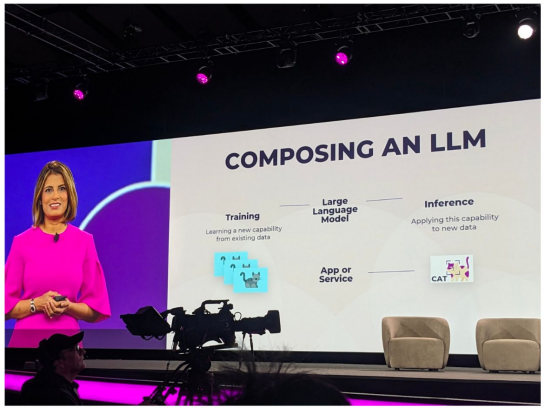
Finetuning LLama 2 LLM on GKE.

AI on Kubernetes	01
LLM's and fine-tuning	02
Fine-tuning code walk-through	03
Dynamic Workload Scheduling	04
Q&A	05

Trending Topics



- **AI**
- Platform Engineering
- Developer Experience
- Environmental Sustainability
- WebAssembly
- eBPF
- Security
- Observability
- Beginner-friendly content
- OSS Maintainership/Contribution

Gari's Hot Take

- The Kubernetes / CNCF community and ecosystem are thriving!
- Great mix of attendees
  - More end users than contributors / vendors
  - Great showing from university students
  - Many corporate users just getting started
- People are interested in AI / ML, but don't really understand it or know how they'll use it
- DevOps is dead; Long Live Platform Engineering!
- Don't forget about Developer Experience
- Increased focus on application development and integration

Why Kubernetes for AI?

Kubernetes is a battle-test foundation for your AI Platform

Portability

Write Once.
Run Everywhere.

Flexibility

Choose the right framework(s) for the job

Scalability and Performance

Fine tune performance and scale the platform.

Cost and Efficiency

Pay for what you need when you need it

Credits

Injae Kwak

Marcel Wysocki

https://github.com/saltysoup/ai-on-gke/tree/demo/tutorials/finetuning-llama-7b-on-l4

https://bit.ly/bos-k8s-meetup-llm

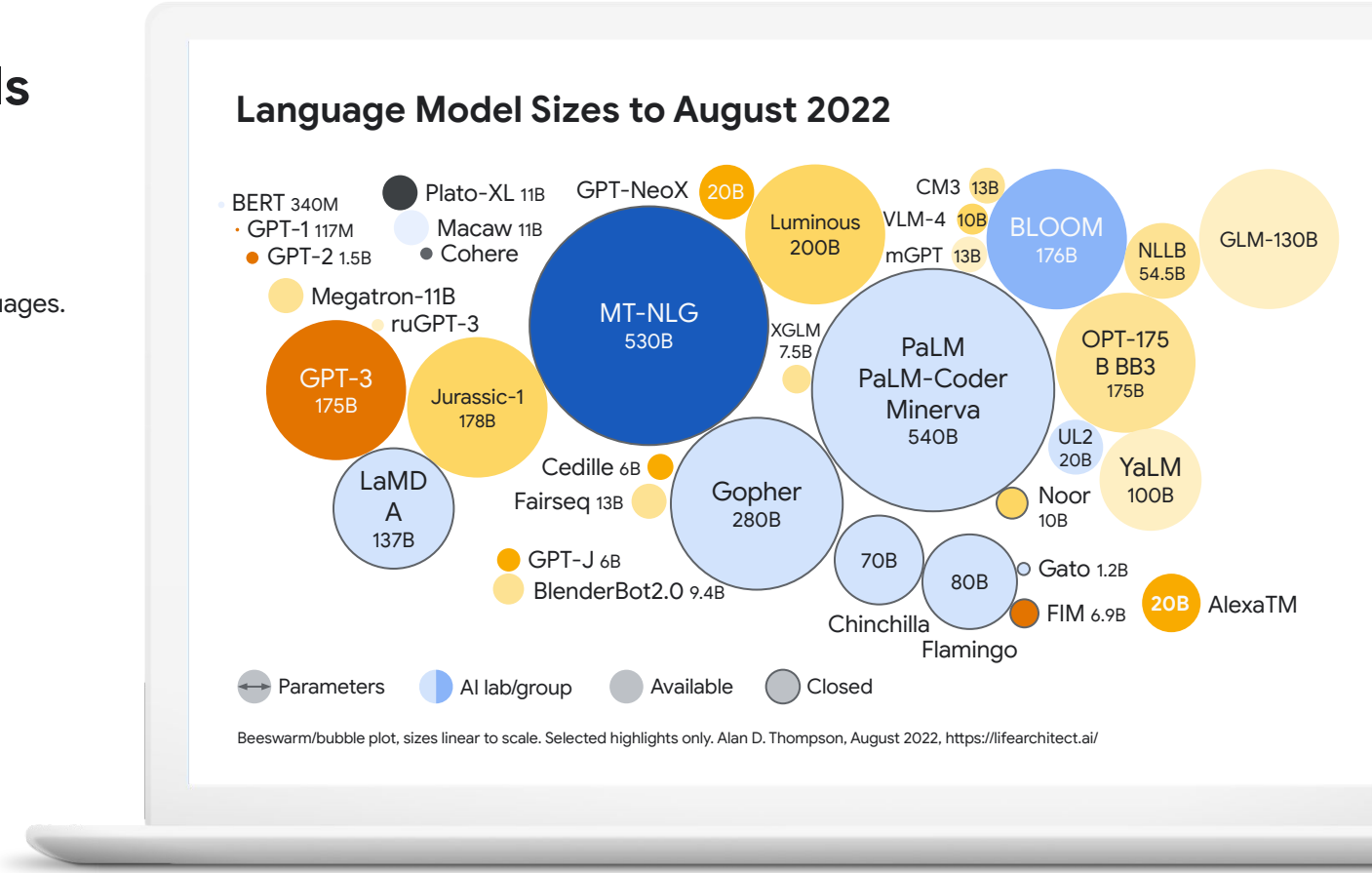
LLM’s & Finetuning

What are Large Language Models (LLMs) ?

ML algorithms that can recognize, predict, and generate human languages.

Pre-trained on petabyte scale text-based datasets resulting in large models with 10s to 100s of billions of parameters.

LLMs are normally pre-trained on a large corpus of text followed by fine-tuning on a specific task.



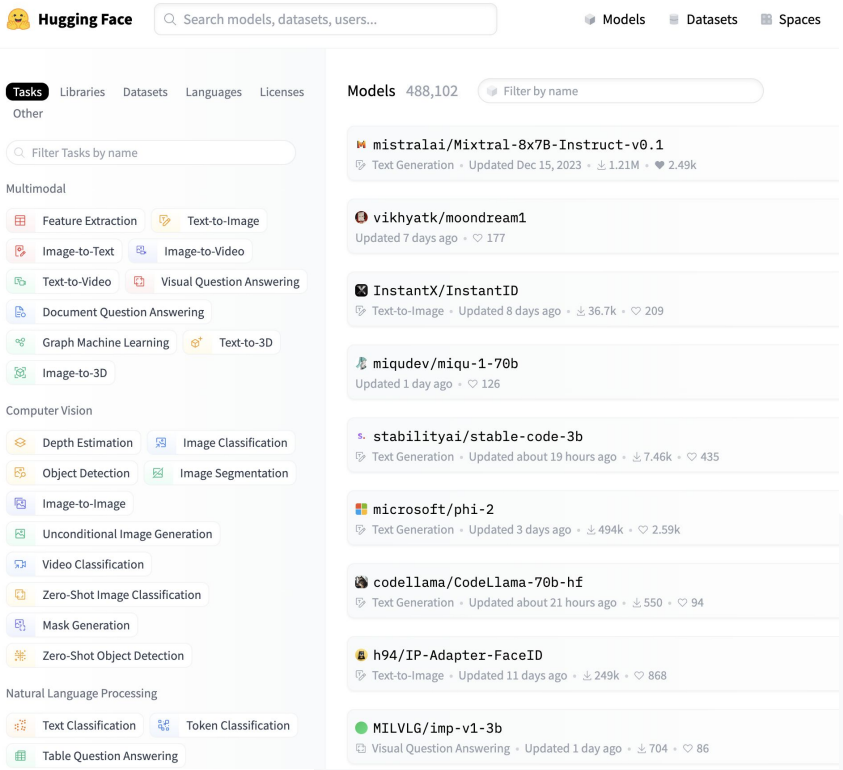
Open Source LLMs

Llama 2

Llama 2 is a collection of pretrained and fine-tuned generative text models ranging in scale from 7 billion to 70 billion parameters. Our fine-tuned LLMs, called Llama-2-Chat, are optimized for dialogue use cases. Llama-2-Chat models outperform open-source chat models on most benchmarks we tested, and in our human evaluations for helpfulness and safety, are on par with some popular closed-source models like ChatGPT and PaLM.

BLOOM
FLAN-T5
SDXL
Falcon 7B/40B
MPT
Llama V2
Code Llama
Mistral

...many more



Finetuning data

Dataset Viewer API Go to dataset viewer

Subset: all_years

Split: 1774

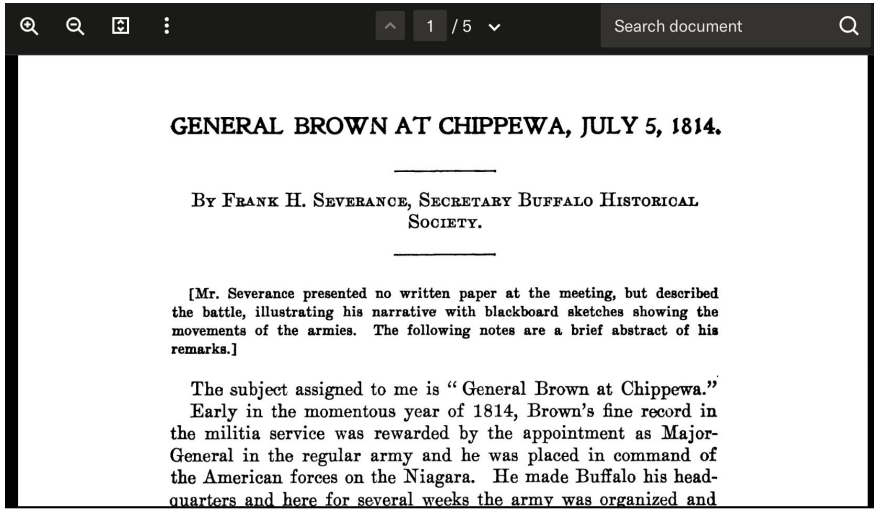
Search this dataset

▶ The dataset viewer is not available for this split.

Dataset Card for the American Stories dataset

Dataset Summary

The American Stories dataset is a collection of full article texts extracted from historical U.S. newspaper images. It includes nearly 20 million scans from the public domain Chronicling America collection maintained by the Library of Congress. The dataset is designed to address the challenges posed by complex layouts and low OCR quality in existing newspaper datasets. It was created using a novel deep learning pipeline that incorporates layout detection, legibility classification, custom OCR, and the association of article texts spanning multiple bounding boxes. It employs efficient architectures specifically designed for mobile phones to ensure high scalability. The dataset offers high-quality data that can be utilized for various purposes. It can be used to pre-train large language models and improve their understanding of historical English and world knowledge. The dataset can also be integrated into retrieval-augmented language models, making historical information more accessible, including interpretations of political events and details about people's ancestors. Additionally, the structured article texts in the dataset enable the use of transformer-based methods for applications such as detecting reproduced content. This significantly enhances accuracy compared to relying solely on existing OCR techniques. The American Stories dataset serves as an invaluable resource for developing multimodal layout analysis models and other multimodal applications. Its vast size and silver quality make it ideal for innovation and research in this domain.



Finetune.py - loading the finetuning data

```
from pathlib import Path
from datasets import load_dataset, concatenate_datasets
from transformers import AutoTokenizer, AutoModelForCausalLM, Trainer, TrainingArguments, DataCollatorForLanguageModeling
from peft import get_peft_model, LoraConfig, prepare_model_for_kbit_training
import torch

# /gcs-mount will mount the GCS bucket created earlier
model_path = "/gcs-mount/llama2-7b"
finetuned_model_path = "/gcs-mount/llama2-7b-american-stories"

tokenizer = AutoTokenizer.from_pretrained(model_path, local_files_only=True)
model = AutoModelForCausalLM.from_pretrained(
        model_path, torch_dtype=torch.float16, device_map="auto", trust_remote_code=True)

dataset = load_dataset("dell-research-harvard/AmericanStories",
   "subset_years",
   year_list=["1809", "1810", "1811", "1812", "1813", "1814", "1815"]
)
dataset = concatenate_datasets(dataset.values())

data = dataset.map(lambda x: tokenizer(
   x["article"], padding='max_length', truncation=True))
```

Finetune.py - loading the model and preparing for finetuning

```
from pathlib import Path
from datasets import load_dataset, concatenate_datasets
from transformers import AutoTokenizer, AutoModelForCausalLM, Trainer, TrainingArguments, DataCollatorForLanguageModeling
from peft import get_peft_model, LoraConfig, prepare_model_for_kbit_training
import torch

# /gcs-mount will mount the GCS bucket created earlier
model_path = "/gcs-mount/llama2-7b"
finetuned_model_path = "/gcs-mount/llama2-7b-american-stories"

tokenizer = AutoTokenizer.from_pretrained(model_path, local_files_only=True)
model = AutoModelForCausalLM.from_pretrained(
        model_path, torch_dtype=torch.float16, device_map="auto", trust_remote_code=True)

dataset = load_dataset("dell-research-harvard/AmericanStories",
    "subset_years",
    year_list=["1809", "1810", "1811", "1812", "1813", "1814", "1815"]
)
dataset = concatenate_datasets(dataset.values())

data = dataset.map(lambda x: tokenizer(
    x["article"], padding='max_length', truncation=True))
```

Finetune.py - training the model.

```
# add LoRA adaptor
model = get_peft_model(model, lora_config)
model.print_trainable_parameters()

training_args = TrainingArguments(
        per_device_train_batch_size=1,
        gradient_accumulation_steps=4,
        warmup_steps=2,
        num_train_epochs=1,
        learning_rate=2e-4,
        fp16=True,
        logging_steps=1,
        output_dir=finetuned_model_path,
        optim="paged_adamw_32bit",
)

trainer = Trainer(
    model=model,
    train_dataset=data,
    args=training_args,
    data_collator=DataCollatorForLanguageModeling(tokenizer, mlm=False),
)
model.config.use_cache = False  # silence the warnings. Please re-enable for inference!

trainer.train()
```

Finetune.py - saving the finetuned model

```
# Merge the fine tuned layer with the base model and save it
# you can remove the line below if you only want to store the LoRA layer
model = model.merge_and_unload()

model.save_pretrained(finetuned_model_path)
tokenizer.save_pretrained(finetuned_model_path)
```

Finetune.py - sample prompt

```
# Beginning of story in the dataset
prompt = """
In the late action between Generals

Brown and Riall, it appears our men fought
with a courage and perseverance, that would
"""
input_ids = tokenizer(prompt, return_tensors="pt").input_ids

gen_tokens = model.generate(
    input_ids,
    do_sample=True,
    temperature=0.8,
    max_length=100,
)
print(tokenizer.batch_decode(gen_tokens)[0])
```

Finetuned response...

<s>
In the late action between Generals
Brown and Riall, it appears our men fought
with a courage and perseverance, that would
do credit to the best troops in Europe. But there is no use in gloriously stating facts. The whole of our army was defeated, and we are now in the hands of the enemy. If this war is to be carried on, we must endeavour to make peace.

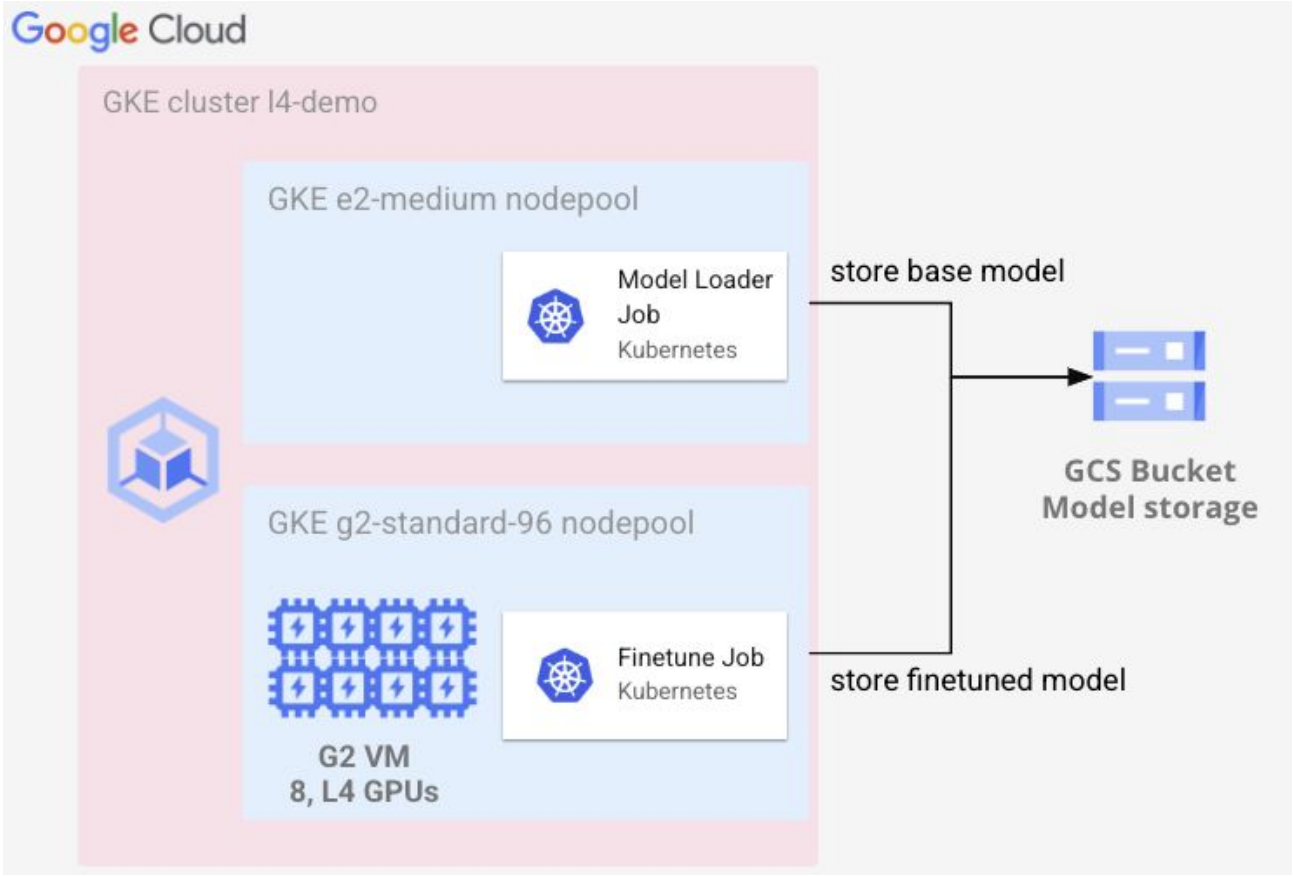
Llama2 7b chat response ...

In the late action between Generals
Brown and Riall, it appears our men fought
with a courage and perseverance, that would

be an interesting sentence to continue. Here are a few options:

- ...become legendary in military history.
- ...inspire their comrades and turn the tide of battle.
- ...earn them the respect and admiration of their enemies.
- ...prove crucial in securing a decisive victory for their side.
- ...leave lasting marks on the landscape, as they fought to hold their ground against overwhelming odds.

Kubernetes Jobs



Create the gpu enabled nodepool

```
gcloud container node-pools create g2-standard-96 --cluster l4-demo \
  --accelerator type=nvidia-l4,count=8,gpu-driver-version=latest \
  --machine-type g2-standard-96 \
  --ephemeral-storage-local-ssd=count=8 \
  --enable-autoscaling --enable-image-streaming \
  --num-nodes=0 --min-nodes=0 --max-nodes=3 \
  --shielded-secure-boot \
  --shielded-integrity-monitoring \
  --node-locations ${REGION}-a,${REGION}-b --region ${REGION}
```

Google Cloud VM g2-standard-96

Technical facts about the Google Compute Engine machine type g2-standard-96.

Series	G2
Family	Accelerator-optimized
vCPU	96
Memory	384 GB
CPU Manufactur	Intel
CPU Platform	Intel Cascade Lake
CPU Base Frequency	2.2 GHz
CPU Turbo Frequency	2.9 GHz
CPU Max. Turbo Frequency	3.7 GHz
Accelerator (GPUs)	8
Accelerator Type	NVIDIA L4

Price per Hour

	Min.	Avg.	Max.
Standard price per hour	$7.9771	$8.9141	$10.2779
Spot provisioning model (Spot VM)	$2.4925	$2.8942	$3.3968
Discount Spot VM vs. standard	$5.059 (63%)	$6.0199 (67%)	$7.2249 (73%)

Price per Month

	Min.	Avg.	Max.
Price per month	$5823.3	$6507.32	$7502.85
Spot provisioning model (Spot VM)	$1819.54	$2112.8	$2479.64

Download Model job


```
apiVersion: batch/v1
kind: Job
metadata:
  name: model-loader
spec:
  template:
    metadata:
    spec:
      restartPolicy: OnFailure
      containers:
      - name: loader
        image: python:3.11
        command:
        - /bin/bash
        - -c
        - |
          pip install huggingface_hub
          mkdir -p /gcs-mount/llama2-7b
          python3 - << EOF
          from huggingface_hub import snapshot_download
          model_id="meta-llama/Llama-2-7b-hf"
          snapshot_download(repo_id=model_id,
local_dir="/gcs-mount/llama2-7b",
                            local_dir_use_symlinks=False, revision="main",
                            ignore_patterns=["*.safetensors",
"model.safetensors.index.json"])
          EOF
        imagePullPolicy: IfNotPresent
        env:
        - name: HUGGING_FACE_HUB_TOKEN
          valueFrom:
            secretKeyRef:
              name: l4-demo
              key: HF_TOKEN
        volumeMounts:
        - name: gcs-fuse-csi-ephemeral
          mountPath: /gcs-mount
      serviceAccountName: l4-demo
      volumes:
      - name: gcs-fuse-csi-ephemeral
        csi:
          driver: gcsfuse.csi.storage.gke.io
          volumeAttributes:
            bucketName: ${BUCKET_NAME}
            mountOptions: "implicit-dirs"
```

Finetune job

```
apiVersion: batch/v1
kind: Job
metadata:
  name: finetune-job
  namespace: default
spec:
  backoffLimit: 2
  template:
    metadata:
      annotations:
        kubectl.kubernetes.io/default-container: finetuner
        gke-gcsfuse/volumes: "true"
        gke-gcsfuse/memory-limit: 400Mi
        gke-gcsfuse/ephemeral-storage-limit: 30Gi
    spec:
      terminationGracePeriodSeconds: 60
      containers:
      - name: finetuner
        image:
us-docker.pkg.dev/google-samples/containers/gke/llama-7b-fine-tune-example
        resources:
          limits:
            nvidia.com/gpu: 8
        volumeMounts:
        - name: gcs-fuse-csi-ephemeral
          mountPath: /gcs-mount
      serviceAccountName: l4-demo
      volumes:
      - name: gcs-fuse-csi-ephemeral
        csi:
          driver: gcsfuse.csi.storage.gke.io
          volumeAttributes:
            bucketName: $BUCKET_NAME
            mountOptions: "implicit-dirs"
      nodeSelector:
        cloud.google.com/gke-accelerator: nvidia-l4
      restartPolicy: OnFailure
```

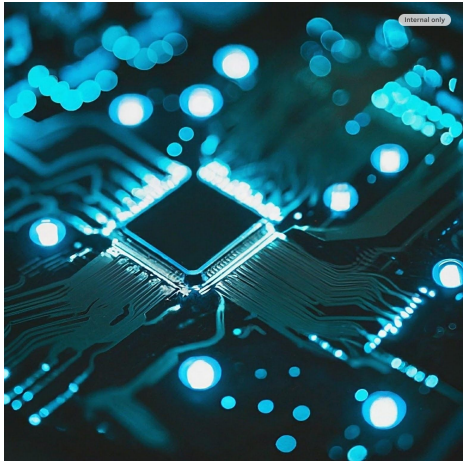
Review Logs

Dynamic Workload Scheduling (DWS)

Accelerators and scale-out workloads are difficult to provision

Common challenges today

- Popular accelerators like A100, H100, or TPUs
- Distributed training across a large number of nodes
- No preemptions during training and fine tuning runs
- Reliable capacity for experimenting with a new model



Dynamic Workload Scheduler (DWS)

New obtainability capabilities for accelerators

Works across GCP

- Managed Instance Groups on GCE
- Batch on GCE
- GKE
- Vertex AI

Calendar Mode:
Job start times assurance with Future Reservations

Use Cases:
(re)training, recurring fine-tuning

GPUs

Flex Start Mode:
Optimized economics and higher obtainability for on-demand resources
(uses dedicated capacity pool)

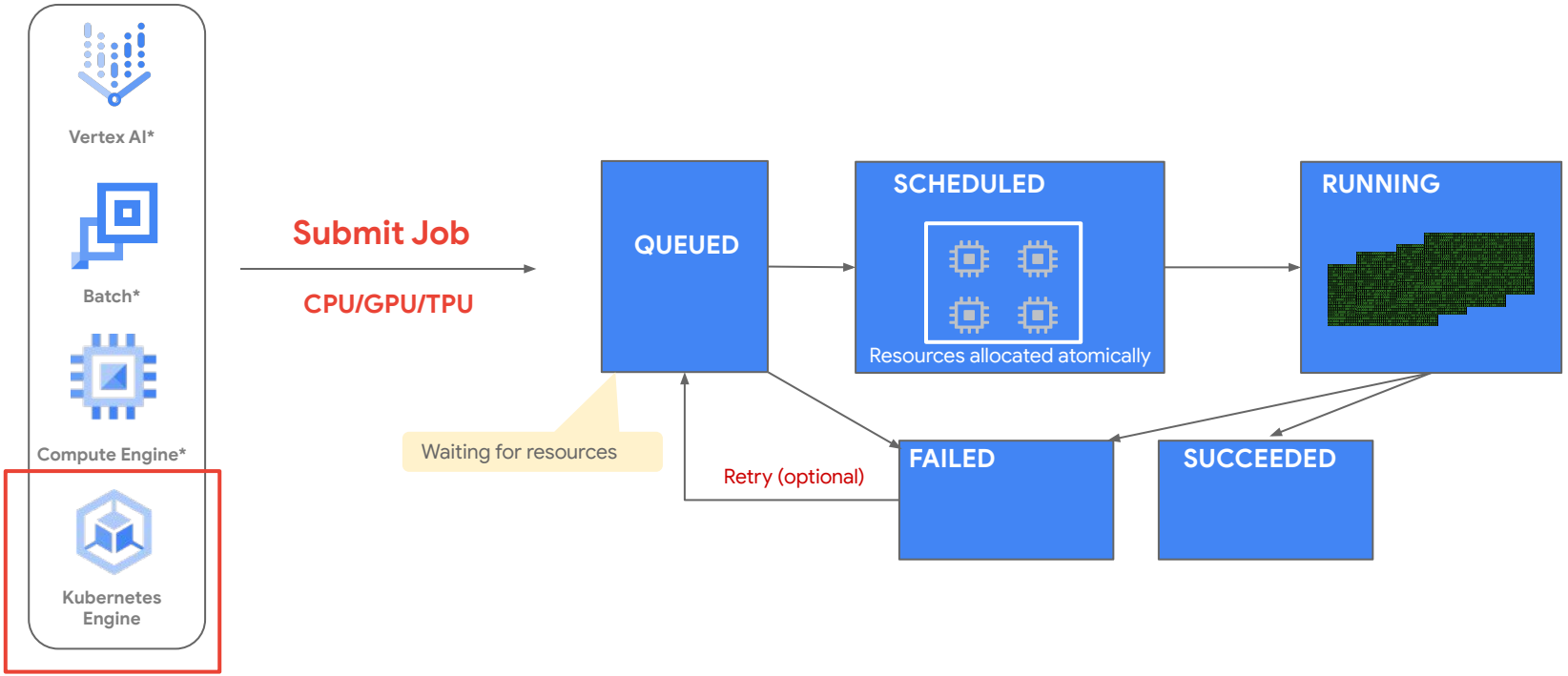
Use Cases:
time flexible experiments, fine tuning, batch inference

GPUs & TPUs

"The new DWS scheduling capabilities have been a game-changer in procuring sufficient GPU capacity for our training runs. We didn't have to worry about wasting money on idle GPUs while refreshing the page hoping for sufficient compute resources to become available."

- Sahil Chopra, Co-Founder & CEO, Linum AI

Lifecycle of a Batch job through DWS



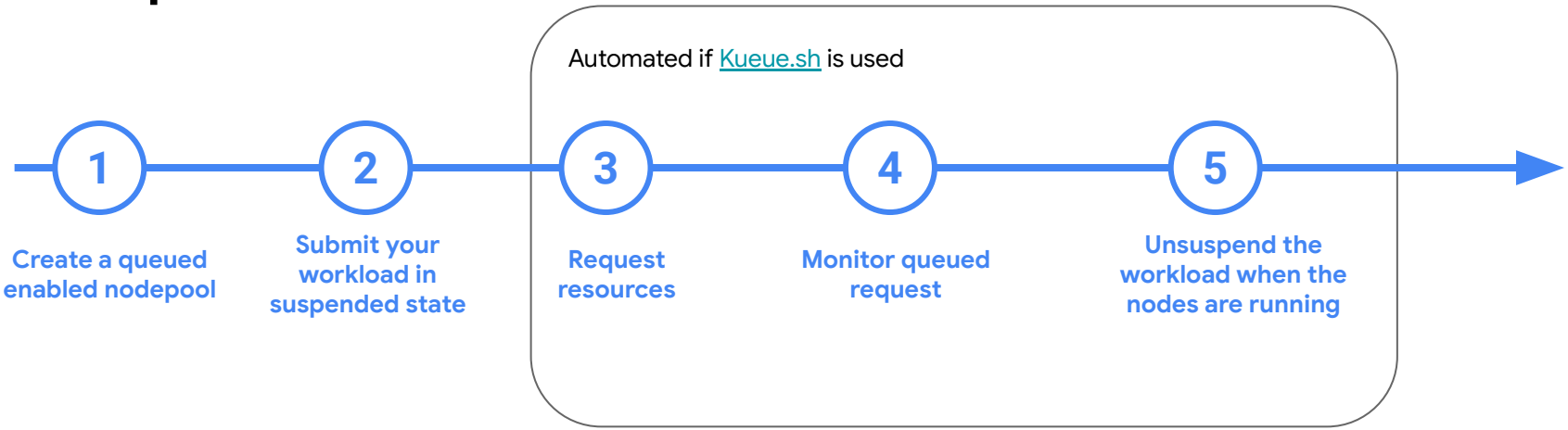
** preview access in Jan '24*

Similar interface and job configs for all jobs

Support for multi-node parallel jobs

Job level observability

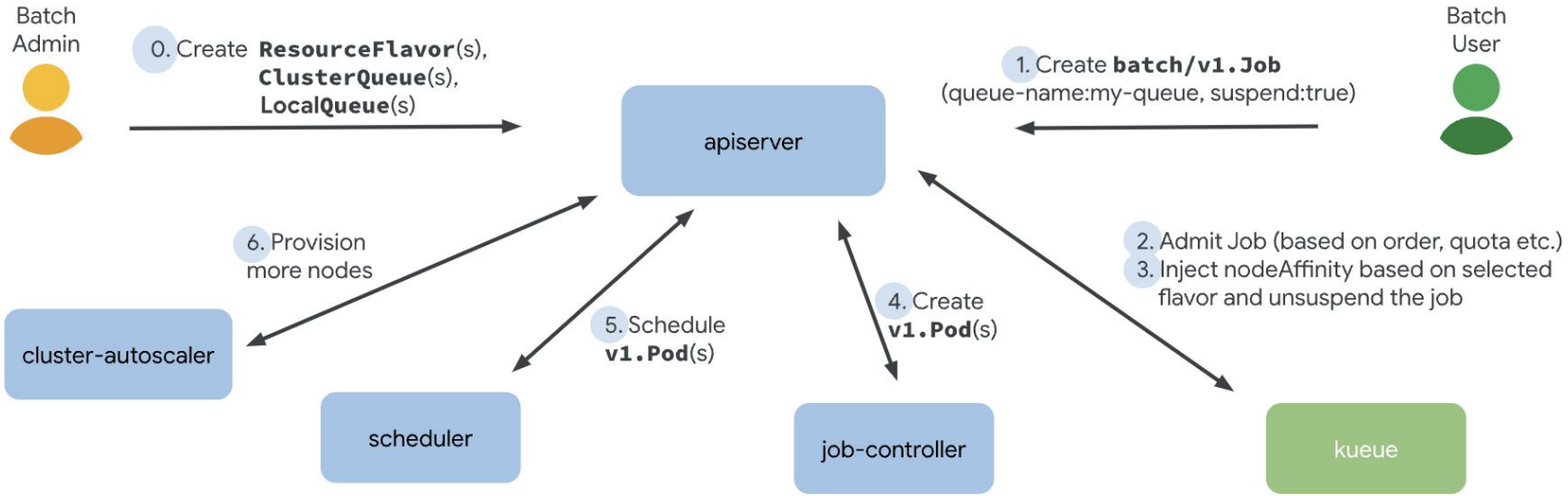
Flow simplification with Kueue



- ✓ Kueue automates steps in the flow
- ✓ Users need to install Kueue once, create a namespace queue (default comes with installation) and when creation jobs - make them suspended and add the label indicating the queue to be used.

Kueue (kueue.sigs.k8s.io)

Kueue is a kubernetes-native system that manages **quotas** and how jobs consume them. Kueue decides when a job should **wait**, when a job should be admitted to **start** (as in pods can be created) and when a job should be **preempted** (as in active pods should be deleted).



```
apiVersion:
kueue.x-k8s.io/v1beta1
kind: ResourceFlavor
metadata:
  name: "dws"
spec:
  Nodelabels:
cloud.google.com/gke-nodepool:
a3-highgpu-8g
```

```
apiVersion:
kueue.x-k8s.io/v1beta1
kind: ClusterQueue
metadata:
  name: "dws-cluster-queue"
spec:
  namespaceSelector: {}
  resourceGroups:
  - coveredResources: ["cpu",
"memory", "nvidia.com/gpu"]
    flavors:
    - name: "dws"
      resources:
      - name: "nvidia.com/gpu"
        nominalQuota: 10000  #
Infinite quota.
  admissionChecks:
  - dws-prov
```

```
apiVersion:
kueue.x-k8s.io/v1beta1
kind: LocalQueue
metadata:
  namespace: "default"
  name: "dws-local-queue"
spec:
  clusterQueue:
"dws-cluster-queue"
```

Create the gpu enabled nodepool

```
# H100-80gb configs
ACCELERATOR_ARG="type=nvidia-h100-80gb,count=8,gpu-driver-version=latest"
MACHINE_TYPE="a3-highgpu-8g"

#Deploy H100 Nodepool
gcloud beta container node-pools create dws-nodepool --zone ${ZONE} --cluster ${CLUSTER_NAME} --project ${PROJECT_ID}
--enable-autoupgrade     --enable-autorepair --accelerator ${ACCELERATOR_ARG} --machine-type ${MACHINE_TYPE}
--ephemeral-storage-local-ssd count=16 --enable-queued-provisioning  --location-policy=ANY --reservation-affinity=none
--enable-autoscaling --num-nodes=0 --total-max-nodes 1  --enable-gvnic --scopes "https://www.googleapis.com/auth/cloud-platform"
```

Google Cloud VM a2-highgpu-8g

Technical facts about the Google Compute Engine machine type a2-highgpu-8g.

Series	A2
Family	Accelerator-optimized
vCPU	96
Memory	680 GB
CPU Manufactur	Intel
CPU Platform	Intel Cascade Lake

Price per Month

	Min.	Avg.	Max.
Price per month	$21452.57	$22881.83	$25258.4
Spot provisioning model (Spot VM)	$7042	$8981.37	$10103.43
Discount Spot VM vs. month	$12871.54 (60%)	$13900.46 (61%)	$15154.97 (68%)

Finetune job (kueue)

```
apiVersion: batch/v1
kind: Job
metadata:
  name: finetune-llama2-13b
  namespace: default
  labels:
    kueue.x-k8s.io/queue-name: dws-local-queue
spec:
  backoffLimit: 2
  suspend: true
  template:
    metadata:
      annotations:
        kubectl.kubernetes.io/default-container: finetuner
        gke-gcsfuse/volumes: "true"
        gke-gcsfuse/memory-limit: 4Gi
        gke-gcsfuse/ephemeral-storage-limit: 30Gi
    spec:
      tolerations:
      - key: "nvidia.com/gpu"
        operator: "Exists"
        effect: "NoSchedule"
      containers:
      - name: finetuner
        Image:
us-central1-docker.pkg.dev/injae-sandbox-340804/llama2-dws-demo/llama2-dws-demo:v4
        env:
        - name: MODEL_NAME
          value: "llama2-13b"
        - name: PER_DEVICE_TRAIN_BATCH_SIZE
          value: "3"
        - name: GRADIENT_ACCUMULATION_STEPS
          value: "3"
        resources:
          requests:
            nvidia.com/gpu: 8
          limits:
            nvidia.com/gpu: 8
        volumeMounts:
        - name: gcs-fuse-csi-ephemeral
          mountPath: /gcs-mount
      serviceAccountName: workload-identity-k8-sa
      volumes:
      - name: gcs-fuse-csi-ephemeral
        csi:
          driver: gcsfuse.csi.storage.gke.io
          volumeAttributes:
            bucketName: "dws-demo"
            mountOptions: "implicit-dirs"
      restartPolicy: Never
      terminationGracePeriodSeconds: 60
```

Let's get started

Building your next-gen **AI Platform on Google Kubernetes Engine**

Learn

Learn at your own pace on how to build high performance AI Platforms and Apps with GCP

Docs

Doc links

Hands-on labs

Labs links

Engage

Engage with our account teams to assess and design your goals

Contact us

Talk to a Google Cloud Specialist

Build

Work with our professional services and partners to build your next big thing

Google Professional Services consulting

GCP Consulting

Partner programs

Find a partner

Docs: g.co/cloud/gke-aiml

Tutorials: github.com/GoogleCloudPlatform/ai-on-gke