Solution Brief

Cross-Industry Architecture
Intel® Optimized Cloud Modules

intel software

Intel® Optimized Cloud Modules for AWS*: Stable Diffusion Distributed Training

Author: Eduardo Alvarez

Date: Dec 22, 2023

Fine-tune Stable Diffusion models in a distributed architecture on Azure with 4th Generation Intel® Xeon ® Scalable Processors

Discover the power of AI model optimization with the Intel® Optimized Cloud Modules for AWS, designed explicitly for the Stable Diffusion model. This hands-on module guides you through fine-tuning the cutting-edge Stable Diffusion v1.5 model for textual-inversion tasks, harnessing the robust capabilities of 4th Generation Intel® Xeon® Scalable Processors on AWS.

Key Features:

- **Tailored Fine-Tuning**: Learn to customize the Stable Diffusion model for your unique needs using the DICOO image dataset, all within a distributed system architecture.
- **Advanced Computing Selection**: Understand how to choose the right compute resources that exploit the Intel Extension for PyTorch, enabling efficient mixed-precision training with bf16 Torch data types.
- **Enhanced Performance**: Experience improved workload performance thanks to this specialized training approach.
- **Versatile Application**: Apply these skills to fine-tune stable diffusion models using datasets critical to your operations, utilizing widespread CPU compute capabilities available on the AWS cloud.

Who needs it?

- **AI Researchers and Academics**: Those involved in artificial intelligence and machine learning research, especially in neural networks and image processing, would find this solution valuable. It provides a platform for experimenting with and advancing state-of-the-art techniques in AI.
- **Data Scientists and Machine Learning Engineers:** Professionals developing, training, and deploying machine learning models, particularly those involving image data and deep learning, would benefit from the enhanced computing power and optimization capabilities.
- **Cloud Engineers and IT Professionals:** Those responsible for managing cloud infrastructure, especially in environments using AWS, would find this solution helpful for setting up and maintaining efficient, scalable distributed systems for AI model training.
- **Technology Companies and Startups:** Companies focusing on AI-driven products or services, especially those in image recognition, computer vision, or automated content generation, would benefit from the optimized training capabilities to improve their models and services.
- **Enterprise Clients with AI Needs:** Larger enterprises that leverage AI for various applications, such as predictive analytics, customer experience enhancement, or automated decision-making, could use this solution to fine-tune their models more effectively.
- **Innovators in Creative Industries:** Professionals in creative fields, such as graphic design or digital content creation, exploring AI-assisted tools to enhance their work, could also find this solution beneficial.

What it does

By leveraging this module, you can fine-tune stable diffusion with a dataset meaningful to your operations with ubiquitous CPU compute on the AWS cloud. The distributed setup with Hugging Face accelerate allows developers to scale this workload without complex logic typically required with Kubernetes applications. We leverage the Intel® Extension for PyTorch® and Intel® oneAPI Collective Communications Library (oneCCL) to optimize the solution further and support the distributed collective communication operations.

- **Intel Extension for PyTorch** delivers the latest and greatest from Intel before it reaches the upstream branch of stock PyTorch. This solution enables auto-mixed precision with bf16 (half-precision data type).
- **Advanced Matrix Extensions** in 4th Generation Xeon Processors are enabled in this solution. It allows us to perform improved memory management, loading larger matrices per core and delivering faster matrix operations.
- **Hugging Face Accelerate** provides a simple distributed system configurator in the command line to simplify setup. Accelerate has intel-optimized integrations to easily select the Intel Extension for PyTorch and lower precision data types for CPU-distributed training.

Cloud Solution Architecture

This solution utilizes EC2 compute instances within a distributed system, where the Rank 0 node establishes the foundational image. This image is then registered as an AMI and used to construct the Rank 1 and 2 nodes. The entire workload is orchestrated through the Rank 0 node, a process depicted in Figure 1. However, Figure 1 does not show the integral software components that facilitate distributed training. These include the Hugging Face Accelerate API and the oneCCL collective operations, essential for enabling communication and synchronization between the nodes, ensuring consistent updates to the model's state throughout the fine-tuning process.

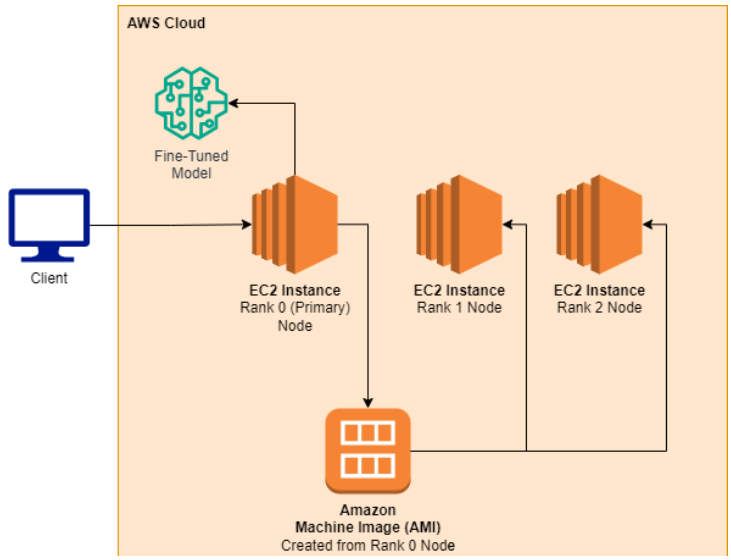

Figure 1 AWS Cloud Optimization Module Solution Architecture. Image by Author.

Code Highlights

Enable the Intel Extension for PyTorch

The Intel Extension for PyTorch elevates PyTorch performance on Intel hardware with the integration of the newest features and optimizations that have not yet been incorporated into open source PyTorch. This extension efficiently utilizes Intel hardware capabilities, such as Intel AVX-512 and Intel AMX on Intel Xeon CPUs. Unleashing this power is straightforward – just wrap your model and optimizer objects with ipex.optimize.

```
# activating ipex optimizations – code can be found in the textual_inversion_icom.py script
import intel_extension_for_pytorch as ipex
unet = ipex.optimize(unet, dtype=weight_dtype)
vae = ipex.optimize(vae, dtype=weight_dtype)
```

Gradient Accumulation with Hugging Face Accelerate

The Accelerate library by Hugging Face streamlines the gradient accumulation process. This package helps to abstract away the complexity of supporting multi-CPUs/GPUs and provides an intuitive, user-friendly API, making gradient accumulation and clipping hassle-free during the training process.

The code snippet below are generic implementations of Accelerate, not the ones specifically found in this solution.

```
# Specifying instance type for estimator data preprocessing pipeline component.
 # Initializing Accelerator object
 self.accelerator = Accelerator(
        gradient_accumulation_steps=gradient_accumulation_steps,
        cpu=True,
 )
# Gradient Accumulation
 with self.accelerator.accumulate(self.model):
        with self.autocast_ctx_manager:
                _, loss = self.model(X, Y)
        self.accelerator.backward(loss)
        loss = loss.detach() / gradient_accumulation_steps

# Gradient Clipping
 self.accelerator.clip_grad_norm_(
        self.model.parameters(), self.trainer_config.grad_clip
 )
```

Executing Distributed Training

For distributed training, we utilized oneCCL. With optimized communication patterns, oneCCL enables developers and researchers to train newer and deeper models more quickly across multiple nodes. OneCCL leverages Intel's Message Passing Interface (MPI) to launch distributed training across system.

```
mpirun -f ./hosts -n 3 -ppn 1 accelerate launch textual_inversion_icom.py --
pretrained_model_name_or_path="runwayml/stable-diffusion-v1-5" --train_data_dir="./dicoo/" --
learnable_property="object"   --placeholder_token="<dicoo>" --initializer_token="toy" --resolution=512  --
train_batch_size=1  --seed=7  --gradient_accumulation_steps=1 --max_train_steps=30 --learning_rate=2.0e-03 --scale_lr -
-lr_scheduler="constant" --lr_warmup_steps=3 --output_dir=./textual_inversion_output --mixed_precision bf16 --
save_as_full_pipeline
```

Next Steps

Download the module from GitHub ›

Register for office hours to get help on your implementation ›

Check out the full suite of Intel Cloud Optimization Modules ›

Come chat with us on our DevHub Discord server to keep interacting with other developers ›