

## Context & Challenge

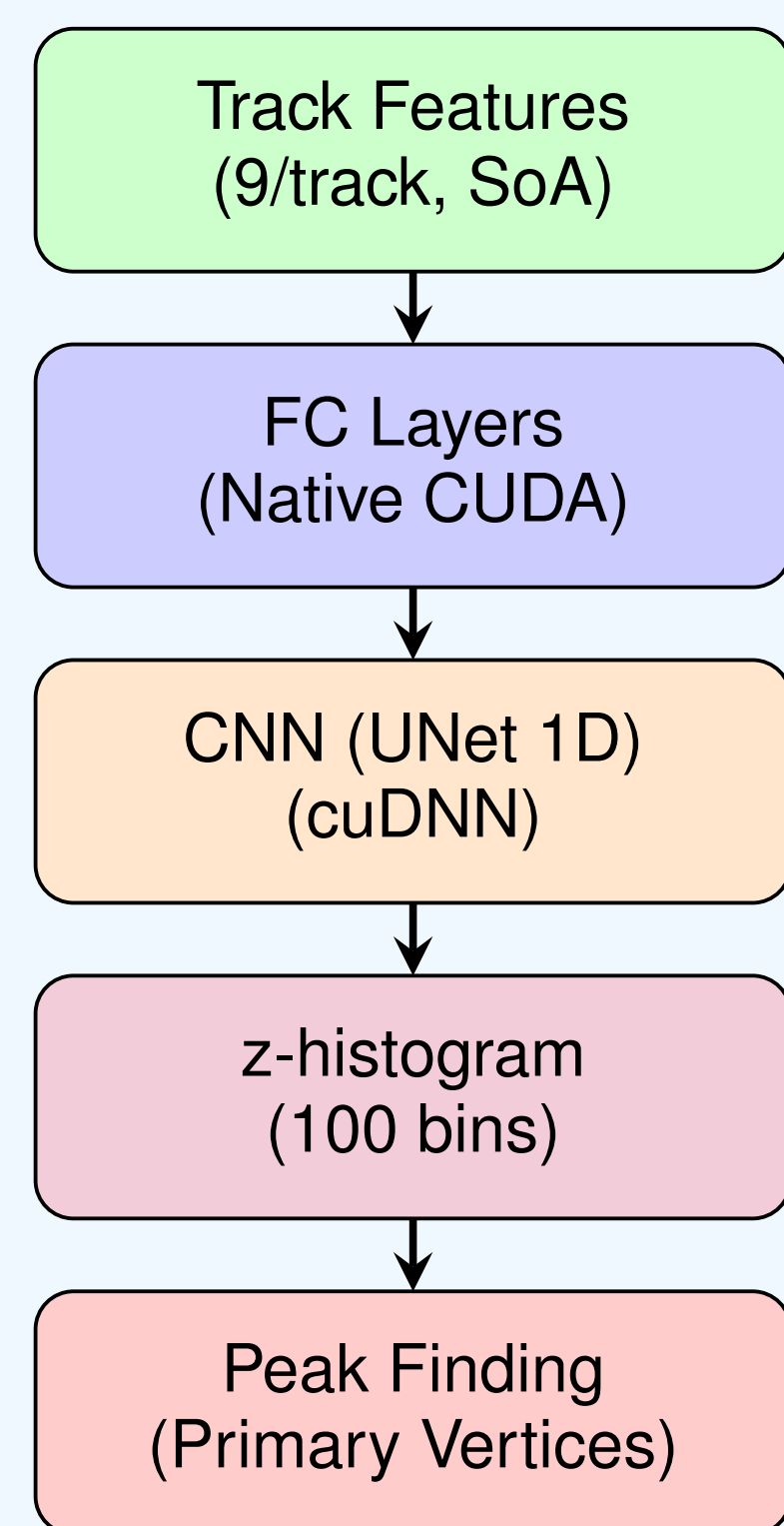
### LHCb Run 3 Requirements:

- 30 MHz input rate → 2 kHz output
- Per-event budget: <math><400 \mu s</math> on GPU
- Deploy ML inside **Allen** (GPU-only HLT1)

### The Challenge:

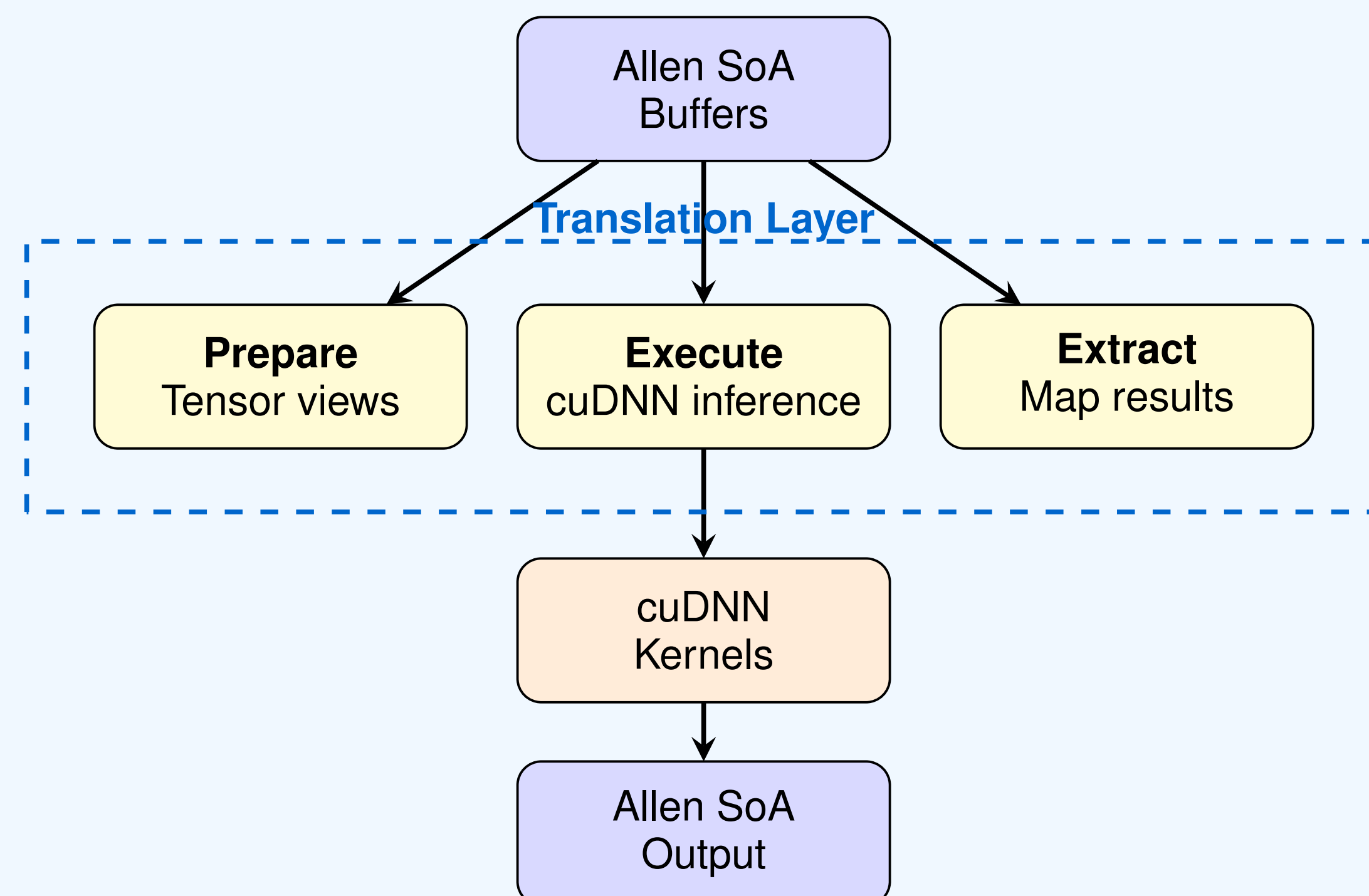
- Native CUDA (Allen) vs cuDNN (ML)
- Memory model incompatibility
- Different Data Structures
- Real-time constraints

## PVFinder Architecture



## Translation Layer Solution

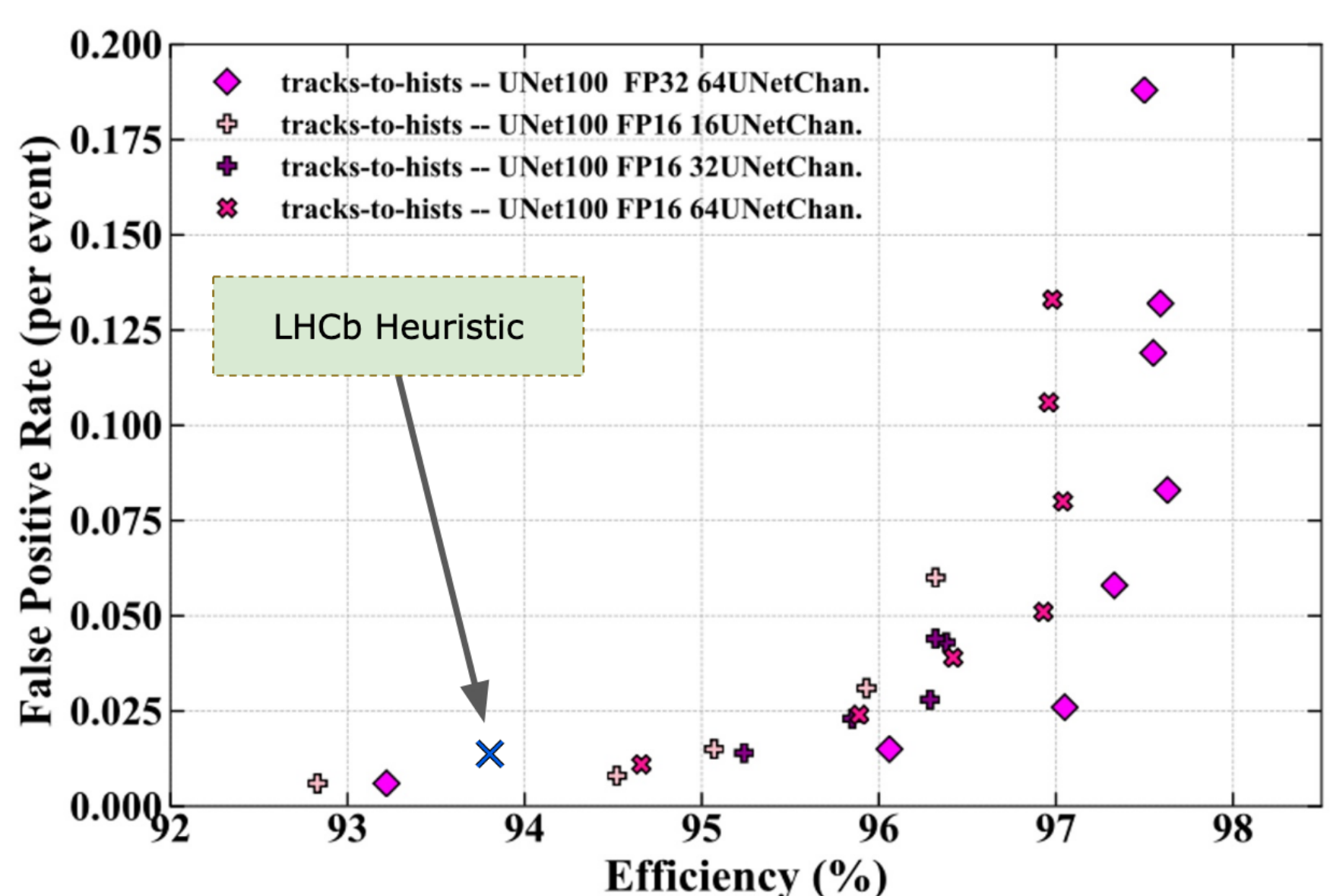
**Challenge:** Bridge native CUDA (Allen) ↔ cuDNN library (ML inference)



### Key Features:

- Zero-copy tensor views on Allen buffers
- Single GPU stream, no dynamic allocation

## Physics Performance



### Results (FP32, 64-channel):

- Efficiency: >97% at low FP rates
- False positives: 0.03/event (0.6%/PV)
- FP16: Minimum performance hit

## Performance Results

### Throughput Impact:

Config	Throughput
Baseline	100%
+ FC	~95%
+ Full	~25%

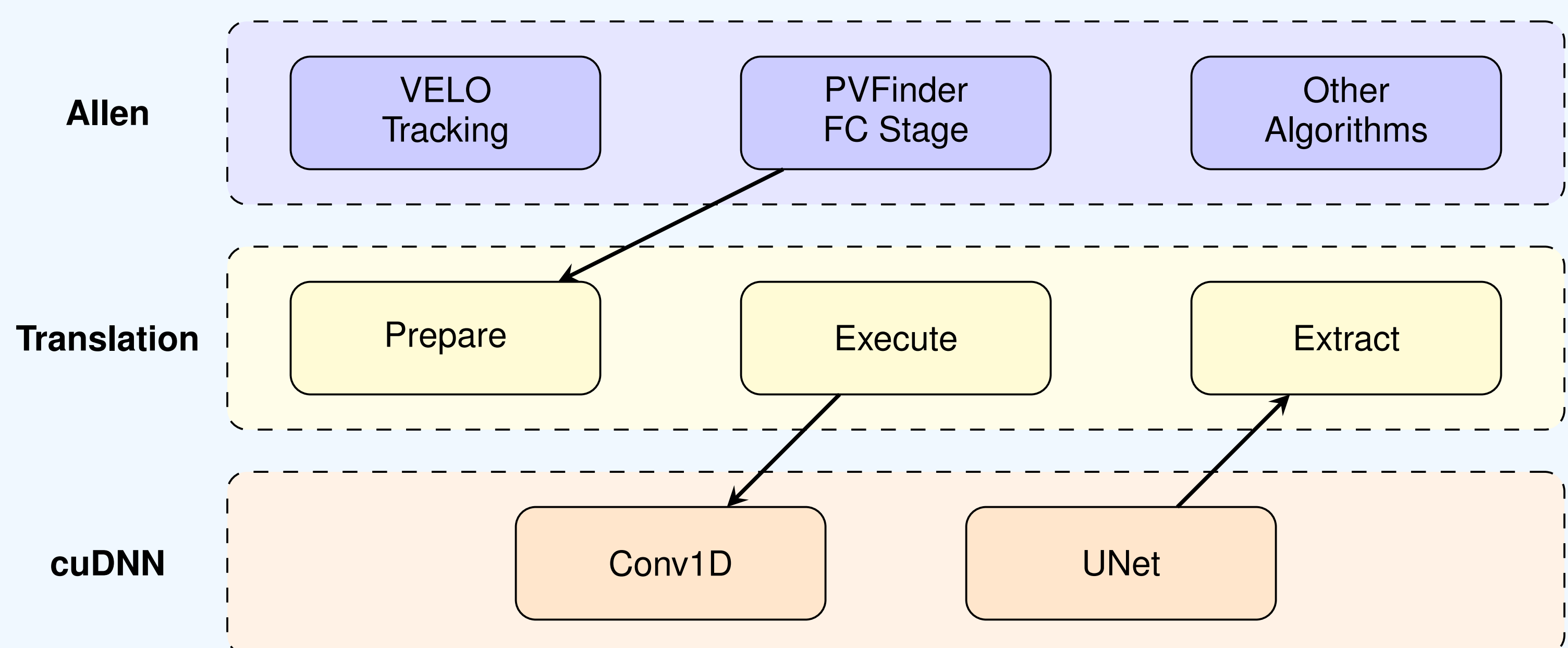
### What this means:

- CNN stage dominates overhead
- Order-of-magnitude improvement needed

### Status

CNN stage overhead  
 Target: <math><5\%</math> for production

## Complete Architecture

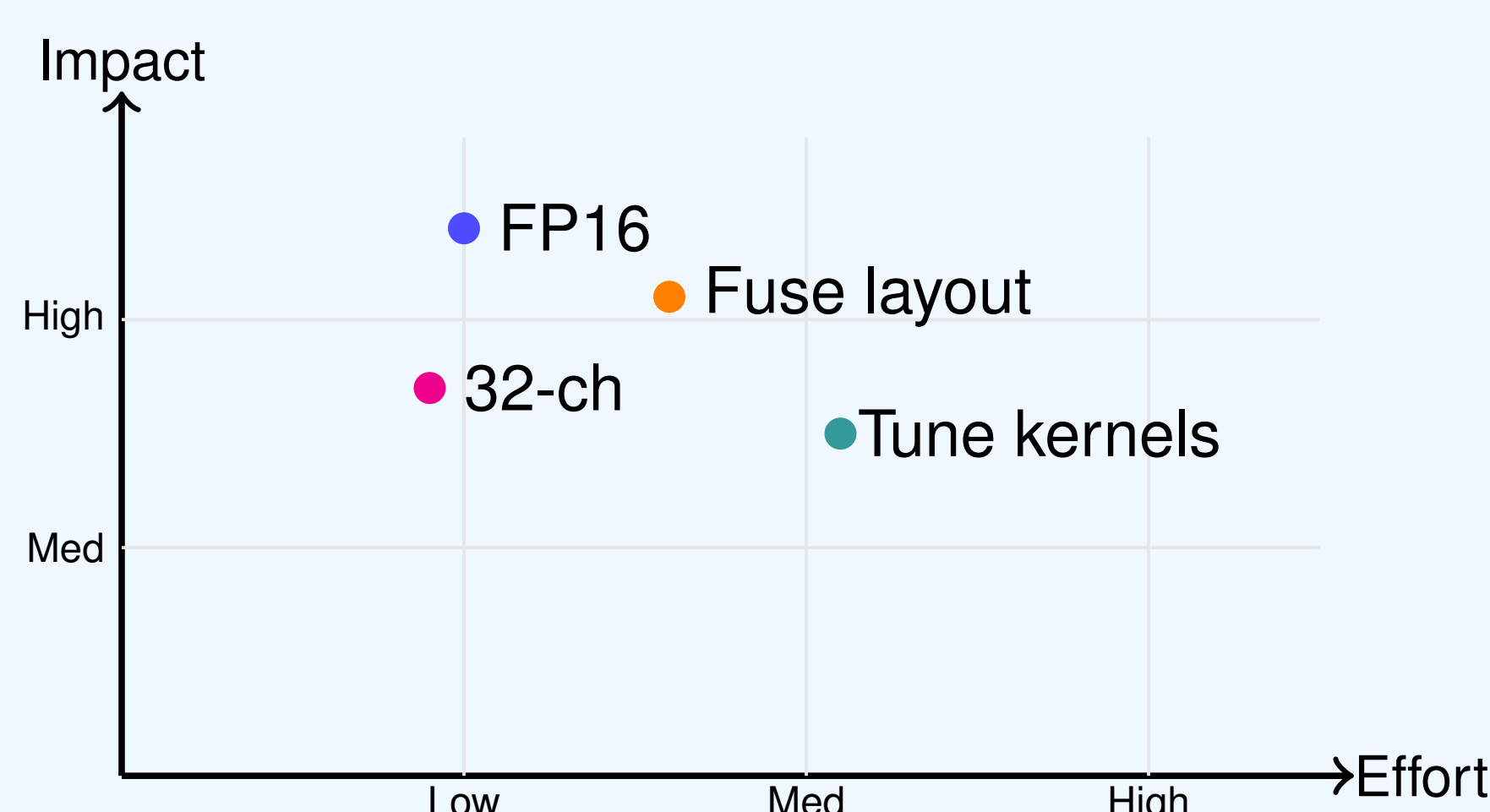


### Design Benefits:

- Each layer speaks native language
- Preserves Allen's deterministic model
- Enables ML without framework changes

## Optimization Roadmap

### High-Impact Steps:



### Priority Actions:

- FP16 + 32-channel: 8x speedup expected
- Fuse layout conversions: reduce memory traffic
- Optimize workspace reuse and coalescing

**Target:** 10x speedup → <math><5\%</math> HLT1 overhead

## Broader Impact

### Future Directions:

- Standardize for LHCb ML deployment
- Generic backends (ONNX Runtime)
- Adapt pattern for other experiments
- Cross-experiment knowledge transfer

