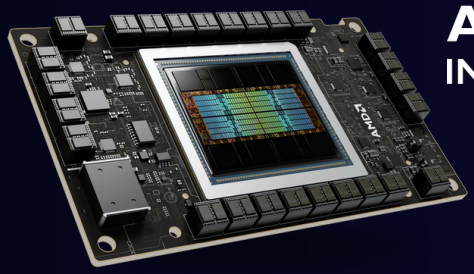
Delivering Choice For Enterprise AI: Multi-Node Fine-Tuning On Dell PowerEdge™ XE9680 With AMD Instinct™ MI300X Accelerators

| May 2024

In this blog, Metrum AI and Dell have partnered to show you how to use domain-specific data to fine-tune the Llama 3 8B Model with BF16 precision on a distributed system of Dell PowerEdge XE9680 Servers equipped with AMD Instinct MI300X Accelerators.



AMD INSTINCT

DELL Technologies AMD METRUM AI Hugging Face

Introduction

Large language models (LLMs) have been a significant breakthrough in AI and demonstrated remarkable capabilities in understanding and generating human-like text across a wide range of domains. The first step in approaching an LLM-assisted AI solution is generally **pre-training,** during which an untrained model learns to anticipate the next token in a given sequence using information acquired from various massive datasets, followed by **fine-tuning**, which involves adapting the pre-trained model for a domain specific task by updating a task-specific layer on top.

Fine-tuning, however, still requires a lot of time, computation, and RAM. One approach to reducing computation time is **distributed fine-tuning**, which allows computational resources to be more efficiently utilized by parallelizing the fine-tuning process across multiple GPUs or devices.

Metrum AI showcased various industry-leading capabilities of Dell PowerEdge XE9680 Servers paired with AMD Instinct MI300X Accelerators on a distributed fine-tuning task by uncovering these key value drivers:

- Developed a distributed finetuning software stack on the flagship Dell PowerEdge XE9680 Server equipped with eight AMD Instinct MI300X Accelerators.
- Fine-tuned Llama 3 8B with BF16 precision using the PubMedQA medical dataset on two Dell PowerEdge XE9680 Servers each equipped with AMD Instinct MI300X Accelerators.
- Deployed a fine-tuned model in an enterprise chatbot scenario & conducted side by side tests with the Llama 3 8B model.
- Released distributed fine-tuning stack with support for Dell PowerEdge XE9680 Servers equipped with AMD Instinct MI300X Accelerators and NVIDIA H100 Tensor Core GPUs to offer enterprise choice.

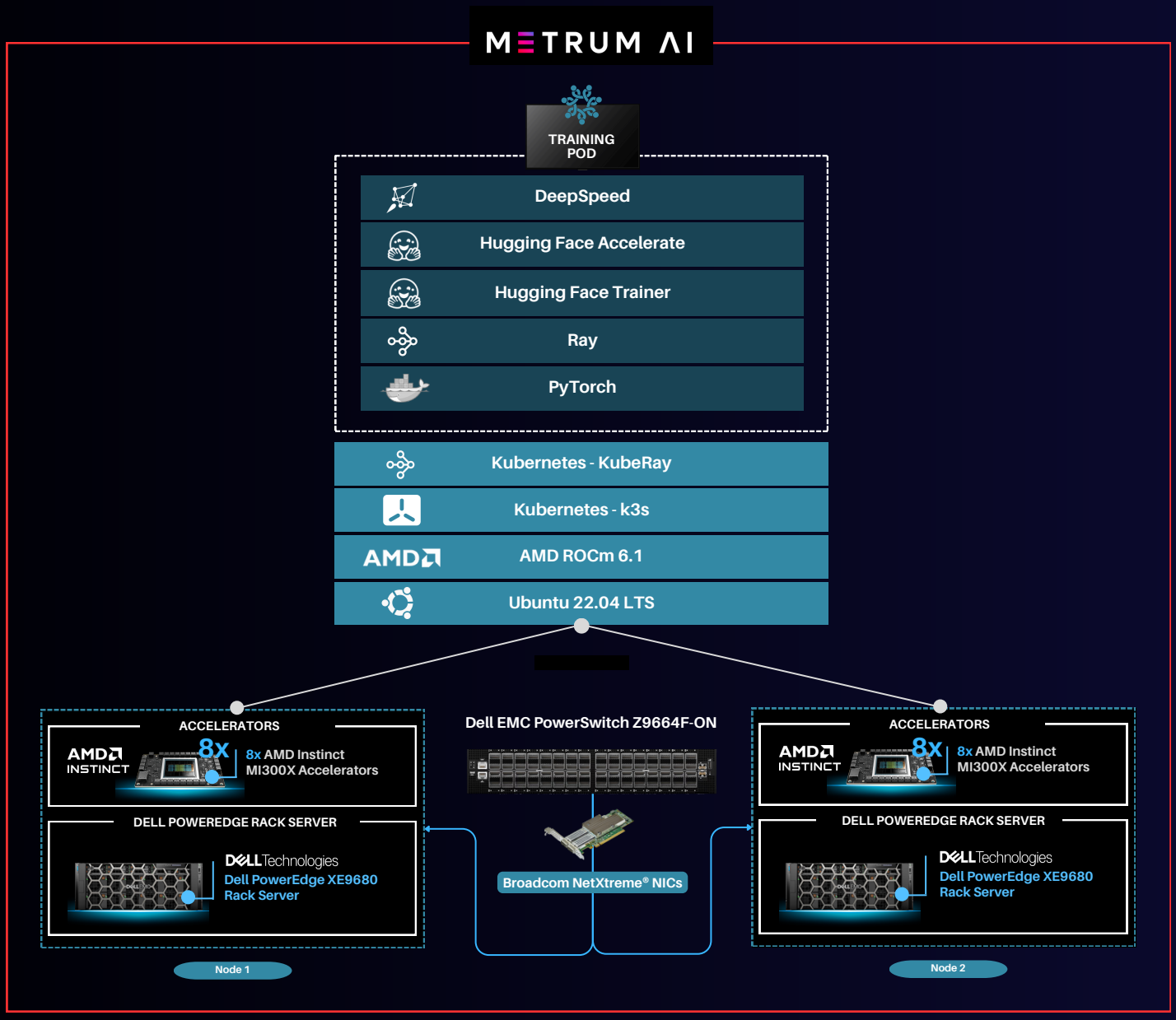
The Software Stack

This solution stack leverages Dell PowerEdge Rack Servers, coupled with Broadcom Ethernet NICs for providing high-speed inter-node communications needed for distributed computing as well as Kubernetes for scaling. Each Dell PowerEdge server contains AI accelerators, specifically AMD Instinct Accelerators to enhance LLM fine-tuning.

The architecture diagram provided below illustrates the configuration of two Dell PowerEdge XE9680 servers with eight AMD Instinct MI300X accelerators each.

Leveraging Dell PowerEdge, Dell PowerSwitch, and high-speed Broadcom Ethernet Network adaptors, the software platform integrates Kubernetes (K3S), Ray, Hugging Face Accelerate, Microsoft DeepSpeed, with other AI libraries and drivers including AMD ROCm™ and PyTorch.

See diagram on next page.


METRUM AI
TRAINING POD
DeepSpeed
Hugging Face Accelerate
Hugging Face Trainer
Ray
PyTorch
Kubernetes - KubeRay
Kubernetes - k3s
AMD ROCm 6.1
Ubuntu 22.04 LTS
ACCELERATORS
AMD INSTINCT
8x
8x AMD Instinct MI300X Accelerators
DELL POWEREDGE RACK SERVER
DELL Technologies
Dell PowerEdge XE9680 Rack Server
Node 1
Dell EMC PowerSwitch Z9664F-ON
Broadcom NetXtreme® NICs
ACCELERATORS
AMD INSTINCT
8x
8x AMD Instinct MI300X Accelerators
DELL POWEREDGE RACK SERVER
DELL Technologies
Dell PowerEdge XE9680 Rack Server
Node 2

Step-by-Step Guide

Step 1. Set up the distributed cluster.

Follow the k3s setup and introduce additional parameters for the k3s installation script. This involves configuring flannel, the networking fabric for kubernetes, with a user-selected specified network interface and utilizing the "host-gw" backend for networking. Then, Helm, the package manager for Kubernetes, will be used, and AMD plugins will be incorporated to grant access to AMD Instinct MI300X GPUs for the cluster pods.

Step 2. Install KubeRay and configure Ray Cluster.

The next steps include installing Kuberay, a Kubernetes operator, using Helm. The core of KubeRay comprises three Kubernetes Custom Resource Definitions (CRDs):

- **RayCluster**: This CRD enables KubeRay to fully manage the lifecycle of a RayCluster, automating tasks such as cluster creation, deletion, and autoscaling, while ensuring fault tolerance.
- **RayJob**: KubeRay streamlines job submission by automatically creating a RayCluster when needed. Users can configure RayJob to initiate job deletion once the task is completed, enhancing operational efficiency.
- **RayService**: RayService is made up of two parts: a RayCluster and a Ray Serve deployment graph. RayService offers zero-downtime upgrades for RayCluster and high availability.

```
helm repo add kuberay https://ray-project.github.io/kuberay-helm/
helm install kuberay-operator kuberay/kuberay-operator --version 1.0.0
```

This RayCluster consists of a head node followed by 1 worker node. In a YAML file, the head node is configured to run Ray with specified parameters, including the dashboard host and the number of GPUs, as shown in the excerpt below. Here, the worker node is under the name "gpu-group".

```
  ...
 headGroupSpec:
  rayStartParams:
   dashboard-host: "0.0.0.0"
   # setting num-gpus on the rayStartParams enables
   # head node to be used as a worker node
   num-gpus: "8"
  ...
```

The Kubernetes service is also defined to expose the Ray dashboard port for the head node. The deployment of the Ray cluster, as defined in a YAML file, will be executed using kubectl.

```
kubectl apply -f cluster.yml
```

Step 3. Fine-tune Llama 3 8B Model with BF16 Precision.
You can either create your own dataset or select one from Hugging Face. The dataset must be available as a single json file with the specified format below.

{"question":"Is pentraxin 3 reduced in bipolar disorder?", "context":"Immunologic abnormalities have been found in bipolar disorder but pentraxin 3, a marker of innate immunity, has not been studied in this population.", "answer":"Individuals with bipolar disorder have low levels of pentraxin 3 which may reflect impaired innate immunity."}

Jobs will be submitted to the Ray Cluster through the Ray Python SDK utilizing the Python script, job.py, provided below.

```
# job.py

from ray.job_submission import JobSubmissionClient

# Update the <Head Node IP> to your head node IP/Hostname
client = JobSubmissionClient("http://<Head Node IP>:30265")

fine_tuning = (
    "python3 create_dataset.py \
     --dataset_path /train/dataset.json \
     --prompt_type 5 \
     --test_split 0.2 ;"
    "python3 train.py \
     --num-devices 16 \ # Number of GPUs available
     --batch-size-per-device 12 \
     --model-name meta-llama/Meta-Llama-3-8B-Instruct \ # model name
     --output-dir /train/ \
     --hf-token <HuggingFace Token> "
)
submission_id = client.submit_job(entrypoint=fine_tuning,)

print("Use the following command to follow this Job's logs:")
print(f"ray job logs '{submission_id}' --address http://<Head Node
IP>:30265 --follow")
```

This script initializes the JobSubmissionClient with the head node IP address, and sets parameters such as prompt_type, which determines how each question-answer datapoint is formatted when inputted into the model, as well as batch size and number of devices for training. It then submits the job with these set parameter definitions.

The initial phase involves generating a fine-tuning dataset, which will be stored in a specified format. Configurations such as the prompt used and the ratio of training to testing data can be added. During the second phase, we will proceed with fine-tuning the model. For this fine-tuning, configurations such as the number of GPUs to be utilized, batch size for each GPU, the model name as available on Hugging Face, Hugging Face API Token, and the number of epochs to fine-tune can all be specified.

Finally, in the third phase, we can start fine-tuning the model.

```
python3 job.py
```

The fine-tuning jobs can be monitored using Ray CLI and Ray Dashboard.

- Using Ray CLI:
  - Retrieve submission ID for the desired job.
  - Use the command below to track job logs.

```
ray job logs <Submission ID> --address http://<Head Node IP>:30265 --follow
```

Ensure to replace <Submission ID> and <Head Node IP> with the appropriate values.

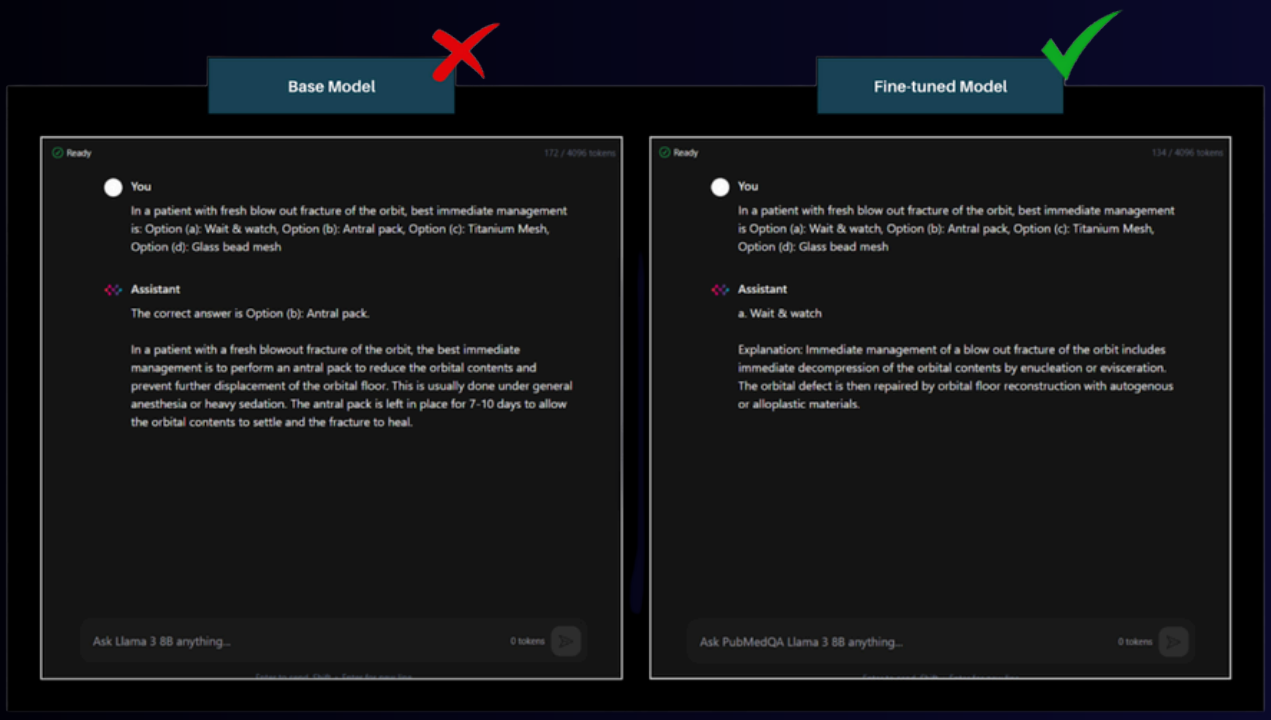
- Using Ray Dashboard:
  - To check the status of fine-tuning jobs, simply visit the Jobs page on your Ray Dashboard at <Head Node IP>:30265 and select the specific job from the list.

The reference code for this solution can be found here.

Industry Specific Medical Use Case

Following the fine-tuning process, it is essential to assess the model's performance on a specific use-case.

This solution uses the PubMedQA medical dataset to fine-tune a Llama 3 8B model on BF16 precision for our evaluation. The process was conducted on a distributed setup, utilizing a batch size of 12 per device, with training performed over 25 epochs. Both the base model and fine-tuned model are deployed in the Metrum AI enterprise chatbot to compare performance. The example below prompts the chatbot with a question from the MedMCQA dataset available on Hugging Face, for which the correct answer is "a."



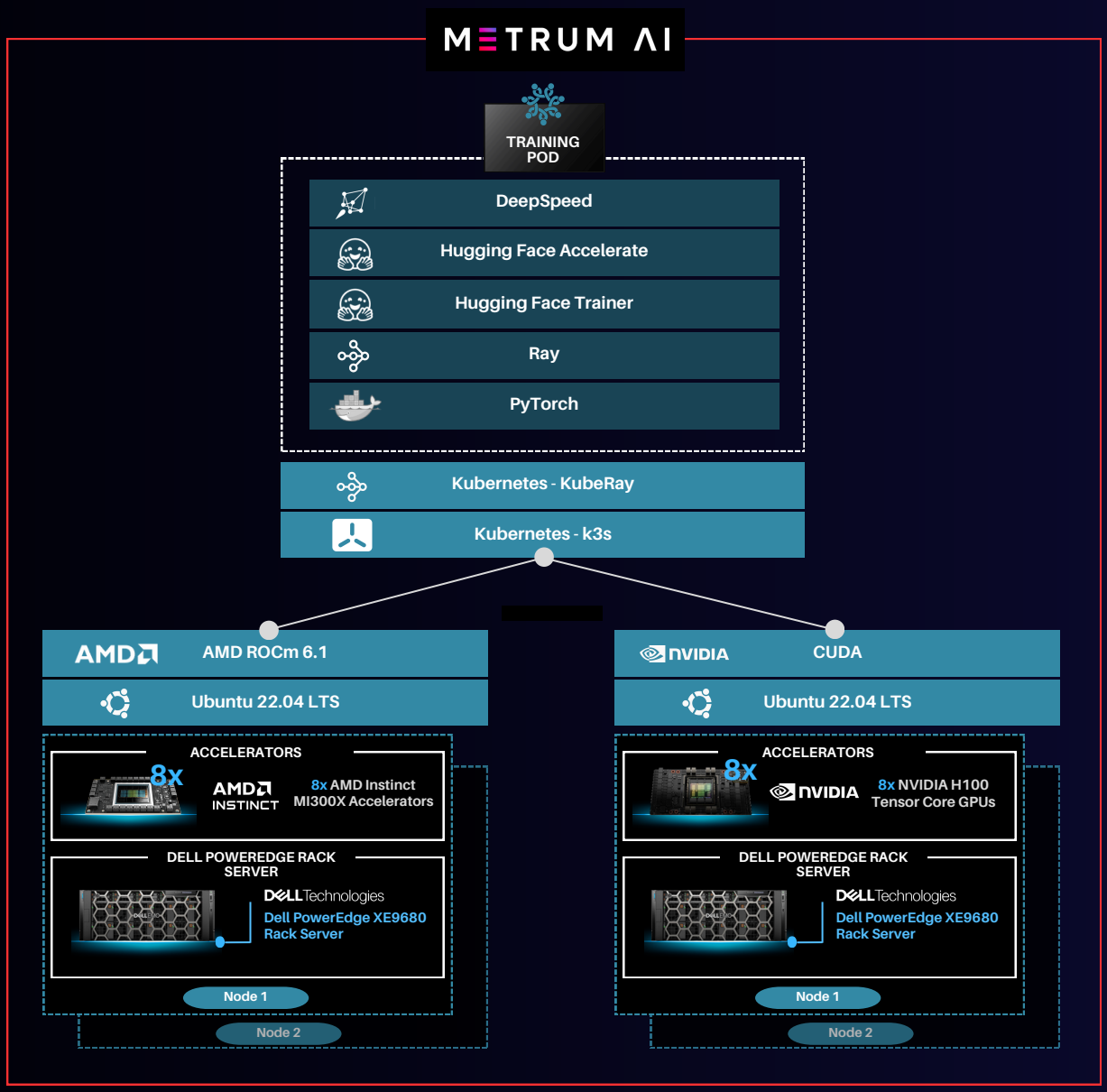
As shown on the left, the response generated by the base Llama 3 8B model is unstructured and vague, and returns an incorrect answer. On the other hand, the fine-tuned model returns the correct answer and also generates a thorough and detailed response to the instruction while demonstrating an understanding of the specific subject matter, in this case medical knowledge, relevant to the instruction.

Enterprise Choice in Industry Leading Accelerators

To deliver enterprise choice, this distributed fine-tuning software stack supports both AMD Instinct MI300X Accelerators as well as NVIDIA H100 Tensor Core GPUs. Below, we show a visualization of the unified software and hardware stacks, running seamlessly with the Dell PowerEdge XE9680 Server.

"Metrum AI is thrilled to offer choice in distributed fine-tuning across both leading AI GPUs in the industry on the flagship PowerEdge XE9680."

- Steen Graham, CEO at Metrum AI



Summary

Dell PowerEdge XE9680 Server, featuring AMD Instinct MI300X Accelerators, provides enterprises with cutting-edge infrastructure for creating industry-specific AI solutions using their own proprietary data. In this blog, we showcased how enterprises deploying applied AI can take advantage of this unified AI ecosystem by delivering the following critical solutions:

- Developed a distributed finetuning software stack on the flagship Dell PowerEdge XE9680 Server equipped with eight AMD Instinct MI300X Accelerators.
- Fine-tuned Llama 3 8B with BF16 precision using the PubMedQA medical dataset on two Dell PowerEdge XE9680 Servers each equipped with eight AMD Instinct MI300X Accelerators.
- Deployed fine-tuned model in an enterprise chatbot scenario & conducted side by side tests with Llama 3 8B.
- Released distributed fine-tuning stack with support for Dell PowerEdge XE9680 Servers equipped with AMD Instinct MI300X Accelerators and NVIDIA H100 Tensor Core GPUs to offer enterprise choice.

Metrum AI is excited to see continued advancements from Dell and AMD on hardware and software optimizations in the future, including an upcoming RAG (retrieval augmented generation) offering running on the Dell PowerEdge XE9680 Server with AMD Instinct MI300X Accelerators at Dell Tech World '24.

References

- Llama 3 Meta AI Blog
- PubMedQA

AMD images: AMD Library, https://library.amd.com/account/dashboard/
NVIDIA images: Nvidia.com
Dell images: Dell.com

***DISCLAIMER - Performance varies by hardware and software configurations, including testing conditions, system settings, application complexity, the quantity of data, batch sizes, software versions, libraries used, and other factors. The results of performance testing provided are intended for informational purposes only and should not be considered as a guarantee of actual performance.