Lecture 9:

System Support for Curating Supervision

Parallel Computing
Stanford CS348K, Spring 2020

Note

- Much of this class involved discussing the Snorkel paper(s)

- I am posting these slides as some were used during parts of the discussion

Today's theme

- Data is not precious. Today, in many domains large collections of *unlabeled data* are readily accessible

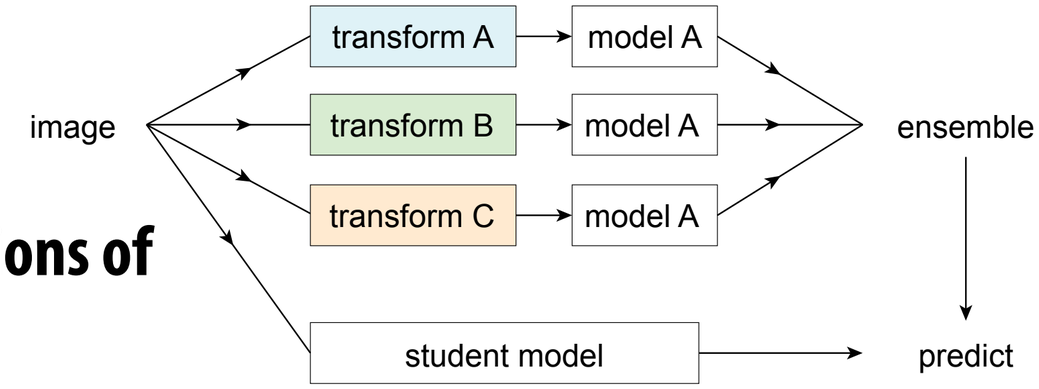
- But labels (supervision) for this data are extremely precious

- Implication: ML engineers are interesting in using any means necessary to acquire sources of supervision

Obtaining supervision using knowledge contained in preexisting models

Using model to supervise itself

- **Example: omni-supervised learning**
- **Evaluate existing model on different augmentations of unlabeled image**
  - **Original model trained using labeled training set**
  - **Ensemble predictions to estimate label for image**
- **Re-train model on both labeled images AND estimated label images**



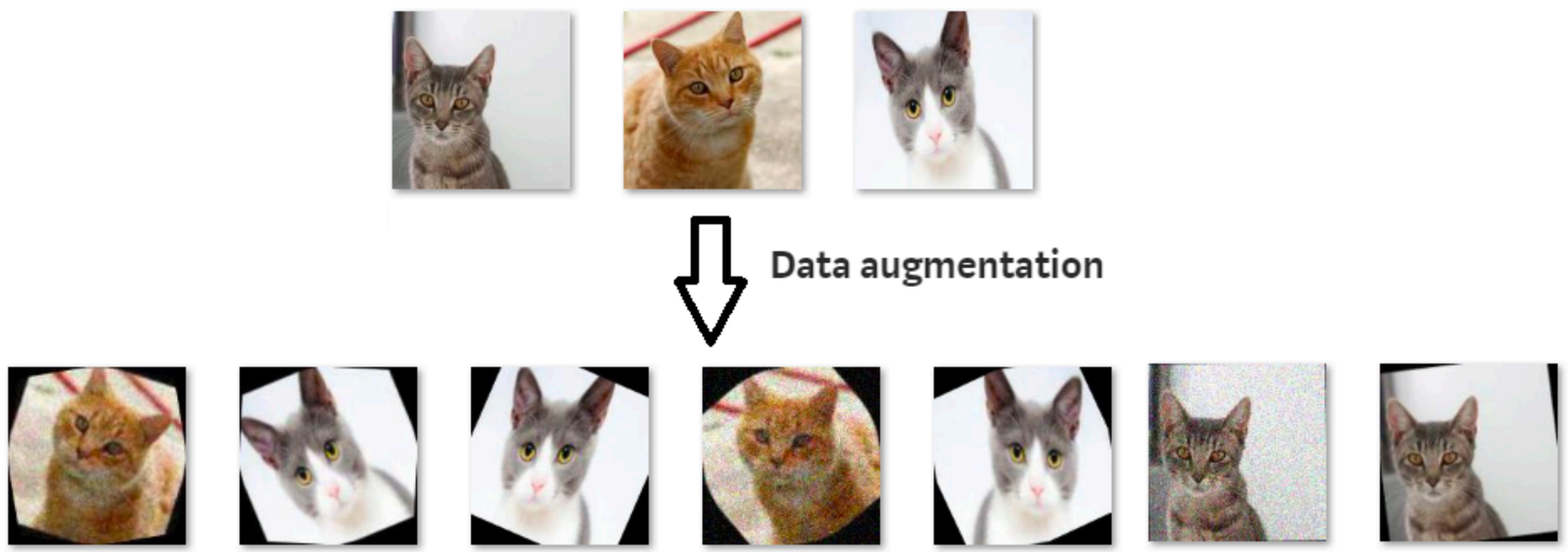
backbone	DD	AP	AP_{50}	AP_{75}	AP_S	AP_M	AP_L
ResNet-50		37.1	59.1	39.6	20.0	40.0	49.4
ResNet-50	✓	**37.9**	**60.1**	**40.8**	**20.3**	**41.6**	**50.8**
ResNet-101		39.2	61.0	42.3	21.7	42.9	52.3
ResNet-101	✓	**40.1**	**62.1**	**43.5**	**21.7**	**44.3**	**53.7**
ResNeXt-101-32×4		40.1	62.4	43.2	22.6	43.7	53.7
ResNeXt-101-32×4	✓	**41.0**	**63.3**	**44.4**	**22.9**	**45.5**	**54.8**

[Source: Radosavovic et al. CVPR 2018]

Key idea: data augmentation

Make machine-provided labels more robust by ensembling output from multiple perturbations of the image

Another augmentation example:



[Source: https://medium.com/@thimblot/data-augmentation-boost-your-image-dataset-with-few-lines-of-python-155c2dc1baec]

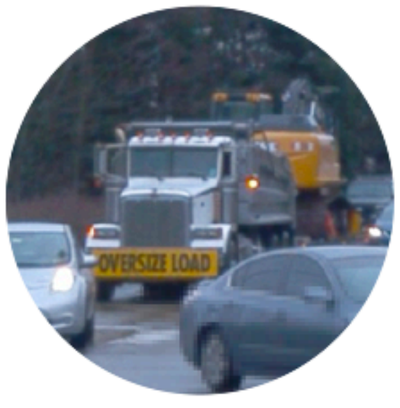
Label transfer via visual similarity

- If I know this image contains a cactus, then visually similar images likely also contain cacti as well.

Saguaro cactus

visually similar

- What are good ways to define similar?



"Oversize load"

has same text

https://blog.waymo.com/2020/02/content-search.html

Providing supervision by writing programs

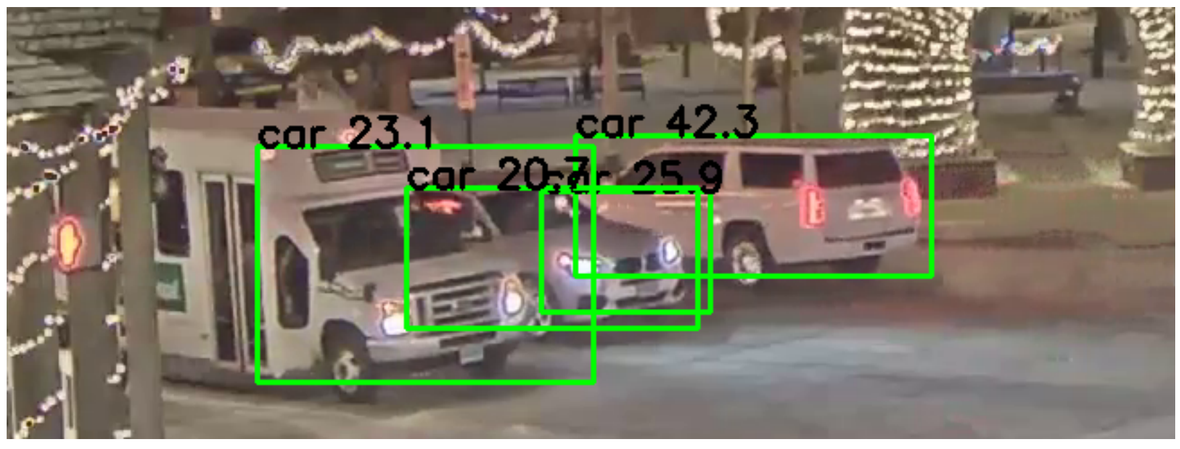
Encode external priors in programs

- **Example: temporal consistency prior: state of world should not change significantly from frame to frame**

(a) Frame 1, SSD (b) Frame 2, SSD (c) Frame 3, SSD

- **Example: domain-knowledge prior: objects cannot overlap in space**

(a) Example error 1.



(b) Example error 2.

[Source: Kang et al. MLSys 2020]

DB queries are “detectors”

(find elements in database matching this predicate)

Video Collection

Basic Annotations

Face Detections

```
3:15-3:16: BERNIE...
5:18-5:20: THANK YOU...
9:15-9:17: TODAY IN...
```

Captions

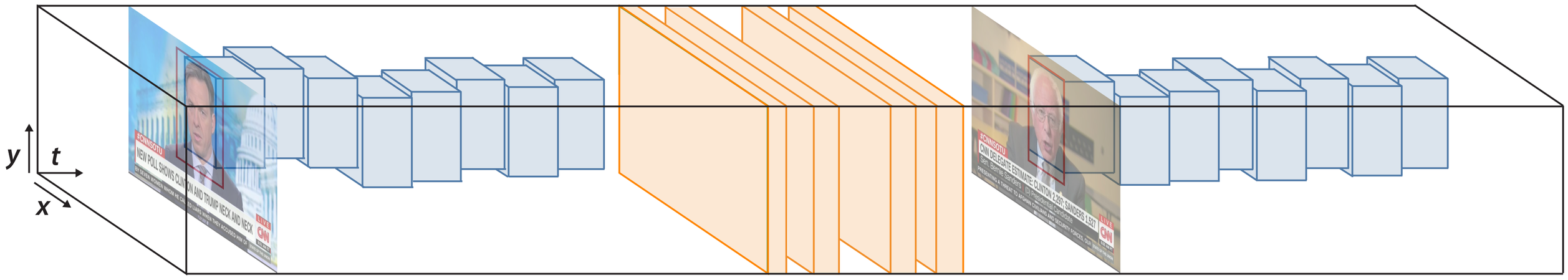
Analyst

```
def bernie_and_jake(faces):
  bernie = faces
    .filter(face.name == “Bernie”)
  jake = faces
    .filter(face.name == “Jake”)

  bernie_and_jake = bernie
      .join(jake,
        predicate = time_overlaps,
        merge_op = span)

  return bernie_and_jake
```

[Source: Fu et al. 2019]

Data model: spatiotemporal labels

All video annotations represented by labels associated with spatiotemporal volume in video



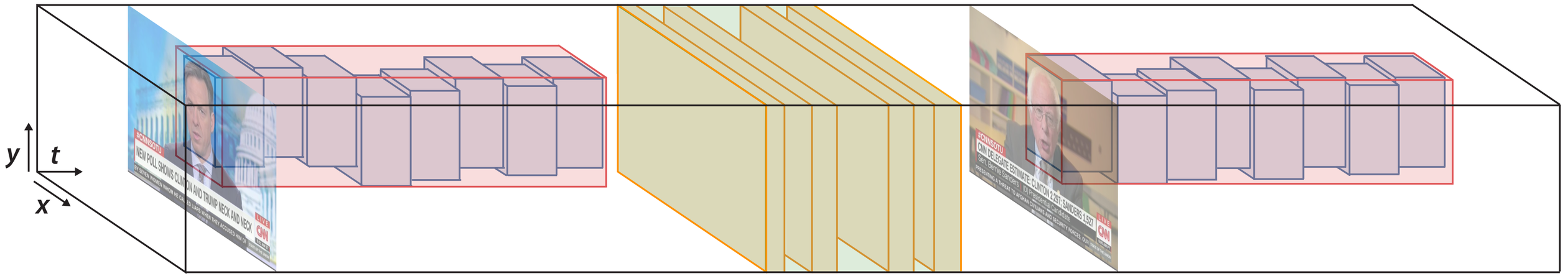
Per-frame face detection:
Jake Tapper

And joining me now

Per-frame face detection:
Bernie Sanders

Data model: spatiotemporal labels

Labels can be nested (hierarchical)

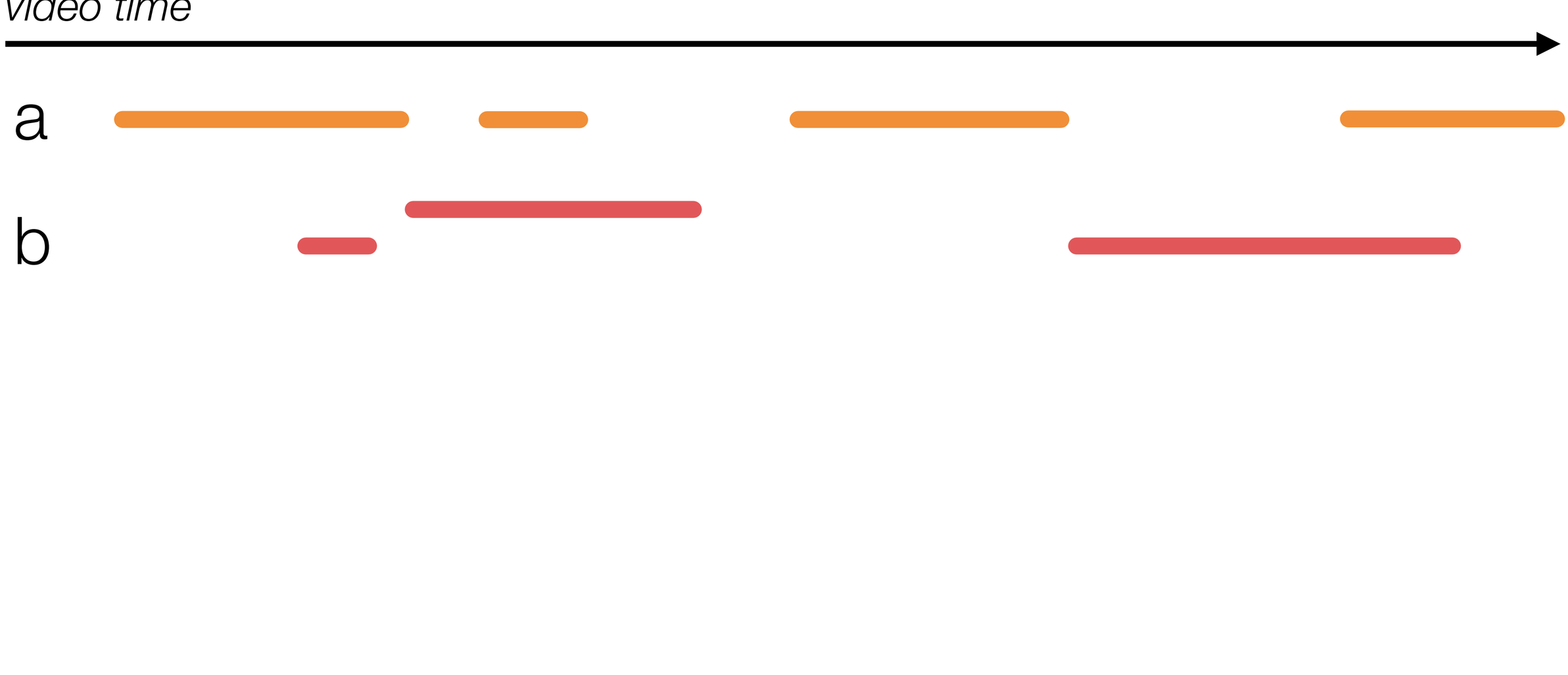


Programming model: create new labels via operations on sets of spatiotemporal labels

Spatiotemporal join:

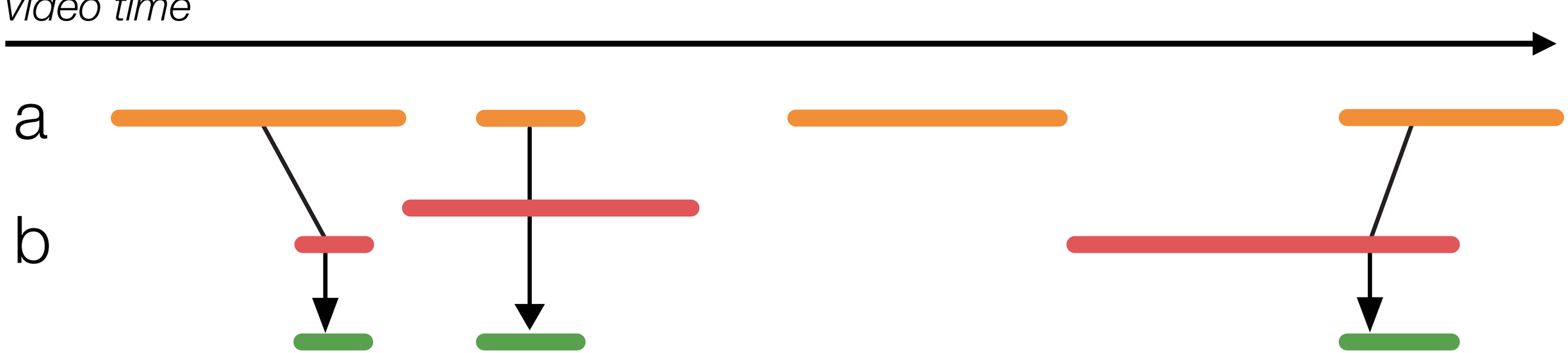
Programming model: create new labels via operations on sets of spatiotemporal labels

Spatiotemporal join:



Programming model: create new labels via operations on sets of spatiotemporal labels

Spatiotemporal join:

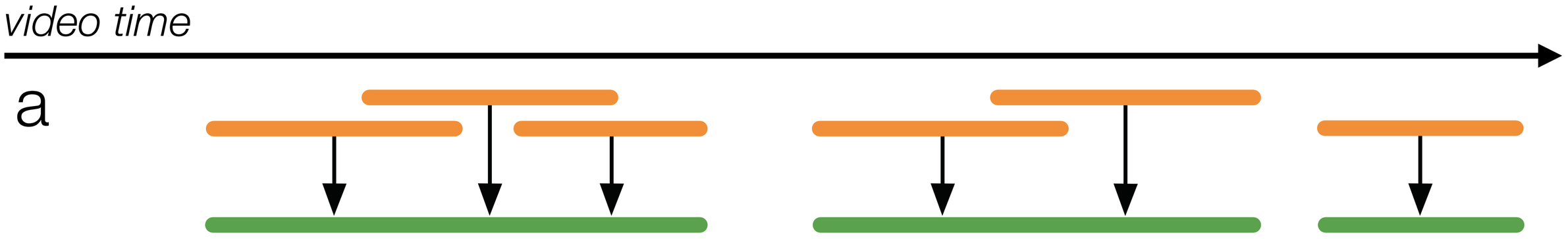


Three-person panels

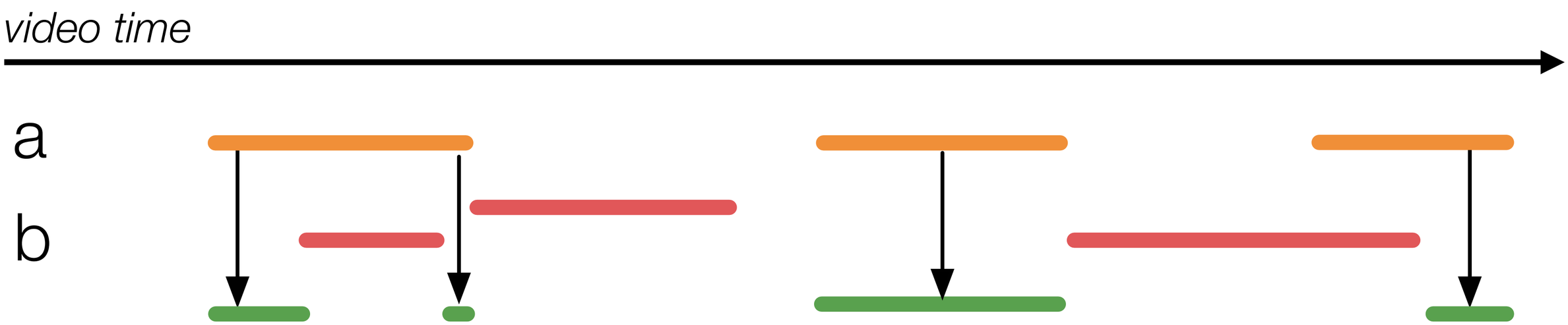
(three faces, bounding boxes greater than 30% of screen height, in horizontal alignment)

Additional label composition operators

Coalesce: merging nearby labels



Minus: remove volume of one label set from another



Interview event detection example

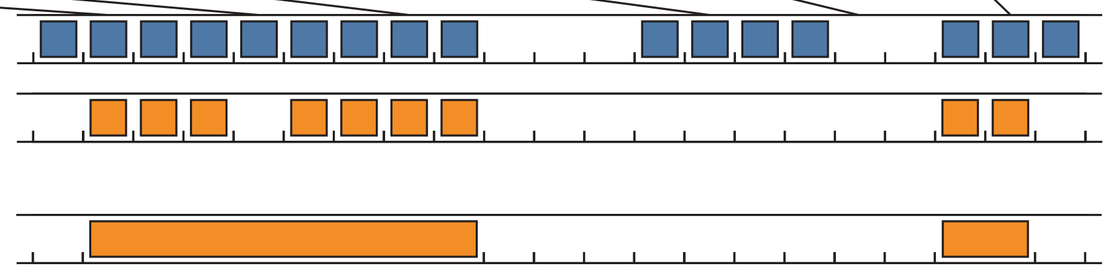

```
faces = rekall.ingest(database.table("faces"), 3D)
```

Input: labels for per-frame face detections

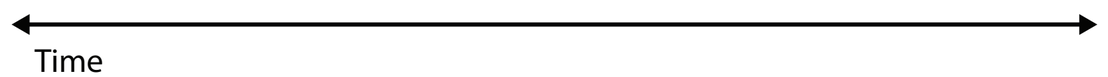
Time

Interview detection example


```
faces = rekall.ingest(database.table("faces"), 3D)

sanders = faces
    .filter(λ face: face.name == "Bernie Sanders")

sanders_segs = sanders
    .coalesce(
      predicate = time_gap < 30 seconds,
      merge = time_span)
```

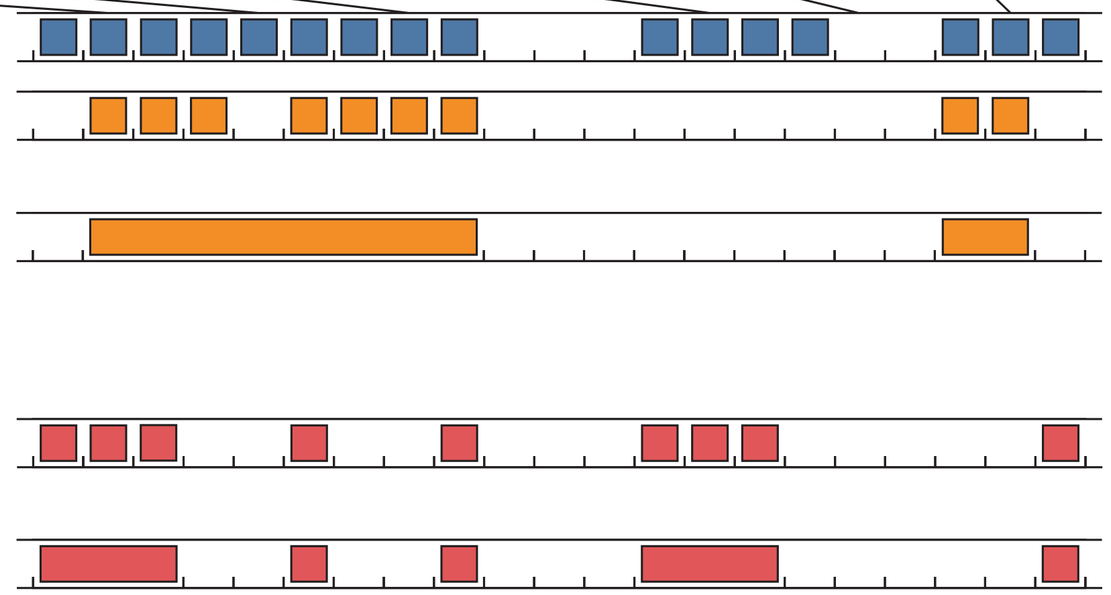


Segments where Sanders is on screen (FILTER, COALESCE)

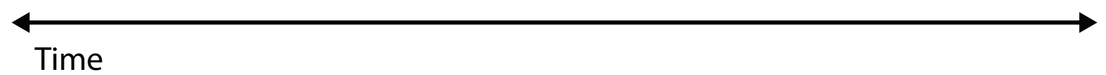


Interview detection example


```
faces = rekall.ingest(database.table("faces"), 3D)

sanders = faces
    .filter(λ face: face.name == "Bernie Sanders")

sanders_segs = sanders
    .coalesce(
      predicate = time_gap < 30 seconds,
      merge = time_span)

tapper = faces
    .filter(λ face: face.name == "Jake Tapper")

tapper_segs = tapper
    .coalesce(
      predicate = time_gap < 30 seconds,
      merge = time_span)
```

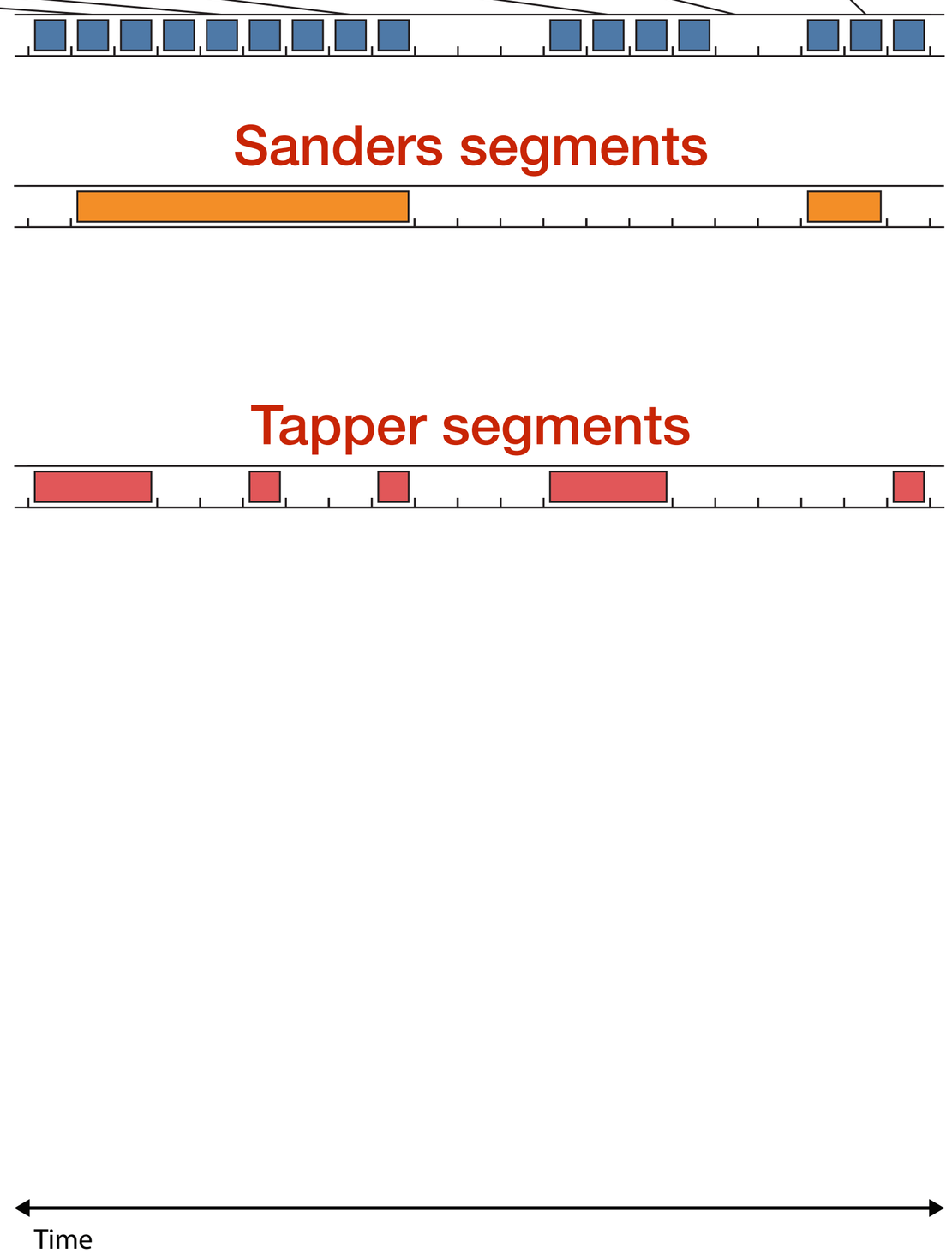


Segments where Tapper is on screen (FILTER, COALESCE)



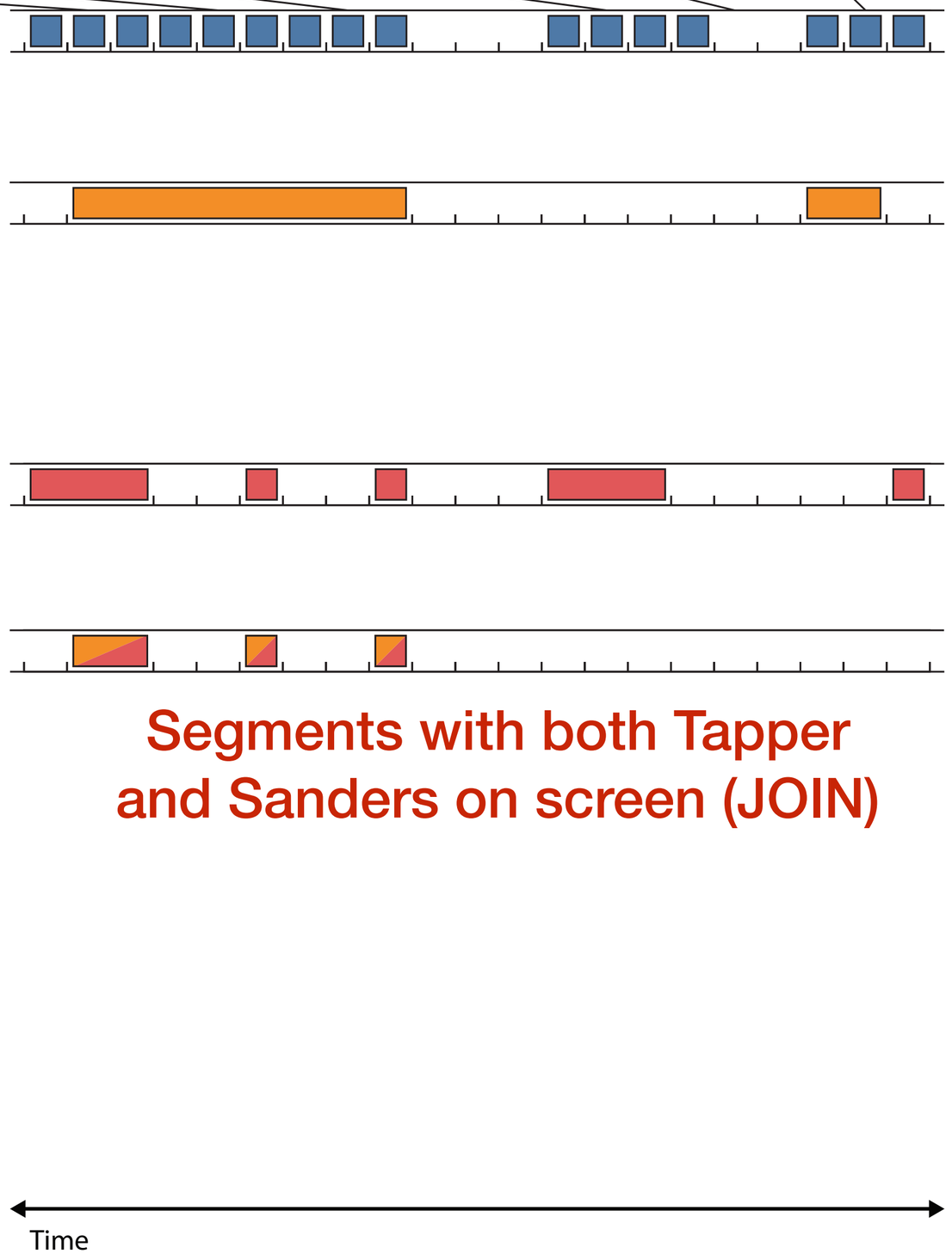
Interview detection example


```
faces = rekall.ingest(database.table("faces"), 3D)

sanders = faces
    .filter(λ face: face.name == "Bernie Sanders")

sanders_segs = sanders
    .coalesce(
      predicate = time_gap < 30 seconds,
      merge = time_span)

tapper = faces
    .filter(λ face: face.name == "Jake Tapper")

tapper_segs = tapper
    .coalesce(
      predicate = time_gap < 30 seconds,
      merge = time_span)
```



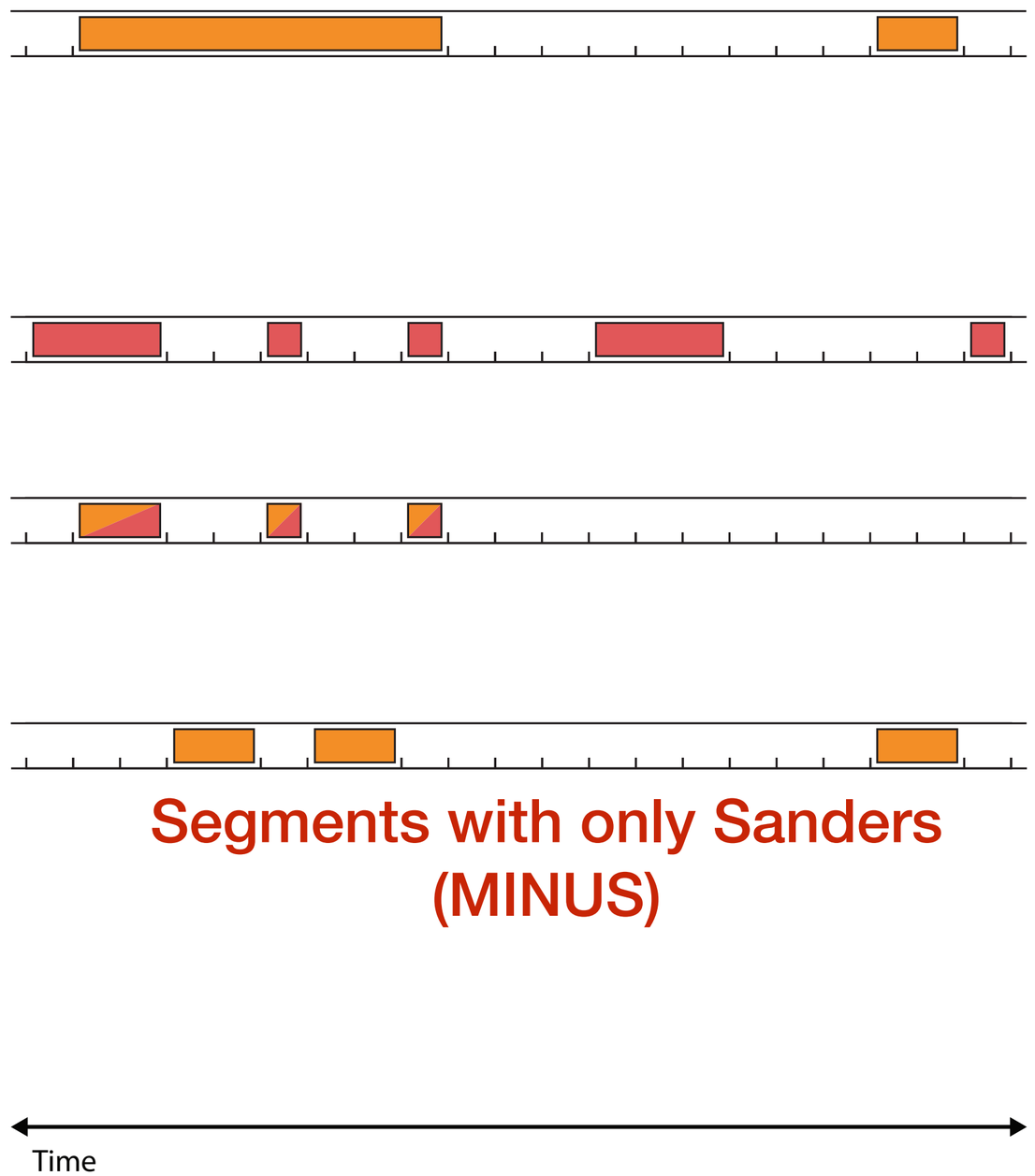
Interview detection example


```
faces = rekall.ingest(database.table("faces"), 3D)

sanders = faces
    .filter(λ face: face.name == "Bernie Sanders")

sanders_segs = sanders
    .coalesce(
      predicate = time_gap < 30 seconds,
      merge = time_span)

tapper = faces
    .filter(λ face: face.name == "Jake Tapper")

tapper_segs = tapper
    .coalesce(
      predicate = time_gap < 30 seconds,
      merge = time_span)

sanders_and_tapper_segs = sanders_segs
    .join(
      tapper_segs,
      predicate = time_overlaps,
      merge = time_intersection)
```



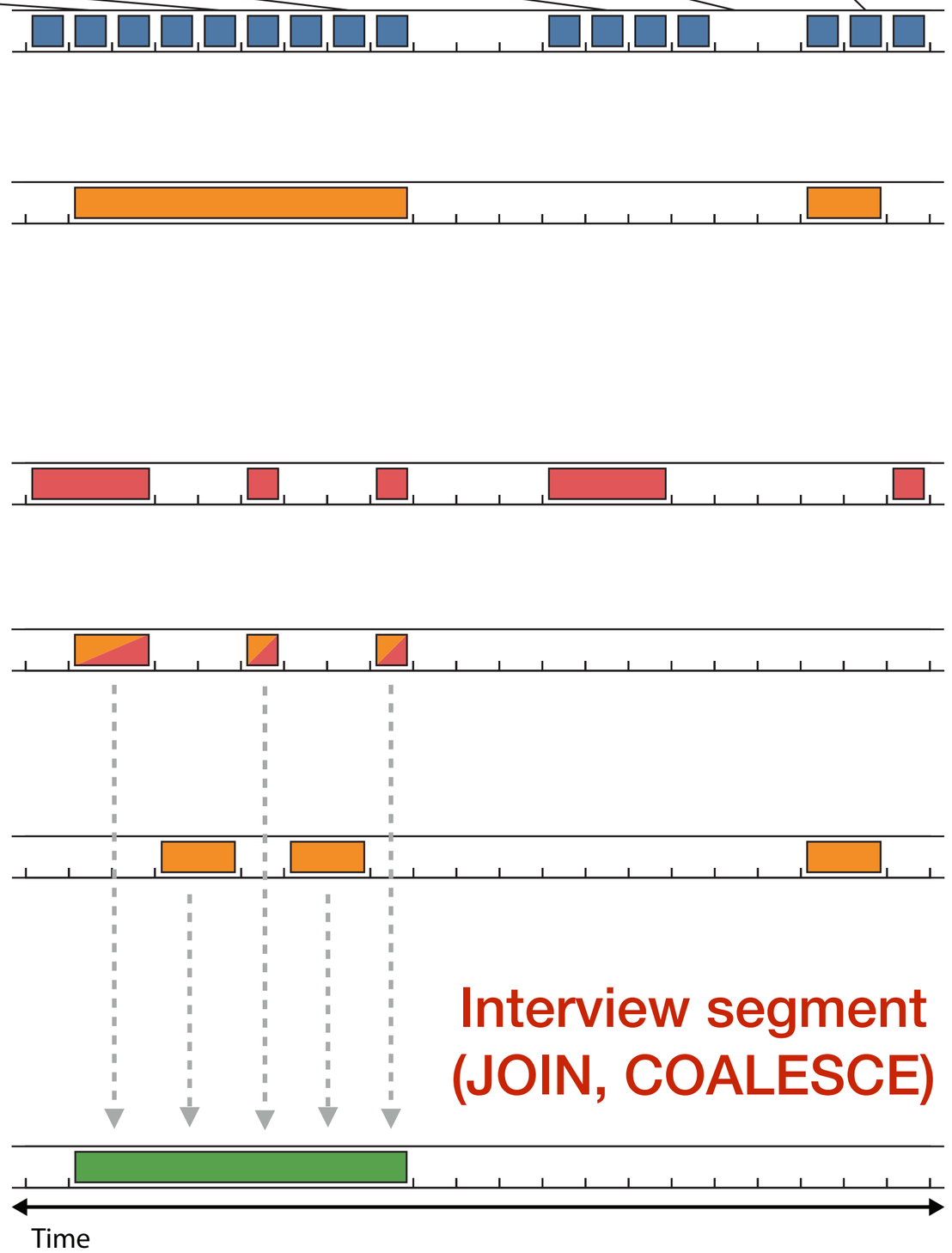
Interview detection example


```
faces = rekall.ingest(database.table("faces"), 3D)

sanders = faces
    .filter(λ face: face.name == "Bernie Sanders")

sanders_segs = sanders
    .coalesce(
      predicate = time_gap < 30 seconds,
      merge = time_span)

tapper = faces
    .filter(λ face: face.name == "Jake Tapper")

tapper_segs = tapper
    .coalesce(
      predicate = time_gap < 30 seconds,
      merge = time_span)

sanders_and_tapper_segs = sanders_segs
    .join(
      tapper_segs,
      predicate = time_overlaps,
      merge = time_intersection)

sanders_alone_segs = sanders_segs
    .minus(sanders_and_tapper_segs)
```



Interview detection example


```
faces = rekall.ingest(database.table("faces"), 3D)

sanders = faces
    .filter(λ face: face.name == "Bernie Sanders")

sanders_segs = sanders
    .coalesce(
      predicate = time_gap < 30 seconds,
      merge = time_span)

tapper = faces
    .filter(λ face: face.name == "Jake Tapper")

tapper_segs = tapper
    .coalesce(
      predicate = time_gap < 30 seconds,
      merge = time_span)

sanders_and_tapper_segs = sanders_segs
    .join(
      tapper_segs,
      predicate = time_overlaps,
      merge = time_intersection)

sanders_alone_segs = sanders_segs
    .minus(sanders_and_tapper_segs)

interview_segs = sanders_and_tapper_segs
    .join(
      sanders_alone_segs,
      predicate = before or after,
      merge = time_span)

interviews = interview_segs.coalesce()
```



Value of leveraging domain knowledge

In label-data poor scenarios, Rekall queries often on par with, and sometimes significantly more accurate than, deep learned approaches

Task	RN-50	RN-50, Smoothed	Conv3D	Rekall
Interview	78.3 +/- 7.6	88.6 +/- 5.3	17.7 +/- 18.3	**95.5**
Commercial	90.9 +/- 1.0	90.0 +/- 0.9	88.6 +/- 0.4	**94.9**
Conversation	65.2 +/- 3.5	66.1 +/- 3.5	**79.4 +/- 2.3**	71.8
Shot Detect	--	--	83.2 +/- 1.0	**84.1**
Shot Scale	67.3 +/- 1.0	68.1 +/- 1.2	70.1 +/- 0.8	**96.2**

… and often can be written by domain experts
(with little Rekall programming experience) in an afternoon

Human-in-the-loop curation tasks

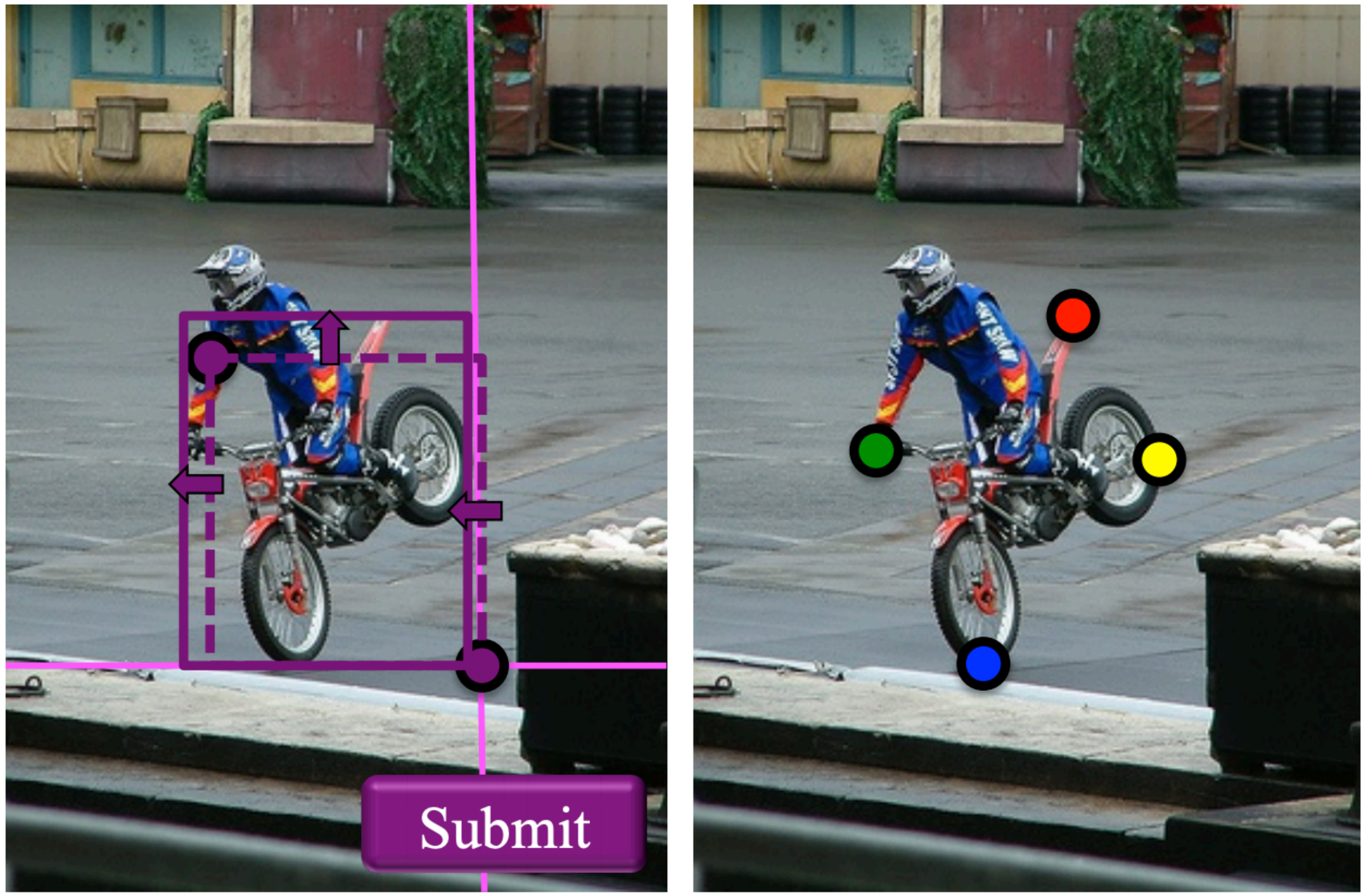
- Queries for curating training data for data labelers
  - Focus labeling effort on important scenes

- Queries for debugging model failures
  - Find situations where a model is likely wrong (disagreement, "flicker”, etc..)

- Queries for curating video for content creation
  - Video designer looking for appropriate “B-roll”

Better interfaces for human labeling

- Example: extreme clicking defines an object bounding box AND ALSO gives four points on the object's silhouette



5x faster for humans to label

[Source: Papadopoulos et al. ICCV 2017]

Many, many ways to find, generate, and operationalize supervision

- Multiple-modalities of data, knowledge in prior models, weak sources of supervision, return to basic heuristics, etc.

- Programming is going into curating and generating training data!

- Programming is going into curating and generating training data! It does seem like better platform and system support would be helpful here!