
[Sc]
STANDARD COGNITION

Building the interface of retail

1. e-commerce level convenience for shoppers

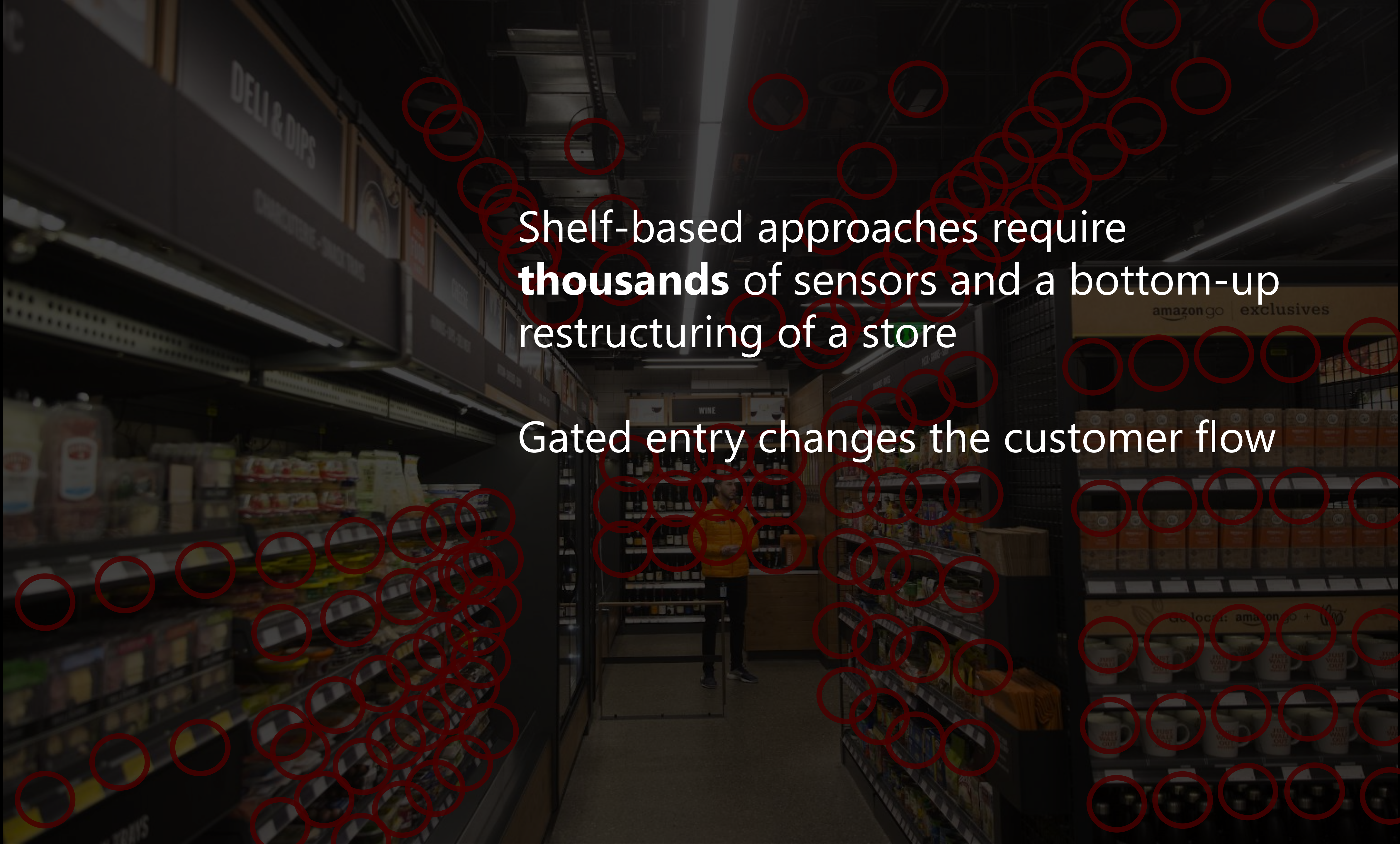
2. e-commerce level automation and insight for retailers


Market

Amazon Go and other shelf-based approaches provide a great proof of concept, but have large drawbacks.


DELI & DIPS
CHARCUTERIE • SNACK TRAYS
CHEESE
WINE
EXIT
amazon go | exclusives
Go local: amazon go +
JUST WALK OUT
TRAYS


DELI & DIPS
CHARCUTERIE • SNACK TRAYS
CHEESE
WINE
EXIT
amazon go | exclusives
JUST WALK OUT



Shelf-based approaches require **thousands** of sensors and a bottom-up restructuring of a store

Gated entry changes the customer flow

Our proof of concept store on Market st

27 total sensors

Our partners have consistently requested a **ceiling-only solution**

Standard Market is a **1,900 sq foot convenience store**

It’s powered by **27 overhead cameras**

No shelf sensors, depth sensors, RFID, biometric trackers, or turnstiles

Major Research Challenges in Autonomous Checkout

Tracking

Action Recognition

Item Classification

Major Research Challenges in Autonomous Checkout

Who

Who has what?

What

Major Research Challenges in Autonomous Checkout

Tracking

Action Recognition

Item Classification


[Sc]
STANDARD COGNITION


Joint work between Karl Obermeyer, Kyle Dorman,
Warren Green, Juan Lasheras, Dave Valdman,
Jordan Fisher
[Sc]
STANDARD COGNITION

Major Research Challenges in Autonomous Checkout

Tracking

Action Recognition

Item Classification

Major Research Challenges in Autonomous Checkout

Tracking

- Dense, multi-object tracking in the wild
- Multi-view consensus
- Constant partial and full occlusions
- Has to run in real time
- Can't use facial recognition
- Off the shelf, cheap hardware
- Has to be nearly 100% accurate

Major Research Challenges in Autonomous Checkout

Tracking

- Dense, multi-object tracking in the wild
- Multi-view consensus
- Constant partial and full occlusions
- Has to run in real time
- Can’t use facial recognition
- Off the shelf, cheap hardware
- Has to be nearly 100% accurate

Peet's
COLD BREW
Home Goods
Bath & Body
Snacks
Drinks


Peet's
COLD BREW

High level components of a tracker

Feature Extraction

Association

High level components of a tracker

Joint Association

Temporal Association

Spatial Association

High level components of a tracker

Feature Extraction

Association

You don’t necessarily want to isolate these systems!

General Approach

- Figure out your metric
- Get good data
- Invest in infrastructure
- Hedge your research bets
- Evaluate true metric
- Productionize

General Approach

- **Figure out your metric**
- Get good data
- Invest in infrastructure
- Hedge your research bets
- Evaluate true metric
- Productionize

Figure out your metric

- You don’t get to pick your metric, you need to determine it
- Whatever metric you pick, it will be leaky. Be prepared
- The standard metric in the literature is probably not what you want

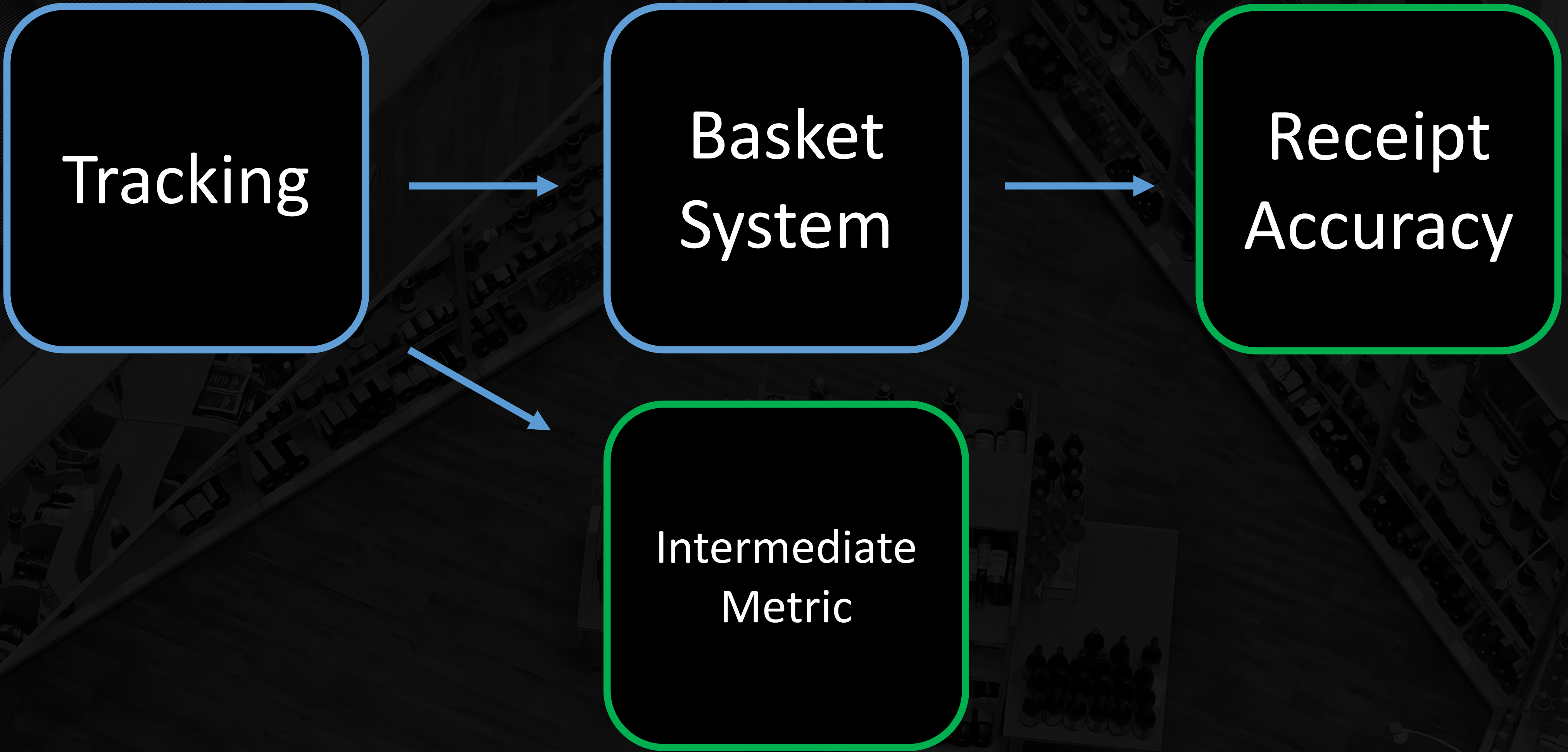
Correct Metric for Cascading Models

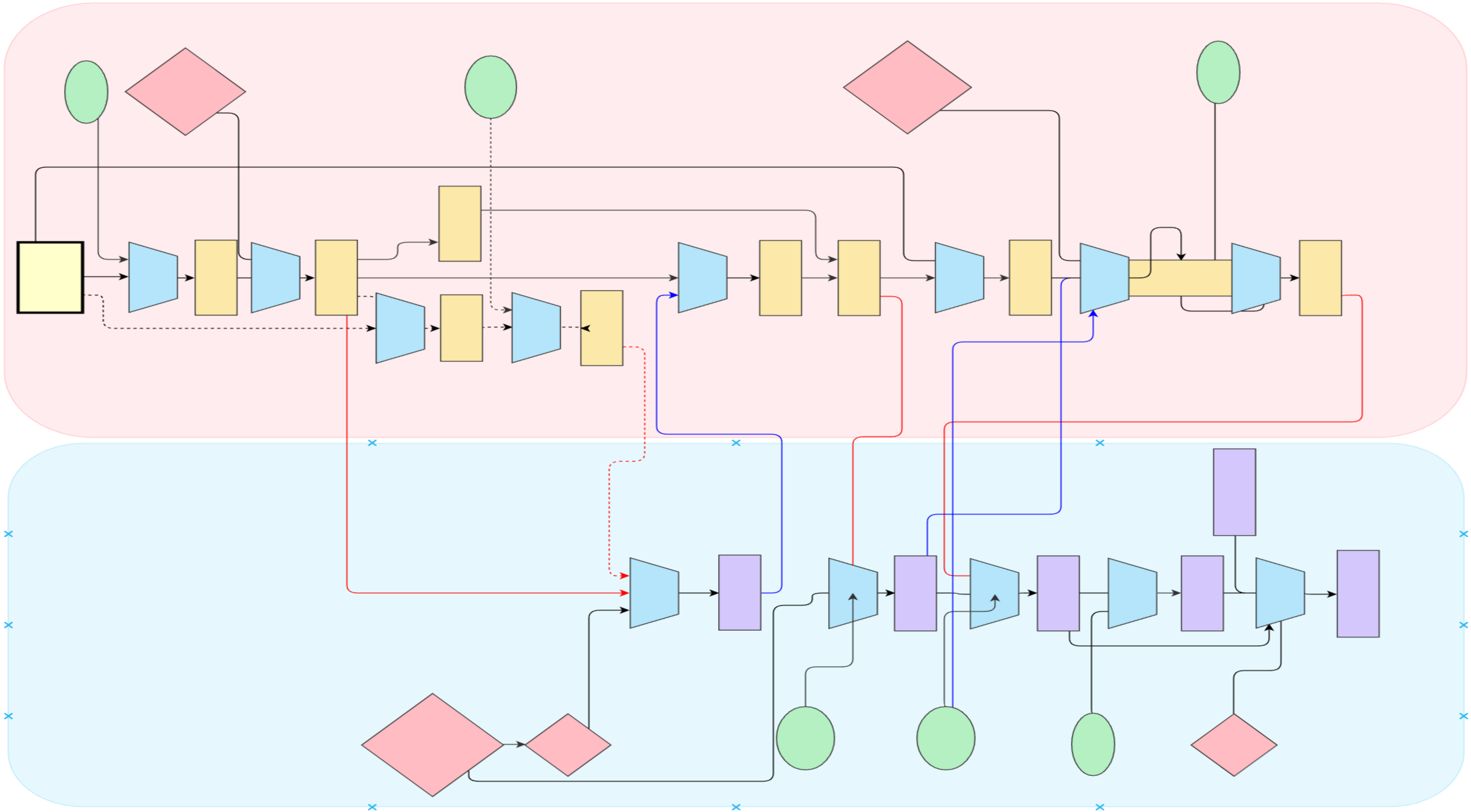
Model 1 → Model 2 → Final Metric

Correct Metric for Cascading Models

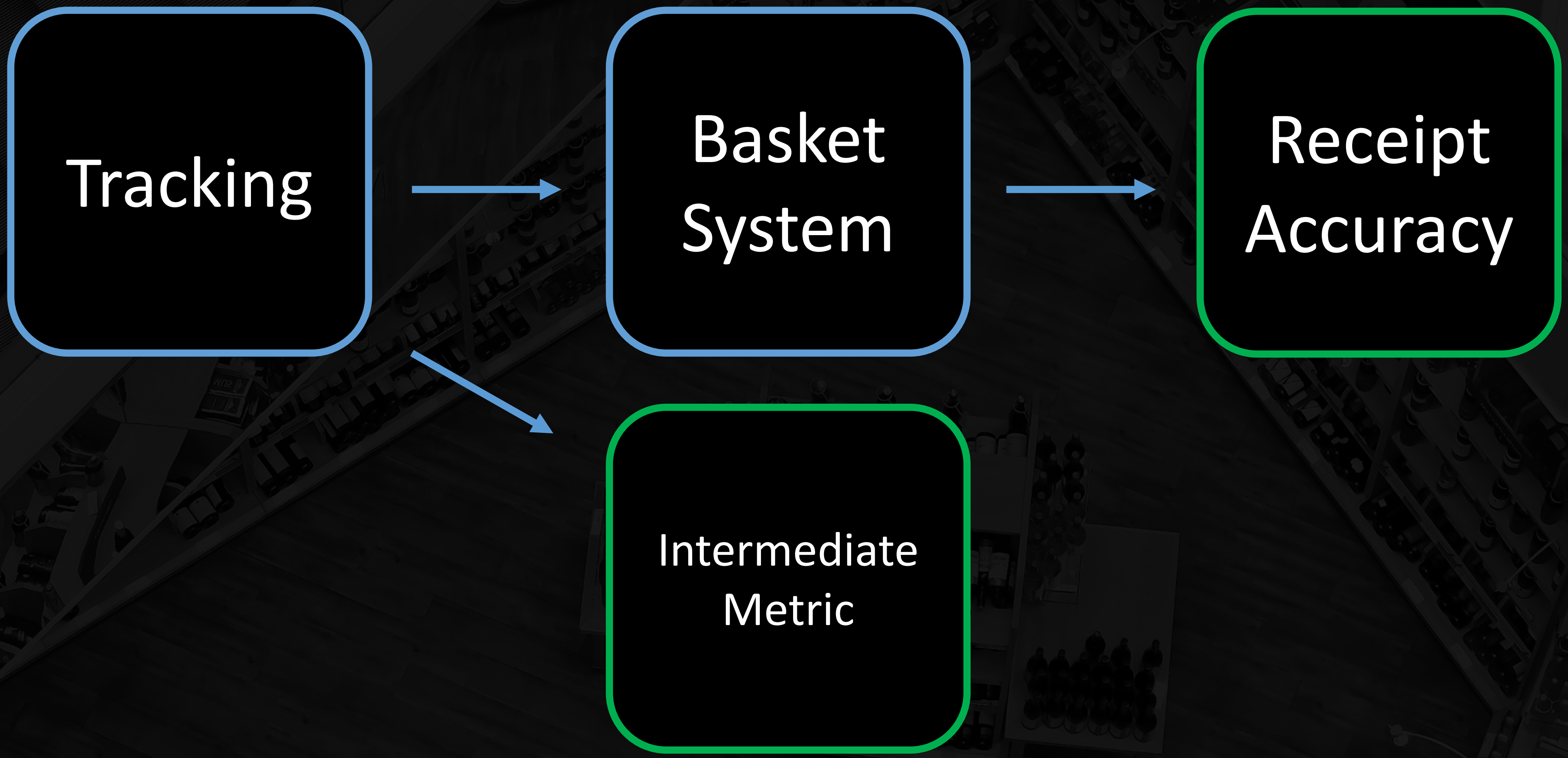
Tracking → Basket System → Receipt Accuracy

Correct Metric for Cascading Models





Correct Metric for Cascading Models



Determining your intermediate metric

- Improving the metric should almost always improve your final metric, potentially with the need to retrain downstream models (We call this Firewalling)
- Should be able to be optimized
- Maximize one thing, satisfice everything else. Alternatively, use blended metric.
- Other metrics should be considered “debug” metrics

Things we might care about

- False negatives
- False positives
- Concentration of false negatives per person
- Image plane swaps
- True swaps

- Impossible to optimize everything simultaneously. How do we proceed?

Blended metrics, or utility functions

$$MOTA = 1 - \frac{\sum_t FN_t + FP_t + IDSW_t}{\sum_t GT_t}$$

Blended metrics, or utility functions

$$MOTA = 1 - \frac{\sum_t FN_t + FP_t + IDSW_t}{\sum_t GT_t}$$

Does this assign the right amount of utility to each individual metric? *Probably not.*

Does this firewall our final metric? *Probably not*

Maximize and satisfice

- For all metrics identify the minimum reasonable requirements
- Identify the one additional metric that improving beyond the minimum would yield continued improvement for the downstream systems

Maximize and satisfice

- Satisfice
  - Swaps = 0
  - Untracked people = 0
  - Dropped tracks = 0
- Maximize
  - Sum of image plane MOTA’s
- Debug metric
  - Image plane swaps, false positives

General Approach

- **Figure out your metric**
- Get good data
- Invest in infrastructure
- Hedge your research bets
- Evaluate true metric
- Productionize

General Approach

- **Figure out your metric**
- Get good data
- Invest in infrastructure
- Hedge your research bets
- **Evaluate true metric**
- Productionize

General Approach

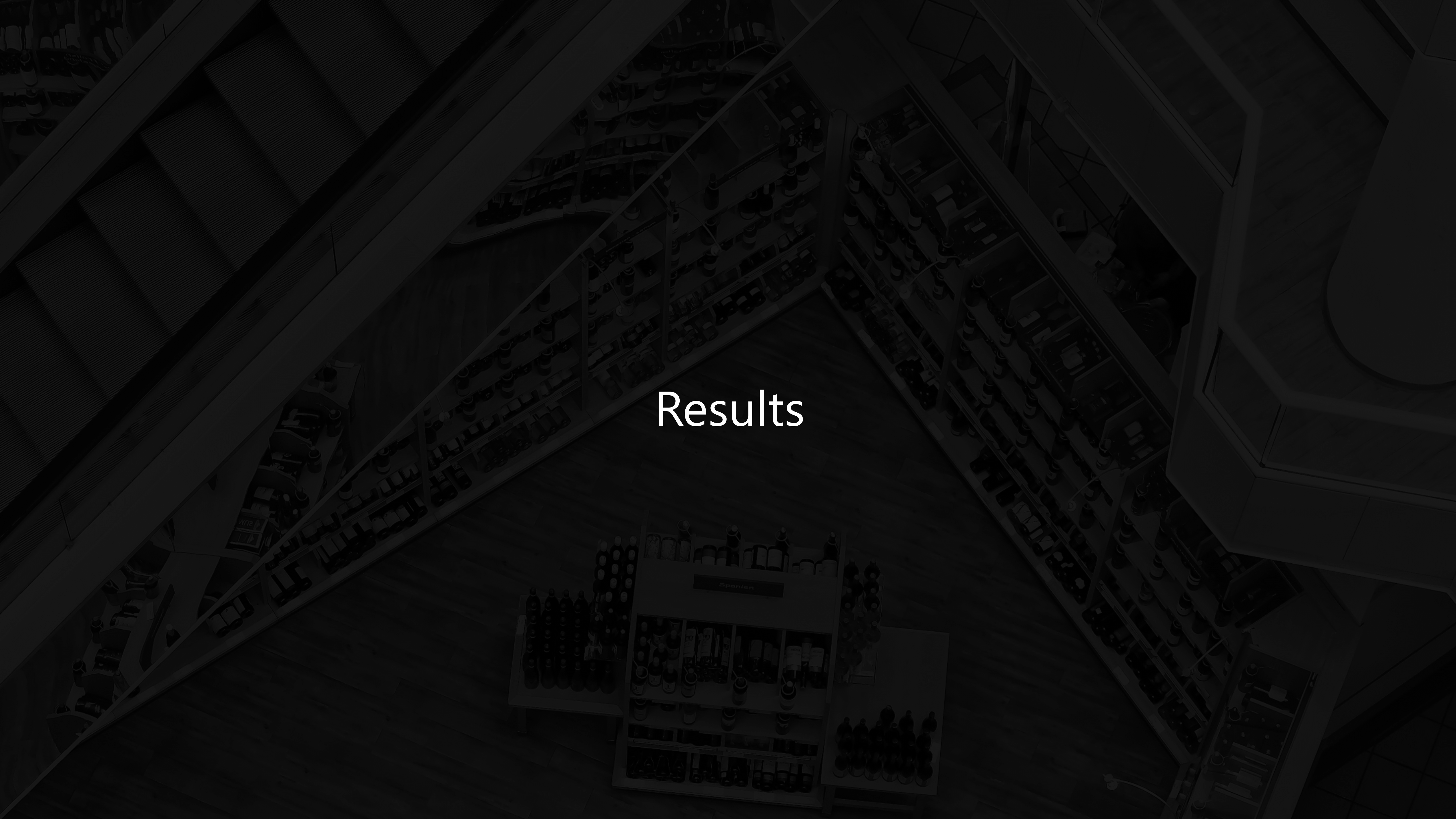
- Figure out your metric
- **Get good data**
- Invest in infrastructure
- Hedge your research bets
- Evaluate true metric
- Productionize

General Approach

- Figure out your metric
- Get good data
- **Invest in infrastructure**
- Hedge your research bets
- Evaluate true metric
- Productionize

General Approach

- Figure out your metric
- Get good data
- Invest in infrastructure
- **Hedge your research bets**
- Evaluate true metric
- Productionize



Results


Peet's
COLD BREW
Home Goods
Snacks


Peet's
COLD BREW

General Approach

- **Figure out your metric**
- Get good data
- Invest in infrastructure
- Hedge your research bets
- **Evaluate true metric**
- Productionize

General Approach

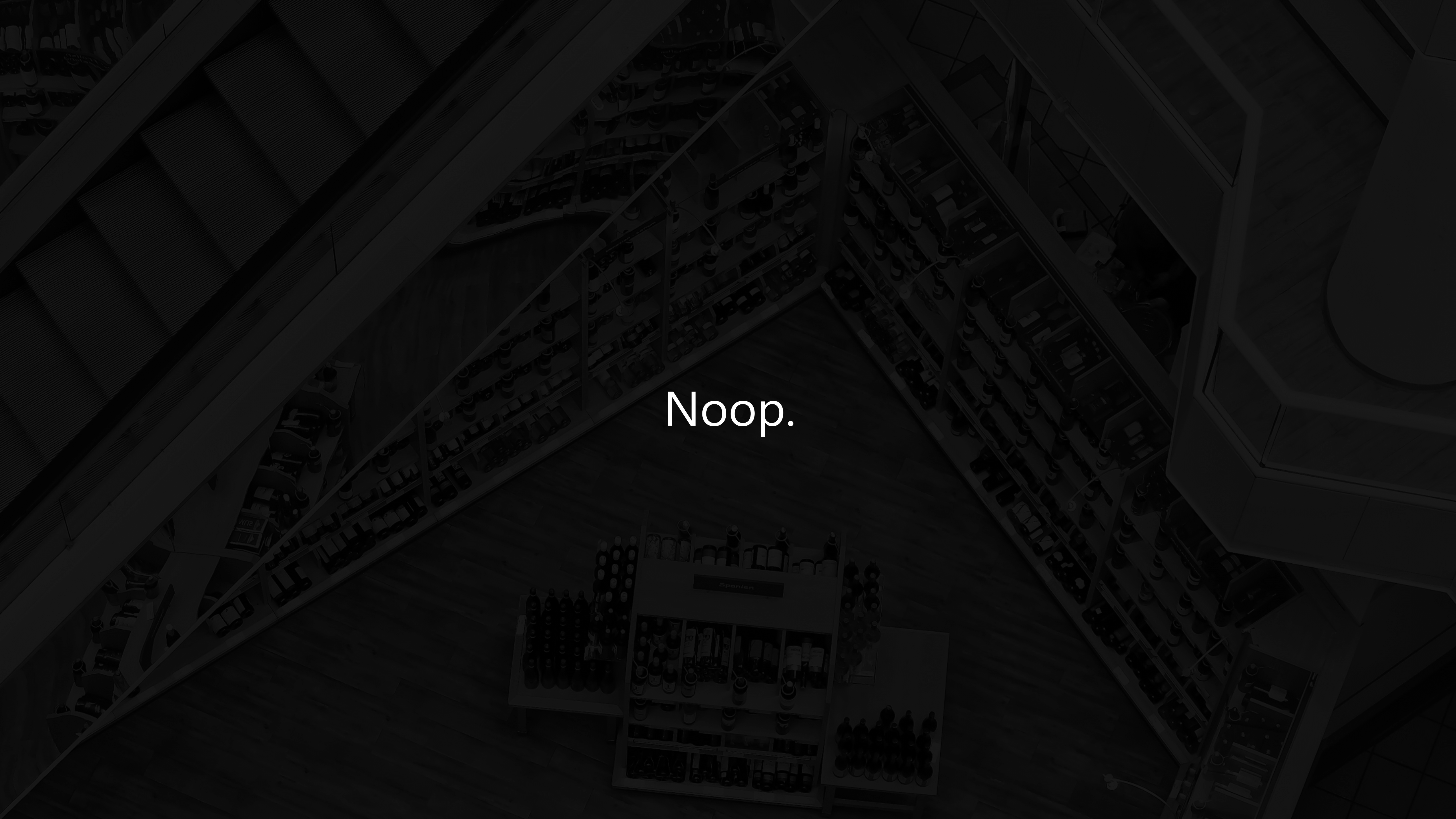
- Figure out your metric
- Get good data
- Invest in infrastructure
- Hedge your research bets
- Evaluate true metric
- **Productionize**

Problem

- Algorithm is $O(n^2 * p^2)$
- Runs at 0.5 FPS, needs to run at 30 FPS

Solution

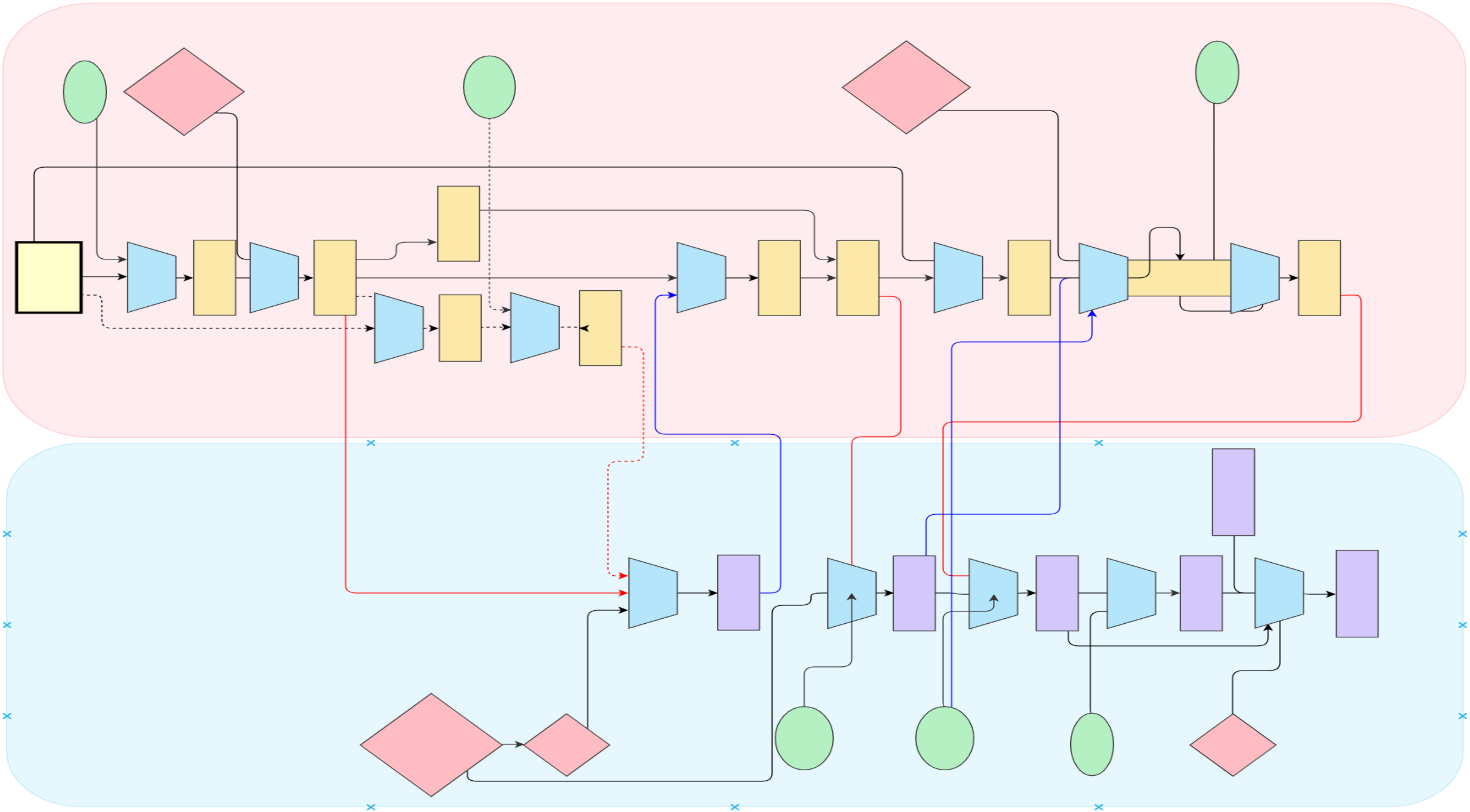
- Modify the algorithm to reduce runtime complexity?


Noop.

If we can get a 100x speedup, we don't need to modify the algorithm.

Why Rust?

1. Very fast
2. Fearless parallelism
3. Easier to maintain
4. Language of choice



Why ***not*** Rust?

1. Poor support for scientific computing
2. Hard to learn
3. Smells “shiny”

Case study

We’re using Rust for high performance system code, but not yet for complex models

Wanted a case study to demonstrate feasibility

Approach

1. Test harness
2. Restructure code to be Rustic
3. Full mypy type coverage
4. Automatic transpilation
5. Iterate with the Rust compiler
6. Hand fix the rest
7. Build needed library FFI’s
8. `dbg!` and `print` pairs to isolate output divergence

```python
class SimpleClass:
    """
    This is a simple class.

    Args:
        x: Some number here!
    """
    def __init__(self, x) -> None:
        self.x = x

    def some_function(self):
        return self.x
```

```python
class SimpleClass:
    """
    This is a simple class.

    Args:
        x: Some number here!
    """
    def __init__(self, x: int) -> None:
        self.x = x

    def some_function(self) -> float:
        return self.x
```

```rust
/// This is a simple class.
pub struct SimpleClass {
    x: usize,
}

impl SimpleClass {
    /// Return a new SimpleClass.
    ///
    /// # Arguments
    ///
    /// * `x` - Some number here!
    pub fn new(x: usize) -> SimpleClass {
        SimpleClass {
            x: x,
        }
    }

    pub fn some_function(&self) -> f64 {
        self.x
    }
}
```

pyout.py output

transpiling rocks!

Approach

1. Test harness
2. Restructure code to be Rustic
3. Full mypy type coverage
4. Automatic transpilation
5. Iterate with the Rust compiler
6. Hand fix the rest
7. Build needed library FFI’s
8. `dbg!` and `print` pairs to isolate output divergence

Approach

1. Test harness
2. Restructure code to be Rustic
3. Full mypy type coverage
4. Automatic transpilation
5. Iterate with the Rust compiler
6. Hand fix the rest
7. Build needed library FFI’s
8. `dbg!` and `print` pairs to isolate output divergence

Results

- 30+ FPS for 20 people and 20 cameras on a single core!
- No parallelism! No algorithmic changes!

What sucked

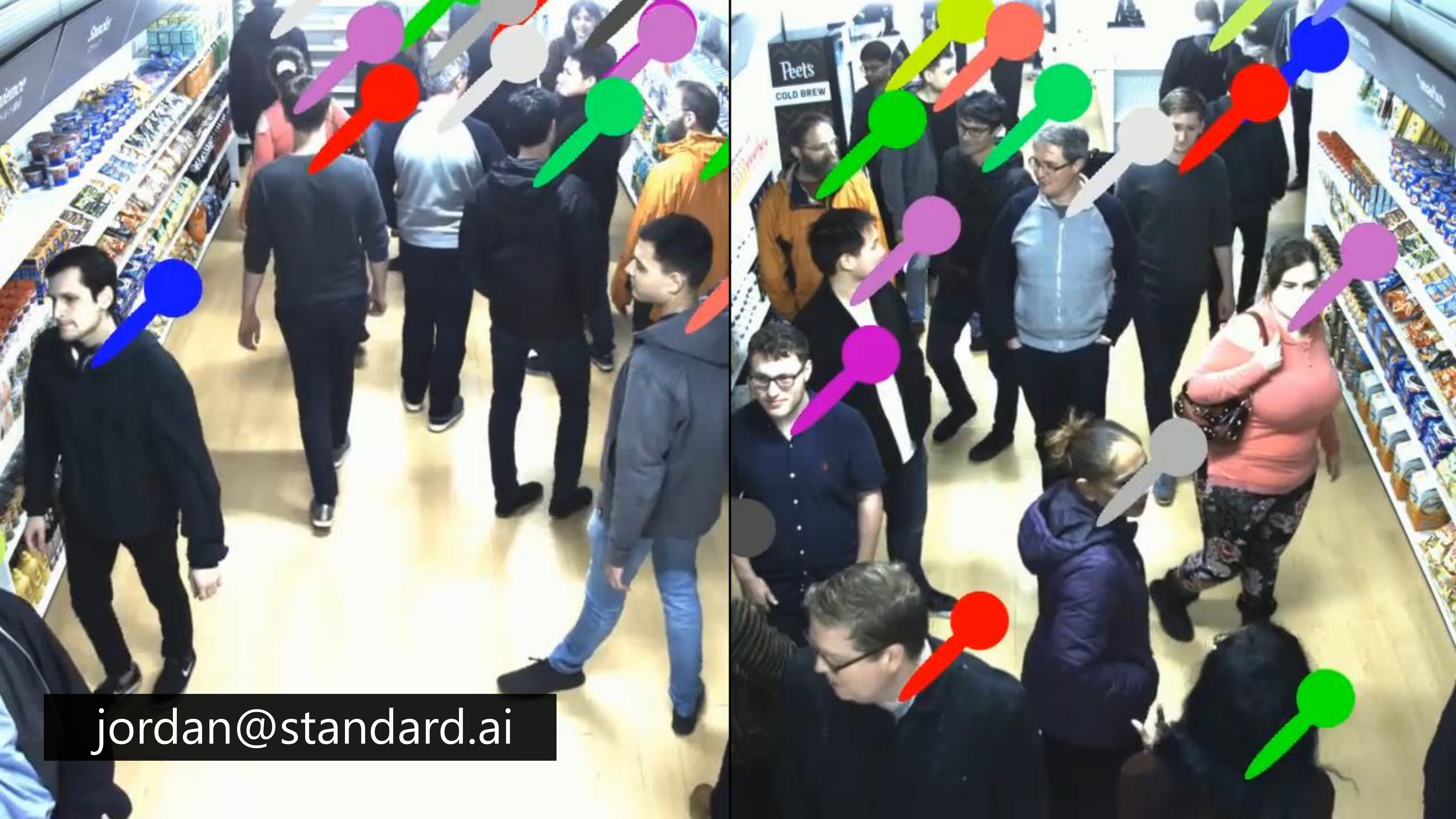
1. Library ecosystem
2. Poor opencv support, had to hand wrap FFI calls
3. Poor, unergonomic multidimensional array support

Major Research Challenges in Autonomous Checkout

Tracking

Action Recognition

Item Classification



jordan@standard.ai