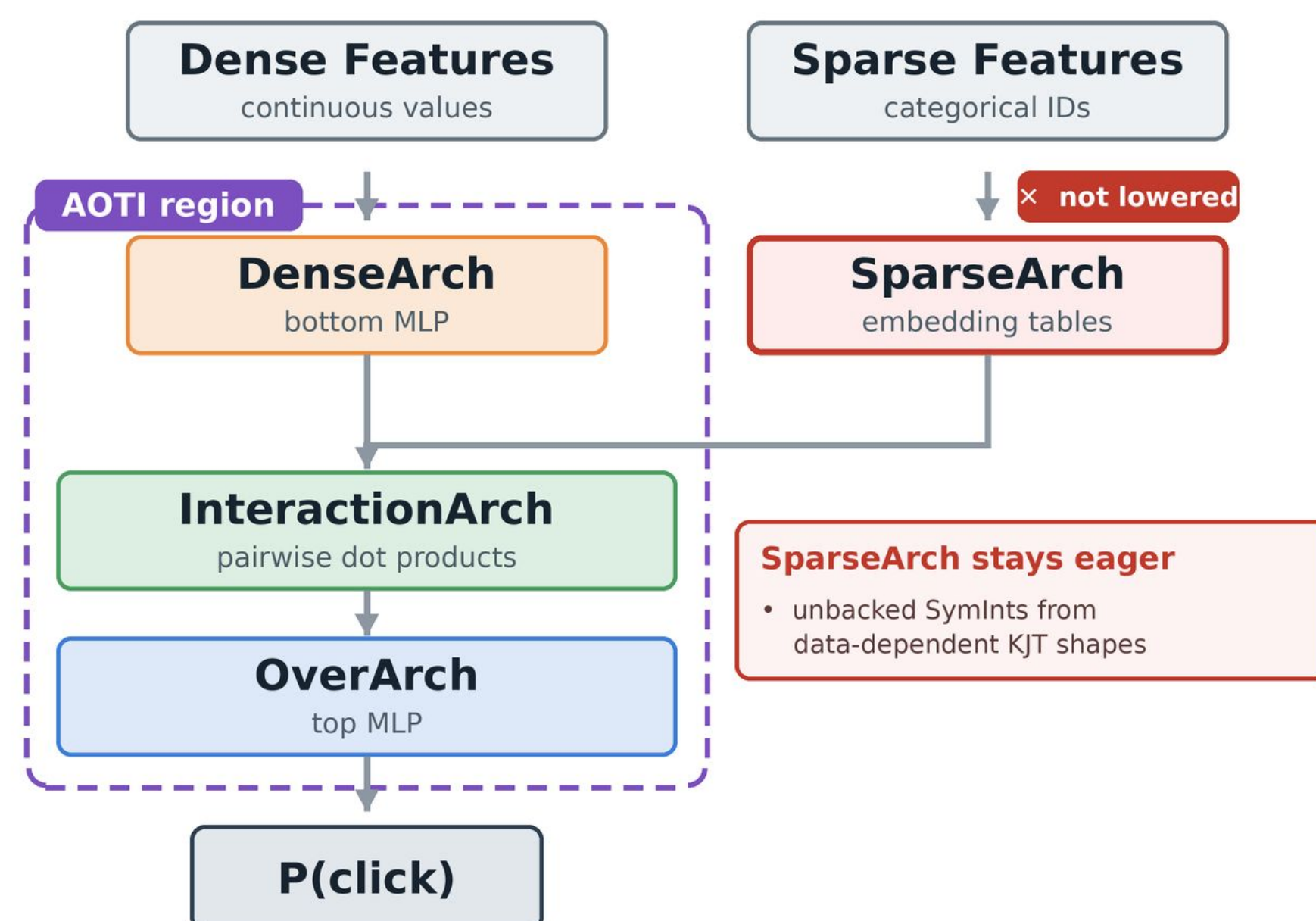


AOTI: torch.export (capture) + aot_compile (generate code)

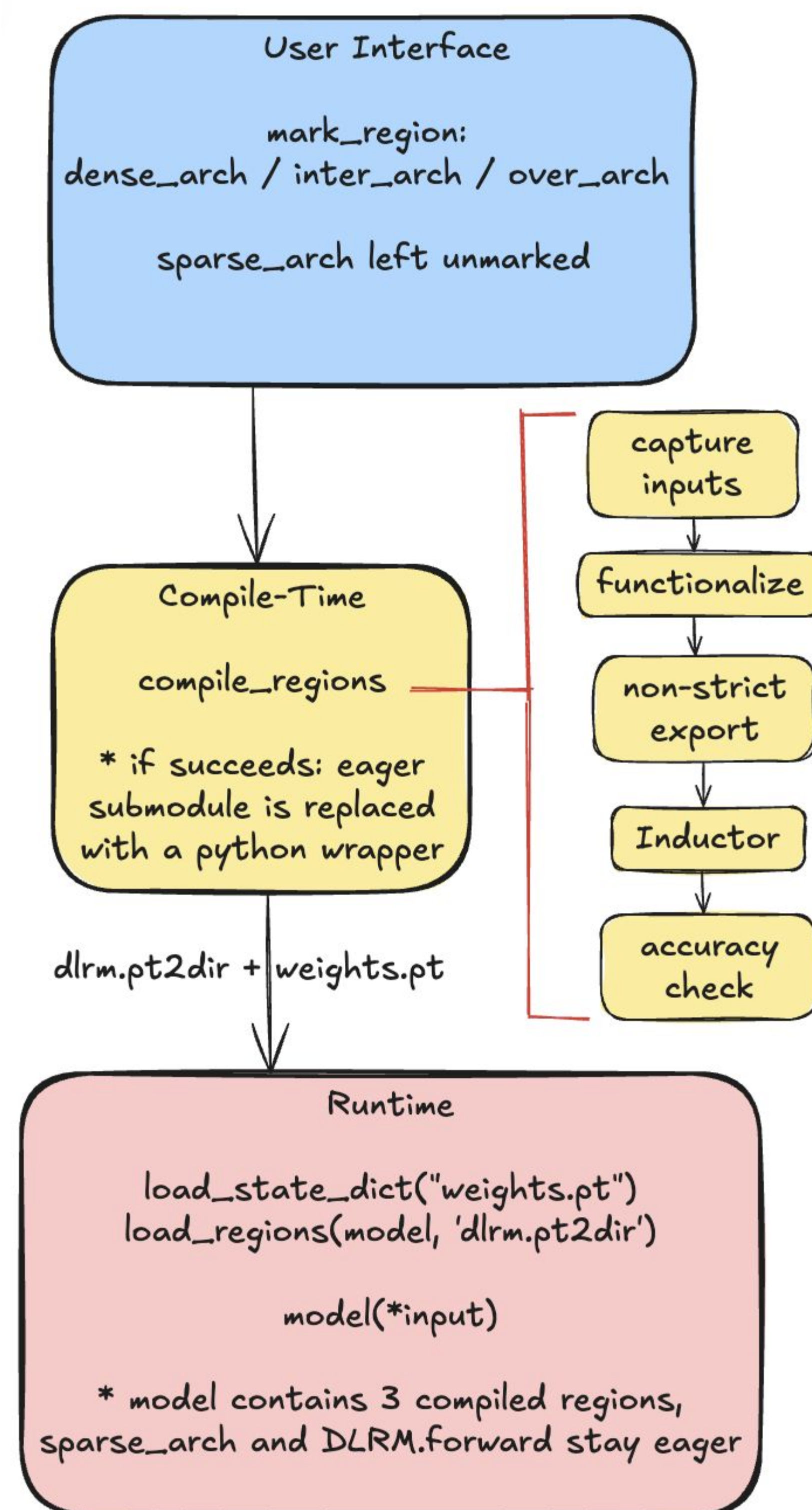
Today: one unsupported op blocks the whole model.

Regional AOTI: compile what you can, keep the rest eager.

DLRM



DLRM example:



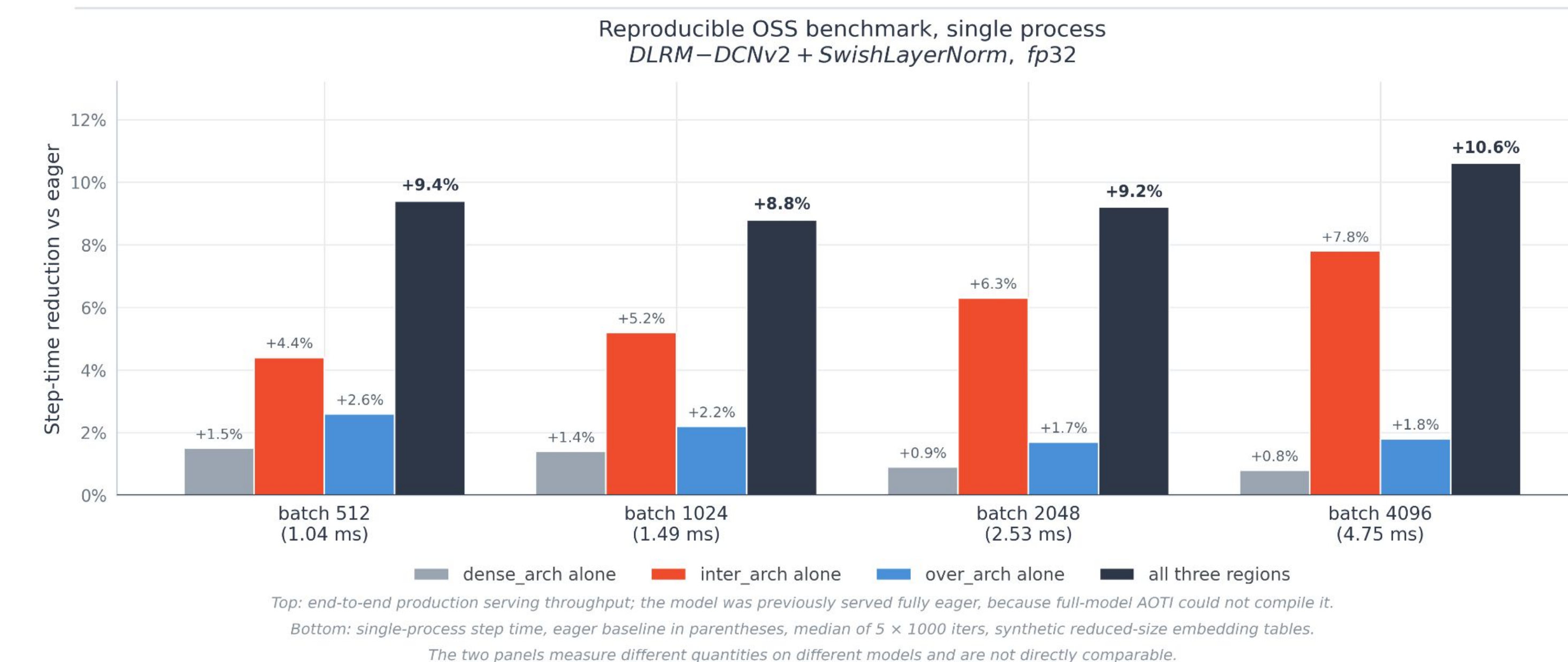
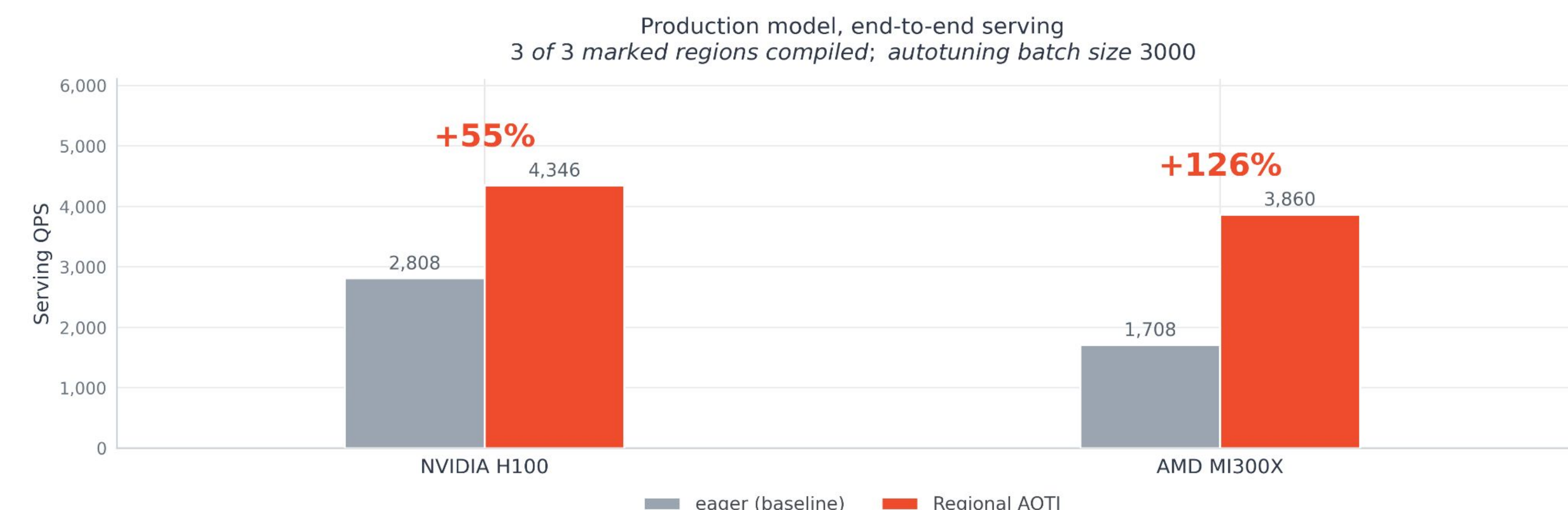
Code Example:

```
1 from torch._inductor import regional_aoti as ra
2
3 # ---- Compile: pick the regions, skip what can't be exported ----
4 model = DLRM(...).eval()
5
6 ra.mark_region(model.dense_arch)
7 ra.mark_region(model.inter_arch)
8 ra.mark_region(model.over_arch, config=ra.RegionConfig(
9     dynamic_shapes={"features": {0: batch}}))
10 # sparse_arch stays eager - unbacked symbols from jagged inputs
11
12 result = ra.compile_regions(model, inputs, package_path="dlrm.pt2dir")
13 print(result.report()) # dense_arch, inter_arch, over_arch: compiled
14 torch.save(result.model.state_dict(), "weights.pt")
15
16 # ---- Serve: only dlrm.pt2dir + weights.pt travel ----
17 model = DLRM(...).eval()
18 model.load_state_dict(torch.load("weights.pt", weights_only=True))
19 model = ra.load_regions(model, "dlrm.pt2dir").model
20
21 logits = model(dense_features, sparse_features) # 3 compiled + eager glue
```

Common blockers for AOTI

- Custom ops with no fake kernel
- Concrete values pulled from symbolic tensors
- Data-dependent output shapes
- Data-dependent control flow

Evaluation



DLRM paper - <https://arxiv.org/pdf/1906.00091>

TorchRec DLRM - <https://github.com/meta-pytorch/torchrec/blob/main/torchrec/models/dlrm.py>

Regional AOTI Prototype - <https://github.com/pytorch/pytorch/pull/190100>