Trace time performance notebook

WARNING: some of the cost accounting in this notebook is 2x what it should be; I thought entire_frame_compile and backend_compile were disjoint measurements

Baseline bd4a5b400aa32092b41a6c39004dc0e5bb62883d

$ python benchmarks/dynamo/torchbench.py --accuracy --timing --backend aot_eager --dynamic-shapes --float32 --only hf_Reformer
cuda eval hf_Reformer   PASS
TIMING: entire_frame_compile:49.54918 backend_compile:8.16953
STATS: call_* op count: 533FakeTensor.__torch_dispatch__:13616 | ProxyTorchDispatchMode.__torch_dispatch__:1105

TensorIterator style short circuiting https://gist.github.com/ad050229c9d1686762e24d80eaef541b

(/home/ezyang/local/a/pytorch-env) [ezyang@devgpu020.ftw1 ~/local/a/pytorch (bd4a5b40)]$ python benchmarks/dynamo/torchbench.py --accuracy --timing --backend aot_eager --dynamic-shapes --float32 --only hf_Reformer
cuda eval hf_Reformer   PASS
TIMING: entire_frame_compile:49.64644 backend_compile:7.93716
STATS: call_* op count: 533FakeTensor.__torch_dispatch__:14651 | ProxyTorchDispatchMode.__torch_dispatch__:1105

More info:

TIMING: entire_frame_compile:49.7405 backend_compile:7.6652
STATS: call_* op count: 533 | FakeTensor.__torch_dispatch__:14651 | FakeTensorMode.__torch_dispatch__:24921 | ProxyTorchDispatchMode.__torch_dispatch__:1105 | attempt fast:630 | fast is_contiguous:93 | slow scalars and tensors:204 | slow shape mismatch:327 | slow ending:6

five slowest aot_eager dynamic training models today

pnasnet5large
timm_efficientdet
volo_d1_224
tf_mixnet_l
hrnet_w18

the five slowest in static baseline is

hrnet_w18
cait_m36_384
AllenaiLongformerBase
res2net101_26w_4s
MobileBertForQuestionAnswering

though i don't know if this is training or eval, voz's sheet doesn't say

a very unscientific napkin math says that for every operator in the original dynamo graph, expect 20ms of backend compile time for aot_eager. That's... really fucking high.

that's like, disk seek amount of time

also, typical operator count is O(1000), so that's 20s of compile time

but the big models are as much as 5k ops, so now you're in minutes (and you've blown out our compile budget, without even including inductor/triton)

Horace He

related, but take a guess

for hf_Bert, post dynamo, we have 570 nodes

I guess a 10x node expansion

Horace He

(and inference)

take a guess, 1. how many nodes we have after aotautograd

2. How many times we hit fake tensor's *_torch_dispatch_*

and another 10x on top of that for fake

Horace He

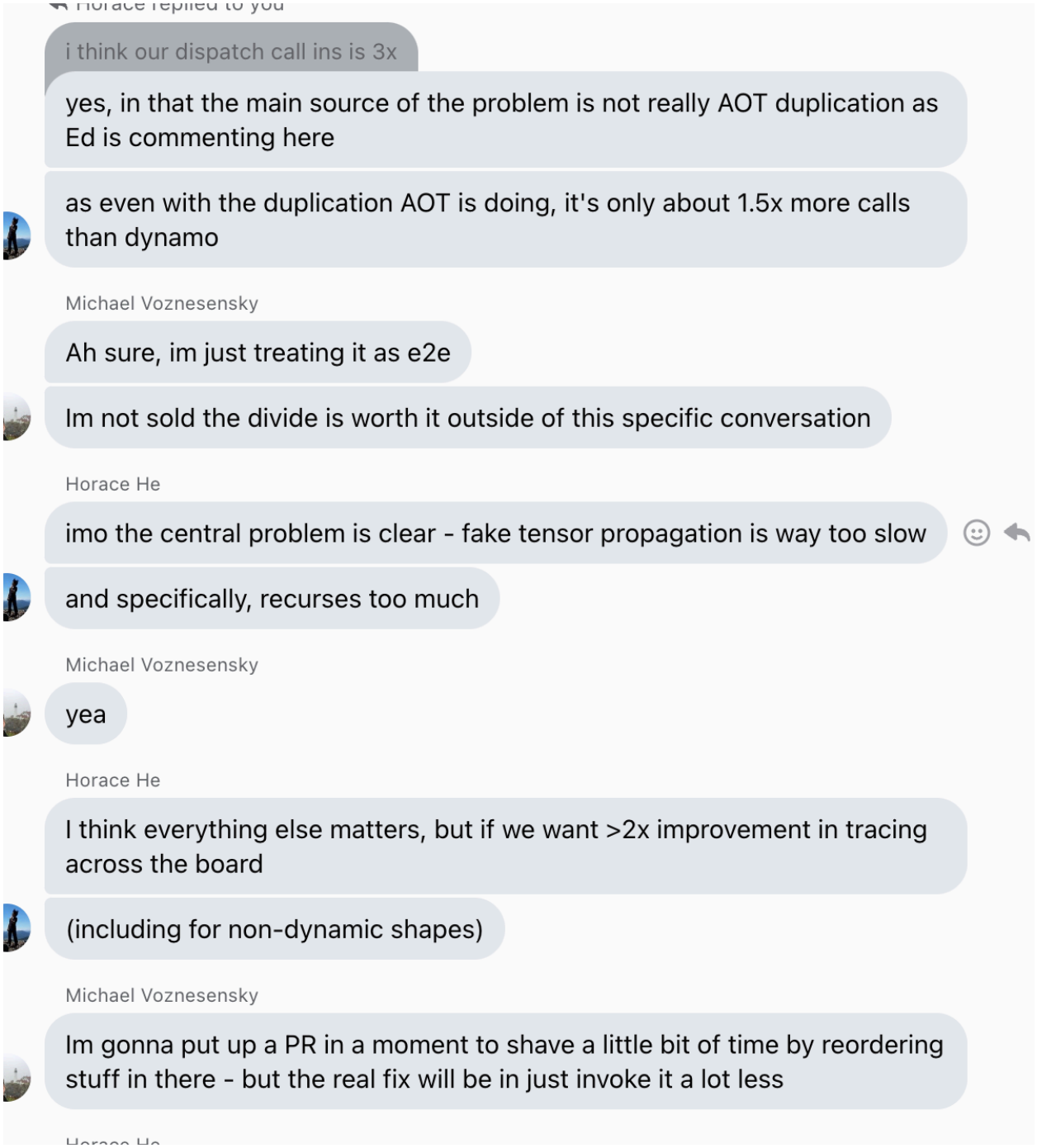
so

off by a factor of 3x 🙂

but in different directions

post dynamo: 570 nodes
post aotautograd: 1528 nodes
fake tensor *_torch_dispatch_*: 47000 invocations

😮😢 2

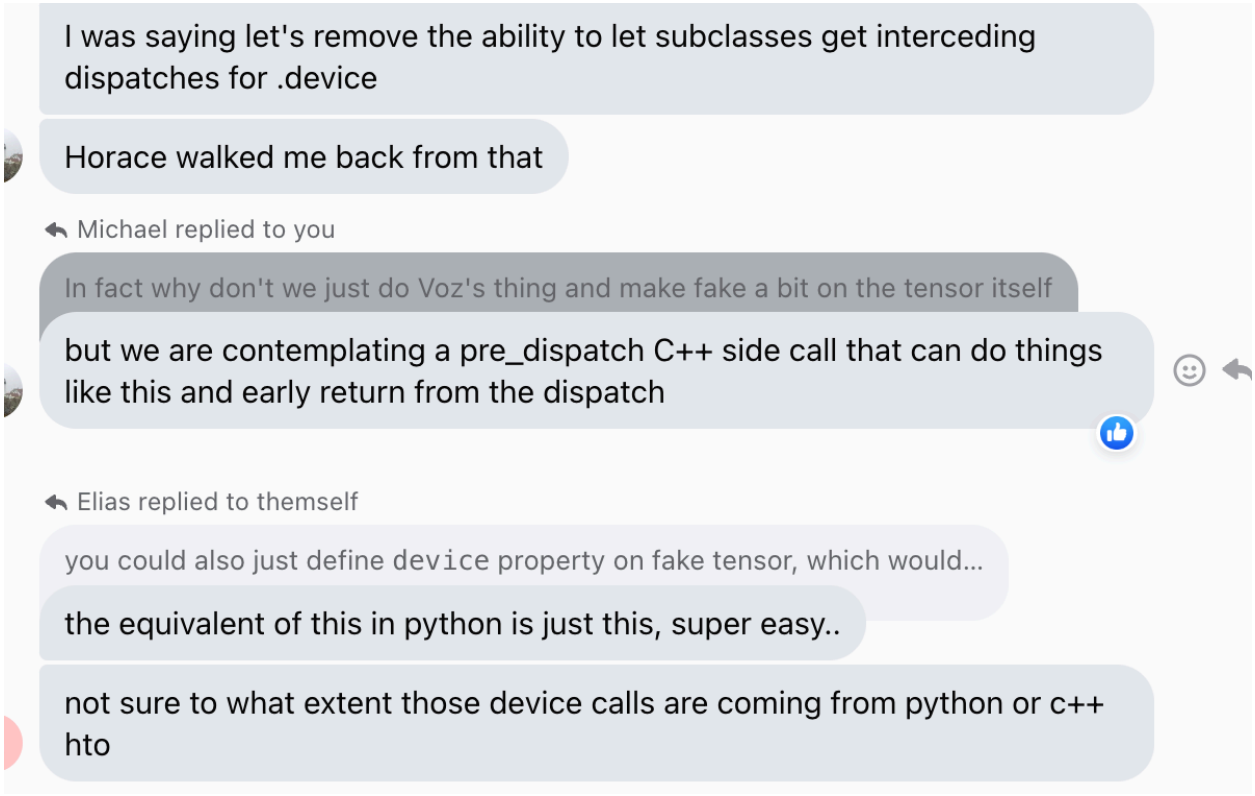


i think a few different things could work.

- benchmark which operators are taking the longest, and write a higher level rule instead of recursion.

- when we run an operator, cache on non-Tensor arguments and calling signature (e.g. memory_format=None vs memory_format=torch.contiguous_format), and record the output and what guards were added to the shape environment. Then, when you run the operator, see if its inputs are in the cache and pass all of the guards and re-use shape propagation.

TBH, i bet you could get pretty far with just the first one.

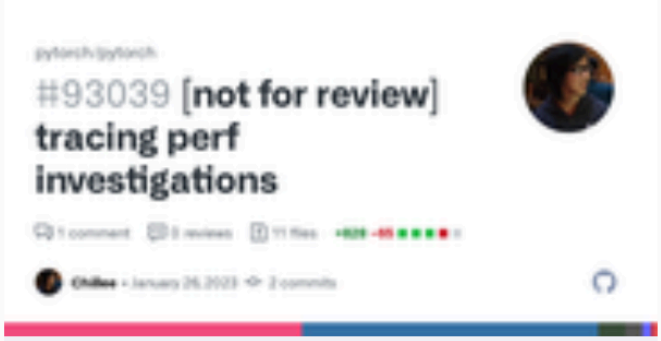



```
./run_all.sh --backend aot_eager --dynamic-shapes --ci --timing
```

haven't run it yet

I put up a (more lightweight) PR for using optree in tracing though

https://github.com/pytorch/pytorch/pull/93039



Use optree for pytrees by Chillee · Pull Request #93039 · pytorch/pytorch

See 10-15% reductions across board

Horace He

Some updates on performance investigations (with Voz).

Our 2 main bottlenecks are fake tensor propagation + new allocations of symbolic tensors (i.e. through empty_strided).

For hrnet_w18, we go from 80 seconds down to 50 seconds with these lines

```
def inner_dispatch(self, func, types, args=(), kwargs=None):
    kwargs = kwargs if kwargs else {}
    with no_dispatch():
        if func in {aten.mul.Tensor, aten.add.Tensor, aten.sub.Tensor,
        aten.relu.default}:
            return FakeTensor(self, torch.empty(args[0].shape,
            device='meta'), device='cuda')
```

2

Michael Voznesensky

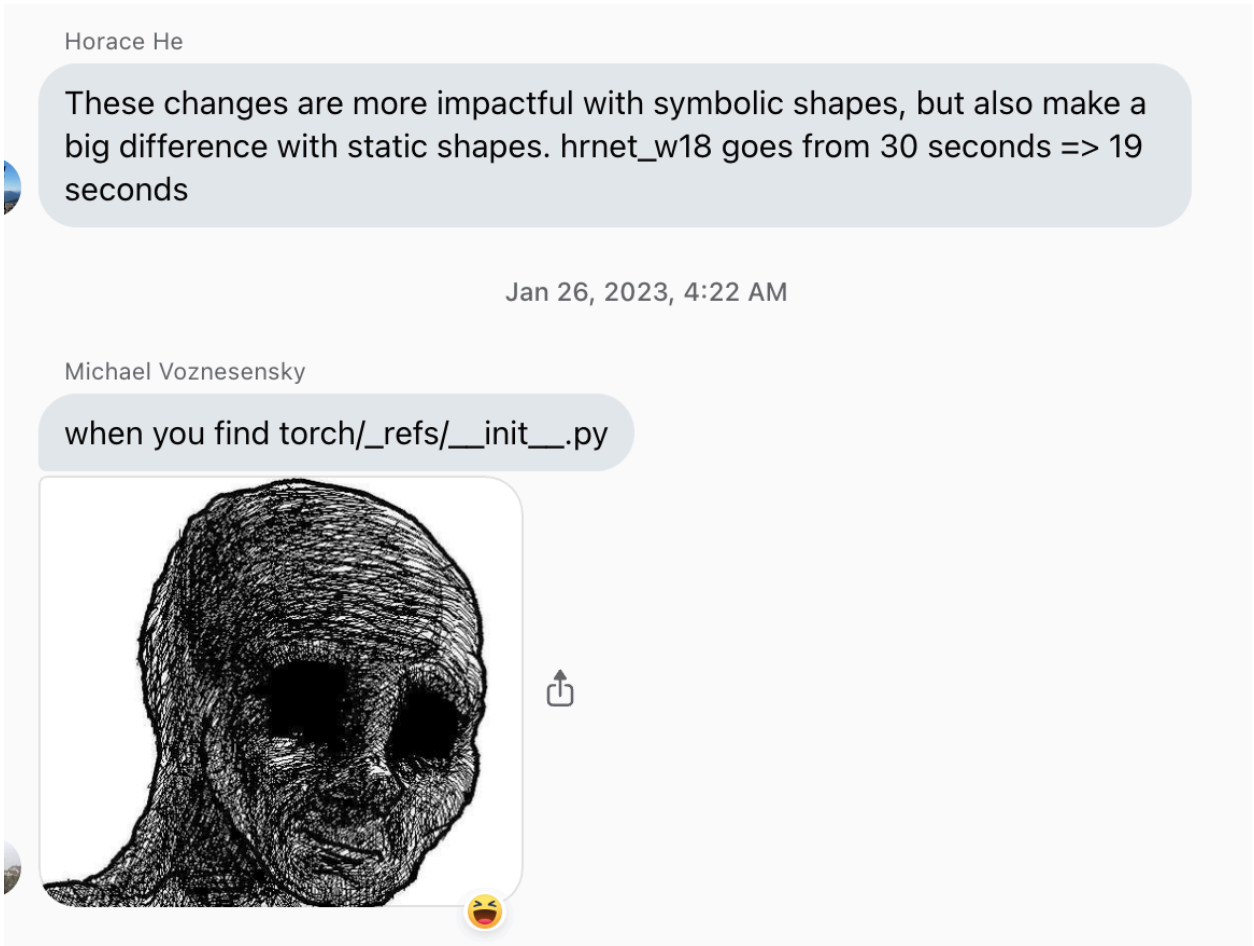
I didnt do anything here

Horace He

And then, we go from 50 seconds down to 36 seconds with this small change.

```
def inner_dispatch(self, func, types, args=(), kwargs=None):
    kwargs = kwargs if kwargs else {}
    with no_dispatch():
        if func in {aten.mul.Tensor, aten.add.Tensor, aten.sub.Tensor,
        aten.relu.default}:
            return args[0]
```

Michael Voznesensky



Reproducing Horace/Voz experiment

Baseline

(/home/ezyang/local/a/pytorch-env) [ezyang@devgpu020.ftw1 ~/local/a/pytorch (ab0e3db0)]$ python benchmarks/dynamo/timm_models.py --accuracy --timing --backend aot_eager --dynamic-shapes --float32 --only hrnet_w18
cuda eval hrnet_w18   PASS
TIMING: entire_frame_compile:54.19504 backend_compile:33.86702
STATS: call_* op count: 1369 | FakeTensor.__torch_dispatch__:72549 | FakeTensorMode.__torch_dispatch__:115542 | ProxyTorchDispatchMode.__torch_dispatch__:3103

Optimized https://gist.github.com/61a4f858882799719801e18073e6dc4a

(/home/ezyang/local/a/pytorch-env) [ezyang@devgpu020.ftw1 ~/local/a/pytorch (ab0e3db0)]$ python benchmarks/dynamo/timm_models.py --accuracy --timing --backend aot_eager --dynamic-shapes --float32 --only hrnet_w18

cuda eval hrnet_w18   PASS
TIMING: entire_frame_compile:34.99378 backend_compile:21.44599
STATS: call_* op count: 1369 | FakeTensor.__torch_dispatch__:54978 |
FakeTensorMode.__torch_dispatch__:71740 | ProxyTorchDispatchMode.__torch_dispatch__:3103

This is roughly consistent

Threats to validity:

- VERY common to broadcast; not even clear you got the correct output shape here

Some ablations:

Remove no_dispatch

(/home/ezyang/local/a/pytorch-env) [ezyang@devgpu020.ftw1 ~/local/a/pytorch (ab0e3db0)]$ python benchmarks/dynamo/timm_models.py --accuracy --timing --backend aot_eager --dynamic-shapes --float32 --only hrnet_w18
cuda eval hrnet_w18   PASS
TIMING: entire_frame_compile:34.71792 backend_compile:21.25611
STATS: call_* op count: 1369 | FakeTensor.__torch_dispatch__:54978 |
FakeTensorMode.__torch_dispatch__:71740 | ProxyTorchDispatchMode.__torch_dispatch__:3103

NO IMPACT

Remove relu

(/home/ezyang/local/a/pytorch-env) [ezyang@devgpu020.ftw1 ~/local/a/pytorch (ab0e3db0)]$ python benchmarks/dynamo/timm_models.py --accuracy --timing --backend aot_eager --dynamic-shapes --float32 --only hrnet_w18
cuda eval hrnet_w18   PASS
TIMING: entire_frame_compile:37.09323 backend_compile:23.43348
STATS: call_* op count: 1369 | FakeTensor.__torch_dispatch__:54978 |
FakeTensorMode.__torch_dispatch__:77628 | ProxyTorchDispatchMode.__torch_dispatch__:3103

5 SEC IMPACT (OUT OF 30 SEC)

Hypothesis: need to increase hit rate for fast path

Baseline (copied for ease of view)

(/home/ezyang/local/a/pytorch-env) [ezyang@devgpu020.ftw1 ~/local/a/pytorch (ab0e3db0)]$ python benchmarks/dynamo/timm_models.py --accuracy --timing --backend aot_eager --dynamic-shapes --float32 --only hrnet_w18
cuda eval hrnet_w18   PASS
TIMING: entire_frame_compile:54.19504 backend_compile:33.86702

STATS: call_* op count: 1369 | FakeTensor.__torch_dispatch__:72549 | FakeTensorMode.__torch_dispatch__:115542 | ProxyTorchDispatchMode.__torch_dispatch__:3103

Recomparing https://gist.github.com/85c5253e4226f05d6bb7db423ff17131

(/home/ezyang/local/a/pytorch-env) [ezyang@devgpu020.ftw1 ~/local/a/pytorch (ab0e3db0)]$ python benchmarks/dynamo/timm_models.py --accuracy --timing --backend aot_eager --dynamic-shapes --float32 --only hrnet_w18
cuda eval hrnet_w18   PASS
TIMING: entire_frame_compile:83.27334 backend_compile:48.46633
STATS: call_* op count: 1369 | FakeTensor.__torch_dispatch__:73960 | FakeTensorMode.__torch_dispatch__:123465 | attempt fast:6352 | slow scalars and tensors:2142 | slow shape mismatch:3903 | fast is_contiguous:307 | ProxyTorchDispatchMode.__torch_dispatch__:3103

WORSE THAN BASELINE

Ablation: always run slow

(/home/ezyang/local/a/pytorch-env) [ezyang@devgpu020.ftw1 ~/local/a/pytorch (ab0e3db0)]$ python benchmarks/dynamo/timm_models.py --accuracy --timing --backend aot_eager --dynamic-shapes --float32 --only hrnet_w18
cuda eval hrnet_w18   PASS
TIMING: entire_frame_compile:56.0579 backend_compile:35.4219
STATS: call_* op count: 1369 | FakeTensor.__torch_dispatch__:73039 | FakeTensorMode.__torch_dispatch__:125307 | ProxyTorchDispatchMode.__torch_dispatch__:3103

Hypothesis: difference due to not restoring fake mode

Restore mode before slow, always run slow

(/home/ezyang/local/a/pytorch-env) [ezyang@devgpu020.ftw1 ~/local/a/pytorch (ab0e3db0)]$ python benchmarks/dynamo/timm_models.py --accuracy --timing --backend aot_eager --dynamic-shapes --float32 --only hrnet_w18
cuda eval hrnet_w18   PASS
TIMING: entire_frame_compile:55.11766 backend_compile:34.50054
STATS: call_* op count: 1369 | FakeTensor.__torch_dispatch__:62784 | FakeTensorMode.__torch_dispatch__:125307 | ProxyTorchDispatchMode.__torch_dispatch__:3103

Not exactly the same as baseline.

Audited refs individually: mul is not registered!

Falsified by code reading

Hypothesis: python argument parsing

[NEW BASELINE] Experiment: @property device for fake tensor

(/home/ezyang/local/a/pytorch-env) [ezyang@devgpu020.ftw1 ~/local/a/pytorch (ab0e3db0)]$ python benchmarks/dynamo/timm_models.py --accuracy --timing --backend aot_eager --dynamic-shapes --float32 --only hrnet_w18
cuda eval hrnet_w18   PASS
TIMING: entire_frame_compile:53.97591 backend_compile:33.60832
STATS: call_* op count: 1369 | FakeTensor.__torch_dispatch__:4995 | FakeTensorMode.__torch_dispatch__:89985 | ProxyTorchDispatchMode.__torch_dispatch__:3010

Hypothesis: has_symbolic_shapes

Only fastpath if has_symbolic_shapes

(/home/ezyang/local/a/pytorch-env) [ezyang@devgpu020.ftw1 ~/local/a/pytorch (f13028ce)]$ python benchmarks/dynamo/timm_models.py --accuracy --timing --backend aot_eager --dynamic-shapes --float32 --only hrnet_w18
cuda eval hrnet_w18   PASS
TIMING: entire_frame_compile:54.08163 backend_compile:33.70638
STATS: call_* op count: 1369 | FakeTensor.__torch_dispatch__:4995 | FakeTensorMode.__torch_dispatch__:89985 | ProxyTorchDispatchMode.__torch_dispatch__:3010

BINGO!

@property device for fake tensor with cached meta

(/home/ezyang/local/a/pytorch-env) [ezyang@devgpu020.ftw1 ~/local/a/pytorch (f13028ce)]$ python benchmarks/dynamo/timm_models.py --accuracy --timing --backend aot_eager --dynamic-shapes --float32 --only hrnet_w18
cuda eval hrnet_w18   PASS
TIMING: entire_frame_compile:55.28344 backend_compile:34.49498
STATS: call_* op count: 1369 | FakeTensor.__torch_dispatch__:4995 | FakeTensorMode.__torch_dispatch__:89985 | ProxyTorchDispatchMode.__torch_dispatch__:3010

Pending experiments:

- Improve fastpath speed

hf_Reformer WITH voz patch

(/home/ezyang/local/a/pytorch-env) [ezyang@devgpu020.ftw1 ~/local/a/pytorch (f13028ce)]$ pp python benchmarks/dynamo/torchbench.py --accuracy --timing --backend aot_eager --dynamic-shapes --float32 --only hf_Reformer
cuda eval hf_Reformer   PASS
TIMING: entire_frame_compile:48.50746 backend_compile:7.17334
STATS: call_* op count: 533 | FakeTensorMode.__torch_dispatch__:19506 | FakeTensor.__torch_dispatch__:1374 | ProxyTorchDispatchMode.__torch_dispatch__:1105

hf_Reformer baseline

(/home/ezyang/local/a/pytorch-env) [ezyang@devgpu020.ftw1 ~/local/a/pytorch (f13028ce)]$ pp python benchmarks/dynamo/torchbench.py --accuracy --timing --backend aot_eager --dynamic-shapes --float32 --only hf_Reformer
cuda eval hf_Reformer   PASS
TIMING: entire_frame_compile:50.66977 backend_compile:7.53315
STATS: call_* op count: 533 | FakeTensorMode.__torch_dispatch__:20454 | FakeTensor.__torch_dispatch__:1017 | ProxyTorchDispatchMode.__torch_dispatch__:1105

hf_Reformer baseline with --training

(/home/ezyang/local/a/pytorch-env) [ezyang@devgpu020.ftw1 ~/local/a/pytorch (f13028ce)]$ pp python benchmarks/dynamo/torchbench.py --accuracy --timing --backend aot_eager --dynamic-shapes --float32 --training --only hf_Reformer
cuda train hf_Reformer   PASS
TIMING: entire_frame_compile:49.43997 backend_compile:9.7174
STATS: call_* op count: 673 | FakeTensorMode.__torch_dispatch__:23766 | FakeTensor.__torch_dispatch__:2989 | ProxyTorchDispatchMode.__torch_dispatch__:1596

hf_Reformer WITH voz patch with training

(/home/ezyang/local/a/pytorch-env) [ezyang@devgpu020.ftw1 ~/local/a/pytorch (f13028ce)]$ pp python benchmarks/dynamo/torchbench.py --accuracy --timing --backend aot_eager --dynamic-shapes --float32 --training --only hf_Reformer
cuda train hf_Reformer   PASS
TIMING: entire_frame_compile:48.38342 backend_compile:8.57134
STATS: call_* op count: 673 | FakeTensorMode.__torch_dispatch__:22006 | FakeTensor.__torch_dispatch__:3341 | ProxyTorchDispatchMode.__torch_dispatch__:1602

hf_Reformer WITH voz patch with training WITH short_circuit_alloc removed

(/home/ezyang/local/a/pytorch-env) [ezyang@devgpu020.ftw1 ~/local/a/pytorch (f13028ce)]$ pp python benchmarks/dynamo/torchbench.py --accuracy --timing --backend aot_eager --dynamic-shapes --float32 --training --only hf_Reformer
cuda train hf_Reformer   PASS
TIMING: entire_frame_compile:48.05059 backend_compile:8.43576
STATS: call_* op count: 673 | FakeTensorMode.__torch_dispatch__:22006 | FakeTensor.__torch_dispatch__:3341 | ProxyTorchDispatchMode.__torch_dispatch__:1602

Buttoning up the fast path

Fast path is hitting https://gist.github.com/7fd74cadd80134b8198e0665afdb62a5

(/home/ezyang/local/a/pytorch-env) [ezyang@devgpu020.ftw1 ~/local/a/pytorch (f13028ce)]$ python benchmarks/dynamo/timm_models.py --accuracy --timing --backend aot_eager --dynamic-shapes --float32 --only hrnet_w18
cuda eval hrnet_w18   PASS
TIMING: entire_frame_compile:40.18931 backend_compile:25.28828
STATS: call_* op count: 1369 | FakeTensor.__torch_dispatch__:4995 | FakeTensorMode.__torch_dispatch__:69478 | attempt fast:4399 | fast is_contiguous:4399 | ProxyTorchDispatchMode.__torch_dispatch__:3010

20k fake tensor dispatch reduction. 4k hits to fastpath. This implies 5 calls elided per. Overall 22s-ish reducion. 10s off from the Horace roofline; some of it (5s) is explainable by missing relu.