Train and deploy large language models on Amazon SageMaker

Dhawal Patel
Principal AI/ML Specialist SA
AWS

Gili Nachum
Senior AI/ML Specialist SA
AWS

Madison Van Horn
NLP Engineer
Mantium
Agenda

1. Training large language models (LLMs)
2. Optimizing LLMs
3. Hosting LLMs (inference)
4. Customer spotlight – LLM MLOps at Mantium
What are large language models (LLMs)?

Input text

LLM

Output text

Embedding
01001101100101
1001101100101
Size of large NLP models is increasing

Models are becoming more complex, with ML practitioners moving from classical ML to deep learning.

State-of-the-art deep learning models are getting larger and larger as we find that larger models generalize better.

They can be applied to a range of tasks to create more powerful digital assistants and generate better search results and product recommendations.

They can be used for multi-modal use cases like answering visual questions from people who are vision-impaired, answering questions visually, emotion recognition, and many more.

15,000x increase in 5 years

Model size

Transformers 65M  BERT 340M  GPT-2 1.5B  GPT-2 8B 8.3B  T5 11B  Turing-NLG 17B  Megatron-Turing 530B  GPT-3 175B  GPT-3 1T 1 trillion

Time

# Pretraining vs. fine-tuning LLMs

<table>
<thead>
<tr>
<th></th>
<th>Pretraining</th>
<th>Fine tuning</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Training duration</strong></td>
<td>Weeks to months</td>
<td>Minutes to hours</td>
</tr>
<tr>
<td><strong>Customization</strong></td>
<td><strong>FULL</strong></td>
<td>Some</td>
</tr>
<tr>
<td></td>
<td>• NN architecture and size</td>
<td>• Specific task tuning</td>
</tr>
<tr>
<td></td>
<td>• Vocabulary size</td>
<td>• Added domain-specific</td>
</tr>
<tr>
<td></td>
<td>• Context length</td>
<td>training data</td>
</tr>
<tr>
<td></td>
<td>• Training data</td>
<td></td>
</tr>
<tr>
<td><strong>Expertize needed</strong></td>
<td>High</td>
<td>Low</td>
</tr>
</tbody>
</table>
Challenges in pretraining LLMs

- Time to train
- Cost to train
- Expertise required
Training LLMs in Amazon SageMaker

DISTRIBUTED TRAINING LIBRARIES

- TensorFlow
- PyTorch
- SageMaker distributed training
- OSS libraries

AMAZON SAGEMAKER PLATFORM

- Large-scale cluster orchestration
- Fault tolerant
- Hugging Face integration
- SageMaker Training Compiler

ML COMPUTE INSTANCES & ACCELERATORS

- 400/800 Gbps EFA networking
- NVIDIA A100 GPUs 40 GB/80 GB
- AWS Trainium
- Habana Gaudi

PyTorch, the PyTorch logo and any related marks are trademarks of Facebook, Inc.
Distributed training methods

Distributed training

Data parallelism

Model parallelism
SageMaker model parallelism options

- Pipeline parallelism
- Tensor parallelism
- Sharded data parallel

Model parallelism leads to:

- Memory optimizations
  - Activations checkpointing/offloading
  - Optimizer state sharding
Optimizing large language models (LLMs) with Amazon SageMaker

Dhawal Patel
Principal AI/ML Specialist SA, AWS
Challenges

1. Model too large to fit in a single GPU for inference
2. Inference latency
3. High cost for model inference
Large model inference optimization techniques

Optimizing model serving

Model compilation
(SageMaker Neo)

Model compression
(DeepSpeed)

Model sharding
(SageMaker, DeepSpeed, Parallelformers)

Quantization
(TensorRT)

Pruning
(DeepSpeed, HF)

Distillation
(SageMaker, HF)

Accuracy:
Performance:

Lossless accuracy/Optimal performance

Loss in accuracy

HF = Hugging Face
Hosting large models with Amazon SageMaker
## SageMaker model deployment stack

### Amazon SageMaker

- **Real-time inference**
- Asynchronous inference
- Serverless inference
- Batch inference
- Multi-model endpoints

### SAGEMAKER STUDIO IDE

- Multi-container endpoints
- Inference DAG and pipelines
- Manage and version models
- MLOps
- Model monitoring
- Metrics and logging in CloudWatch

### FRAMEWORKS

- TensorFlow
- PyTorch
- mxnet
- Keras
- ONNX
- DJL
- BYOC

### MODEL SERVERS

- AWS Deep Learning Containers
- TensorFlow Serving
- TorchServe
- NVIDIA Triton Inference Server
- Multi Model Server (MMS)
- Deep Java Learning Serving (DJLServing)

### ML COMPUTE INSTANCES & ACCELERATORS

- CPUs
- GPUs
- Inferentia & Trainium
- Graviton (ARM)

### DEEP LEARNING COMPILERS AND RUNTIMES

- SageMaker Neo
- NVIDIA TensorRT/cuDNN
- Intel oneDNN
- ARM Compute Library
Hosting GPT-J-6B model on multiple GPUs

- Amazon SageMaker endpoint
  - Amazon EBS volume size setting
  - Timeout setting
- DJLServing (model server)
- DeepSpeed inference
- Model (tensor) parallelism

Access the complete solution
Key takeaways for hosting LLMs

- Use SageMaker **Neo** and **TensorRT** for quick optimization gains
- Use **prebuilt** SageMaker **Hugging Face** Deep Learning Containers
- Use **prebuilt** SageMaker **DJLServing/DeepSpeed**–based deep learning container if model cannot fit on single GPU for supported model architectures
- Use SageMaker **MME** for hosting multiple LLMs*
- Use **AWS Inferentia** for supported **model architectures**

*Applicable for LLMs that can fit in single GPU memory
How we thrive

Veterans of the AI space
Team members from Google, Nexosis, and DataRobot. Founder’s previous startup was acquired by DataRobot.

World-class investors
Round co-led by venture funds Drive Capital and Top Harvest.

Industry and academic relationships
Member of the AWS Partner Network and NVIDIA Inception program. Research collaboration with Mila – Quebec AI Institute.

AI providers
Users have centralized access to many powerful AI models from AWS, AI21, Cohere, EleutherAI, OpenAI, Hugging Face, and more.

Security
Completed SOC 2 Type 2 audit for Data Security Standards.

How we create value

Accelerated AI development and testing
The AI Builder is a low-code solution to building powerful AI workflows that can be prototyped, tested, and deployed quickly.

Powerful integrations
Integrations give enterprises the power to automate virtually any manual task.

Decreased infrastructure complexity
We easily integrate into an enterprise’s existing infrastructure.

Faster prototype to production
AI innovations get into the day-to-day operations of your business faster than ever before.

Purpose
We are determined to remove the barrier between non-technical innovators and the development of artificial intelligence.

Supporting enterprises to implement AI faster than historically possible, allowing people across organizations to abandon time-consuming tasks and focus on strategic contributions, gaining value quickly.

Strategy
Lower the barriers faced by non-technical AI innovators by simplifying AI development and infrastructure complexities.
Mantium MLOps serves two customers – Mantium AI engineers and Mantium users

**Mantium AI engineers**

Need to

- Train models and develop training protocols
- Deploy models
- Conduct inference with models

**Mantium users**

Need to

- Train models
- Conduct inference with models
- Incorporate models into custom workflows
Example MLOps lifecycle for GPT-J and RoBERTa Large

**Select models**
- 6.7B-parameter LLM
- GPT-J
- RoBERTa Large

**Train models**
- Parameter-efficient tuning with soft prompts
  - Distributed model parallelism
  - Tensor parallelism
- Full-model training
  - Distributed model parallelism
  - Tensor parallelism

**Deploy models**
- Multiple models, some with support for soft prompts
  - DeepSpeed inference kernel
  - Variable instance types
- Full-model optimization
  - Distributed model parallelism
  - Tensor parallelism

**Deploy training protocol**
- Parameter-efficient tuning with soft prompts
  - Distributed model parallelism
  - Tensor parallelism
  - Full-model optimization

**Model selection**
- 354M-parameter LLM

**Model training**
- Full-model training
  - Hugging Face estimator
  - Training compiler

**Deploy models**
- Multi-step optimization process
  - 8-bit quantization
  - SageMaker Neo
  - AWS Inferentia

**Deploy training protocol**
- Multi-step optimization process
  - 8-bit quantization
  - SageMaker Neo
  - AWS Inferentia
Training compiler

Artemis
- Config-based and extensible
- Abstraction over estimators
- Abstraction over preprocessing and training scripts

MantiumNLP
- Core Mantium training and inference implementations
- Simple abstractions
- Extensible, designed for rapid dev cycles

SageMaker Pipelines
- Serverless, real-time, async
- Triton, DeepSpeed

Deployment
Mantium uses SageMaker to support state-of-the-art LLM development and deployment for Mantium engineers and Mantium users.

To do this, we leverage the best technical solutions for a specific problem and

We build production-ready abstractions over them in order to minimize barriers between internal research and our user-facing product.
Get started with LLMs on Amazon SageMaker

Training LLMs

- Access end-to-end distributed training LLMs on SageMaker example notebooks
- Explore examples with a variety of frameworks including PyTorch, TensorFlow, and Hugging Face

Hosting LLMs

Access complete solution for how to optimize and host Hugging Face LLM with low latency and high throughput on SageMaker

Hosting LLM on single GPU

Hosting LLM on multiple GPUs
Thank you!

Please complete the session survey in the mobile app.