KubeCon | CloudNativeCon

Europe 2024



# Strategies for Efficient LLM Deployments in Any Cluster

*Angel M De Miguel Meana, Broadcom*
*Francisco Cabrera, Microsoft*

# In this talk



## Goals

- Learn different strategies to run, operate and improve models (fine-tuning, quantization, etc.)
- Setting the appropriate infrastructure:  local container caching, OCI for models, GPU usage, scaling, etc.)
- Small-Medium private Kubernetes on prem deployment
- Deploy your model to on-prem Kubernetes

## Non-Goals

- Become LLM experts
- Non intended for AI service providers, (small clusters, no mega distributed clusters)

# Why should you consider deploying LLMs?

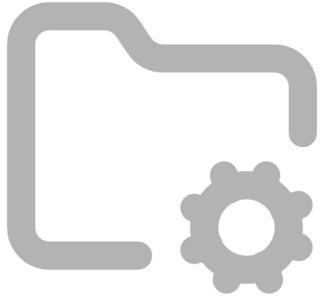




**Control and flexibility**



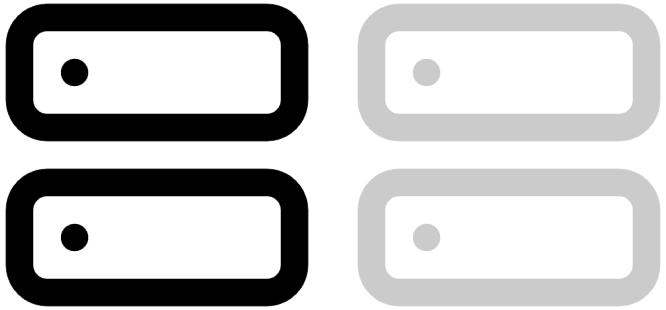
**Private data**



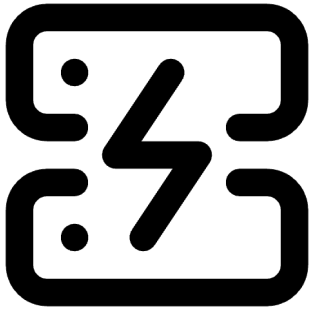
**Fine-tuning**

# What does efficiency mean?





Provision the minimal resources you need to complete a task in a secure and reliable way

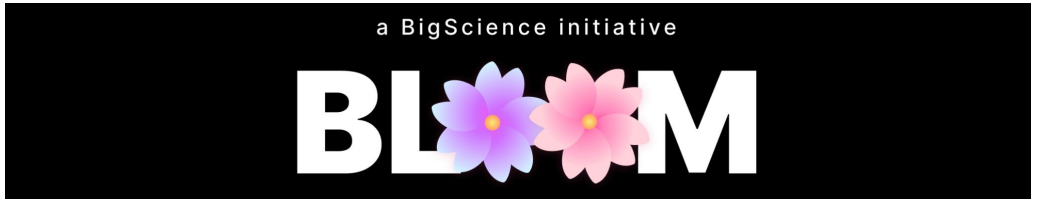


Ensure you use your allocated resources, so they are not idle and consuming energy.

# Energy Consumption

# Training BLOOM






"BLOOM is a **176 billion parameter** language model [...] trained on **1.6 terabytes of data**"

| | |
|---|---|
| Training time | 118 days, 5 hours, 41 min |
| Total energy | **433,196 kWh** |

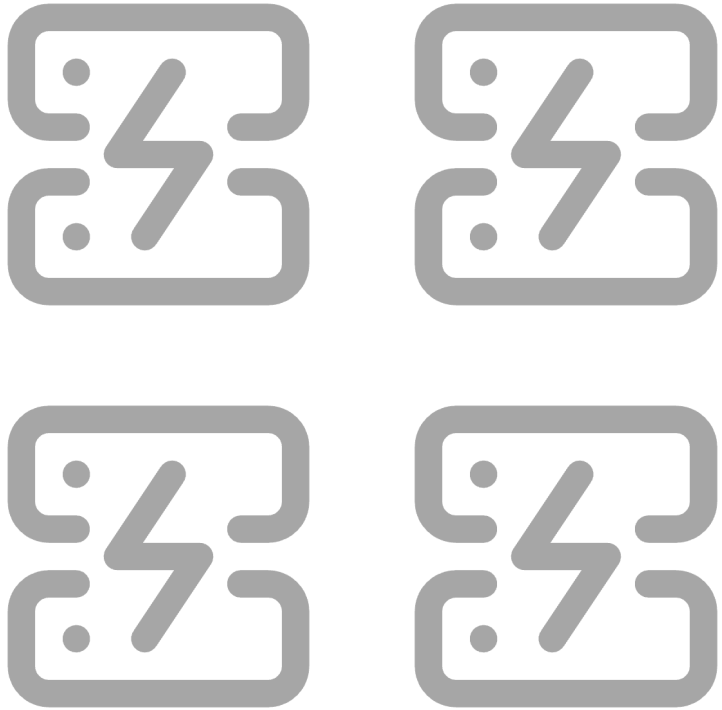
Same energy as 61 French homes in a year 🇫🇷

Luccioni, A.S., Viguier, S. and Ligozat, A.L., 2023. Estimating the carbon footprint of bloom, a 176b parameter language model. Journal of Machine Learning Research, 24(253), pp.1-15.
https://arxiv.org/abs/2211.02001

# Inference



**Optimize for energy efficiency at inference**, so your system require less resources and less energy.



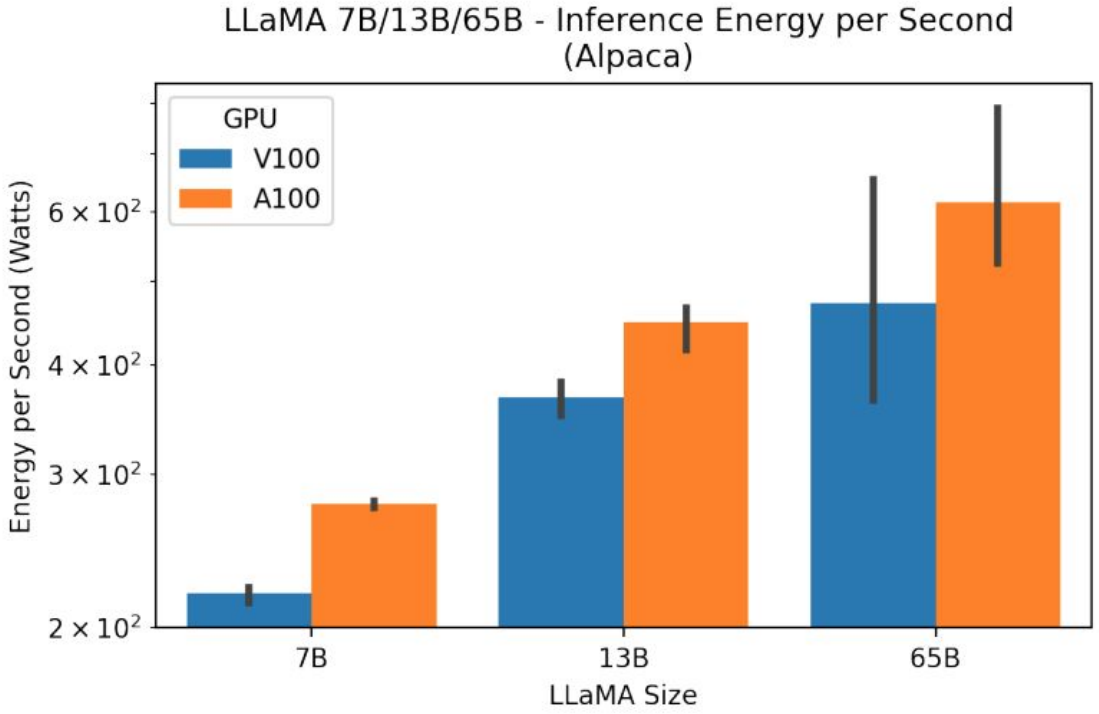
# Efficient LLM deployments

# Model selection



The model you select determines the hardware you need to run it at a scale and the energy it will consume.






Samsi, S., Zhao, D., McDonald, J., Li, B., Michaleas, A., Jones, M., Bergeron, W., Kepner, J., Tiwari, D. and Gadepally, V., 2023, September. From words to watts: Benchmarking the energy costs of large language model inference. In 2023 IEEE High Performance Extreme Computing Conference (HPEC) (pp. 1-9). IEEE.

https://arxiv.org/abs/2310.03003

# Quantization



Quantization techniques reduces the model sizes by reducing the precision.

Quantized models have lower hardware requirements.

| Model | Measure | F16 | Q4_0 | Q4_1 | Q5_0 | Q5_1 | Q8_0 |
|---|---|---|---|---|---|---|---|
| 7B | perplexity | 5.9066 | 6.1565 | 6.0912 | 5.9862 | 5.9481 | 5.9070 |
| 7B | file size | 13.0G | 3.5G | 3.9G | 4.3G | 4.7G | 6.7G |
| 7B | ms/tok @ 4th | 127 | 55 | 54 | 76 | 83 | 72 |
| 7B | ms/tok @ 8th | 122 | 43 | 45 | 52 | 56 | 67 |
| 7B | bits/weight | 16.0 | 4.5 | 5.0 | 5.5 | 6.0 | 8.5 |
| 13B | perplexity | 5.2543 | 5.3860 | 5.3608 | 5.2856 | 5.2706 | 5.2548 |
| 13B | file size | 25.0G | 6.8G | 7.6G | 8.3G | 9.1G | 13G |
| 13B | ms/tok @ 4th | - | 103 | 105 | 148 | 160 | 131 |
| 13B | ms/tok @ 8th | - | 73 | 82 | 98 | 105 | 128 |
| 13B | bits/weight | 16.0 | 4.5 | 5.0 | 5.5 | 6.0 | 8.5 |

https://github.com/ggerganov/llama.cpp
https://kaitchup.substack.com/p/gguf-quantization-for-fast-and-memory

# Quantization



"It shows that BitNet b1.58 starts to match **full precision** LLaMA LLM at 3B model size in terms of perplexity, while being **2.71 times faster** and using **3.55 times less GPU memory**"

| Models | Size | Memory (GB)↓ | Latency (ms)↓ | PPL↓ |
|---|---|---|---|---|
| LLaMA LLM | 700M | 2.08 (1.00x) | 1.18 (1.00x) | 12.33 |
| **BitNet b1.58** | 700M | 0.80 (2.60x) | 0.96 (1.23x) | 12.87 |
| LLaMA LLM | 1.3B | 3.34 (1.00x) | 1.62 (1.00x) | 11.25 |
| **BitNet b1.58** | 1.3B | 1.14 (2.93x) | 0.97 (1.67x) | 11.29 |
| LLaMA LLM | 3B | 7.89 (1.00x) | 5.07 (1.00x) | 10.04 |
| **BitNet b1.58** | 3B | **2.22 (3.55x)** | **1.87 (2.71x)** | **9.91** |
| **BitNet b1.58** | 3.9B | **2.38 (3.32x)** | **2.11 (2.40x)** | **9.62** |

Table 1: Perplexity as well as the cost of BitNet b1.58 and LLaMA LLM.

Ma, S., Wang, H., Ma, L., Wang, L., Wang, W., Huang, S., Dong, L., Wang, R., Xue, J. and Wei, F., 2024. The Era of 1-bit LLMs: All Large Language Models are in 1.58 Bits. arXiv preprint arXiv:2402.17764.
https://arxiv.org/abs/2402.17764

# Small Language Models (SMLs) and fine-tuning



**Clem Delangue** 🤗 • 3er+
Co-founder & CEO at Hugging Face
5 meses •

I've said it and will say it again:

#1 Smaller, cheaper, faster, more customized models will cover 99% of use-cases. You don't need a million dollar formula 1 to get to work everyday and you don't need a banking customer success chatbot to tell you the meaning of life!

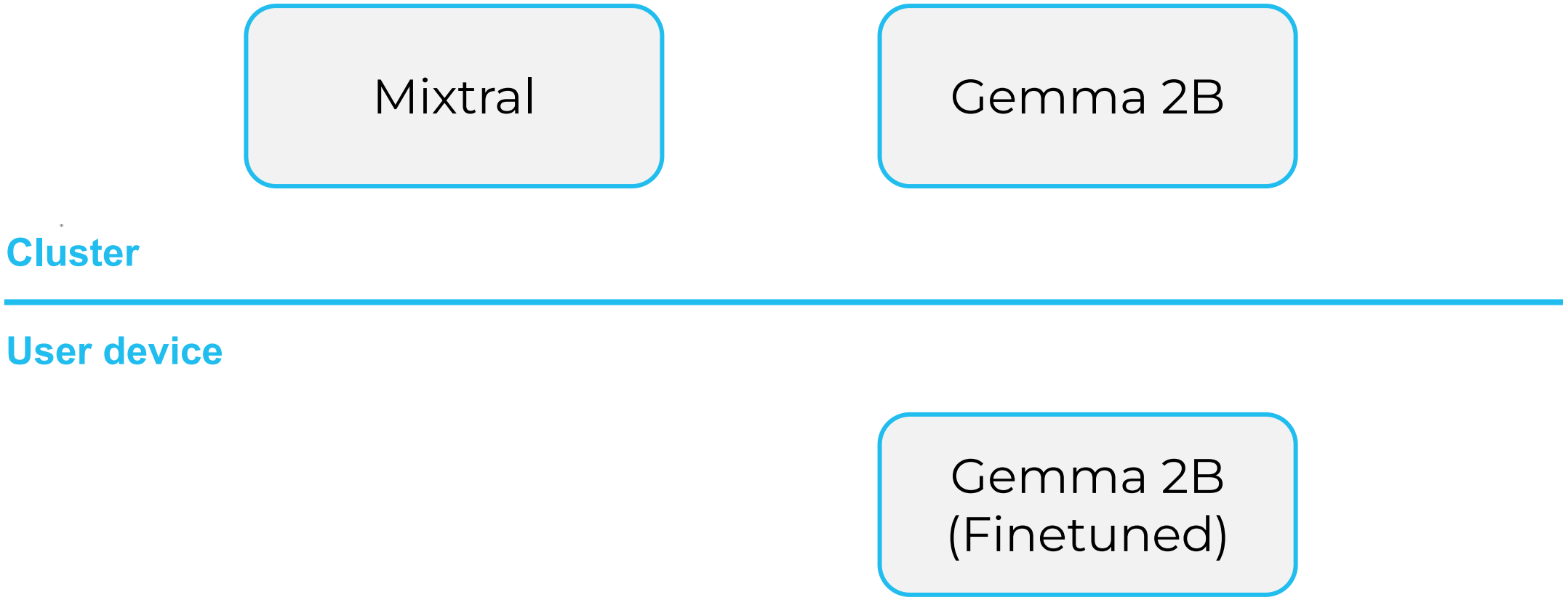
Phi2
(Microsoft)

Gemma 2B
(Google)

Qwen1.5
(Alibaba)

# WebAssembly





# All together on Kubernetes

# Define your solution architecture

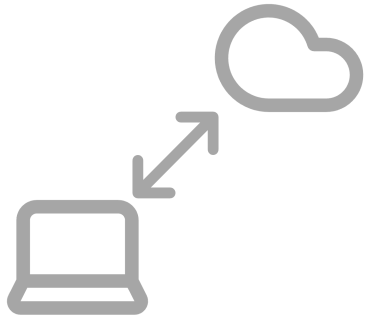




Lower costs



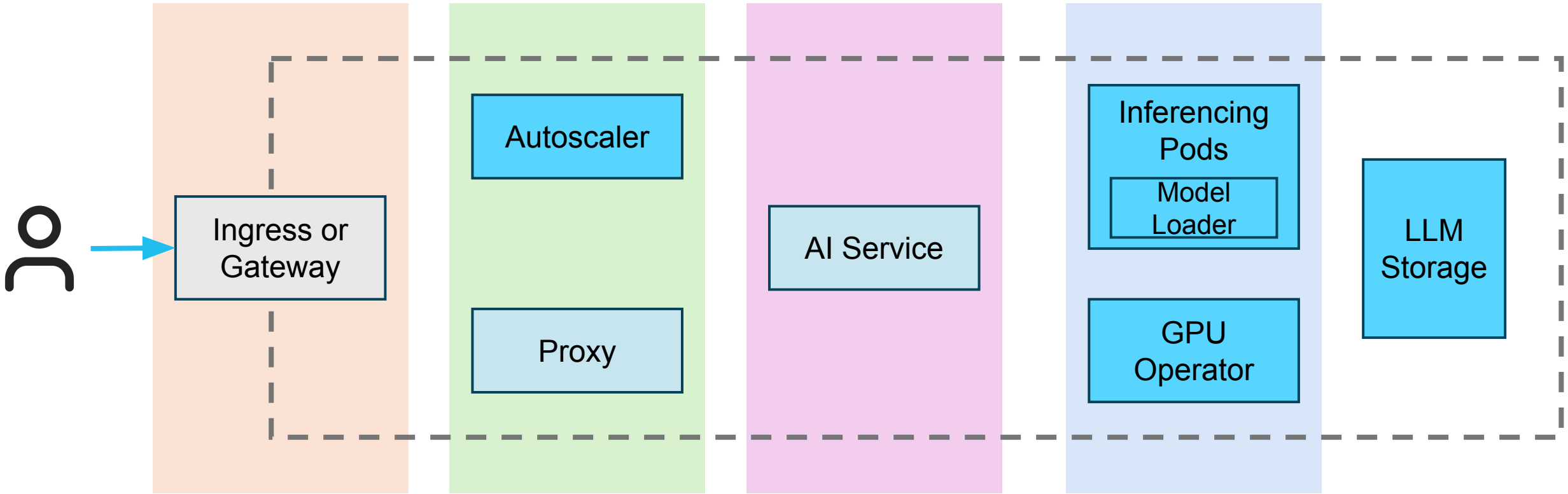
Increase Scalability



Unified Deployment

# Define your architecture





# Model Storage



Models are **big size binary files**.
The closer and the more you cache, the better.

With OCI, you can store models locally in infrastructure you already have to solve multiple issues and get multiple benefits (layering, hashing, Auth & RBAC, retrial mechanism, etc)

| Model | Size (GB) |
|---|---|
| Qwen1.5-1.8B | **~3.7GB** |
| Phi-2 | **~5.5GB** |
| Gemma 2B | **~5GB** |
| Llama2-7B | **~13GB** |
| Mistral-7B | **~14.5GB** |
| Gemma-7B | **~17GB** |
| Llama-70B | **~137GB** |

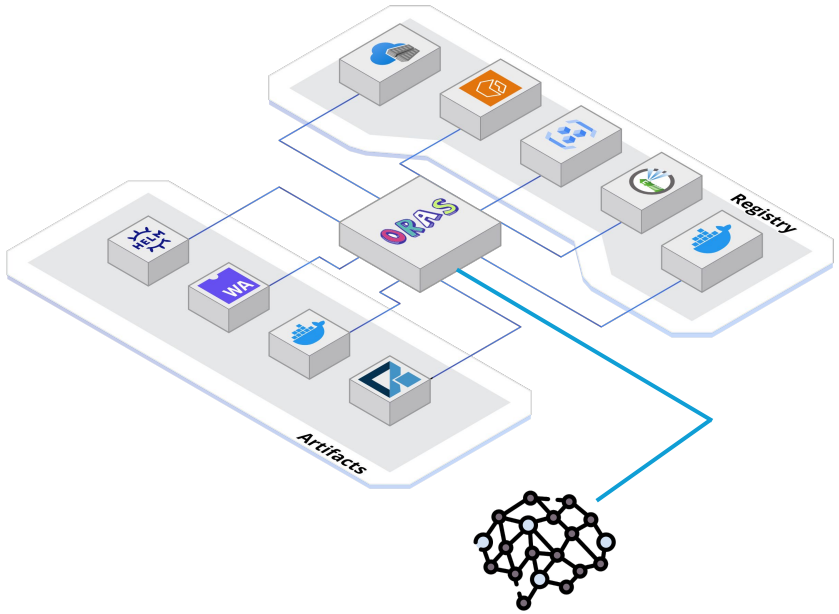
# Uploading and distributing as OCI artifacts



1. Chose your weight format (.pt vs .**safetensors** vs .bin vs .hdf5 vs so many other)
2. Define optimal division size and divide into smaller chunks
3. Upload to Container Registry using ORAS

```
oras push --plain-http localhost:5000/hello-artifact:v1 \
    --artifact-type application/vnd.acme.rocket.config \
    artifact.txt:text/plain
```

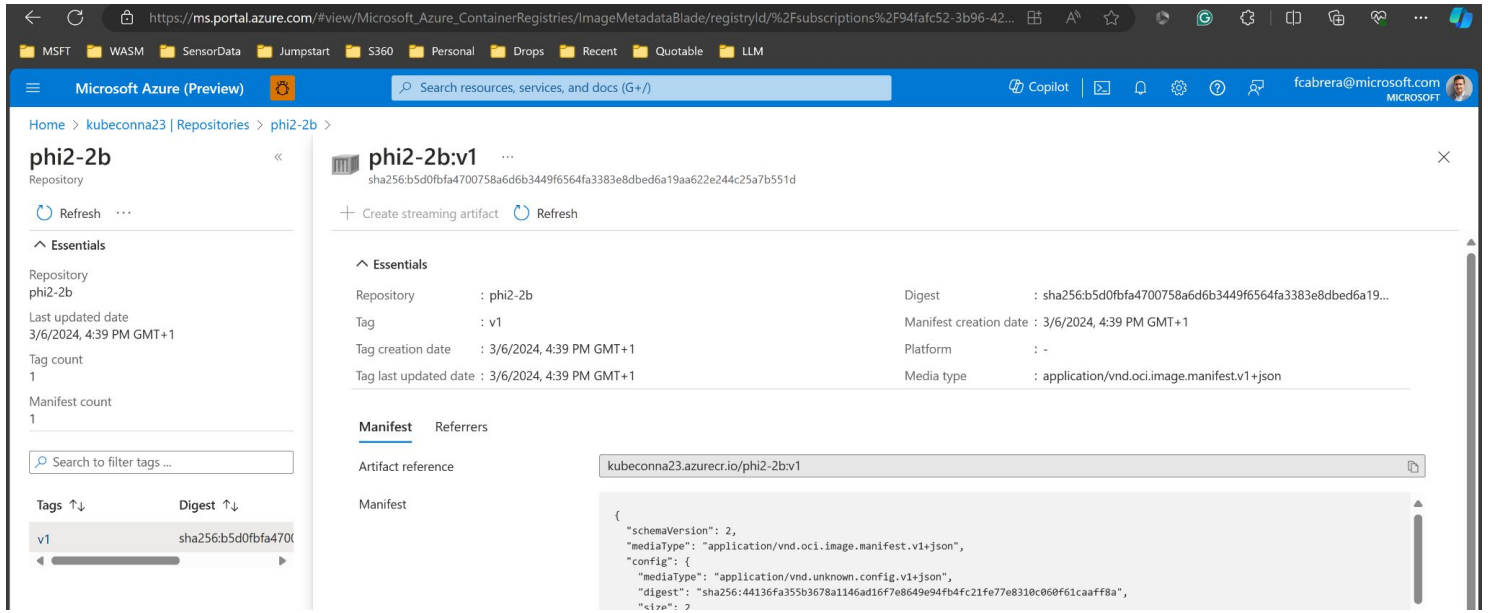
```
Uploading a948904f2f0f artifact.txt
Uploaded  a948904f2f0f artifact.txt
Pushed [registry] localhost:5000/hello-artifact:v1
Digest: sha256:bcdd6799fed0fca0eaedfc1c642f3d1dd7b8e78b43986a89935d6fe217a09cee
```



# Uploading and distributing as OCI artifacts



```
azureuser@artifcat-ubuntu:~/gemma-2b$ split -b 500M model-00001-of-00002.safetensors model-00001-of-00002.safetens
azureuser@artifcat-ubuntu:~/gemma-2b$ ls -la
total 4878520
drwxrwxr-x 2 azureuser azureuser      4096 Mar  5 00:56 .
drwxr-x--- 6 azureuser azureuser      4096 Mar  5 00:56 ..
-rw-rw-r-- 1 azureuser azureuser 524288000 Mar  4 21:24 model-00001-of-00002.safetensors-aa
-rw-rw-r-- 1 azureuser azureuser 524288000 Mar  4 21:24 model-00001-of-00002.safetensors-ab
-rw-rw-r-- 1 azureuser azureuser 524288000 Mar  4 21:24 model-00001-of-00002.safetensors-ac
-rw-rw-r-- 1 azureuser azureuser 524288000 Mar  4 21:24 model-00001-of-00002.safetensors-ad
-rw-rw-r-- 1 azureuser azureuser 524288000 Mar  4 21:24 model-00001-of-00002.safetensors-ae
-rw-rw-r-- 1 azureuser azureuser 524288000 Mar  4 21:24 model-00001-of-00002.safetensors-af
-rw-rw-r-- 1 azureuser azureuser 524288000 Mar  4 21:24 model-00001-of-00002.safetensors-ag
-rw-rw-r-- 1 azureuser azureuser 524288000 Mar  4 21:24 model-00001-of-00002.safetensors-ah
-rw-rw-r-- 1 azureuser azureuser 524288000 Mar  4 21:24 model-00001-of-00002.safetensors-ai
-rw-rw-r-- 1 azureuser azureuser 276992424 Mar  4 21:24 model-00001-of-00002.safetensors-aj
```





```
{
  "schemaVersion": 2,
  "mediaType": "application/vnd.oci.image.manifest.v1+json",
  "config": {
    "mediaType": "application/vnd.unknown.config.v1+json",
    "digest": "sha256:44136fa355b3678a1146ad16f7e8649e94fb4fc21fe77e8310c060f61caaff8a",
    "size": 2
  },
  "layers": [
    {
      "mediaType": "application/vnd.safetensors",
      "digest": "sha256:75ade030c2d8d2648f0a4378164feb094e582651e952b20147cdd72a277cbd34",
      "size": 524288000,
      "annotations": {
        "org.opencontainers.image.title": "model-00001-of-00002.safetensors-aa"
      }
    },
    {
      "mediaType": "application/vnd.safetensors",
      "digest": "sha256:093ab608367081dbace69033a69f5cafe574362a03c8d34085d5cebb74dfc920",
      "size": 524288000,
      "annotations": {
        "org.opencontainers.image.title": "model-00001-of-00002.safetensors-ab"
      }
    },
    {
      "mediaType": "application/vnd.safetensors",
      "digest": "sha256:629558dd17588c4628d809f51195789b1e579b655c962c1a886f6a47679fb23a",
      "size": 524288000,
      "annotations": {
        "org.opencontainers.image.title": "model-00001-of-00002.safetensors-ac"
      }
    },
    {
      "mediaType": "application/vnd.safetensors",
      "digest": "sha256:e4072def6152dace0438ce2449d6a72bed241d31c54b3e25cd8c95dd128cbd84",
      "size": 524288000,
      "annotations": {
        "org.opencontainers.image.title": "model-00001-of-00002.safetensors-ad"
      }
    },
    {
      "mediaType": "application/vnd.safetensors",
      "digest": "sha256:8952c92cb4cfbe6a4177763bb327b0245a5c04ba5ff4732d3df91dd60f9d2d94",
      "size": 524288000,
```

# Uploading and distributing as OCI artifacts



4. Download and recreate the model

5. Load the model and use your custom LORA layer

6. (Optional) Serve multiple LORA layers

```
# Load the adapter config
adapter_config = {}

with open(lora_path, 'r') as f:
    # Read the JSON data into a dictionary
    adapter_config = json.load(f)

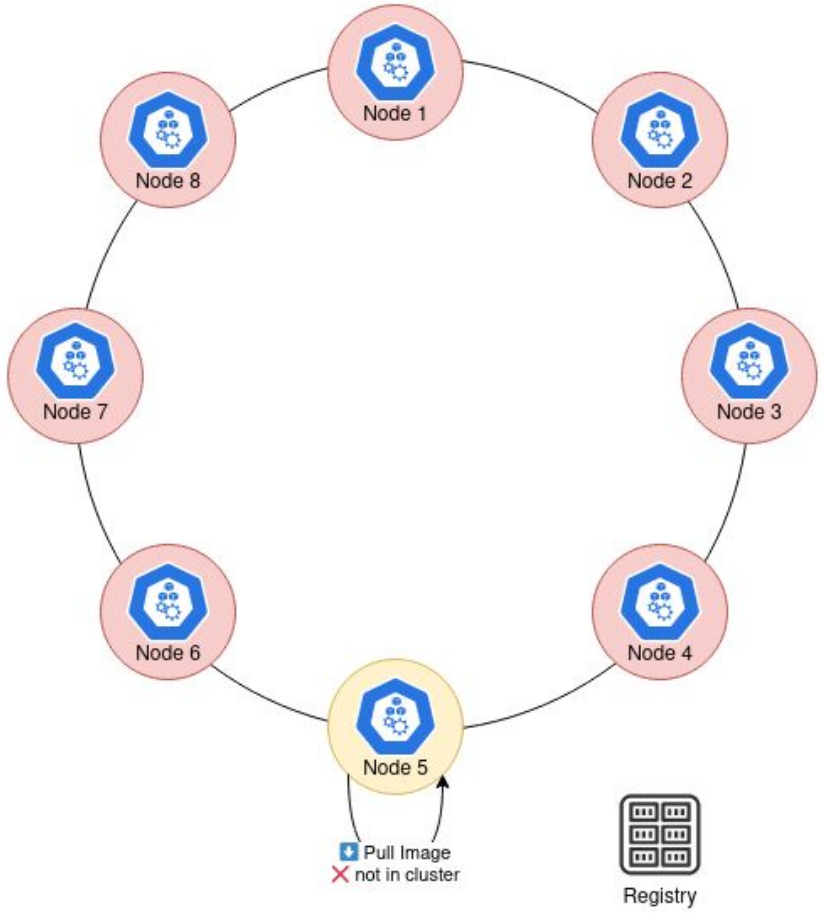
# Initialize the model
base_model_id = adapter_config["base_model_name_or_path"]
base_model = AutoModelForCausalLM.from_pretrained(
    base_model_id,
    low_cpu_mem_usage=True,
    return_dict=True,
    torch_dtype=torch.float16,
    device_map={"": 0},
)
tokenizer = AutoTokenizer.from_pretrained(base_model_id)
tokenizer.add_special_tokens({'pad_token': '[PAD]'})

# Load the base model + LORA Layer configuration
merged_model= PeftModel.from_pretrained(base_model, model_path)
```

# Local caching and P2P alternatives



## Spegel




## PeerD



# Local caching alternatives



## Spegel / PeerD

```
IMAGE                                  TAG                IMAGE ID        SIZE
aksiotdevacr.azurecr.io/pause          3.9                f48ea180879d3   747kB
docker.io/nvidia/cuda                  12.3.2-devel-ubi8  1ddbb83118cf5   3.81GB
ghcr.io/xenitab/spegel                 <none>             75586a67c4737   38.2MB
```

Node 1 - No caching: **~151 seconds**

```
Events:
  Type     Reason     Age    From               Message
  ----     ------     ----   ----               -------
  Normal   Scheduled  2m48s  default-scheduler  Successfully assigned default/cuda-deployment-6987f74877-fptq9 to desktop-5dd5gma-ledge
  Normal   Pulled     16s    kubelet            Successfully pulled image "nvidia/cuda:12.3.2-devel-ubi8" in 2m31.533946335s (2m31.534012288s including waiting)
```

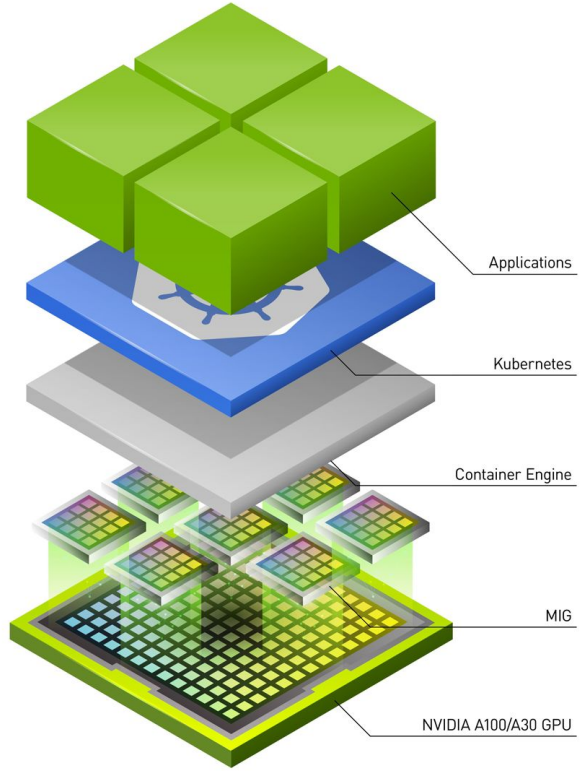
Node 2 - Served by Node 1: **~5.7 seconds -> ~28X faster**

```
Events:
  Type     Reason     Age    From               Message
  ----     ------     ----   ----               -------
  Normal   Scheduled  8m25s  default-scheduler  Successfully assigned default/cuda-deployment-756b5cbdb-5wx4k to desktop-h6iboc8-ledge
  Normal   Pulling    8m24s  kubelet            Pulling image "nvidia/cuda:12.3.2-devel-ubi8"
  Normal   Pulled     8m18s  kubelet            Successfully pulled image "nvidia/cuda:12.3.2-devel-ubi8" in 5.761850981s (5.761855961s including waiting)
```

# Using NVIDIA GPUs



1. Get your GPU ☹️
2. Set up your node – (NVIDIA Drivers, nvidia-container-toolkit and nvidia-container-runtime)
3. Define your Kubernetes device allocation: (k8s-device-plugin, k8s-dra-driver)
4. Define GPU access mechanism: single CUDA Process vs Concurrent Access (MPS, Time-slicing, MIG, vGPUs)
5. Application logic – (GPU/CPU, LoRa, etc)



# Custom scaling



1. Scale up/down based on multiple metrics and events

2. Leverage GPU metrics for HPA (enable DCGM-Exporter)

3. Create your custom scaling logic and use custom metrics (e.g latency, #tokens/sec, GPU utilization)



# Demo

# Demo



- You are working for an HR company that needs to process people's introductions and extract certain information. You want to automate the process.
- You need to extract different kind of information from multiple sources in different languages.

```
Hi! My name is Angel and I'm 32 years old. I currently live in Sevilla and I'm working as a software engineer.
```



```
{ "name": "Angel", "age": "32", "role": "software engineer", "location": "Sevilla" }
```

# Demo

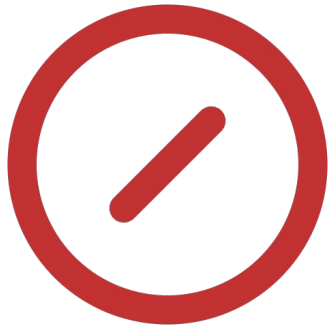




Mixtral 8x7B



Gemma-7b-it



Phi-2 /
Gemma-2B

# Demo





## Phi-2

- Create a synthetic dataset (for demo).
- Fine-tune the Microsoft / Phi2 model with the data.
- Run the fine-tuned model in a cluster.

# Related links

# Related links



All the following links refer to the demo:

- **Synthetic dataset:** https://huggingface.co/datasets/angelmmiguel/synthetic-introduction-extraction
- **Fine-tuned model:** https://huggingface.co/angelmmiguel/phi2-intro-finetuned-demo
- **Repository:** https://github.com/Angelmmiguel/kubecon-2024-demo

# Thanks