Monitoring and Quality Assurance in Complex ML Deployments with Assertions

Daniel Kang
Stanford University

STANFORD DAWN

Errors in ML models lead to downstream consequences

Self-Driving Uber Car Kills Pedestrian in Arizona, Where Robots Roam

Autonomous vehicles have already been involved in fatal accidents

- Errors can have extreme consequences
- No standard way of monitoring / quality assurance

Software 1.0 is also deployed in mission-critical settings!

Software powers
medical devices, etc.

Important software is monitored and has rigorous QA

- Assertions
- Unit tests
- Regression tests
- Fuzzing
- …

Our research:

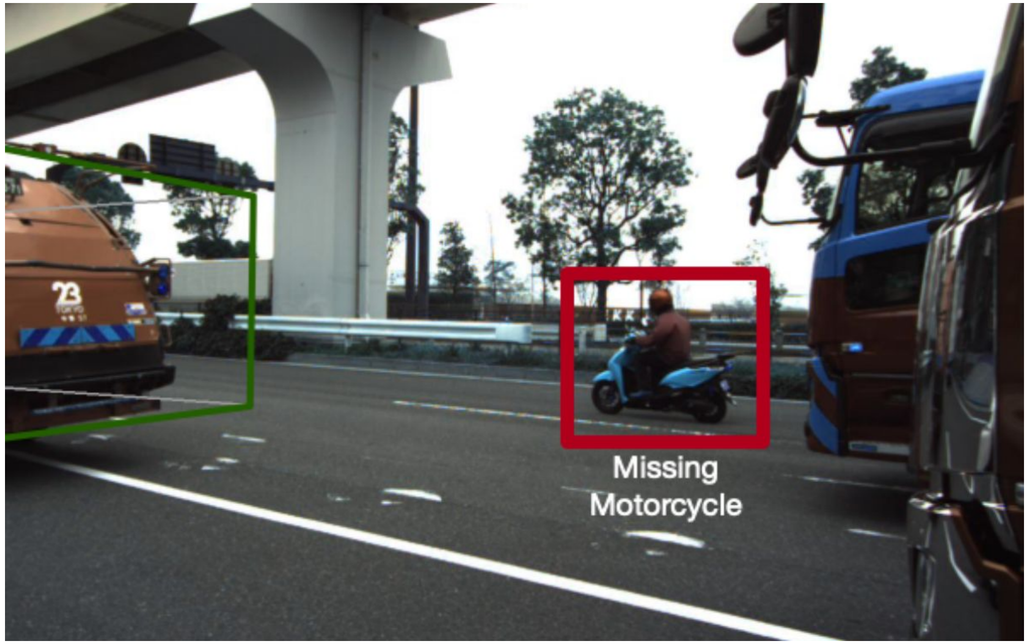
Can we design monitoring / QA methods that work across the ML deployment stack?

This talk:

Abstractions for finding errors in ML deployments and in labeling pipelines

Errors in ML models and labels can be systematic

Cars should not flicker in and out of a video



Labelers consistently miss certain objects

THE VERGE

TECH \ TRANSPORTATION \ UBER

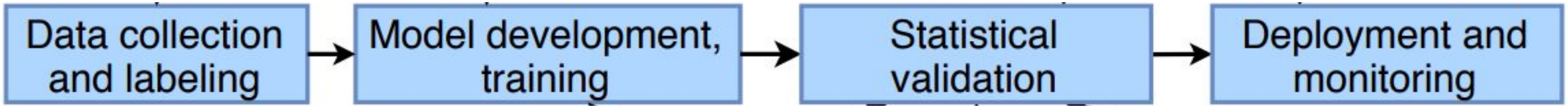
Serious safety lapses led to fatal self-driving crash, new documents suggest

“As the [automated driving system] **changed the classification** of the pedestrian several times—**alternating between vehicle, bicycle, and an other** — the system was unable to correctly predict the path of the detected object,” the board’s report states.

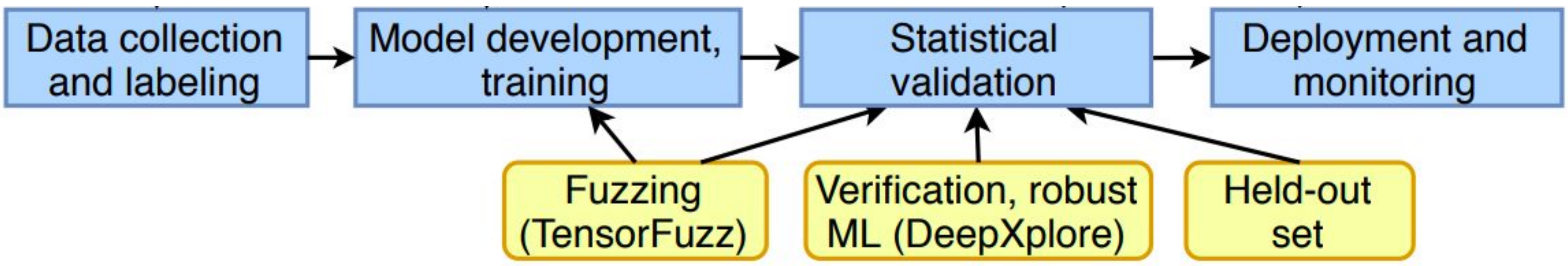
Outline

- Motivation
- Model assertions
- Learned observation assertions (LOA)

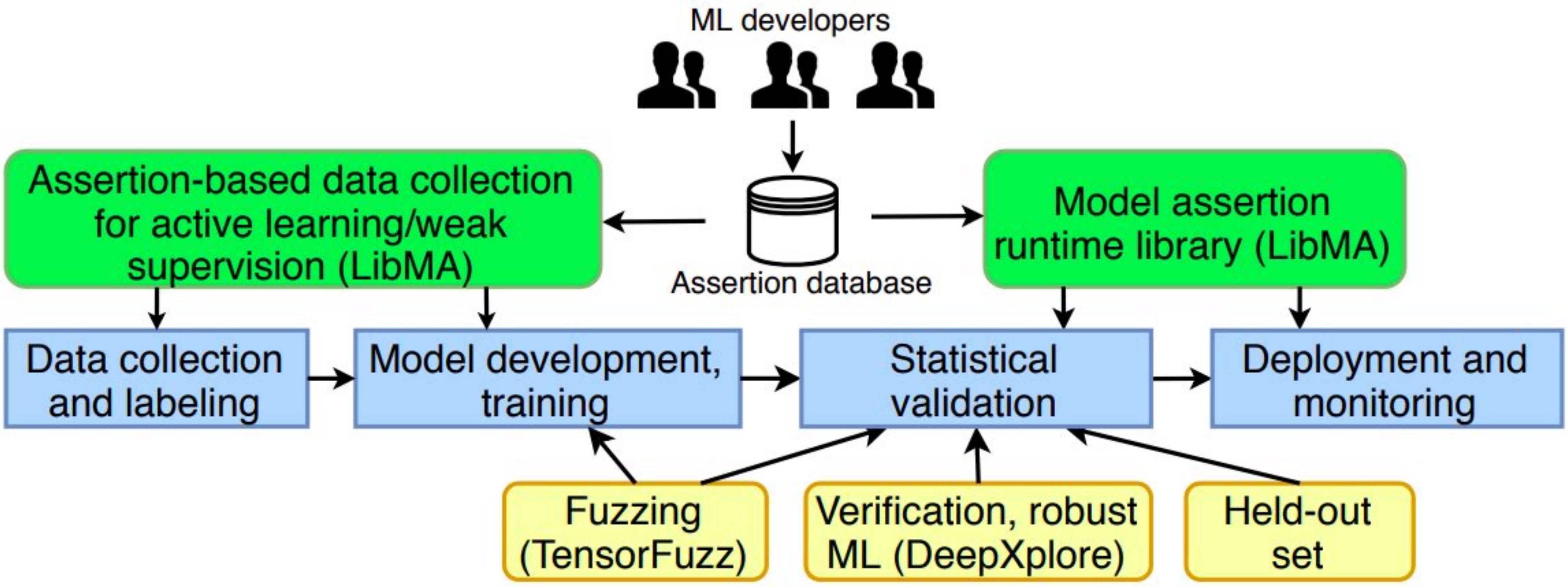
Model assertions in context



Model assertions in context



Model assertions in context



Many users, potentially not the model builders, can collaboratively add assertions

Model assertions for finding errors and improving ML models [MLSys '20]

```
def flickering(
    recent_frames: List[PixelBuf],
    recent_outputs: List[BoundingBox]
) -> Float
```

Model assertion inputs are a history of inputs and predictions

Model assertions output a severity score, where a 0 is an abstention

Predictions from different AV sensors should agree

Assertions can be specified in little code

```
def sensor_agreement(lidar_boxes,
camera_boxes):
  failures = 0
  for lidar_box in lidar_boxes:
    if no_overlap(lidar_box, camera_boxes):
      failures += 1
  return failures
```

Specifying model assertions: consistency API

Identifier	Time stamp	Attribute 1 (gender)	Attribute 2 (hair color)
1	1	M	Brown
1	2	M	Black
1	4	F	Brown
2	5	M	Grey

Transitions cannot happen too quickly

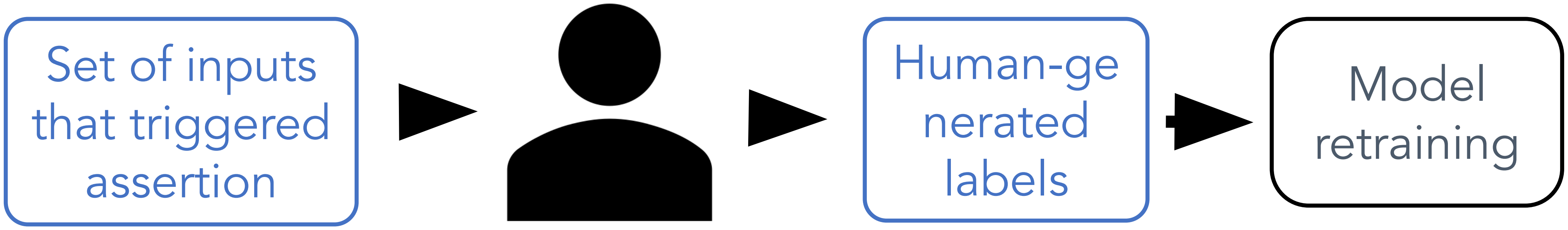
Attributes with the same identifier must agree

Model assertions for TV news analytics

Overlapping boxes in the same scene should agree on attributes

Automatically specified via consistency assertions

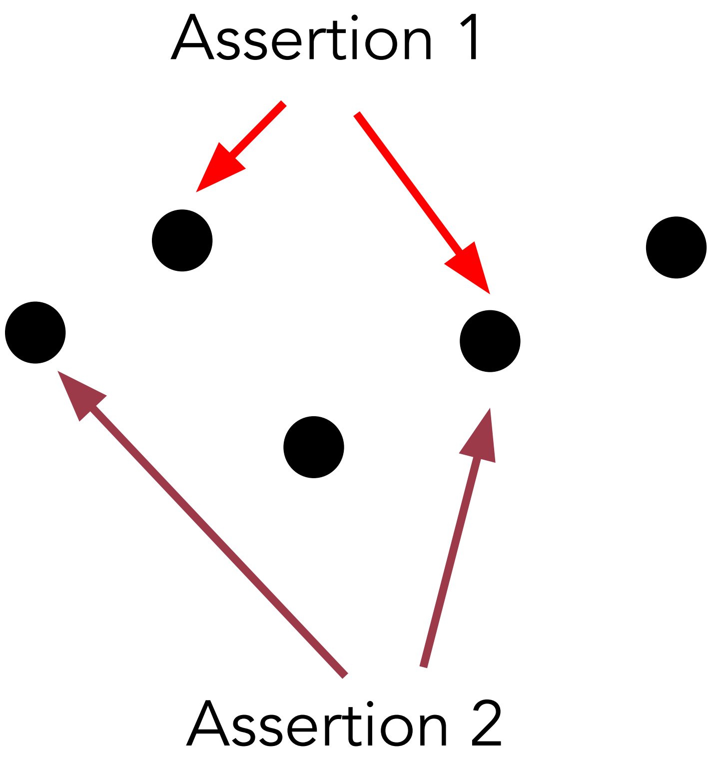
Training models via model assertions



Agnostic to data type, task, and model!

New data collection API

How should we select data points to label for active learning?



- Many assertions can flag the same data point
- The same assertion can flag many data points
- Which points should we label?

Model assertion-based bandit algorithm

Evaluation setting

- Deployed MAs on real world datasets (more in paper)
  - Video analytics
  - Self-driving cars
  - ECG readings

Goals:
Find errors
Retrain models

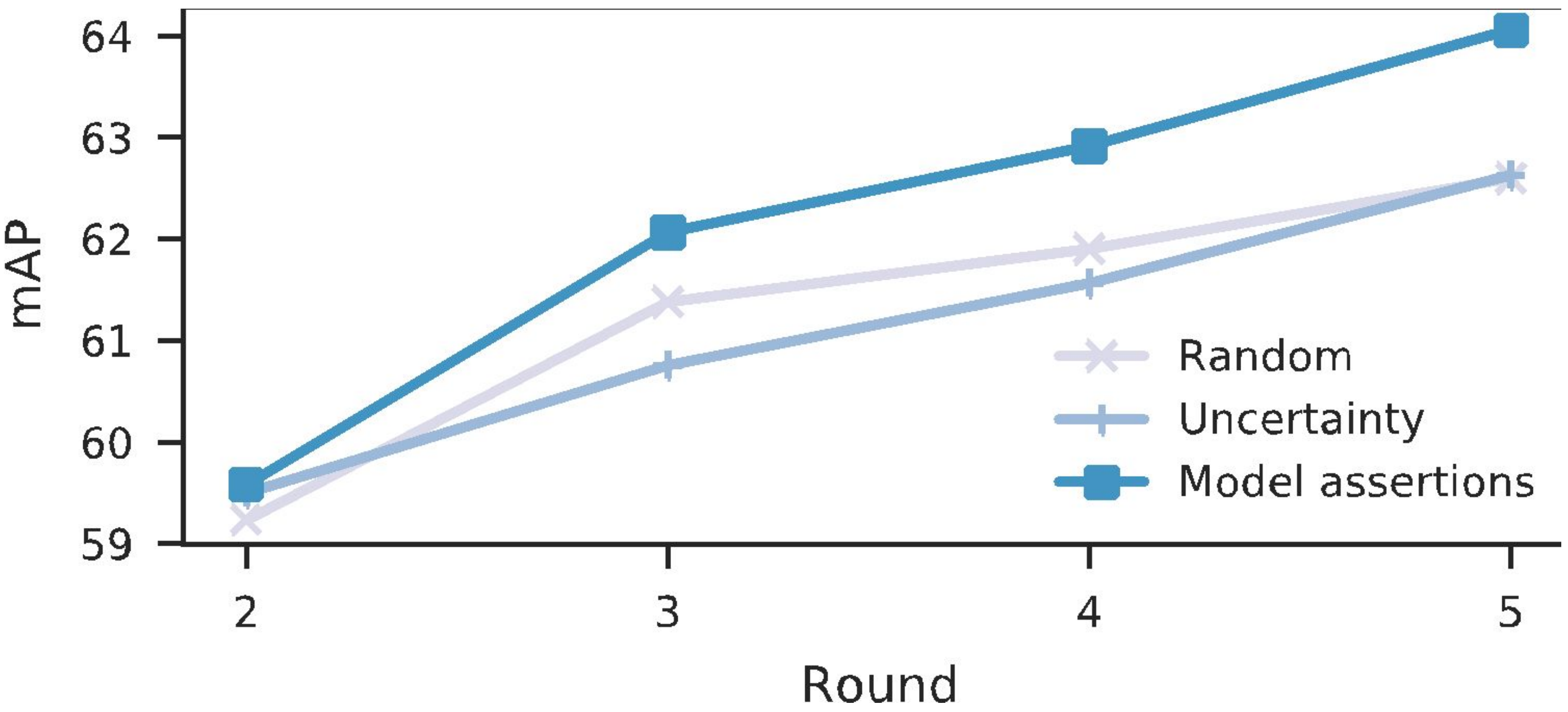
→

Metrics:
Precision
mAP

Model assertions can find errors with high true positive rate

Setting	Assertion	True Positive Rate	LOC
Video analytics	Flickering	96%	18
Video analytics	Multibox	100%	14
Video analytics	No phantom cars	88%	18
AV	LIDAR/camera match	100%	11
Medical	ECG classification shouldn’t vary too quickly	100%	23

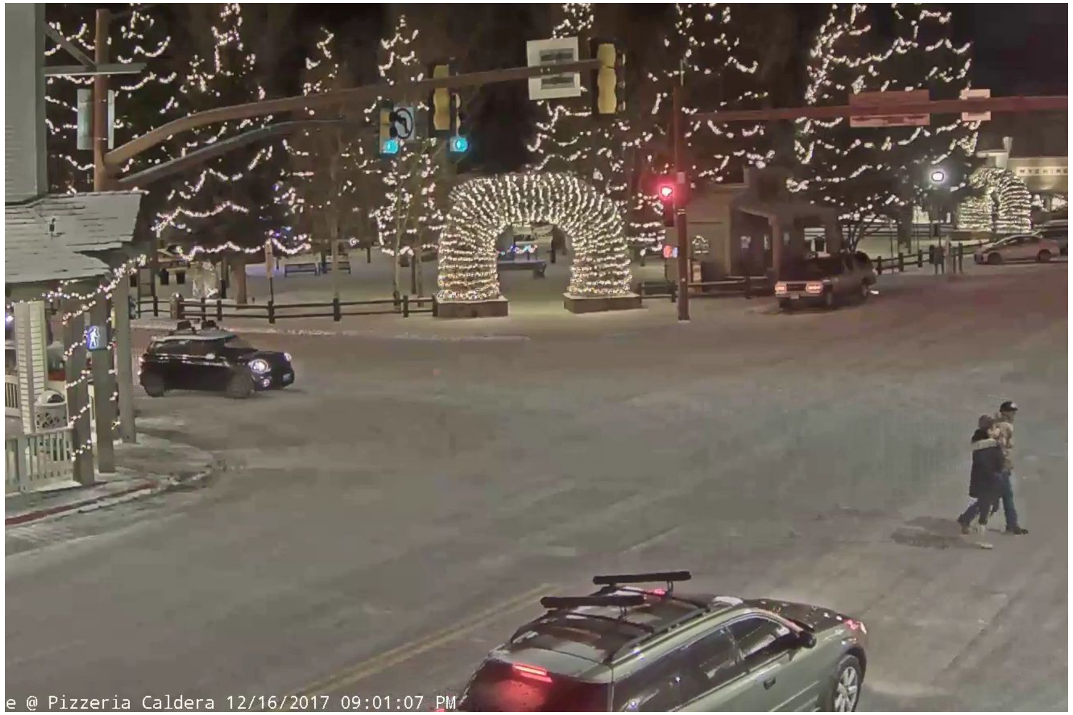
Assertion-based AL outperforms baselines



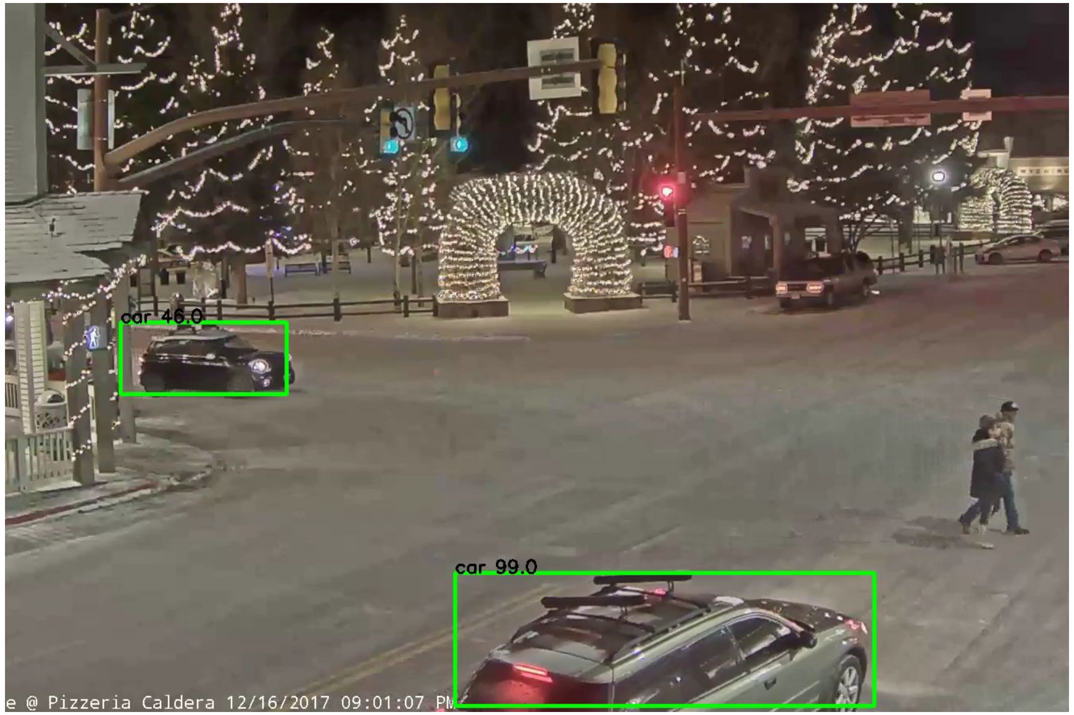
Using assertions outperforms uncertainty and random sampling

Evaluating Model Quality after Retraining: Qualitative Improvement

Original SSD



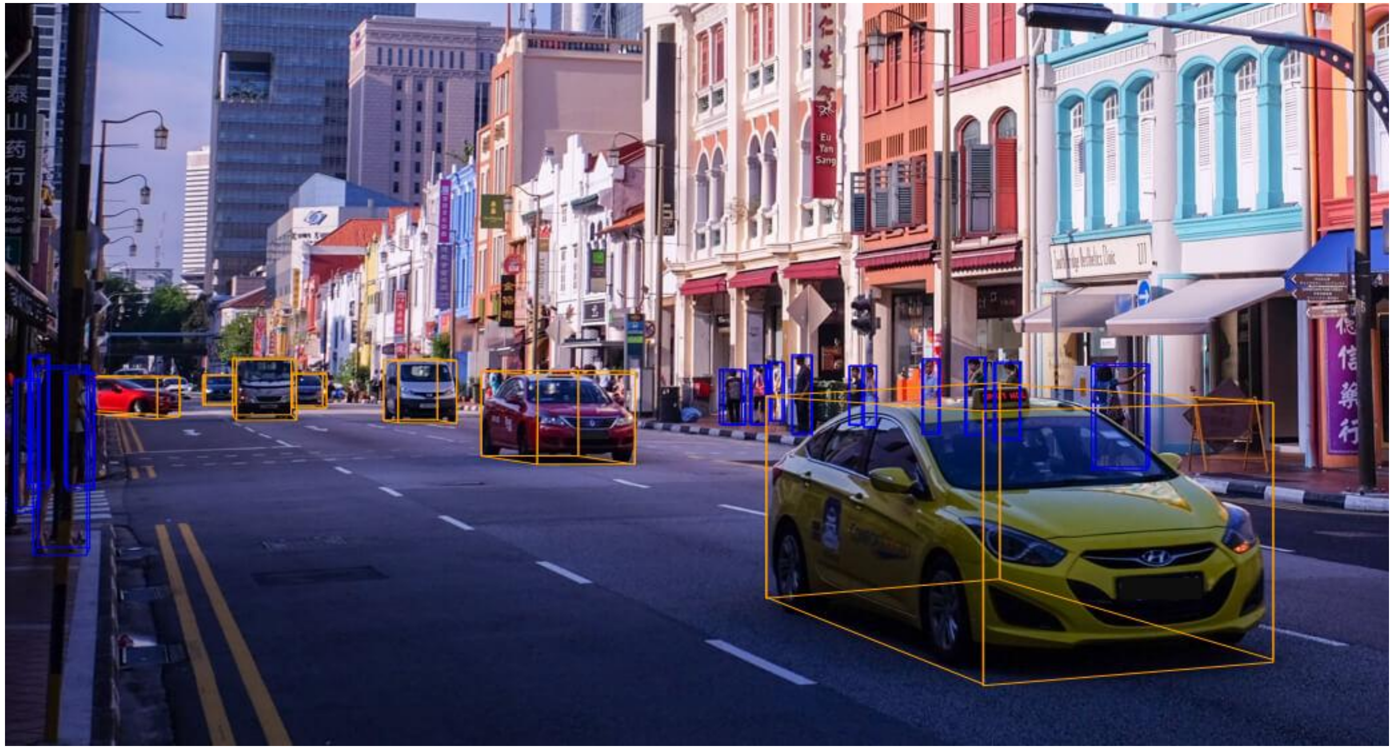
Best Retrained SSD



Outline

- Motivation
- Model assertions
- Learned observation assertions (LOA)

ML models for perception are exploding



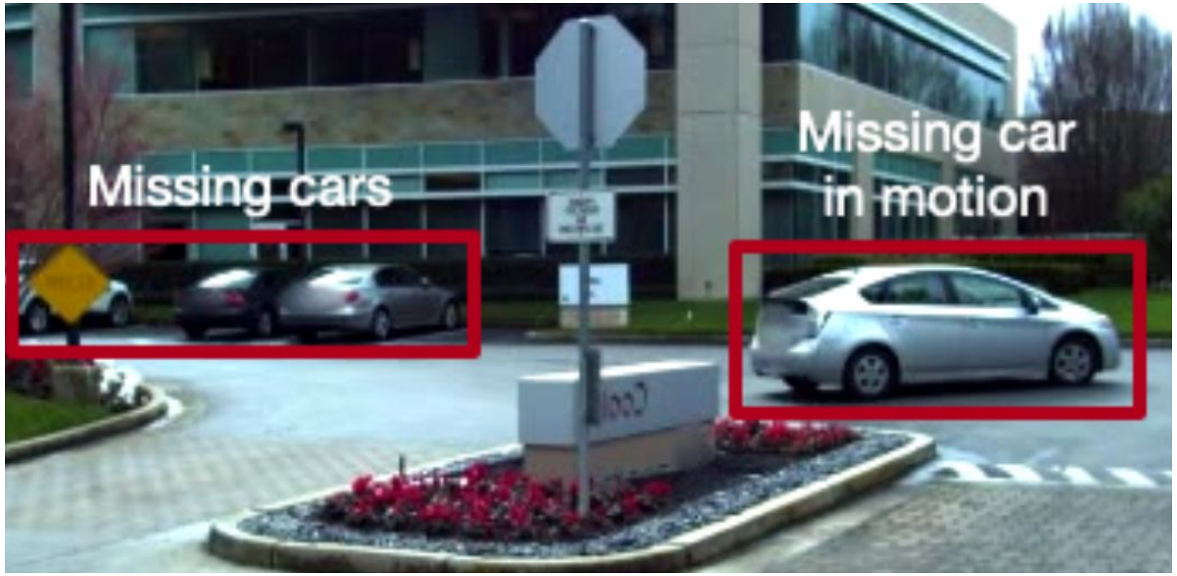
Autonomous vehicles, smart cities, …

ML models for perception require data!

Toyota Research	Voyage	Standard Cognition
OpenAI	Skydio	Otto Motors
Embark	Skip	Optimus Ride
Lyft	Sea Machines	States Title

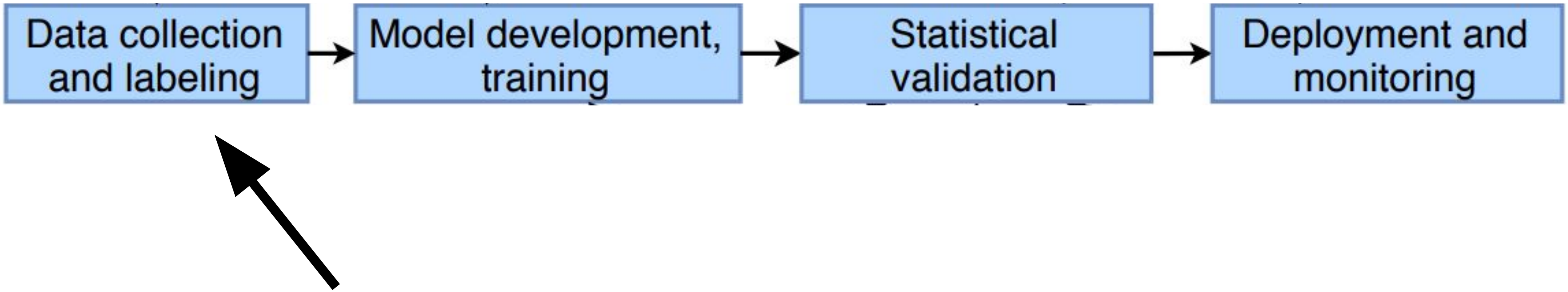
Labeling vendor (e.g., Scale AI) have millions in revenue, hundreds of customers!

Training data is rife with errors!



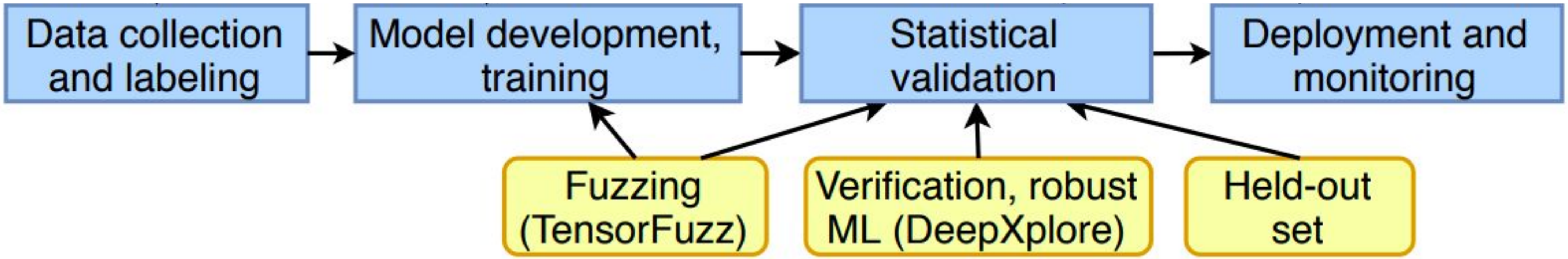
Even the best-in-class labeling services
misses critical labels!

ML pipelines require data

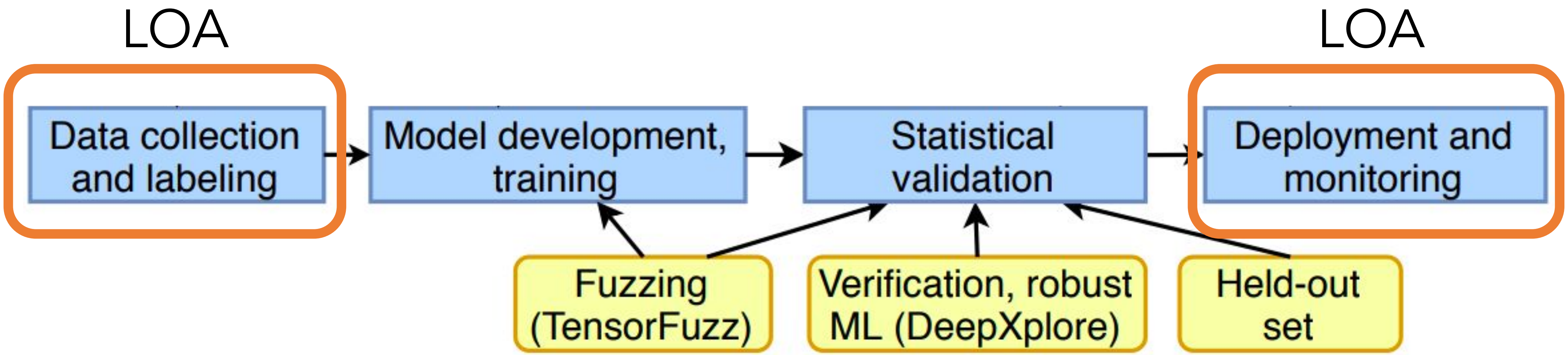


Critical component for ML deployments!

LOA in context

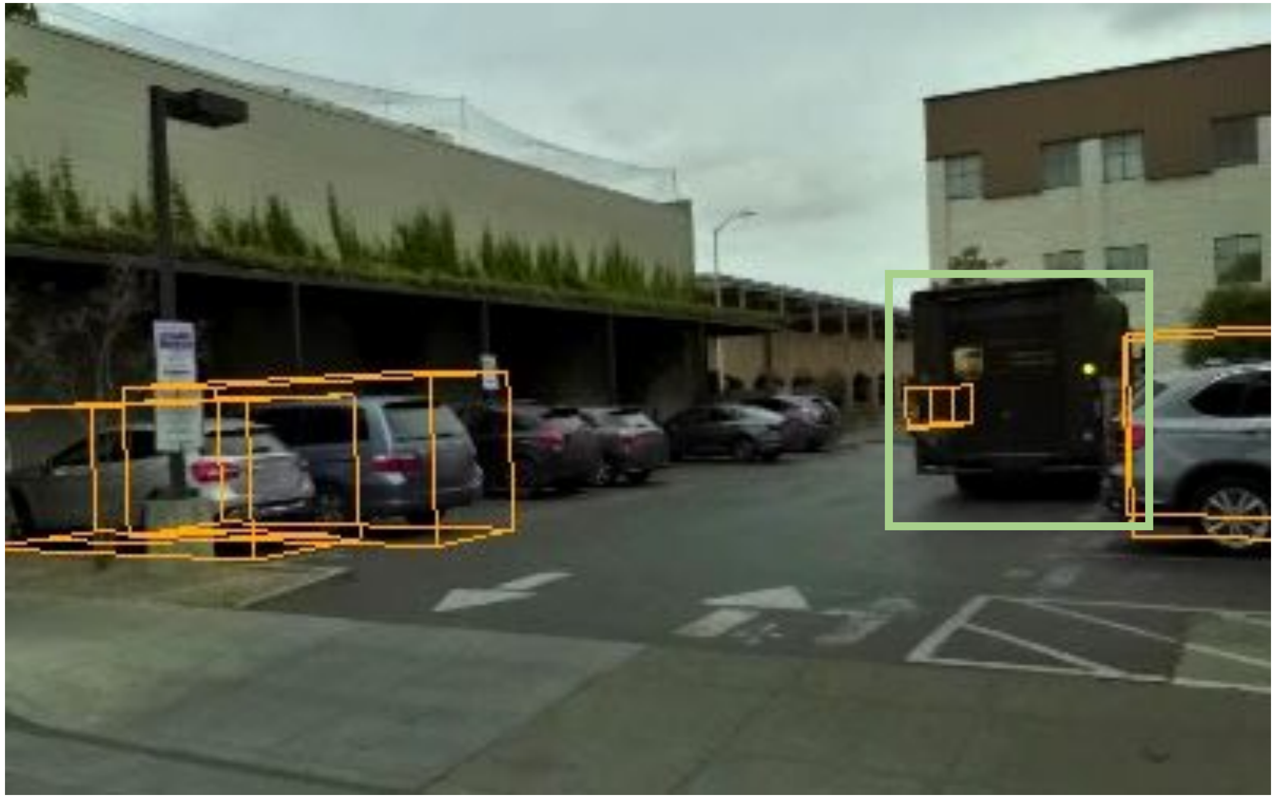


LOA in context



Vetting training data is critical for safety and liability reasons

Finding errors in labels via ML models

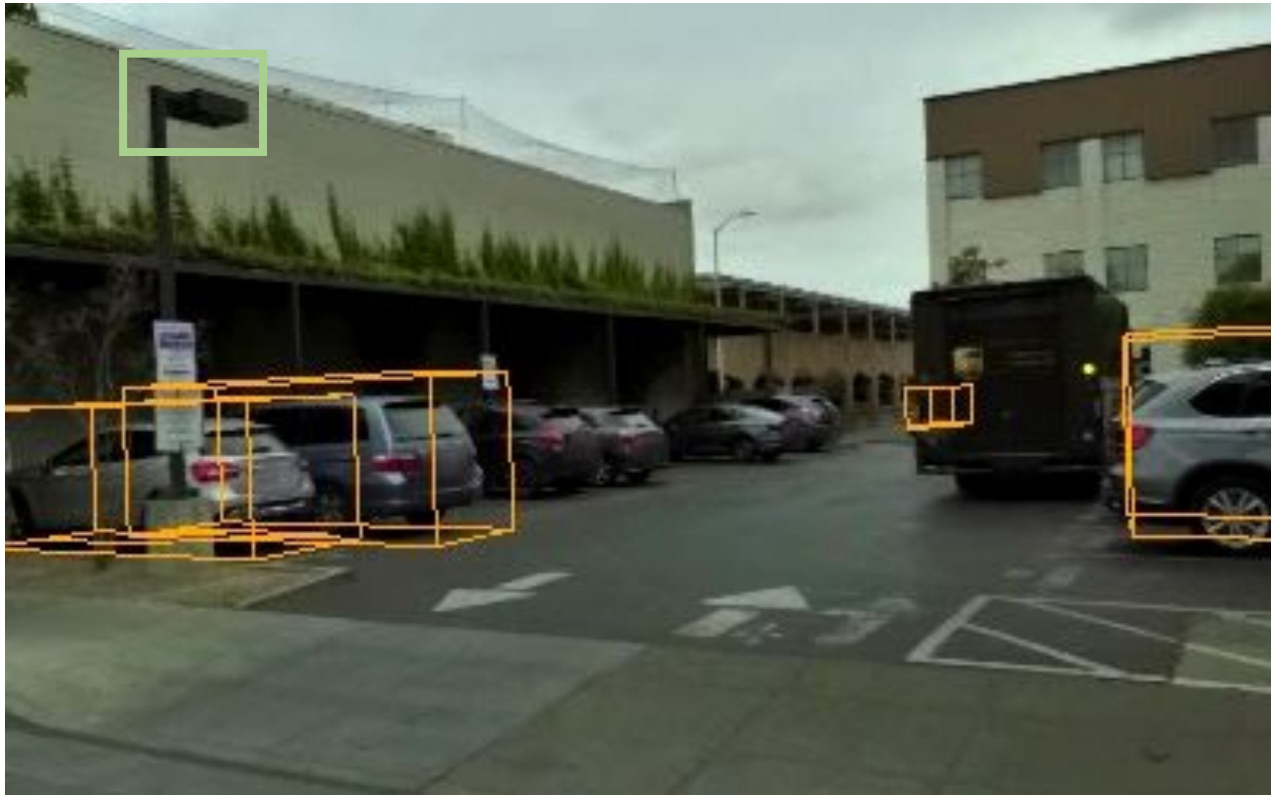


Human annotation

Model prediction

Model is correct, human label is incorrect

Challenge: models can be unreliable!



Human annotation

Model prediction

Model is incorrect, human label is correct

How can we specify which model predictions are likely errors?

Inputs to LOA

Application user:

- Features
- Associations

System administrator:

- ML model predictions
- Existing labels

LOA example: features

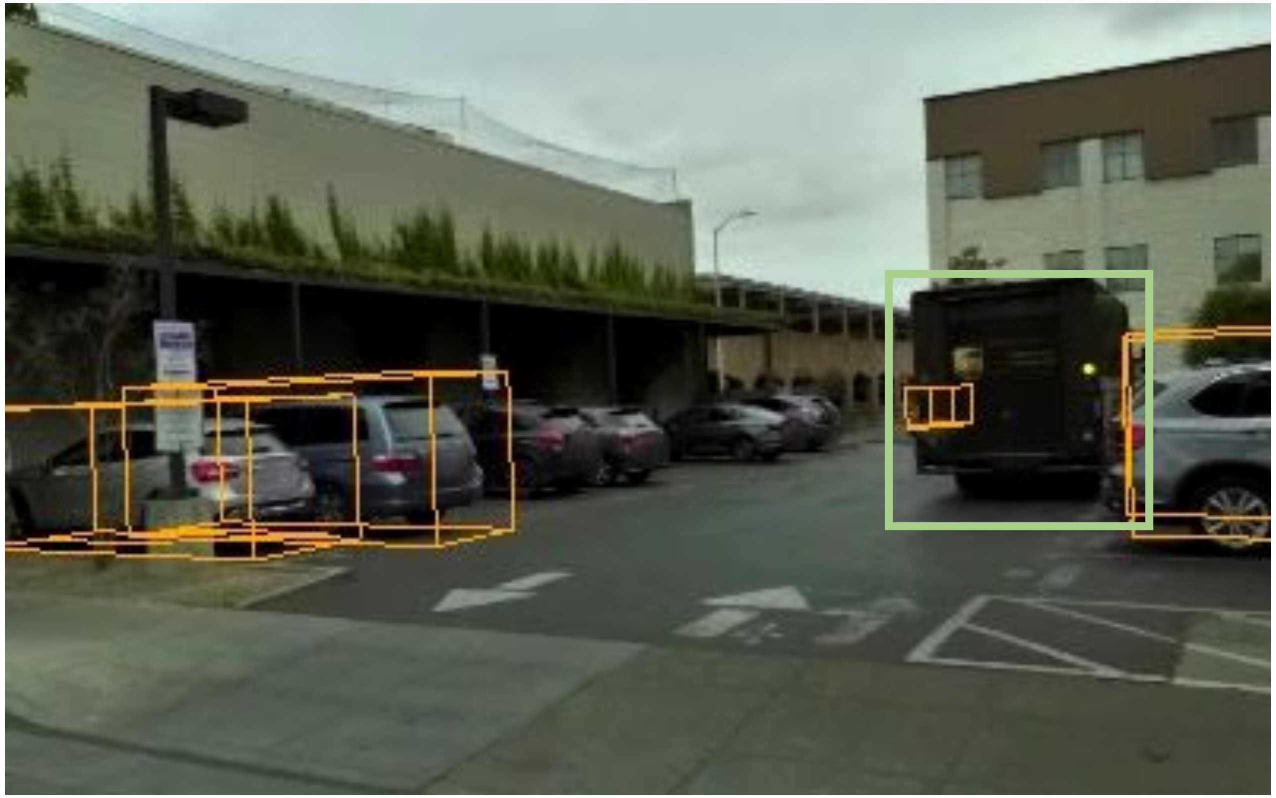
```
def VolumeFeature(box):
  return box.width * box.height * box.length

def VelocityFeature(box1, box2, time):
  return (box1.center – box2.center) / time
```

LOA example: associations

```
def Association(box1, box2):
  return overlaps(box1, box2)
```

Organizational resources: ML models

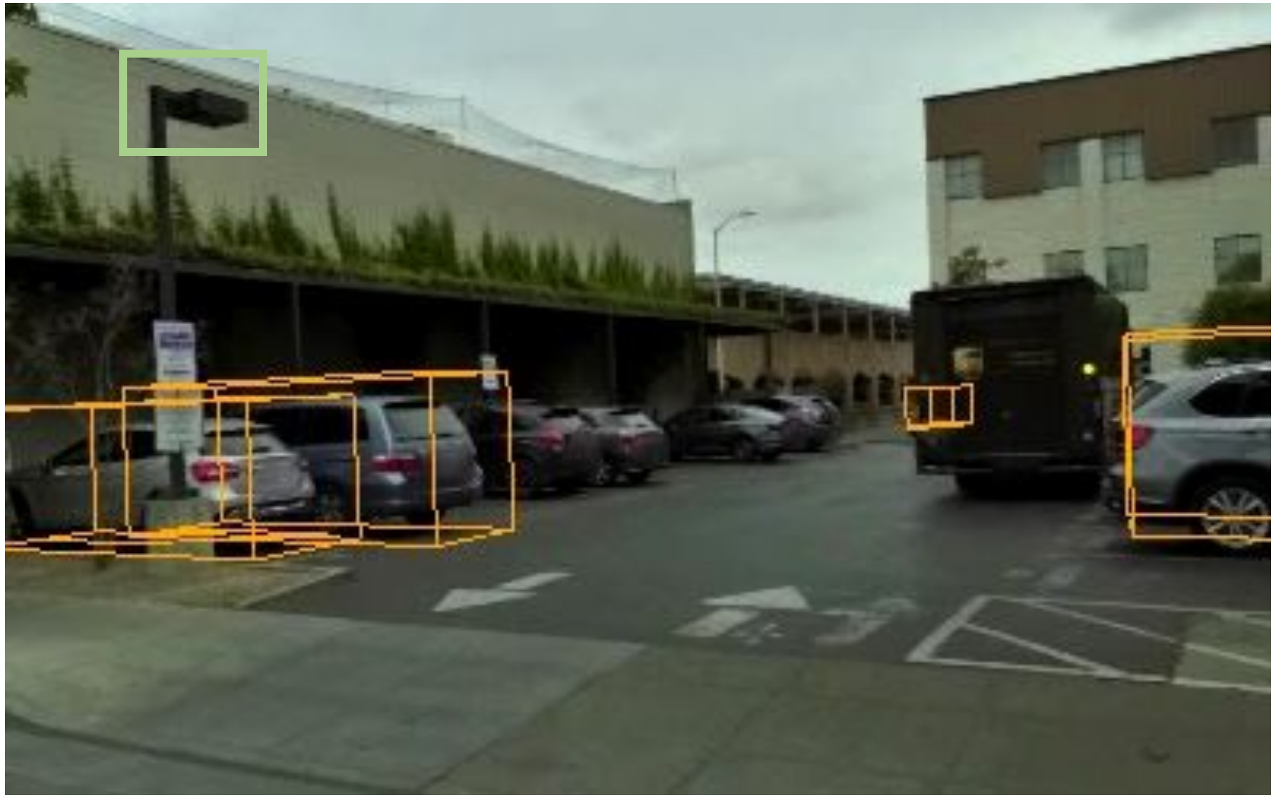


Human annotation

Model prediction

ML models can provide information about potentially missing tracks

Challenge: models can be unreliable!

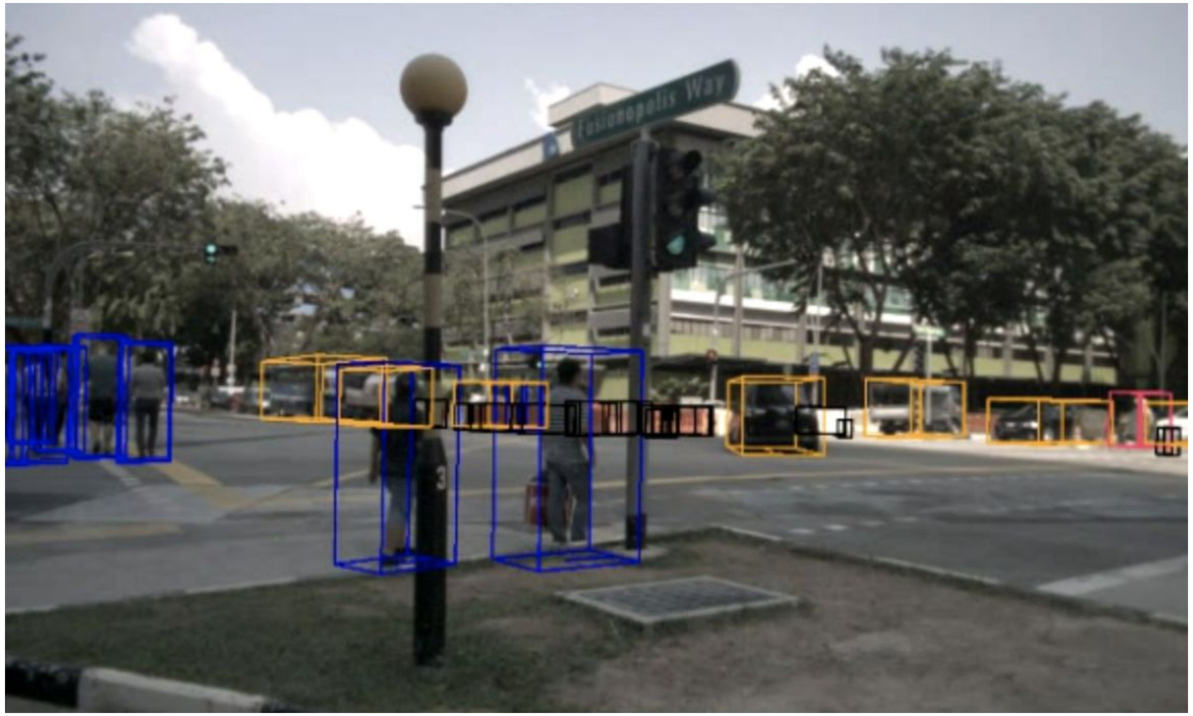


Human annotation

Model prediction

Model is incorrect, human label is correct

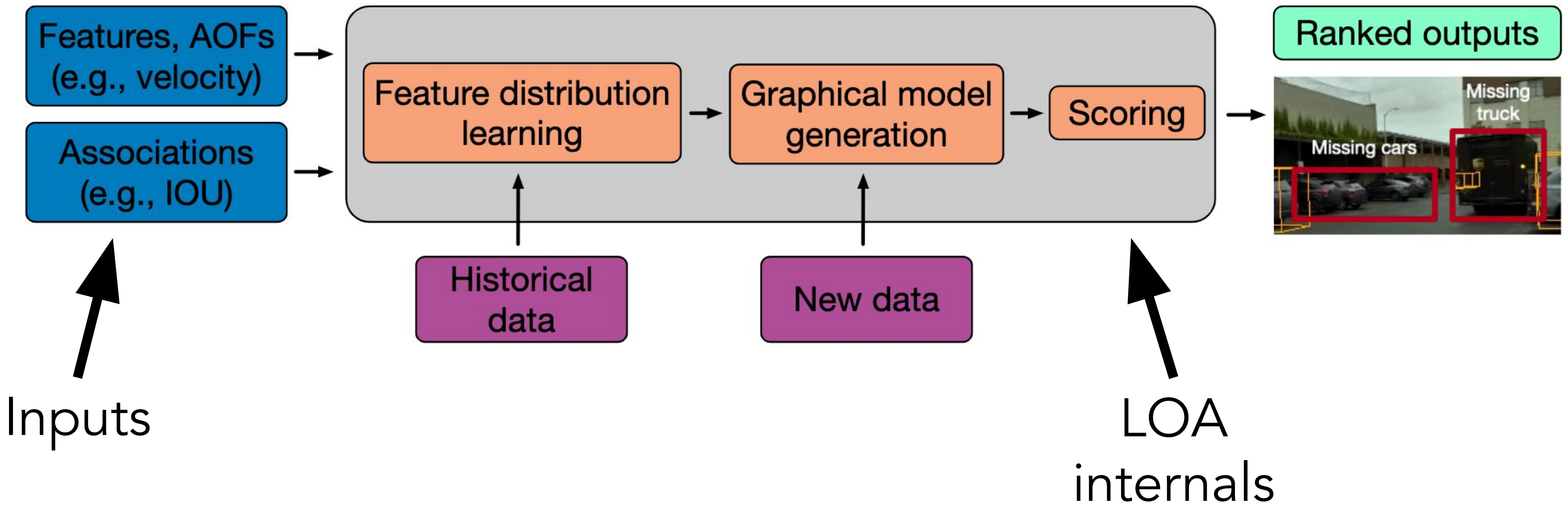
Organizational resources: existing human labels



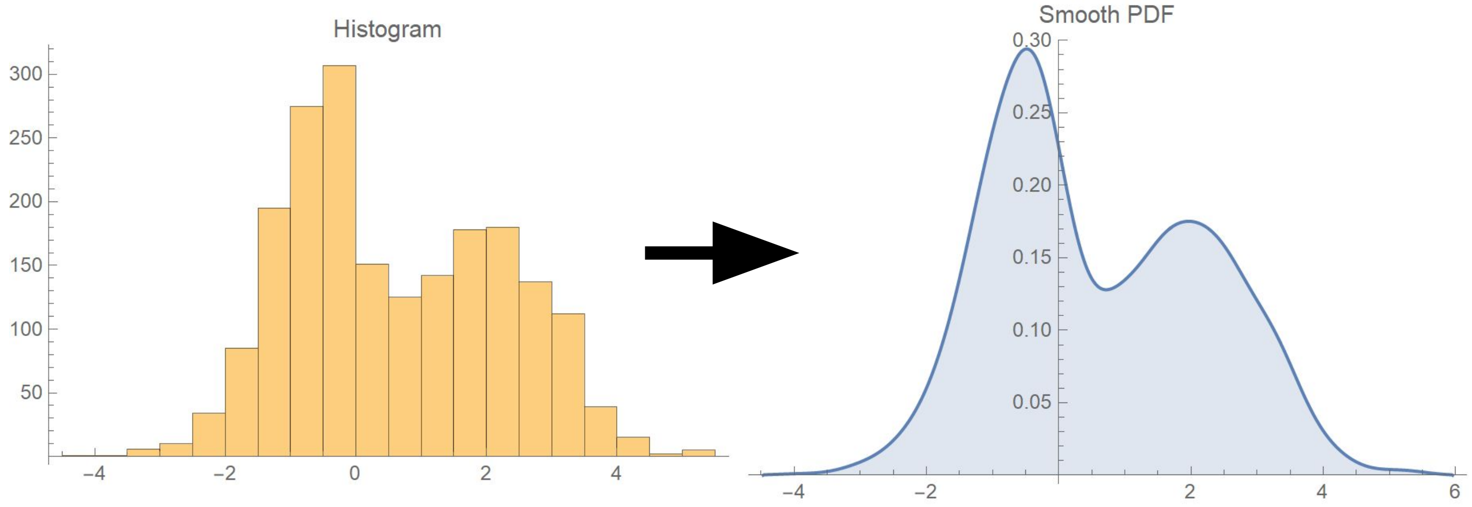
Existing labels can provide examples of expected behavior:

- Box volume
- Velocity
- Track lengths
- …

LOA workflow

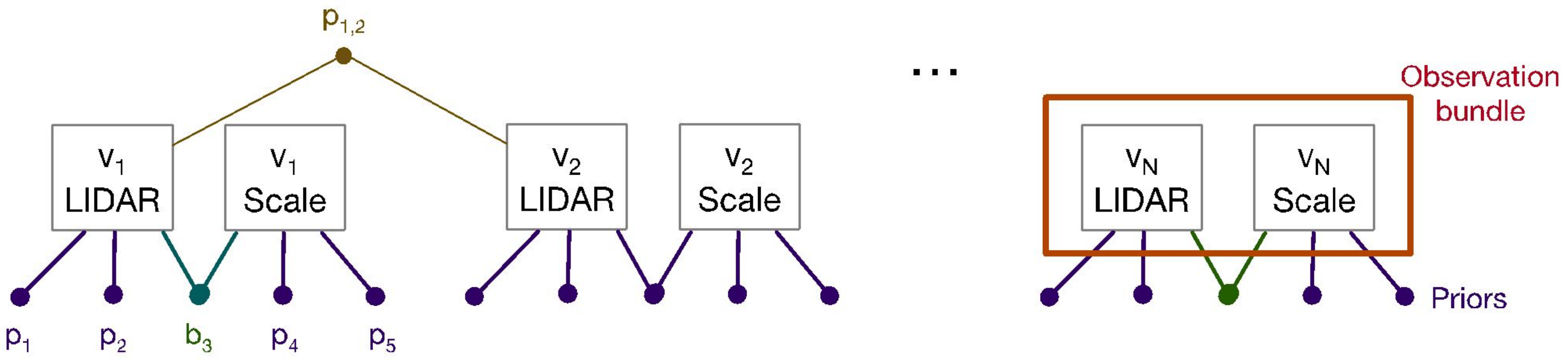


Learning feature distributions



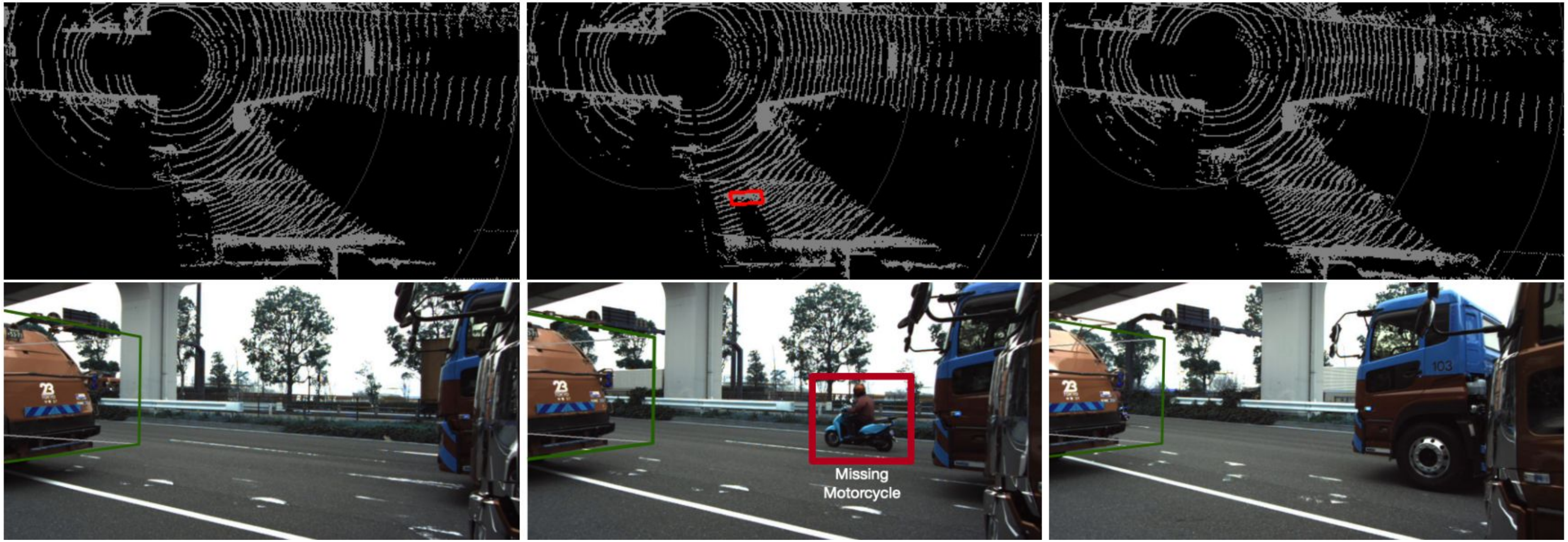
Use existing labels to learn probabilities of expected and unexpected values

Finding errors in labels with LOA



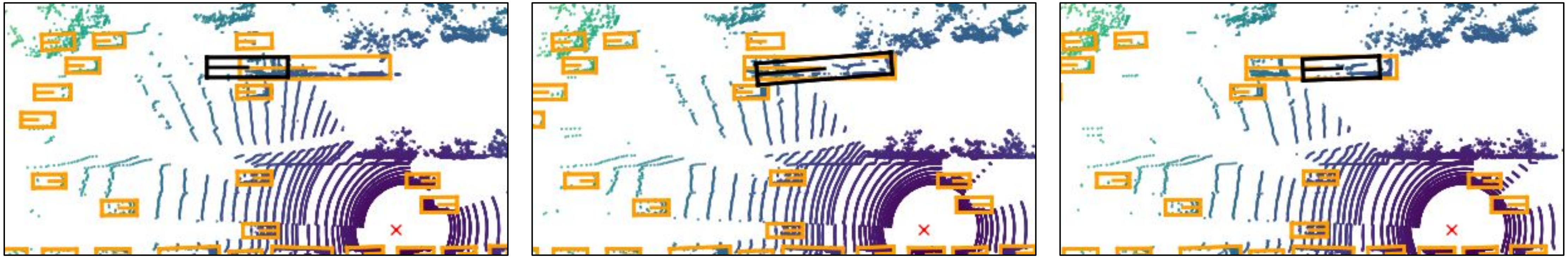
LOA automatically constructs graphical model from features

Proposing missing tracks

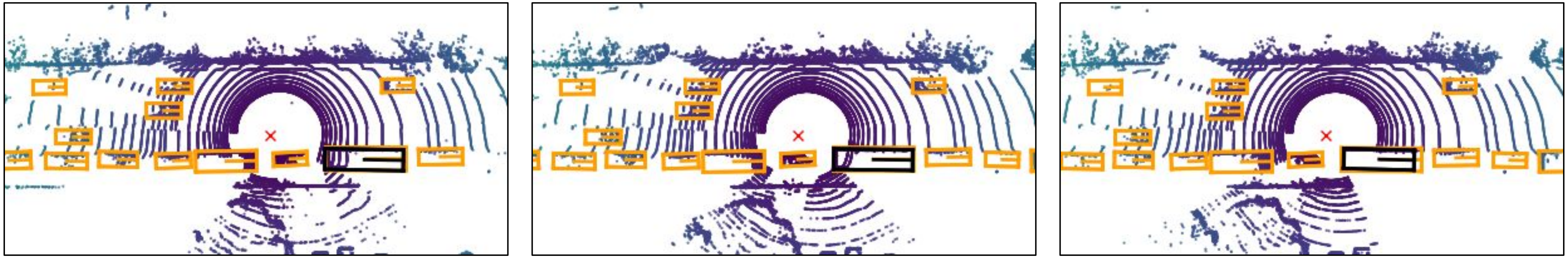


Find differences between labels and model predictions

Unlikely track: inconsistent box volumes



Likely track: consistent features



Scoring tracks

$$\sum_{\pi_i \in \pi(\omega)} \ln\left(f_i(\pi_i(\omega))\right)$$

Aggregating features

Score from feature

Evaluation setting: human labeling errors

Two real autonomous vehicle datasets

- Lyft Level 5 (publicly available)
- Toyota Research Institute (TRI) internal dataset

Goals:		Metrics:
Find errors	→	Recall
Without spurious predictions		Precision

Evaluation setting: human labeling errors

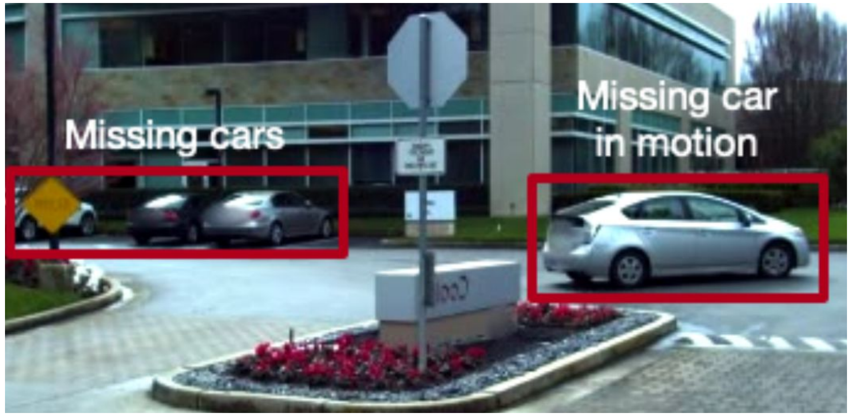
Baseline (model assertions):

- Select model predictions not present in human labels
- Rank randomly or by confidence

LOA:

- Five total features
- <10 LOC per feature

LOA identifies errors in *human labels* in real-world datasets: Lyft Level 5



- Deployed LOA per scene (5-15s clip)
- Found errors in 70% of the Lyft validation scenes (via expert auditor)

Dataset used to train models, host competitions, cited hundreds of times!

LOA identifies errors in human labels in real-world datasets: TRI

- Labels generated from leading vendor!
- Recall of 75% for errors on an exhaustively examined scene (compared to expert auditor)

TOYOTA RESEARCH INSTITUTE

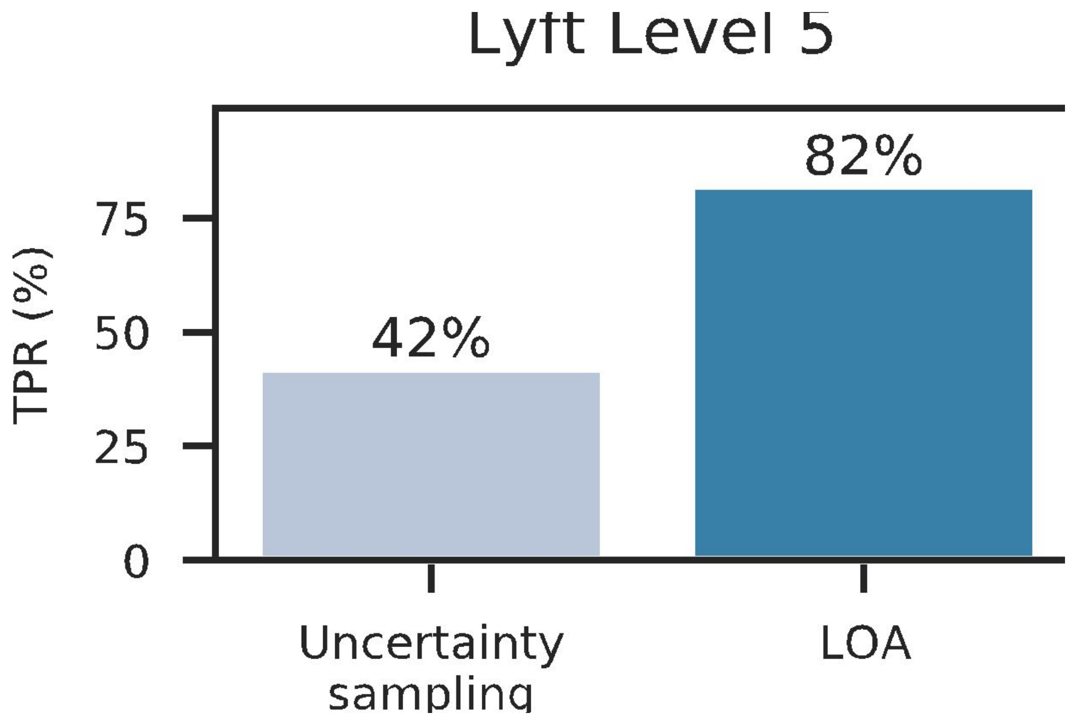
LOA can find errors with high precision

Dataset	Method	Precision at top 5 (across scenes)
Lyft	LOA	**70%**
Lyft	Ad-hoc MA (random)	30%
Lyft	Ad-hoc MA (confidence)	40%
Internal	LOA	**100%**
Internal	Ad-hoc MA (random)	64%
Internal	Ad-hoc MA (confidence)	86%

Evaluation setting: model errors

- Two real autonomous vehicle datasets
  - Lyft Level 5 (publicly available)
  - Toyota Research Institute (TRI) internal dataset
- Exclude errors found by ad-hoc model assertions

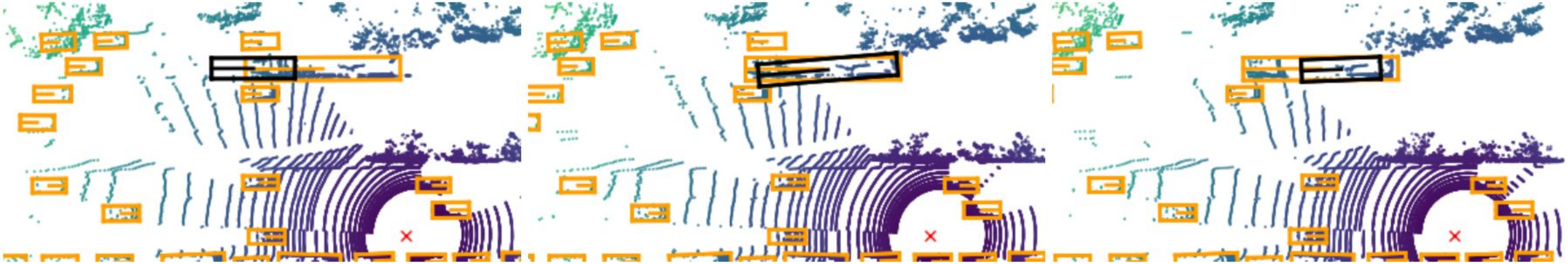
LOA can find errors in ML models not found by model assertions



Excluded model errors found by model assertions

Outperforms uncertainty sampling by ~2x!

Examples of errors in ML models



(a) $t = 0$ (b) $t = 1$ (c) $t = 2$

LOA finds overlapping, but unlikely tracks,
not found by model assertions

Links

- Model assertions paper: https://ddkang.github.io/papers/2020/ma-sysml20.pdf
- Model assertions code: https://github.com/stanford-futuredata/omg
- LOA paper: https://ddkang.github.io/papers/2022/loa-sigmod.pdf
- LOA code: https://github.com/stanford-futuredata/loa

Conclusion

- Errors are rife in both training data and for ML models at deployment time
- We present model assertions and LOA, two abstractions for finding errors in ML pipelines
- We need more work for the ML deployment stack beyond training!

ddkang@stanford.edu

@daniel_d_kang