DL training and inference optimization library towards speed and scale

Samyam Rajbhandari, Jeff Rasley, Olatunji Ruwase, Reza Yazdani Aminabadi, Elton Zheng, Minjia Zhang, Niranjan Uma Naresh, Shaden Smith, Ammar Ahmad Awan, Conglong Li, Cheng Li, Zhewei Yao, Jeffrey Zhu, Yuxiong He
DL System Challenges and Capability

**Challenges**

- Too slow to train high-quality models on massive data
  - More hardware ≠ higher throughput, bigger model
  - Higher throughput ≠ better accuracy, faster convergence
  - Better techniques ≠ handy to use
- Less data / smaller models, tradeoff accuracy for training time
- Slow and expensive to deploy the models

**Desired Capability of DeepSpeed**

- **Efficiency:** Efficient use of hardware for high throughput and scalability
- **Effectiveness:** High accuracy and fast convergence, lowering cost
- **Easy to use:** Improve development productivity of model scientists
**DL Training and Inference Optimization: DeepSpeed**

### Bert - Original

```python
# Construct distributed model
model = BertMultiTask(...)
model = DistributedDataParallel(model)

... # Construct FP16 optimizer
optimizer = FusedAdam(model.parameters, ...)
optimizer = FP16_Optimizer(optimizer, ...)

# Forward pass
loss = model(batch)

# Backward pass
optimizer.backward(loss)

# Parameter update
optimizer.step()
```

### Bert – w. DeepSpeed

```python
# Construct Bert model
model = BertMultiTask(...)

# Wrap to get distributed model and FP16 optimizer
model, optimizer, _, _ = deepspeed.initialize(
    args=ARGS,
    model=model,
    model_parameters=model.parameters,
    ...)

# Forward pass
loss = model(batch)

# Backward pass
model.backward(loss)

# Parameter update
model.step()
```

---

**Minimal code change**

**Efficiency + Effectiveness**

**Speed + Scale**
DeepSpeed
https://github.com/microsoft/DeepSpeed
System capability to efficiently train models with **20 trillion** parameters

DeepSpeed key technologies:
- ZeRO: Zero Redundancy Optimizer
- 3D parallelism: data parallelism, pipeline, and model parallelism
- ZeRO-Infinity

**Model Scale**
- 10 Trillion parameters

**Speed**
- Fast & scalable training

**Democratize AI**
- Bigger & faster for all

**Compressed Training**
- Boosted efficiency

**Accelerated inference**
- Up to 6x faster & cheaper

**Usability**
- Few lines of code changes
# Fastest Transformer Kernels

<table>
<thead>
<tr>
<th>#Devices</th>
<th>Source</th>
<th>Training Time</th>
</tr>
</thead>
<tbody>
<tr>
<td>256 V100 GPUs</td>
<td>Nvidia</td>
<td>236 mins</td>
</tr>
<tr>
<td>256 V100 GPUs</td>
<td>DeepSpeed</td>
<td>144 mins</td>
</tr>
<tr>
<td>1024 TPU3 chips</td>
<td>Google</td>
<td>76 mins</td>
</tr>
<tr>
<td>1024 V100 GPUs</td>
<td>Nvidia</td>
<td>67 mins</td>
</tr>
<tr>
<td>1024 V100 GPUs</td>
<td>DeepSpeed</td>
<td>44 mins</td>
</tr>
</tbody>
</table>

## World Fastest BERT Training

- **#Devices**: 1024 V100 GPUs
- **Source**: Nvidia
- **Training Time**: 67 mins

## Scalable distributed training through ZeRO-powered DP

- Superlinear speedup with increasing #GPUs

## DeepSpeed key technologies

- **Efficiency**: ZeRO, ultra-fast GPU kernels, IO/compute/communication overlapping
- **Effectiveness**: Advanced HP tuning, large-batch scaling

## Model Scale

- **10 Trillion parameters**

## Speed

- **Fast & scalable training**

## Democratize AI

- **Bigger & faster for all**

## Compressed Training

- **Boosted efficiency**

## Accelerated inference

- **Up to 6x faster & cheaper**

## Usability

- **Few lines of code changes**
ZeRO-Infinity: 1 Trillion model on a single GPU, 700x bigger

1-bit Adam: 5x less communication, 3.5x faster training

Model Scale
- 10 Trillion parameters

Speed
- Fast & scalable training

Democratize AI
- Bigger & faster for all

Compressed Training
- Boosted efficiency

Accelerated inference
- Up to 6x faster & cheaper

Usability
- Few lines of code changes
• **Sparse attention**: 10x longer seq, up to 6x faster

• **Progressive Layer Drop**: Compressed robust training
  - 24% faster when training the same number of samples
  - 2.5X faster to get similar accuracy on downstream tasks

Model Scale
- 10 Trillion parameters

Speed
- Fast & scalable training

Democratize AI
- Bigger & faster for all

Compressed Training
- Boosted efficiency

Accelerated inference
- Up to 6x faster & cheaper

Usability
- Few lines of code changes
Accelerated inference for large-scale transformer models

**Up to 6x faster and cheaper**

DeepSpeed key inference technologies:
- Inference-adapted parallelism
- Inference optimized CUDA kernels
- Effective quantize-aware training and efficient quantized kernels

| Model Scale | • 10 Trillion parameters |
| Speed       | • Fast & scalable training |
| Democratize AI | • Bigger & faster for all |
| Compressed Training | • Boosted efficiency |
| Accelerated inference | • Up to 6x faster & cheaper |
| Usability    | • Few lines of code changes |

---

![Graph showing throughput improvement for different models with DeepSpeed Inference vs Baseline.](chart.png)
• Only few lines of code changes to enable DeepSpeed on PyTorch models
• Scalable and convenient data parallelism

- Infrastructure agnostic, supporting AzureML, Azure VMs, local-nodes
- HuggingFace and PyTorch Lightning integrate DeepSpeed as a performance-optimized backend

<table>
<thead>
<tr>
<th>Trainable Model Parameters (Billions)</th>
<th>ZeRO-Powered</th>
<th>PyTorch DP</th>
</tr>
</thead>
<tbody>
<tr>
<td>0</td>
<td>1.4B</td>
<td>1T</td>
</tr>
</tbody>
</table>

Training Massive Models without requiring model parallelism

Model Scale
- 10 Trillion parameters

Speed
- Fast & scalable training

Democratize AI
- Bigger & faster for all

Compressed Training
- Boosted efficiency

Accelerated inference
- Up to 6x faster & cheaper

Usability
- Few lines of code changes
ZeRO-Infinity
Breaking GPU Memory Wall for DL Training
Large model training landscape today

- GPU Memory Wall
  - 1T (10T) params: 800 (8K) V100 GPUs
  - How do we support the growth in model size?

- Accessibility to large model training
  - 256 GPUs to fine-tune GPT-3
  - Limited access to such resources

- Model code refactoring
  - Re-writing the model using 3D parallelism (tensor-slicing + pipeline parallelism)
  - Painful and error prone

---

*AI and Memory Wall. (This blogpost has been written in... | by Amir Gholami | riselab | Medium*
Redefining the landscape with ZeRO-Infinity

• Beyond GPU Memory
  • 50x larger models
  • 32T params on 512 GPUs (instead of 25K)

• Broader access to large model training
  • GPT-3 sized fine-tuning on a single node/GPU (instead of 16 nodes)

• Excellent Throughput and Scalability
  • Comparable to 3D-parallelism

• Ease of Use
  • No model refactoring necessary

Paper: ZeRO-Infinity: Breaking the GPU Memory Wall for Extreme Scale Deep Learning (arxiv.org)
Beyond the GPU Memory

- Modern clusters have heterogeneous memory systems.

- GPU memory comprises a small fraction

- ZeRO-Infinity leverages GPU/CPU/NVMe memory
  - 32T params on 32 nodes
  - 1T params on a single node

- GPT-3 can be fine-tuned on a single node
Bandwidth Requirements

(a) Parameter and Gradient Bandwidth  
(b) Optimizer States bandwidth  
(c) Activation Checkpoint Bandwidth

Figure 3: Impact of bandwidth on efficiency assuming an accelerator with 70 TFlops of single GPU peak achievable throughput.
ZeRO-Infinity Architecture
Evaluation

(a) ZeRO-Infinity efficiently trains 40x larger models than 3D parallelism on 512 GPUs.

(b) ZeRO-Infinity exceeds linear scaling from 64 to 512 GPUs for a 1T parameter model.

(c) ZeRO-Infinity can train up to 1T model on a DGX-2 node without model parallelism.

Figure 5: Efficiency and scalability of ZeRO-Infinity for training multi-trillion parameter models.
ZeRO-Infinity in a nutshell

**Massive Model Scale**
10T - 100T parameters

**Broader Access**
1T parameters on a single GPU

**Excellent Efficiency**
49 TFLOPs per V100 GPU

**Super-linear Scaling**
512 GPUs and beyond