimizing Limit Queries

- The user is interested in finding a limited number of events, typically for manual inspection.
  - Supports limit queries searching for at least N of an object class (e.g., at least one bus and at least five cars).
  - We use specialized NNs to bias which frames to sample:
    - ◆ If there are no instances of the query in the training set, BLAZEIT will default to performing the object detection method over every frame and applying applicable filters as in prior work (random sampling is also possible).
    - ◆ If there are examples, BLAZEIT will train a specialized NN to recognize frames that satisfy the query.
    - ◆ **BLAZEIT rank orders the unseen data by the confidence from the specialized NN.**
    - ◆ BLAZEIT will perform object detection in the rank order until the requested number of events is found.
- 

# Optimizing Limit Queries

- The user is interested in finding a limited number of events, typically for manual inspection.
  - Supports limit queries searching for at least N of an object class (e.g., at least one bus and at least five cars).
  - We use specialized NNs to bias which frames to sample:
    - ◆ If there are no instances of the query in the training set, BLAZEIT will default to performing the object detection method over every frame and applying applicable filters as in prior work (random sampling is also possible).
    - ◆ If there are examples, BLAZEIT will train a specialized NN to recognize frames that satisfy the query.
    - ◆ BLAZEIT rank orders the unseen data by the confidence from the specialized NN.
    - ◆ **BLAZEIT will perform object detection in the rank order until the requested number of events is found.**
- 

# Physical Operator and Selection

- We train the specialized NN to predict counts rather than as a binary classifier of the frames that satisfy the predicate or not
  - ◆ Alleviates class imbalance issue
  - ◆ Allows the trained specialized NN to be reused for other queries such as aggregation.



# Physical Operator and Selection

- We train the specialized NN to predict counts rather than as a binary classifier of the frames that satisfy the predicate or not
  - ◆ Alleviates class imbalance issue
  - ◆ Allows the trained specialized NN to be reused for other queries such as aggregation.

**EX:**

**The user wants to find frames with at least one bus and at least five cars.**

**Then, BLAZEIT trains a single specialized NN to separately count buses and cars. BLAZEIT use the sum of the probability of the frame having at least one bus and at least five cars as its signal.**

**BLAZEIT takes the most confident frames until the requested number of frames is found.**





# Physical Operator and Selection

- Multiple Object Classes: **BLAZEIT trains a single NN to predict each object class separately**
  - ◆ e.g. *Instead of jointly predicting “car” and “bus”, the specialized NN would return a separate confidence for “car” and “bus”*
  - ◆ This results in fewer weights and typically higher performance.
- After results are sorted, **full object detector is applied** until the requested number of events is found or all the frames are searched.



# Limit Queries with Multiple Predicates

- If there is sufficient training data in the TMAS, BLAZEIT can execute the procedure above.
- **If there is not sufficient training data, BLAZEIT will train a specialized NN to search for the most selective set of predicates that contains enough data in a similar fashion to generating an aggregation specialized NN.**



# Implementation

Video ingestion, Specialized NN training, Identifying objects across frames

# Technologies Used

## → Languages

- ◆ Python 3.5 and C++

## → Framework

- ◆ PyTorch v1.0, FGFA, Detectron

## → Libraries

- ◆ OpenCV, MXNet v1.2

## → Algorithms

- ◆ Mask R-CNN

## → Hardware

- ◆ 1 NVIDIA Tesla P100 GPU
- ◆ 2 Intel Xeon E5-2690v4 CPUs (56 threads)
- ◆ System has 504 GB of RAM.



# Implementation

## → Video ingestion

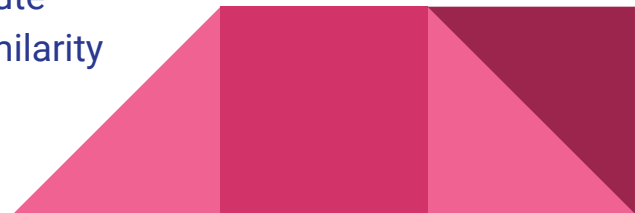
- ◆ Resizes videos for NNs to 65×65 for specialized NNs, short side of 600 pixels for object detection methods using standard ImageNet normalization

## → Specialized NN training

- ◆ Train using PyTorch
- ◆ Use tiny ResNet” architecture
  - Modified version of standard ResNet

## → Identifying objects across frames

- ◆ Use motion IOU to compute `trackid`
  - Ex: with a set of objects in 2 consecutive frames, compute pairwise IOU for each object in two frames to detect similarity





# Evaluation

Experimental Setup, Aggregate Queries, Cardinality-limited Queries, Specialized  
Neural Networks

# Experimental Setup

- **6 videos** scraped from YouTube
- **6-11 hours of video per day** where object detection method can perform well
- Varying in object classes (**car, bus, boat**), occupancy (**12 to 90%**), and average duration (**1.4s to 10.7s**)
- **3 days of video** for each webcam
  - Training labels
  - Threshold Computation
  - Testing

# Experimental Setup

Video name	Object	Occupancy	Avg. duration of object in scene	Distinct count	Resol.	FPS	# Eval frames	Length (hrs)	Detection method	Thresh
taipei	bus	11.9%	2.82s	1749	720p	30	1188k	33	FGFA	0.2
	car	64.4%	1.43s	32367						
night-street	car	28.1%	3.94s	3191	720p	30	973k	27	Mask	0.8
rialto	boat	89.9%	10.7s	5969	720p	30	866k	24	Mask	0.8
grand-canal	boat	57.7%	9.50s	1849	1080p	60	1300k	18	Mask	0.8
amsterdam	car	44.7%	7.88s	3096	720p	30	1188k	33	Mask	0.8
archie	car	51.8%	0.30s	90088	2160p	30	1188k	33	Mask	0.8

## Target object detection methods

- Used pre-trained object detection
- *Mask R-CNN* pretrained on *MS-COCO*, *FGFA* pretrained on *ImageNet-Vid*, and *YOLOv2* pretrained on *MS-COCO*



# Experimental Setup

Video name	Object	Occupancy	Avg. duration of object in scene	Distinct count	Resol.	FPS	# Eval frames	Length (hrs)	Detection method	Thresh
taipei	bus	11.9%	2.82s	1749	720p	30	1188k	33	FGFA	0.2
	car	64.4%	1.43s	32367						
night-street	car	28.1%	3.94s	3191	720p	30	973k	27	Mask	0.8
rialto	boat	89.9%	10.7s	5969	720p	30	866k	24	Mask	0.8
grand-canal	boat	57.7%	9.50s	1849	1080p	60	1300k	18	Mask	0.8
amsterdam	car	44.7%	7.88s	3096	720p	30	1188k	33	Mask	0.8
archie	car	51.8%	0.30s	90088	2160p	30	1188k	33	Mask	0.8

## Data Preprocessing

- Only considered regions where objects are large relative the the size of the frame
- Manually selected confidence thresholds for each video and object class for when to consider an object present

# Experimental Setup

Video name	Object	Occupancy	Avg. duration of object in scene	Distinct count	Resol.	FPS	# Eval frames	Length (hrs)	Detection method	Thresh
taipei	bus	11.9%	2.82s	1749	720p	30	1188k	33	FGFA	0.2
	car	64.4%	1.43s	32367						
night-street	car	28.1%	3.94s	3191	720p	30	973k	27	Mask	0.8
rialto	boat	89.9%	10.7s	5969	720p	30	866k	24	Mask	0.8
grand-canal	boat	57.7%	9.50s	1849	1080p	60	1300k	18	Mask	0.8
amsterdam	car	44.7%	7.88s	3096	720p	30	1188k	33	Mask	0.8
archie	car	51.8%	0.30s	90088	2160p	30	1188k	33	Mask	0.8

## Evaluation Metrics

- Object detection = ground truth
- **Aggregation Queries** -> absolute error & **Limit Queries** -> throughput
  - Throughput was measured by timing the complete end-to-end system excluding the time to decode the video
- Considered accuracy at the frame level

# Aggregate Queries

- 6 aggregate queries across 6 videos

# Aggregate Queries

- 6 aggregate queries across 6 videos
- 5 variants of each query
  1. **Naive** - object detection on every frame

# Aggregate Queries

- 6 aggregate queries across 6 videos
- 5 variants of each query
  1. **Naive** - object detection on every frame
  2. **Binary Oracle** - object detection on every frame with the object class present

# Aggregate Queries

- 6 aggregate queries across 6 videos
- 5 variants of each query
  1. **Naive** - object detection on every frame
  2. **Binary Oracle** - object detection on every frame with the object class present
  3. **Naive AQP** - randomly sampled from the video

# Aggregate Queries

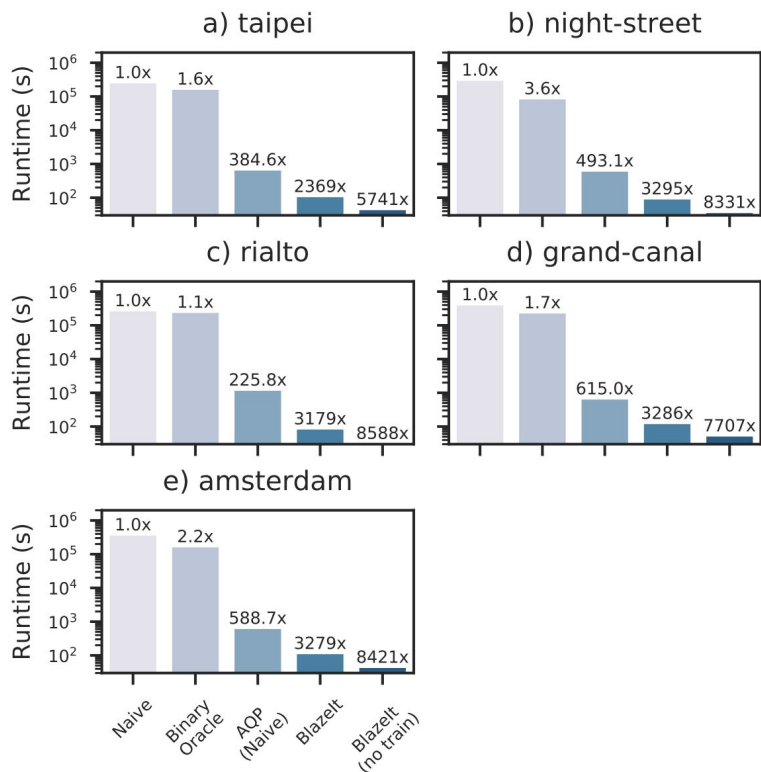
- 6 aggregate queries across 6 videos
- 5 variants of each query
  1. **Naive** - object detection on every frame
  2. **Binary Oracle** - object detection on every frame with the object class present
  3. **Naive AQP** - randomly sampled from the video
  4. **BLAZEIT** - use specialized NNs and control variates

# Aggregate Queries

- 6 aggregate queries across 6 videos
- 5 variants of each query
  1. **Naive** - object detection on every frame
  2. **Binary Oracle** - object detection on every frame with the object class present
  3. **Naive AQP** - randomly sampled from the video
  4. **BLAZEIT** - use specialized NNs and control variates
  5. **BLAZEIT (no train)** - excluded the training time

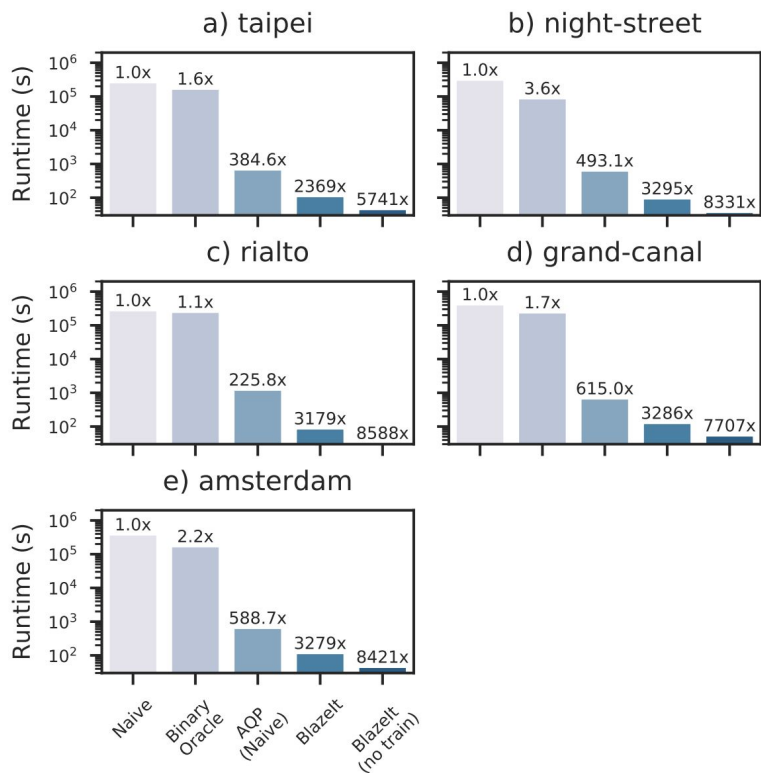


# Aggregate Queries - Query rewriting



- End-to-end runtime of aggregate queries where BLAZEIT rewrote the query with a specialized network
  - measured in seconds (log scale)
  - BLAZEIT outperforms all baselines
- BLAZEIT can outperform naive AQP by up to 14 times

# Aggregate Queries - Query rewriting



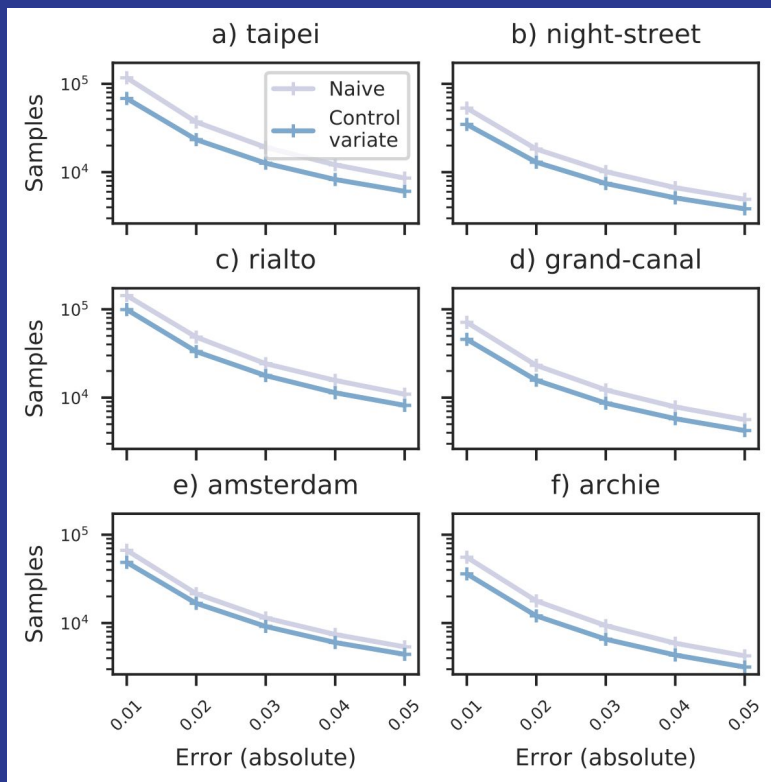
- End-to-end runtime of aggregate queries where BLAZEIT rewrote the query with a specialized network
  - measured in seconds (log scale)
  - BLAZEIT outperforms all baselines
- **BLAZEIT can outperform naive AQP by up to 14 times**

# Using Specialized NNs

**Table 5:** Average error of 3 runs of query-rewriting using a specialized NN for counting. These videos stayed within  $\epsilon = 0.1$ .

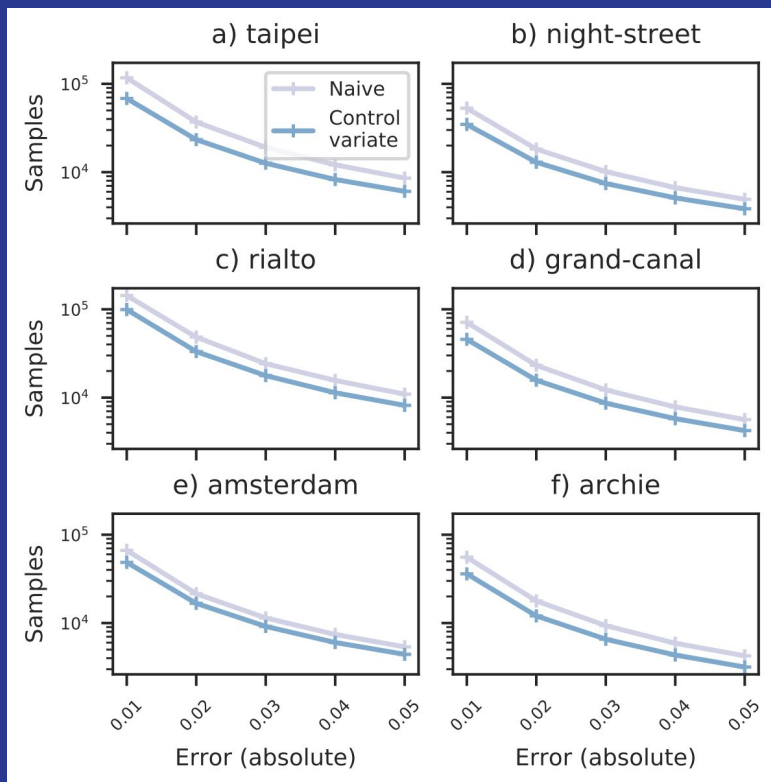
Video Name	Error
taipei	0.043
night-street	0.022
rialto	-0.031
grand-canal	0.081
amsterdam	0.050

# Aggregate Queries - Sampling and control variates



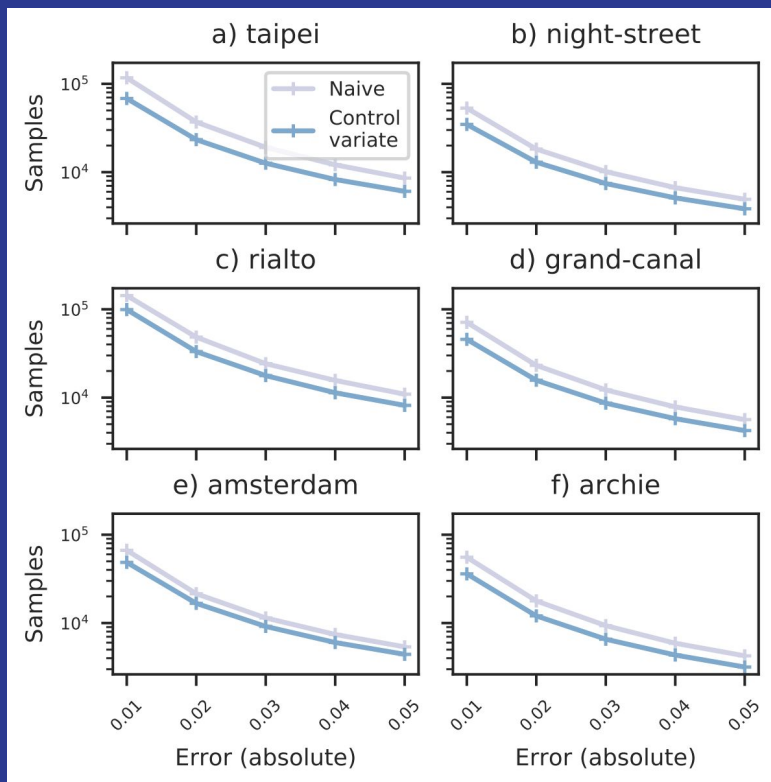
- Sample complexity of random sampling and BLAZEIT with control variates
- Control variates via specialized NNs can deliver up to 1.7 times reduction in sample complexity
- As the correlation between the specialized NNs and object detection method decreases, the sample complexity increases

# Aggregate Queries - Sampling and control variates



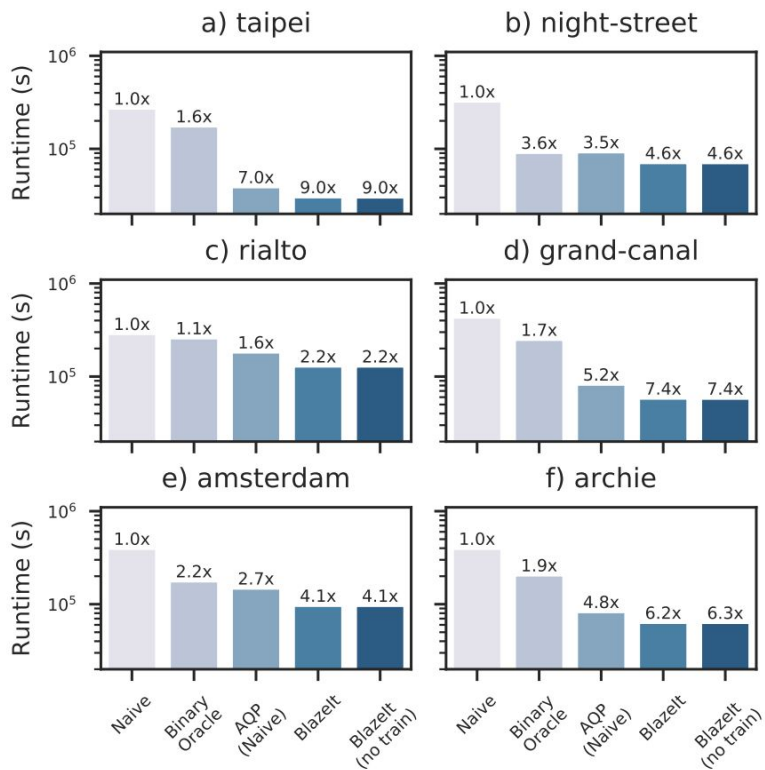
- Sample complexity of random sampling and BLAZEIT with control variates
- Control variates via specialized NNs can deliver up to 1.7 times reduction in sample complexity
- As the correlation between the specialized NNs and object detection method decreases, the sample complexity increases

# Aggregate Queries - Sampling and control variates



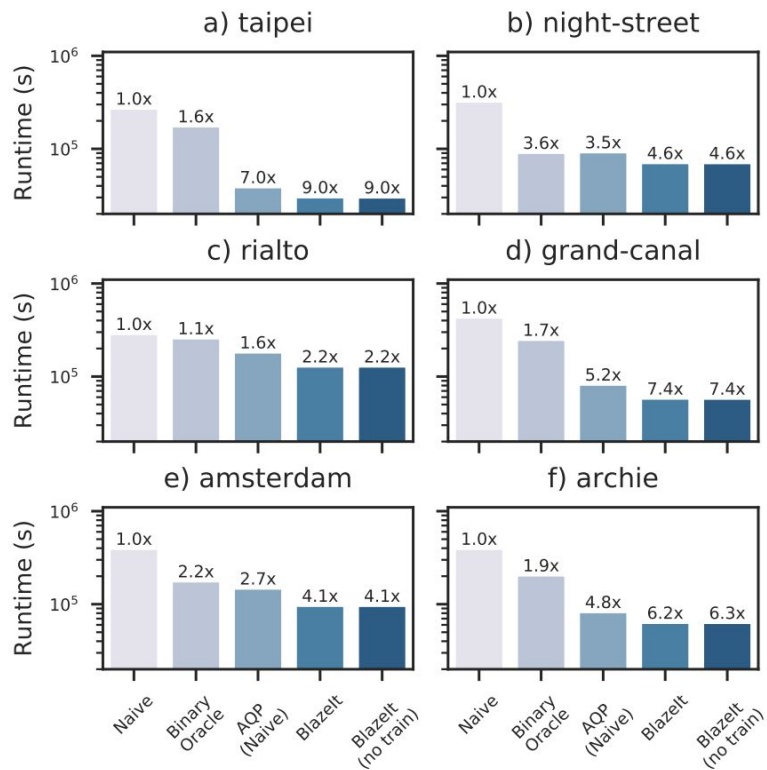
- Sample complexity of random sampling and BLAZEIT with control variates
- Control variates via specialized NNs can deliver up to 1.7 times reduction in sample complexity
- As the correlation between the specialized NNs and object detection method decreases, the sample complexity increases

# Aggregate Queries - Sampling with predicates



- Runtime of BLAZEIT and baselines for aggregation queries with predicates
- **Control variates via specialized NNs can deliver up to 1.5 times speedup compared to naive AQP**

# Aggregate Queries - Sampling with predicates



- Runtime of BLAZEIT and baselines for aggregation queries with predicates
- **Control variates via specialized NNs can deliver up to 1.5 times speedup compared to naive AQP**
- The relative gain of BLAZEIT's control variates depends on the reduction in variance
  - Gains are lower compared to queries with predicates as there is less training data



# Cardinality-limited Queries

- Frames of interest are returned to the user, up to the requested number of frames
- Selected rare events with at least 10 instances
  - If user queries more than max # of frames, BLAZEIT must inspect every frame

Video name	Object	Number	Instances
taipei	car	6	70
night-street	car	5	29
rialto	boat	7	51
grand-canal	boat	5	23
amsterdam	car	4	86
archie	car	4	102

# Cardinality-limited Queries

- Report the runtime and sample complexity

# Cardinality-limited Queries

- Report the runtime and sample complexity
- 5 variants of each query
  1. **Naive** - object detection sequentially until requested # of frames is found

# Cardinality-limited Queries

- Report the runtime and sample complexity
- 5 variants of each query
  1. **Naive** - object detection sequentially until requested # of frames is found
  2. **Binary Oracle** - object detection over the frames containing object class(es) of interest until requested # of frames is found

# Cardinality-limited Queries

- Report the runtime and sample complexity
- 5 variants of each query
  1. **Naive** - object detection sequentially until requested # of frames is found
  2. **Binary Oracle** - object detection over the frames containing object class(es) of interest until requested # of frames is found
  3. **Naive AQP** - randomly sampled the video until requested # of frames is found

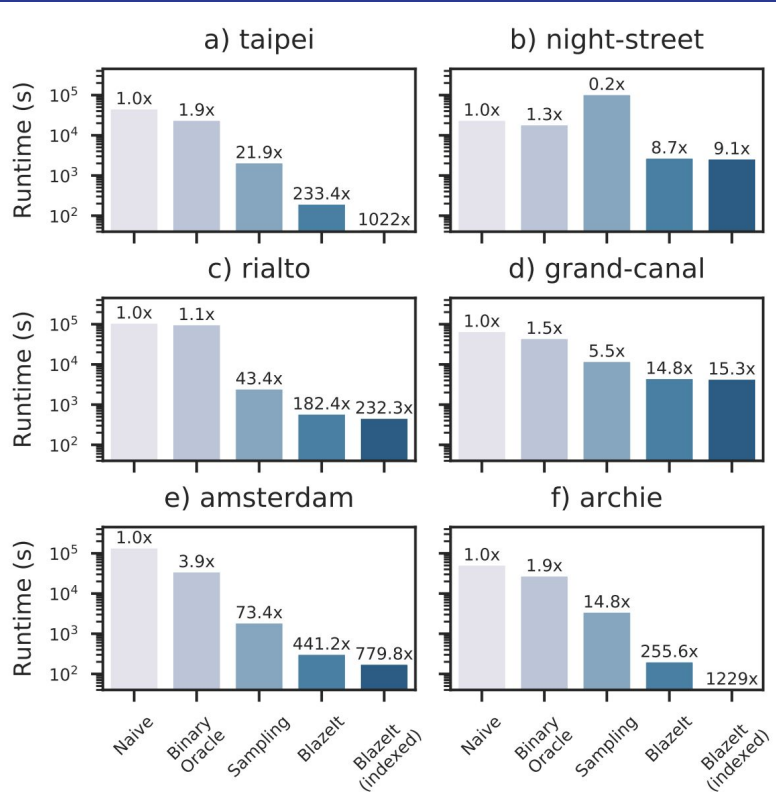
# Cardinality-limited Queries

- Report the runtime and sample complexity
- 5 variants of each query
  1. **Naive** - object detection sequentially until requested # of frames is found
  2. **Binary Oracle** - object detection over the frames containing object class(es) of interest until requested # of frames is found
  3. **Naive AQP** - randomly sampled the video until requested # of frames is found
  4. **BLAZEIT** - specialized NNs as a proxy signal to rank the frames

# Cardinality-limited Queries

- Report the runtime and sample complexity
- 5 variants of each query
  1. **Naive** - object detection sequentially until requested # of frames is found
  2. **Binary Oracle** - object detection over the frames containing object class(es) of interest until requested # of frames is found
  3. **Naive AQP** - randomly sampled the video until requested # of frames is found
  4. **BLAZEIT** - specialized NNs as a proxy signal to rank the frames
  5. **BLAZEIT (no train)** - assume the specialized NN has been trained and run over the remaining data

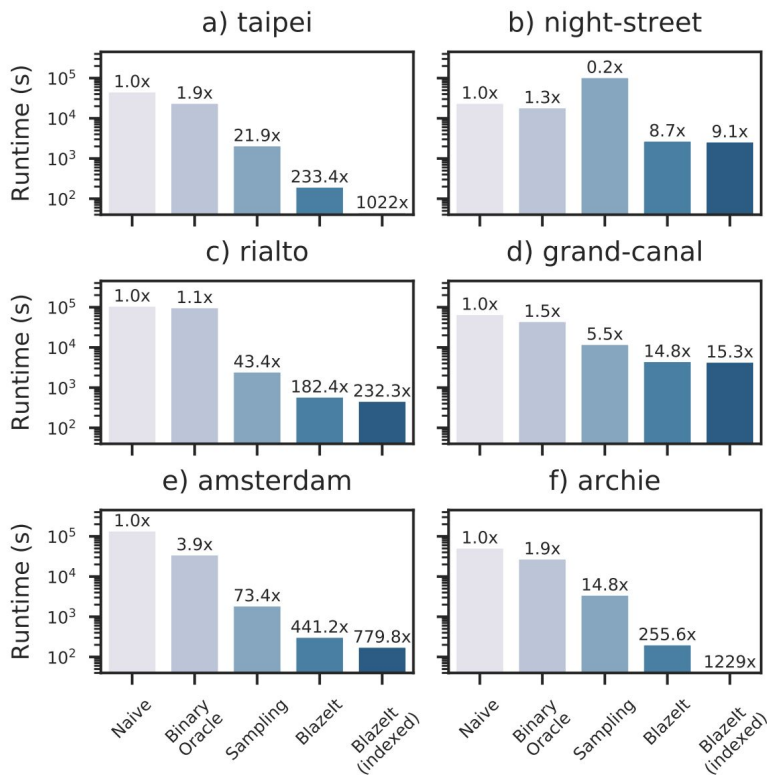
# Cardinality-limited Queries - Single Object Class



- BLAZEIT can achieve over a 1000 times speedup compared to baselines
  - BLAZEIT's specialized NNs can serve as a high-fidelity signal

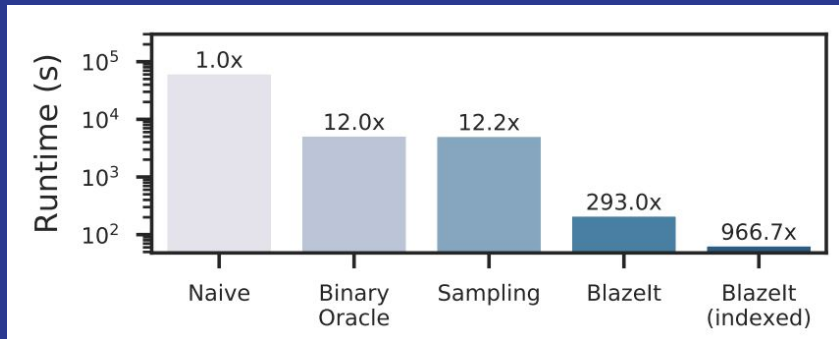


# Cardinality-limited Queries - Single Object Class



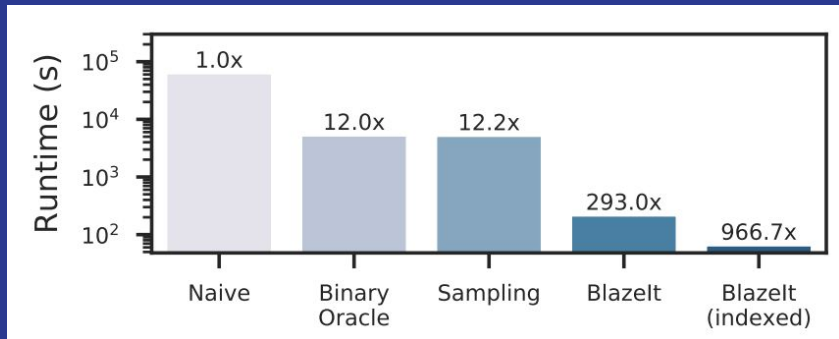
- BLAZEIT can achieve over a 1000 times speedup compared to baselines
  - BLAZEIT's specialized NNs can serve as a high-fidelity signal
- BLAZEIT's sample complexity remains nearly constant for up to 5 cars (search for common objects)
  - Shows efficacy of biased sampling

# Cardinality-limited Queries - Multiple Object Class

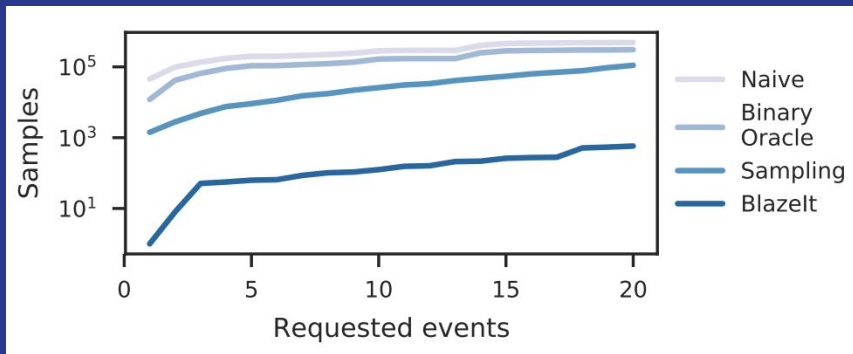


- End-to-end runtime of finding at least 1 bus and 5 cars in 'taipei'
  - **BLAZEIT outperforms the naive baseline by to 966 times**

# Cardinality-limited Queries - Multiple Object Class

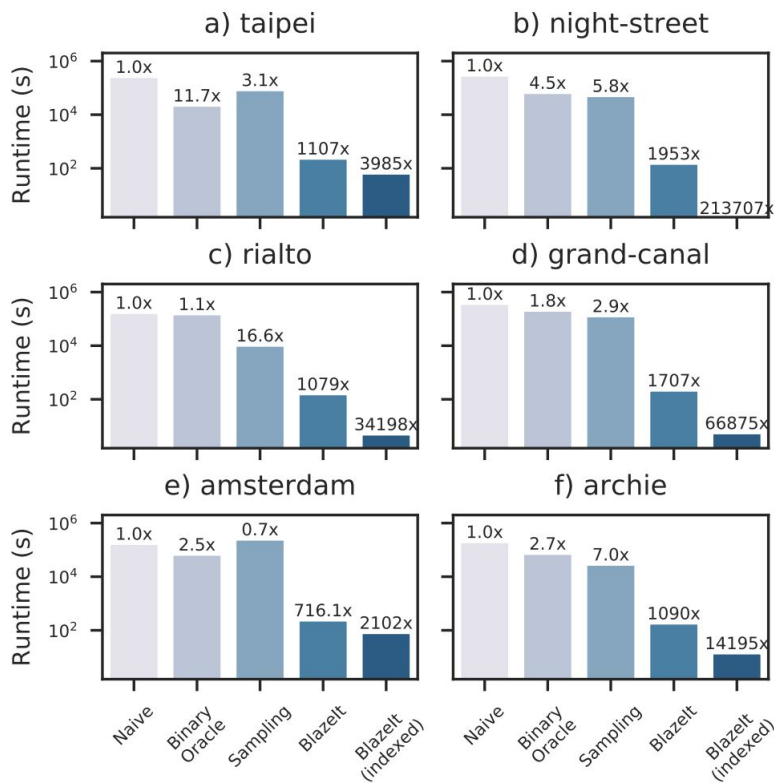


- End-to-end runtime of finding at least 1 bus and 5 cars in 'taipei'
  - **BLAZEIT outperforms the naive baseline by to 966 times**



- Sample complexity when searching for at least 1 bus and 5 cars
  - **BLAZEIT can be up to orders of magnitude more sample efficient over both naive baseline and binary oracle**

# Cardinality-limited Queries - Limit queries with predicates



- Runtime of BLAZEIT and baselines on limit queries with predicates
- **BLAZEIT outperforms all baselines by up to 300 times, even including the proxy model training time**
  - Especially outperforms baselines on queries that have few matches

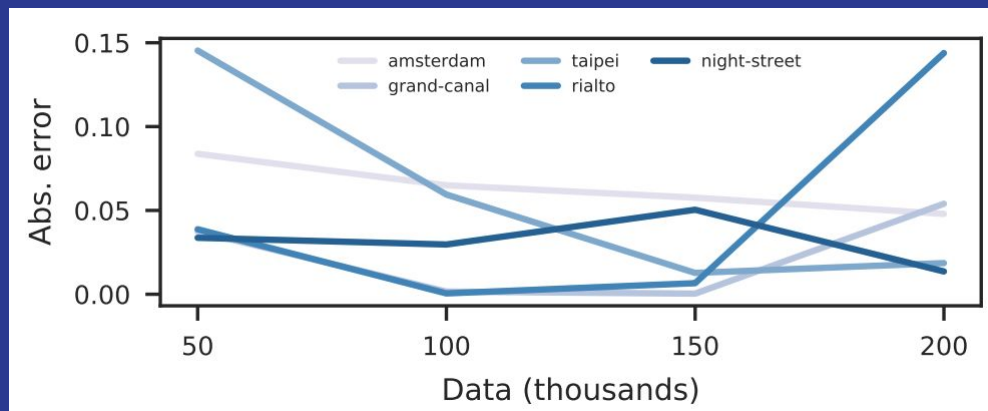
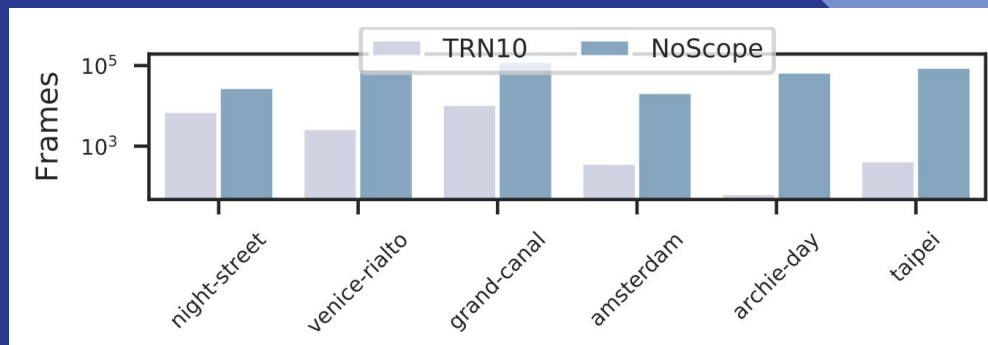
# Specialized Neural Networks

## Effect of NN type

- ResNet (referred to as TRN10)
- Requires **significantly fewer** samples compared to NOSCOPE NN

## Effect of Training Data

- Error decreases until 150,000 training samples
  - Increases potentially due to overfitting



# Results

1. **BLAZEIT** achieves up to **14** times speedup over **AQP** on aggregation queries
2. **BLAZEIT** achieves up to an **83** times speedup compared to the next best method for video limit queries

# Results

1. **BLAZEIT** achieves up to **14** times speedup over **AQP** on aggregation queries
2. **BLAZEIT** achieves up to an **83** times speedup compared to the next best method for video limit queries

# Conclusion



# Blazelt

- There was a need for a speed up in querying videos for semantic information with less emphasis on complex coding



# Blazelt

- There was a need for a speed up in querying videos for semantic information with less emphasis on complex coding
  - ◆ The answer is **BLAZEIT!**



# Blazelt

- There was a need for a speed up in querying videos for semantic information with less emphasis on complex coding
  - ◆ The answer is **BLAZEIT!**
- Uses FrameQL as its declarative language
- Optimizations for aggregation and limit querying
  - ◆ 14x speedup on aggregation queries
  - ◆ 83x speedup on limit queries
- Retains accuracy guarantees despite potential inaccurate specialized NNs



The background is a solid pink color. In the top right corner, there is a decorative graphic consisting of several overlapping geometric shapes, including triangles and squares, in various shades of pink and dark pink.

Thank you!



# Related Works

# Related Works


## → AQP

- ◆ Result of query returned quickly by subsampling the data
- ◆ Blazelt uses variance reduction with control variates through specialized NNs to reduce the cost of creating a tuple

## → Visual Data Management

- ◆ Use classic computer vision techniques for semantic queries
- ◆ Blazelt uses FrameQL, an extension of SQL, to automatically populate these fields

## → Modern Video Analytics

- ◆ NOSCOPE, Focus, and Tahoma cannot adapt to user queries or optimize training time on specialized NNs
  - ◆ Blazelt has optimizations to allow for aggregation and limit querying
- 

# Optimizations

## Query Specific NNs

Lorem ipsum dolor sit amet, consectetur adipiscing elit, sed do eiusmod tempor incididunt ut labore et dolore magna aliqua.

Ut enim ad minim veniam, quis nostrud

## Context

Ut enim ad minim veniam, quis nostrud exercitation

- Duis aute irure dolor in reprehenderit in voluptate velit

## Problem statement

Excepteur sint occaecat cupidatat non proident, sunt in culpa qui officia deserunt mollit anim id est laborum.

# Challenges deep-dive

## Challenge 1

### Expand audience

Lorem ipsum dolor sit amet, consectetur adipiscing elit, sed do eiusmod tempor incididunt ut labore et dolore magna aliqua.

## Challenge 2

### Up 30-day actives

Ut enim ad minim veniam, quis nostrud exercitation

- Duis aute irure dolor in reprehenderit in voluptate velit

## Challenge 3

### Increase conversion

Excepteur sint occaecat cupidatat non proident, sunt in culpa qui officia deserunt mollit anim id est laborum.



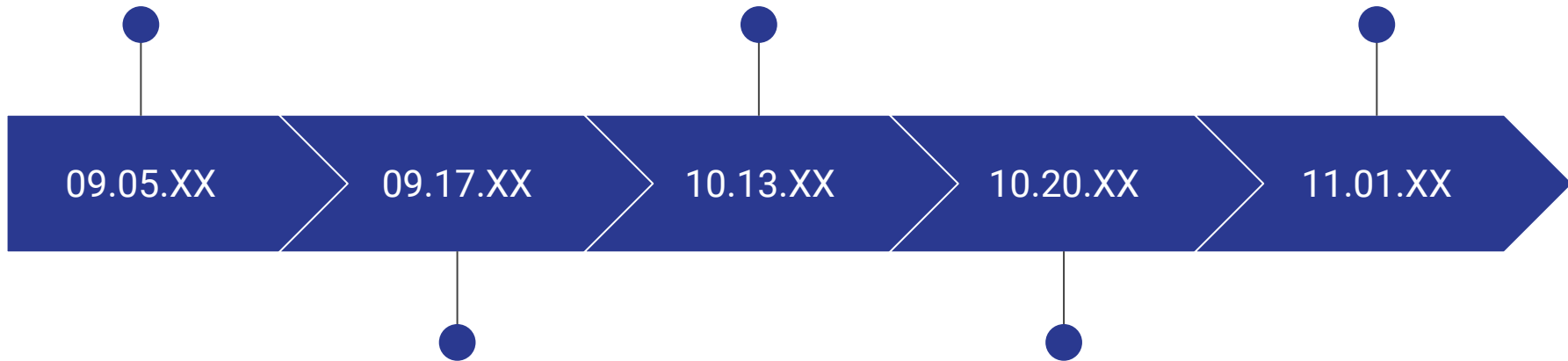
# Use Cases

- Urban planning
- Autonomous vehicle analysis
- Store planning
- Ornithology

Lorem ipsum dolor sit  
amet, consectetur  
adipiscing elit

Lorem ipsum dolor sit  
amet, consectetur  
adipiscing elit

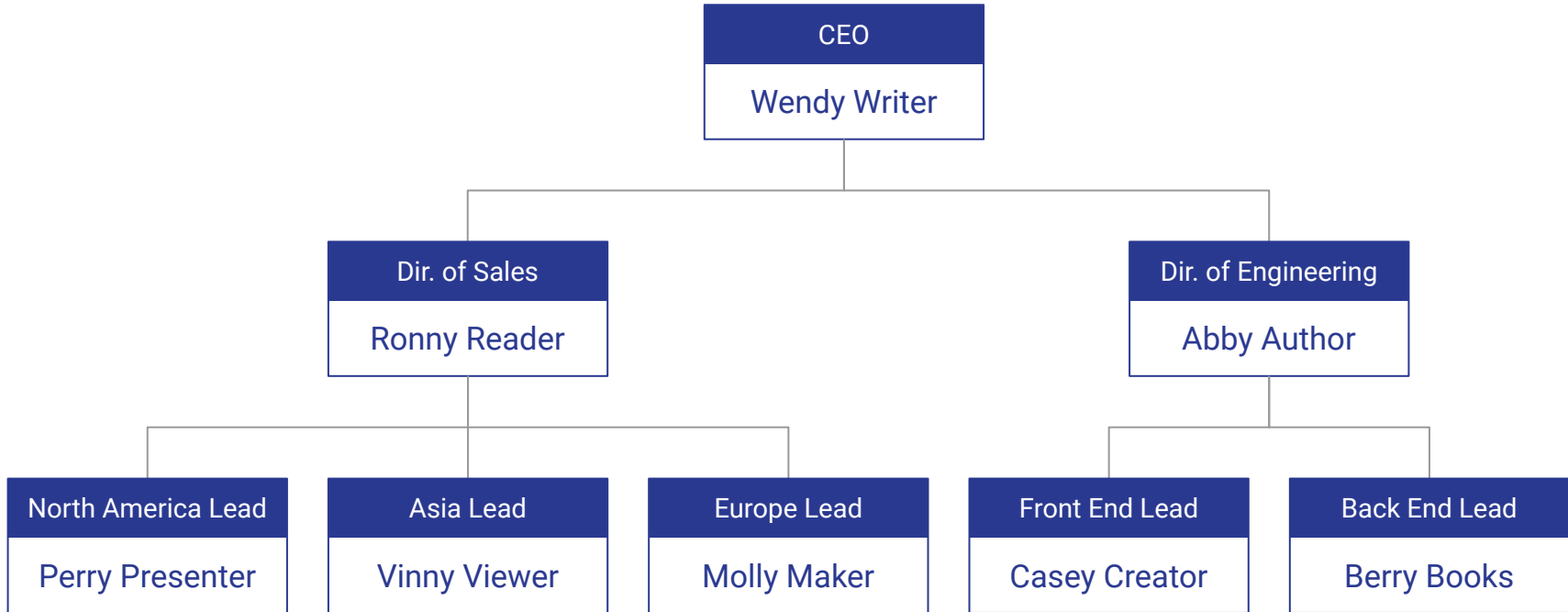
Lorem ipsum dolor sit  
amet, consectetur  
adipiscing elit



Lorem ipsum dolor sit  
amet, consectetur  
adipiscing elit

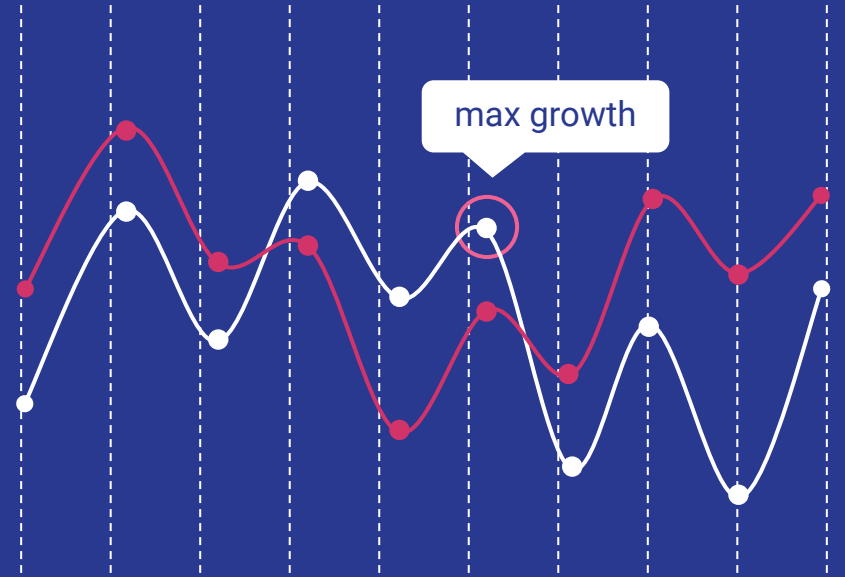
Lorem ipsum dolor sit  
amet, consectetur  
adipiscing elit

# The team



# Impact

XX% sales increase



---

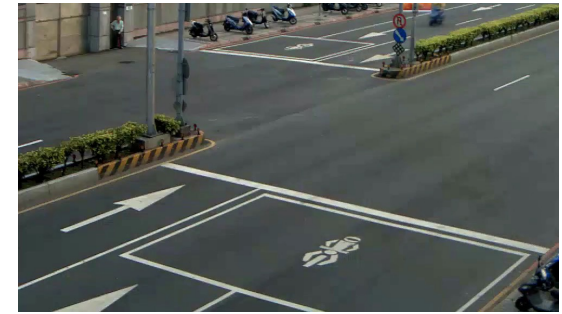
# Why does specialized NN work?

- “I want to know when buses pass by this intersection in Taipei using YOLOv2”
- Who cares about toilets, cats, skis, ...?
- Who cares about buses from different perspectives?



# Why does specialized NN work?

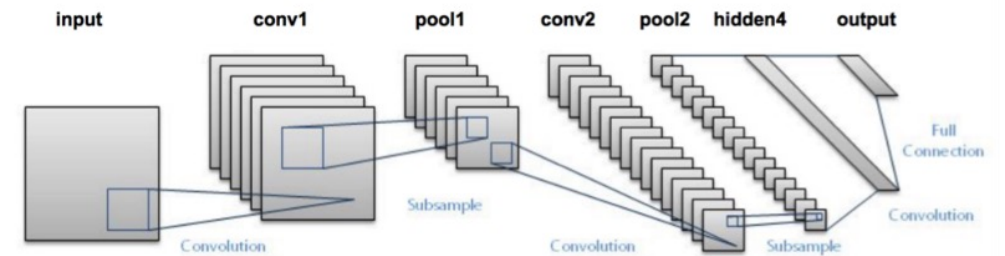
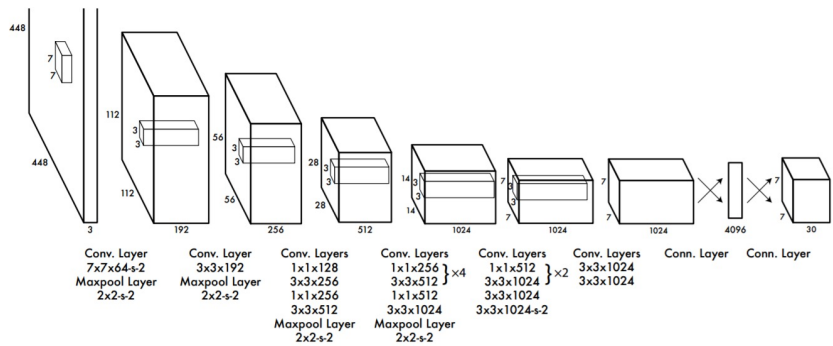
- “I want to know when buses pass by this intersection in Taipei using YOLOv2”



**Buses are similar from the perspective of the webcam!**

# Exploiting scene-specific locality: specialized CNNs

- Idea: train smaller, faster scene-specific CNN
- 1. Run big CNN over stream to obtain labels
- 2. Train smaller, specialized CNN over labels



# Proxy models are much smaller

## Mask R-CNN

- 152 convolutional layers
- 64-1024 filters per layer
- 3 fps

~150 billion FLOPS

## Blazert proxy model

10 convolutional layers  
16-64 filters per layer  
15,000 fps

~10 million FLOPS

**15,000x fewer FLOPS**  
**5,000x faster execution**



# Specialization != Model Compression

**Model compression/distillation:** lossless models

- Goal: smaller model for **same task** as reference model
- Result: typically **2-10x** faster execution

**Specialization:** perform “lossy” compression of reference model

- A specialized model **does not generalize** to other videos...
- ...but is accurate on target video, can be **100-1000x** faster

# Physical Representation-based Predicate Optimization for a Visual Analytics Database

- Modern content extraction techniques are **accurate but expensive** → can we reduce the cost of visual content queries using **inexpensive classifiers**?
- Evaluates a large number of **cascade classifiers** to optimize both the **CNN architecture** and **input data representation**
- Introduces the idea of using **specialized candidate binary-classification CNNs** to reduce inference cost

2019 IEEE 35th International Conference on Data Engineering (ICDE)

## Physical Representation-based Predicate Optimization for a Visual Analytics Database

Michael R. Anderson  
University of Michigan  
mrande@umich.edu

Michael Cafarella  
University of Michigan  
michjc@umich.edu

German Ros\*  
Intel Labs  
german.ros@intel.com

Thomas F. Wenisch  
University of Michigan  
twenisch@umich.edu

*Abstract*—Querying the content of images and video requires expensive content extraction methods. Modern extraction techniques are based on deep convolutional neural networks (CNNs) and can classify objects within images with astounding accuracy. Unfortunately, these methods are slow: processing a single image can take about 10 milliseconds on modern GPU-based hardware. As massive video libraries become ubiquitous, running a content-based query over millions of video frames is prohibitive.

One promising approach to reduce the runtime cost of queries of visual content is to use a hierarchical model, such as a cascade, where simple cases are handled by an inexpensive classifier. Prior work has sought to design cascades that optimize the computational cost of inference by, for example, using smaller CNNs. However, we observe that there are critical factors besides the inference time that dramatically impact the overall query time. Notably, by treating the physical representation of the input image as part of our query optimization—that is, by including image transformations such as resolution scaling or color-depth reduction within the cascade—we can optimize data handling costs and enable drastically more efficient classifier cascades.

In this paper, we propose TAHOMA, which generates and evaluates many potential classifier cascades that jointly optimize the CNN architecture and input data representation. Our experiments on a subset of ImageNet show that TAHOMA's input transformations speed up cascades by up to 35 times. We also find up to a 98x speedup over the ResNet50 classifier with no loss in accuracy and a 280x speedup if some accuracy is sacrificed.

### I. INTRODUCTION

Recent developments in computer vision have made feasible a long-term dream for the database community: a *visual analytics database*, which stores image data and answers user questions about its contents. For example, video frames

CNN was first used [28]. Recent results have lowered the error rate to 2% [20], rivaling or exceeding human performance.

Unfortunately, deep networks pose a considerable computational challenge when deployed in an analytical database system: a model's inference for a single image can require a lengthy series of large tensor multiplications. For example, YOLOv2, an object detection system designed for speed, requires 8.52 billion operations per single 416x416 pixel image, processing about 67 images per second on a modern GPU [38]. Since GPU hardware is far more expensive than most image sensors, data from multi-camera applications will soon outpace processing capabilities. Simply, to query huge amounts of image data, we need drastically lower processing costs.

Processing queries over a corpus of image data fits a more general *loop-and-test* pattern that is common to many machine learning tasks: the processor loops over the data, executing an expensive operator on each element to find those satisfying the task's constraints. In this case, content is extracted from each image by the expensive inference stage of a deep network to determine if the image satisfies a binary predicate specified in a user's query. While a loop-and-test process can be shortened by processing fewer items overall—using simple sampling or more sophisticated input selection [4]—we focus here on speeding up the *test* phase by reducing the per-image inference cost.

Recent work has reduced inference times for these types of deep learning systems (e.g., [17], [26]). However, we note that all of the visual data system optimizations to date suffer from a critical defect: they concentrate only on computation and ignore the inevitable data-handling costs, such as loading and

# NoSCOPE: Optimizing Neural Network Queries over Video at Scale

- Accelerating neural network video analysis via **inference-optimized model search**
- Trains a cascade of **difference detectors** and **specialized models** to mimic the behavior of a reference model but three orders of magnitude faster
- Integrates these two types of models using **cost-based optimization** to match a target accuracy

## NoSCOPE: Optimizing Neural Network Queries over Video at Scale

Daniel Kang, John Emmons, Firas Abuzaid, Peter Bailis, Matei Zaharia  
Stanford InfoLab  
noscope@cs.stanford.edu

### ABSTRACT

Recent advances in computer vision—in the form of deep neural networks—have made it possible to query increasing volumes of video data with high accuracy. However, neural network inference is computationally expensive at scale: applying a state-of-the-art object detector in real time (i.e., 30+ frames per second) to a single video requires a \$4000 GPU. In response, we present NoSCOPE, a system for querying videos that can reduce the cost of neural network video analysis by up to three orders of magnitude via *inference-optimized model search*. Given a target video, object to detect, and reference neural network, NoSCOPE automatically searches for and trains a sequence, or cascade, of models that preserves the accuracy of the reference network but is specialized to the target video and are therefore far less computationally expensive. NoSCOPE cascades two types of models: *specialized models* that forego the full generality of the reference model but faithfully mimic its behavior for the target video and object; and *difference detectors* that highlight temporal differences across frames. We show that the optimal cascade architecture differs across videos and objects, so NoSCOPE uses an efficient cost-based optimizer to search across models and cascades. With this approach, NoSCOPE achieves two to three order of magnitude speed-ups (265-15,500 $\times$  real-time) on binary classification tasks over fixed-angle webcam and surveillance video while maintaining accuracy within 1-5% of state-of-the-art neural networks.

### 1. INTRODUCTION

Video represents a rich source of high-value, high-volume data: video comprised over 70% of all Internet traffic [2] in 2015 and over 300 hours of video are uploaded to YouTube every minute [3]. We can leverage this video data to answer queries about the physical world, our lives and relationships, and our evolving society. It is increasingly infeasible—both too costly and too slow—to rely

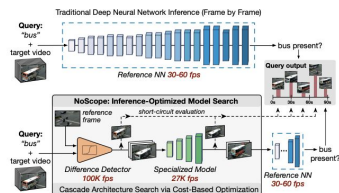


Figure 1: NoSCOPE is a system for accelerating neural network analysis over videos via inference-optimized model search. Given an input video, target object, and reference neural network, NoSCOPE automatically searches for and trains a cascade of models—including difference detectors and specialized networks—that can reproduce the binarized outputs of the reference network with high accuracy—but up to three orders of magnitude faster.

video methods due to their incredible accuracy—often rivaling or exceeding human capabilities—in visual analyses ranging from object classification [82] to image-based cancer diagnosis [31, 95].

Unfortunately, applying NNs to video data is prohibitively expensive at scale. The fastest NNs for accurate object detection run at 30-80 frames per second (fps), or 1-2.5 $\times$  real time (e.g., 50 fps on an NVIDIA K80 GPU, ~\$4000 retail, \$0.70-0.90 per hour on cloud; 80 fps on an NVIDIA P100, ~\$4600 retail) [79-81].<sup>1</sup> Given continued decreases in image sensor costs (e.g., < \$0.65 for a 640x480 VGA CMOS sensor), the computational overheads of NNs lead to a three order-of-magnitude imbalance between the cost of data acquisition and the cost of data processing. Moreover, state-of-the-art NNs continue to get deeper and more costly to evaluate; for example, Google's winning NN in the 2014 ImageNet competition had 22

# Optimizing Video Analytics with Declarative Model Relationships

- Previous optimizations are difficult to use on **complex queries** with multiple predicates and models
- Proposes the idea of **Relational Hints**, which suggests ML model relationships based on domain knowledge (CAN REPLACE / CAN FILTER)
- **VIVA** is a new visual analytics system that uses relational hints to optimize video dataset queries



## Optimizing Video Analytics with Declarative Model Relationships

Francisco Romero\*  
Stanford University  
faromero@stanford.edu

Johann Hauswald\*  
Stanford University &  
Sutter Hill Ventures  
johannh@stanford.edu

Aditi Partap  
Stanford University  
aditi712@stanford.edu

Daniel Kang  
Stanford University  
ddkang@cs.stanford.edu

Matei Zaharia  
Stanford University  
matei@cs.stanford.edu

Christos Kozyrakis  
Stanford University  
christos@cs.stanford.edu

### ABSTRACT

The availability of vast video collections and the accuracy of ML models has generated significant interest in video analytics systems. Since naively processing all frames using expensive models is impractical, researchers have proposed optimizations such as selectively using faster but less accurate models to replace or filter frames for expensive models. However, these optimizations are difficult to apply on queries with multiple predicates and models, as users must manually explore a large optimization space. Without significant systems expertise or time investment, an analyst may manually create an execution plan that is unnecessarily expensive and/or terribly inaccurate.

We propose *Relational Hints*, a declarative interface that allows users to suggest ML model relationships based on domain knowledge. Users can express two key relationships: when a model can replace another (CAN REPLACE) and when a model can be used to filter frames for another (CAN FILTER). We aim to design an interface to express *model relationships informed by domain specific knowledge* and define the constraints by which these relationships hold. We then present the *VIVA video analytics system* that uses relational hints to optimize SQL queries on video datasets. VIVA automatically selects and validates the hints applicable to the query, generates possible query plans using a formal set of transformations, and finds the best performance plan that meets a user's accuracy requirements. VIVA relieves users from rewriting and manually optimizing video queries as new models become available and execution environments evolve. We evaluate VIVA implemented on top of Spark and show that hints improve performance up to 16.6x without sacrificing accuracy.

PVLDB Reference Format:

### 1 INTRODUCTION

Video analytics, the ability to extract insights from video, is enabled by increasingly accurate machine learning (ML) models and access to large archives of professionally produced content or videos captured by devices like cellphones, security cameras, and video-conference systems. While we can already answer queries over videos like “have any cars passed this intersection that match an AMBER alert?”, several challenges remain before video analytics are as practical and as performant over analytics on structured data. For complex video analytics queries with multiple predicates or ML models, users must manually optimize their queries to avoid the high cost of naively executing large models on every frame using expensive hardware. For example, it takes over 14 GPU-months to process 100 camera-months of video using a very accurate YOLOv5 model for object detection [55].

Consider an analyst studying political coverage of major cable news channels that writes a query to find instances of Bernie Sanders, a politician, reacting angrily to Jake Tapper, a TV news host [20]. Their query may use object detection to find scenes with two people, face recognition to find instances of Jake Tapper and Bernie Sanders, and emotion detection to detect angry reactions. This query can take minutes to execute using unnecessarily accurate models, even on small video inputs, making it challenging for the analyst to interactively explore their dataset. To improve performance, the analyst may use domain knowledge to explore the following model optimizations:

- *Replacement*: use a different model for a task, such as a cheaper but less accurate object detector [25, 27, 28, 47].
- *Input Filtering*: use a fast model to filter inputs to an expensive model [26, 36, 63]. For example, insert a binary classifier to detect

# RECL: Responsive Resource-Efficient Continuous Learning for Video Analytics

- Extend visual analytics frameworks to support **continuous learning** capabilities (model reusing and online retraining)
- Creates a “**model zoo**” of previously trained expert models, enabling historical model reuse → selects a highly accurate **expert model** from this model zoo → dynamically **optimizes GPU** allocation for retraining

## RECL: Responsive Resource-Efficient Continuous Learning for Video Analytics

Mehrdad Khani<sup>1,2</sup>, Ganesh Ananthanarayanan<sup>2</sup>, Kevin Hsieh<sup>2</sup>, Junchen Jiang<sup>3</sup>, Ravi Netravali<sup>4</sup>,  
Yuanchao Shu<sup>5</sup>, Mohammad Aliizadeh<sup>1</sup>, Victor Bahl<sup>2</sup>

<sup>1</sup>MIT CSAIL, <sup>2</sup>Microsoft, <sup>3</sup>University of Chicago, <sup>4</sup>Princeton University, <sup>5</sup>Zhejiang University

### Abstract

Continuous learning has recently shown promising results for video analytics by adapting a lightweight “expert” DNN model for each specific video scene to cope with the data drift in real time. However, current adaptation approaches either rely on periodic retraining and suffer its delay and significant compute costs or rely on selecting historical models and incur accuracy loss by not fully leveraging the potential of persistent retraining. Without dynamically optimizing the resource sharing among model selection and retraining, both approaches have a diminishing return at scale. RECL is a new video-analytics framework that carefully integrates model reusing and online model retraining, allowing it to quickly adapt the expert model given any video frame samples. To do this, RECL (i) shares across edge devices a (potentially growing) “model zoo” that comprises expert models previously trained for all edge devices, enabling history model reuse across video sessions, (ii) uses a fast procedure to online select a highly accurate expert model from this shared model zoo, and (iii) dynamically optimizes GPU allocation among model retraining, model selection, and timely updates of the model zoo. Our evaluation of RECL over 70 hours of real-world videos across two vision tasks (object detection and classification) shows substantial performance gains compared to prior work, further amplifying over the system lifetime.

### 1 Introduction

Video analytics with deep neural networks (DNNs) is a promising technology adopted in a wide range of applications such as enterprise security, retail, traffic management, and transportation [1, 2]. Across these applications, it is often imperative to run analytics tasks directly on edge devices

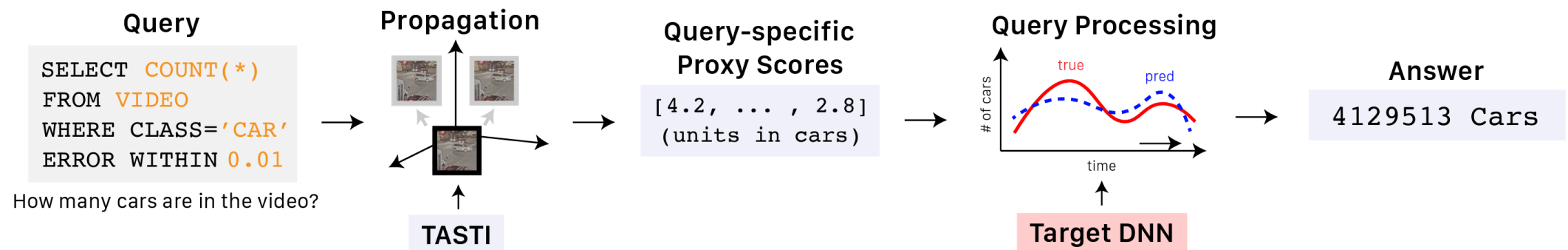
DNNs [15–18]. However, owing to their inherent limits on the number of object appearances and scenes they can learn in their condensed structures, such specialized DNNs require *continuous retraining* to cope with dynamic scenes (data drifts) in order to maintain high inference accuracy. Recent work in the computer vision and systems communities [19–21] has shown the effectiveness of this approach for edge video analytics, delivering both high resource efficiency and accuracy in results.

Though promising, continuous retraining and deploying specialized DNNs has two fundamental limitations. First, continuous retraining consumes the vast majority of *compute resources* in these video analytics systems (70%–90% in our study) [20, 21], making model retraining the key bottleneck in scaling video analytics to more video streams with limited compute resources. Our study (Fig. 2) shows that accuracy drops sharply (by 40% in object detection) as 4× more cameras share the GPU cycles to retrain their models (§2.2). Second, it takes *time* to retrain specialized DNNs, and abrupt video scene changes inevitably lead to drastic accuracy drops until the retraining is completed (see Fig. 3 for an example). Hence, it is fundamentally challenging to uphold the accuracy lower tail during the retraining.

**Our goal** in this work is to address the above two fundamental limitations so that video analytics are scalable with more consistent accuracy. As retraining specialized DNNs requires resources and takes time, we aim to minimize the necessity of retraining by judiciously *reusing* historical specialized DNNs that are trained with past video segments. The intuition behind our approach is that video streams typically exhibit spatio-temporal correlations (e.g., a car drives back on the same street or another car has been on the same street before) [22]. Thus, it is likely that the current video segment

# Semantic Indexes for Machine Learning-based Queries over Unstructured Data [SIGMOD'22]

- Downsides of query-specific proxy models
  - have to be trained per query
  - require non-trivial amount of labels
  - can not easily share computation across different queries or query types.
- TASTI: Learn embeddings to cluster semantically similar records
  - e.g., video frames with similar object types and object positions are close



---

---

# Blazelt: Optimizing Declarative Aggregation and Limit Queries for Neural Network-Based Video Analytics

Industry Practitioner - Ankith Reddy Chitti (Kroger)

---

# What is Blazelt:

Blazelt is a system that optimizes queries concerning spatiotemporal information of objects in video. It uses the novel FrameQL to accept queries and enable video specific query optimization.

Unlike prior work Blazelt is mainly aimed at optimizing aggregation and limit queries:

- Uses Neural Networks as control variates to optimize aggregation queries.
- For limit queries, specialized Neural Networks are used to bias search towards regions with a high probability of the event.

Blazelt focuses on batch setting and can deliver up to 83× speedups over similar systems on video analytics.



# Blazelt for store planning:

## Retail Video Analytics:

- Discovery, interpretation, and communication of meaningful patterns in data from video content, and applying those patterns towards effective decision-making.
- Provides retailers with insightful business intelligence such as store traffic statistics and queue data.

## Benefits:

- Maximize store layout and navigation.
- Manage store traffic.
- Streamline checkout.
- **Optimize promotions and product displays.**



Kroger's spent just **under \$100 million** on promotional material regarding new product launches, cross merchandising and discount sales across all its stores in 2023 alone.

However in the most recent quarter, the company posted a **net loss of \$180 million**. It is imperative we start to look at ways to cut down our losses and generate more revenue.



Blazelt with its optimizations for aggregation and limit queries can provide valuable insights that can be used to formulate effective promotions and display designs.

Both of BLAZEIT's novel optimizations share a key property: **Accuracy guarantees are always upheld.**

Empirical tests show Blazelt runs **upto 83X faster** compared to other similar systems in the domain.

[1] <https://advertisers.mediaradar.com/kroger-advertising-profile#NewProducts>

[2] <https://www.cnbc.com/2023/09/08/kroger-kr-earnings-q2-2023.html>

# Blazelt and in-store marketing campaign

Rapidly changing consumer behavior and preferences are leading retail chains to overhaul their marketing campaigns and In-store marketing tactics.

According to [Gartner](#), nearly 30% of marketing leaders believe lack of agility and flexibility negatively impacts marketing execution. The Gartner survey also concluded that marketers are under immense pressure to deliver insights faster than ever before.

Video analytics via Blazelt not only gives us an accurate, and rapid real-time view of how consumer preferences are changing but also allows us to execute marketing programs with a high degree of confidence.

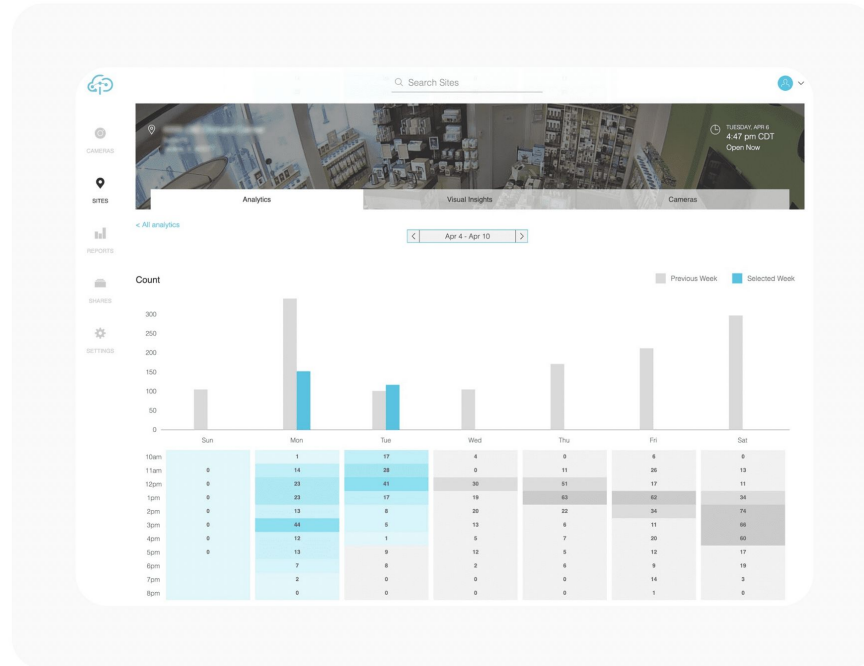
Sample use-case wherein we can get the list of store\_lanes for each hour that had a high customer footfall.

```
SELECT timestamp, store_lane
FROM kroger_atl
GROUP BY store_lane
HAVING COUNT(class='customer')>=15
GAP 108000 // assume fps is 30.
```

This could help us streamline marketing operations and analyze customer dwell times / hotspots at specific locations.

## Another potential use-case

Blazelt can provide reliable data on footfalls for the entire store or for a specific department by date and time. The information can be used to schedule in-house promotional displays effectively.



[1] <https://interfacesystems.com/blog/retail-video-analytics/>

## One potential drawback of Blazelt

Blazelt works best in the batch setting. Due to our continuous inflow of new video data, as the distribution might change, Blazelt is susceptible to degraded execution. Note that the accuracy requirements will always be met irrespective of the data.

This is not a big cause of concern since it can be mitigated by labeling a portion of new data or continuous retraining.

**In summary, by integrating Blazelt, Kroger can improve its overall store efficiency, enhance the shopping experience for its customers, and make data-driven decisions to remain competitive in the highly competitive grocery industry**

**Thank You!**



# Discussion

How are FrameQL queries different from standard aggregate queries in AQP?

- UDFs that extract relational columns from unstructured inputs are often very expensive
  - Traditional query optimization (e.g., predicate pushdown) doesn't work
  - pre-computing is wasteful, especially to support ad-hoc queries
- UDFs and predicates are not deterministic
  - Performance accuracy tradeoff
- Even "ground truth" object detection models contain errors

# Discussion

Is FrameQL/SQL a good language for video analytics?

- Much better than working with raw pixel data
- Does not require knowledge of neural network
- Familiar to SQL users
- Declarativity allows for query optimization opportunities



# Discussion

Is SQL a good language for video analytics?

- Some queries might be hard to specify:
  - Event/Action queries:
    - A player passes the ball to one of his teammates but an opponent player tries to intercept the ball
  - Trajectory queries:
    - A car turns left and then right
- Other query interfaces?
  - Natural language
  - Scene graph