MIXED PRECISION TRAINING FOR SEVERAL CONDITIONAL GANS

Ming-Yu Liu

WHY MIXED PRECISION?

- Generative models mostly operate in the under-fitting region.
  - Mode dropping, blurry outputs, artifacts
- To further improve the performance, one trivial solution is to increase the model capacity or to train with a larger batch size.
- Issue: Both will lead to a larger memory consumption and longer training time.
- Solutions
  - Spend more money to upgrade your machine.
  - Become a CUDA Ninja and write super efficient CUDA kernels.
  - Use AMP to capitalize the TensorCores available in your GPUs.

IMAGINAIRE

- A PyTorch Library for a wide range of conditional GAN models.
  - AMP-ready
  - Speed optimized
- Contain SOTA image and video synthesis models including
  - Supervised image-to-image translation
    - pix2pixHD (CVPR'18), SPADE (CVPR'19)
  - Unsupervised image-to-image translation
    - UNIT (NeurIPS'17), MUNIT (ECCV'18), FUNIT (ICCV'19), COCO-FUNIT (ECCV'20)
  - Video-to-video synthesis
    - vid2vid (NeurIPS'18), few-shot vid2vid (NeurIPS'19), world-consistent vid2vid (ECCV'20)

IMAGINAIRE WITH AMP

```
# AMP initialization.
[net_G, net_D], [opt_G, opt_D] = \
    amp.initialize([net_G, net_D], [opt_G, opt_D],
                   opt_level=cfg.trainer.amp, num_losses=2)
```

```
with amp.scale_loss(total_loss, self.opt_G, loss_id=0) as scaled_loss:
    scaled_loss.backward()
self.opt_G.step()
```

```
with amp.scale_loss(total_loss, self.opt_D, loss_id=1) as scaled_loss:
    scaled_loss.backward()
self.opt_D.step()
```

SPECTRAL NORM

```
weight_mat = self.reshape_weight_to_matrix(weight)
if self.fp16:
    weight_mat = weight_mat.half()
```

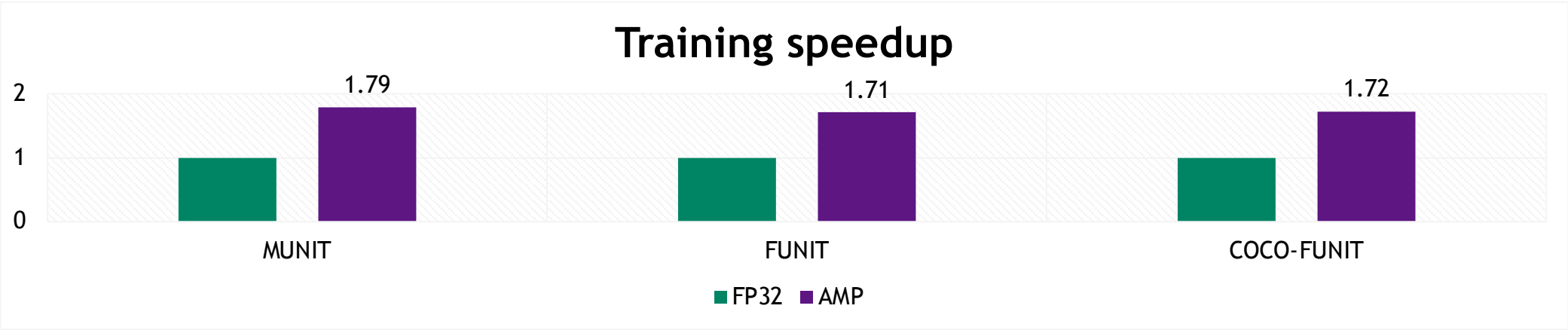
```
@torch.jit.script
def fuse_norm(input: torch.Tensor, eps: float, out: torch.Tensor):
    out = normalize(input, dim=0, eps=eps, out=out)
    return out
```

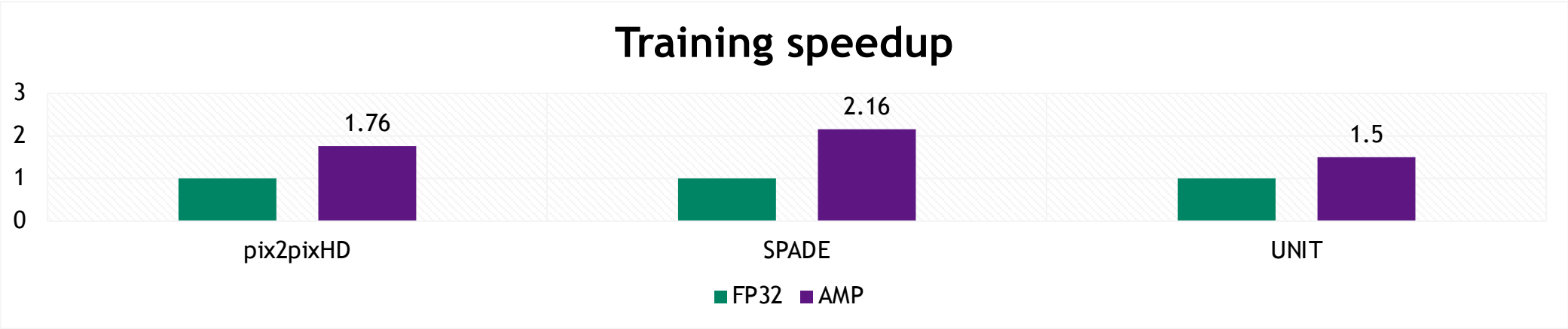
```
v = fuse_norm(
    torch.mv(weight_mat.t(), u),
    self.eps, v)
u = fuse_norm(
    torch.mv(weight_mat, v),
    self.eps, u)
```

```
sigma = torch.dot(u.half(), torch.mv(weight_mat, v.half()))
```

BENEFIT

Same or similar FID but train faster





CONCLUSION

- Speed gain comes free.
- Also save some GPU memory for fitting bigger models.
- The code will be available in September.
- Link: https://github.com/nvlabs/imaginaire


NVIDIA