aws

COMMUNITY DAY

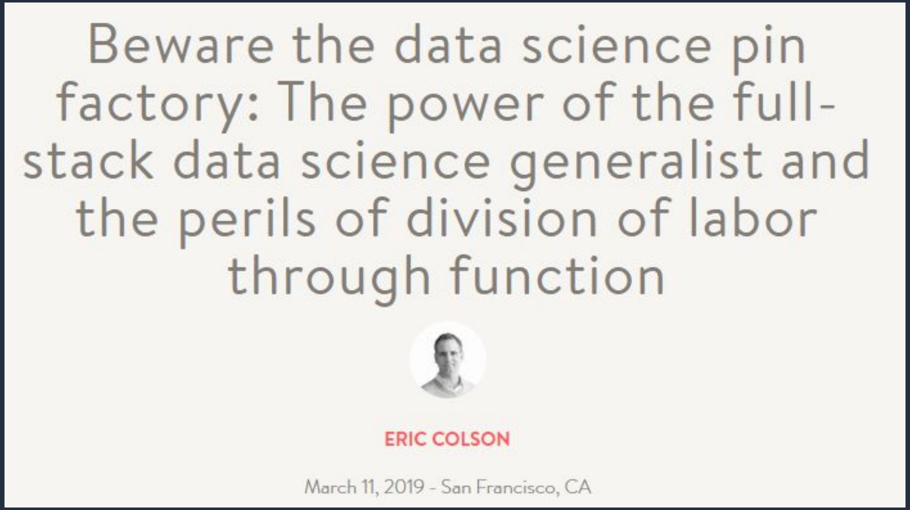
Avoiding Common Pitfalls with Hosting Machine Learning Models

Max De Jong | June 13, 2024

Who Am I?

Applied scientist with academic background

Realized that a “full ML stack” understanding required for maximum impact



Beware the data science pin factory: The power of the full-stack data science generalist and the perils of division of labor through function

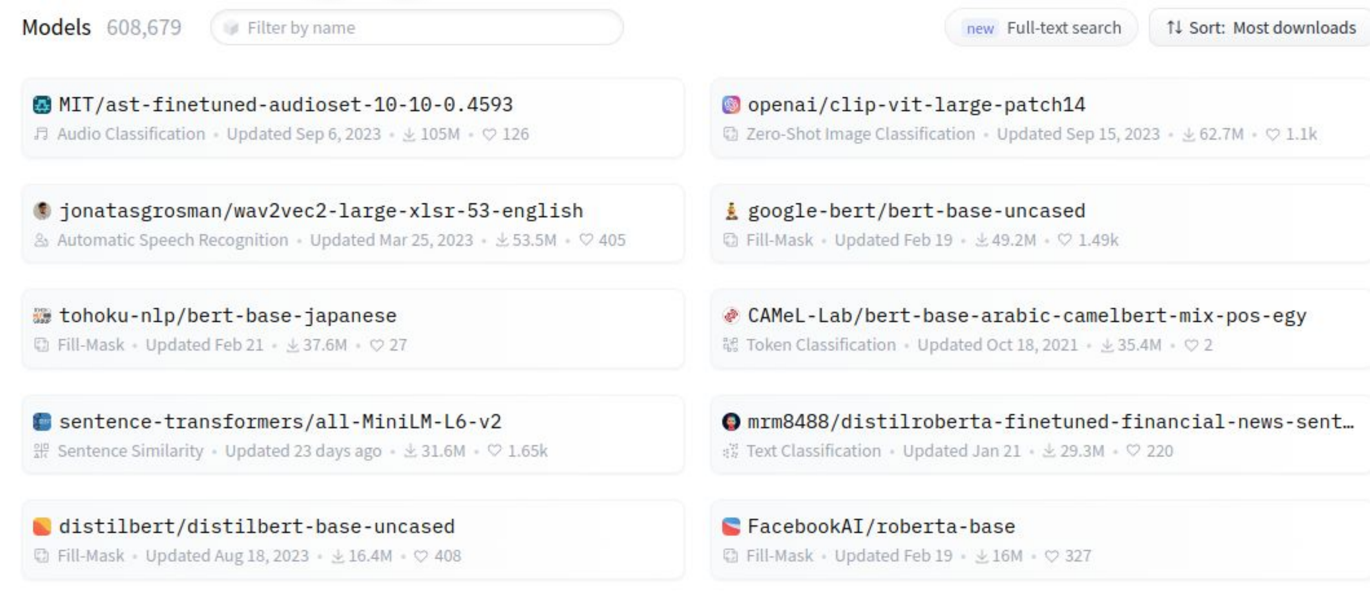
ERIC COLSON

March 11, 2019 - San Francisco, CA

Explosion of Open Source Models

There has never been a better moment to build with machine learning

Hugging Face



Models 608,679 Filter by name | new Full-text search | Sort: Most downloads

MIT/ast-finetuned-audioset-10-10-0.4593
Audio Classification • Updated Sep 6, 2023 • 105M • 126

openai/clip-vit-large-patch14
Zero-Shot Image Classification • Updated Sep 15, 2023 • 62.7M • 1.1k

jonatasgrosman/wav2vec2-large-xlsr-53-english
Automatic Speech Recognition • Updated Mar 25, 2023 • 53.5M • 405

google-bert/bert-base-uncased
Fill-Mask • Updated Feb 19 • 49.2M • 1.49k

tohoku-nlp/bert-base-japanese
Fill-Mask • Updated Feb 21 • 37.6M • 27

CAMeL-Lab/bert-base-arabic-camelbert-mix-pos-egy
Token Classification • Updated Oct 18, 2021 • 35.4M • 2

sentence-transformers/all-MiniLM-L6-v2
Sentence Similarity • Updated 23 days ago • 31.6M • 1.65k

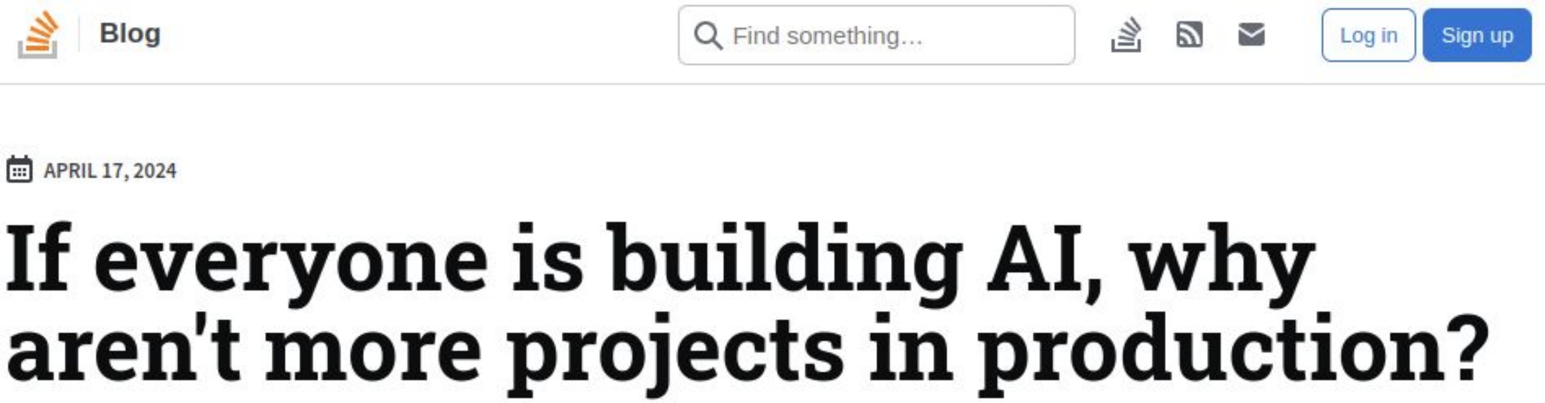
mrm8488/distilroberta-finetuned-financial-news-sent…
Text Classification • Updated Jan 21 • 29.3M • 220

distilbert/distilbert-base-uncased
Fill-Mask • Updated Aug 18, 2023 • 16.4M • 408

FacebookAI/roberta-base
Fill-Mask • Updated Feb 19 • 16M • 327

Yet Something Is Missing…

Breakthroughs in models don’t translate to ML democratization

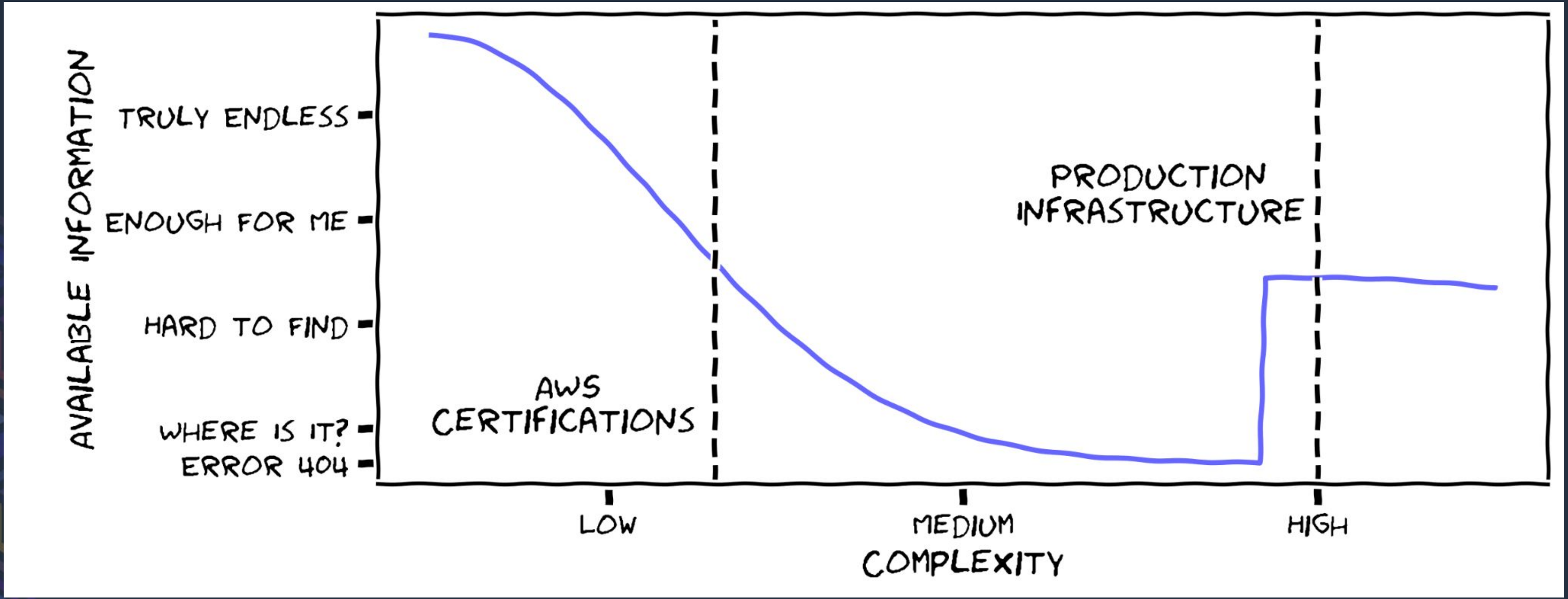


APRIL 17, 2024

If everyone is building AI, why aren't more projects in production?

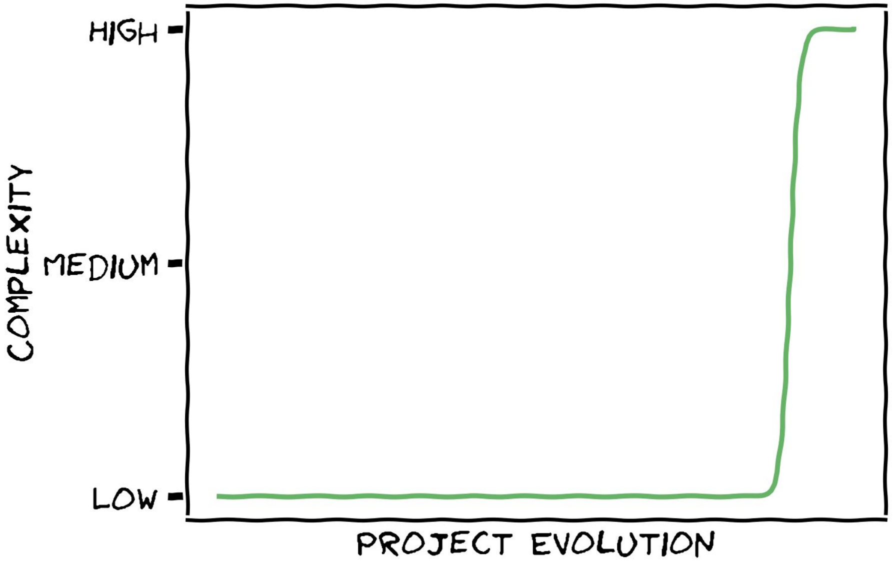
Major Knowledge Gap

Lack of intermediate resources makes learning much harder than necessary



Resulting Difficulty Cliff

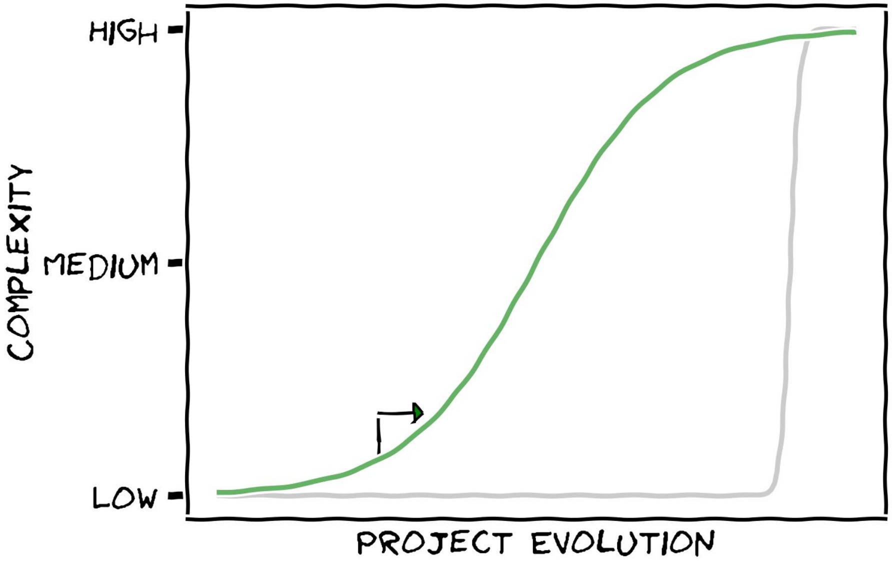
Very hard to transition out of the beginner phase of a project without enough educational resources



Flattening the Difficulty Cliff

Today’s theme: finding atomic, tractable improvements to allow for meaningful iteration

Along the way, identifying pitfalls to avoid



Building Philosophy

Machine learning solutions are costly to properly build and maintain

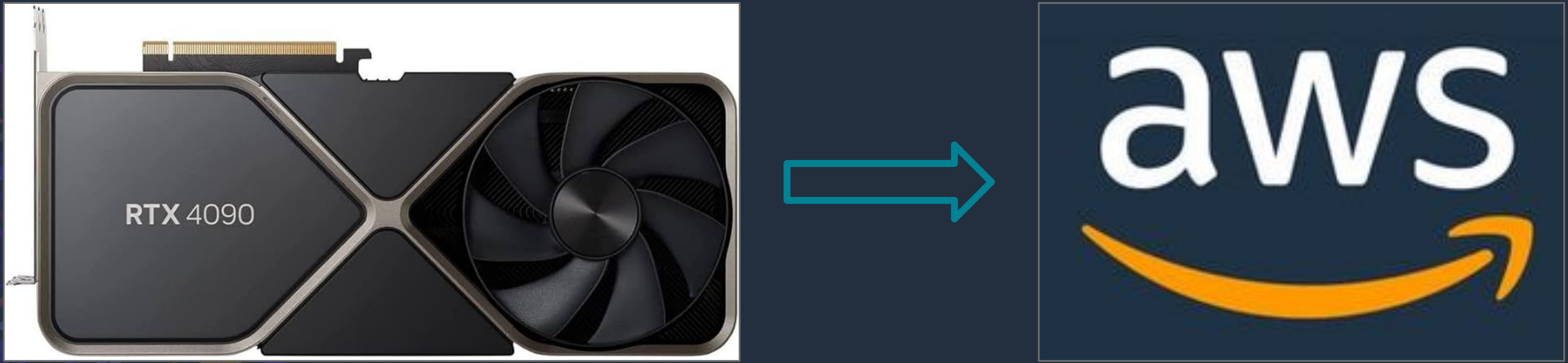
- Lots of models end up not working as intended

Goal: avoid sinking time in untested ideas while avoiding getting crushed by technical debt if we want to scale our solution

- Think “scalable proof of concept”
- It’s ok to have “bad” system designs as long as they can be easily improved

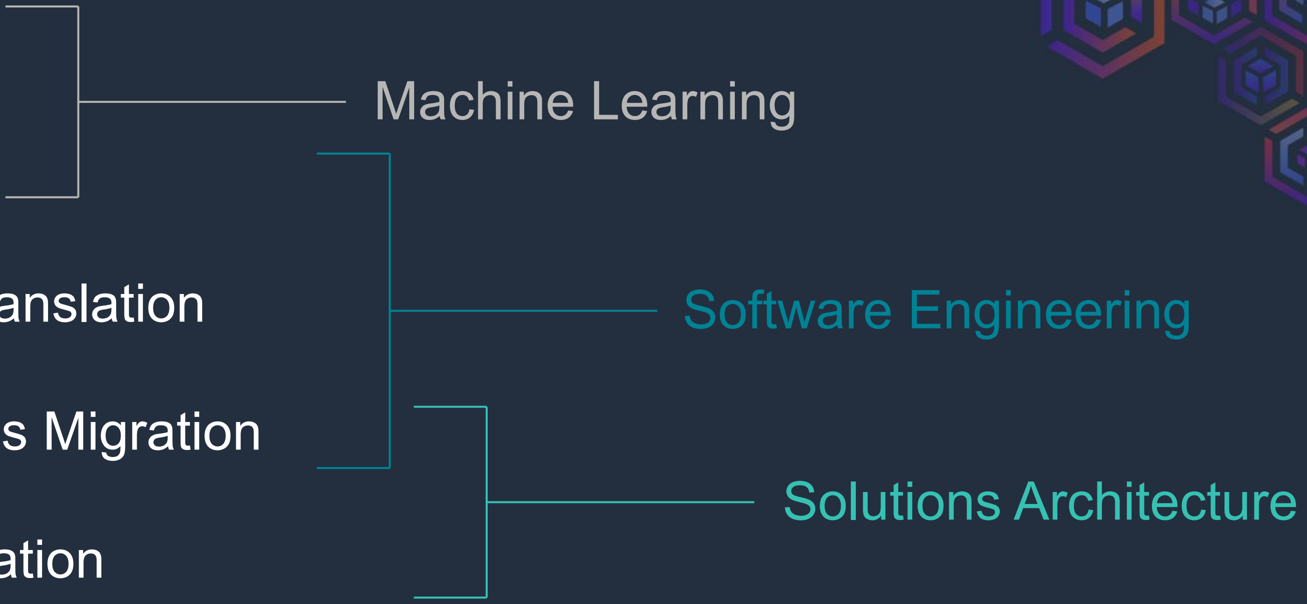
Where to Build

Prototype locally, deploy to AWS



How to Build

1. Fact Finding
2. Bake-off
3. Microservice Translation
4. Cloud Essentials Migration
5. Full Cloud Migration



Machine Learning

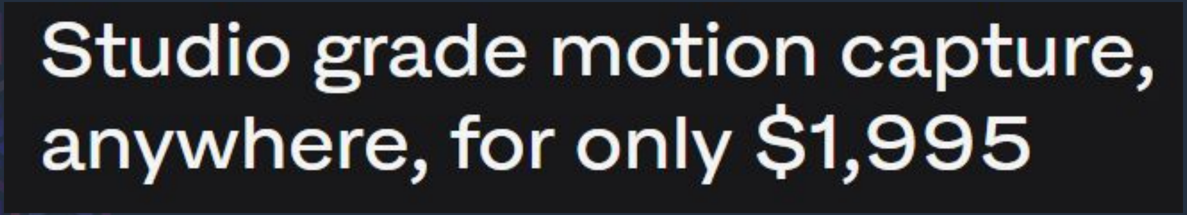
Software Engineering

Solutions Architecture

Specific Application: 3D Pose Estimation

What is possible with open-source models?

- Some benchmarks for general tasks
- Nothing for our specific use case



Studio grade motion capture, anywhere, for only $1,995

Step #1: Fact Finding

Double-pronged investigation through literature and repos

Goals:

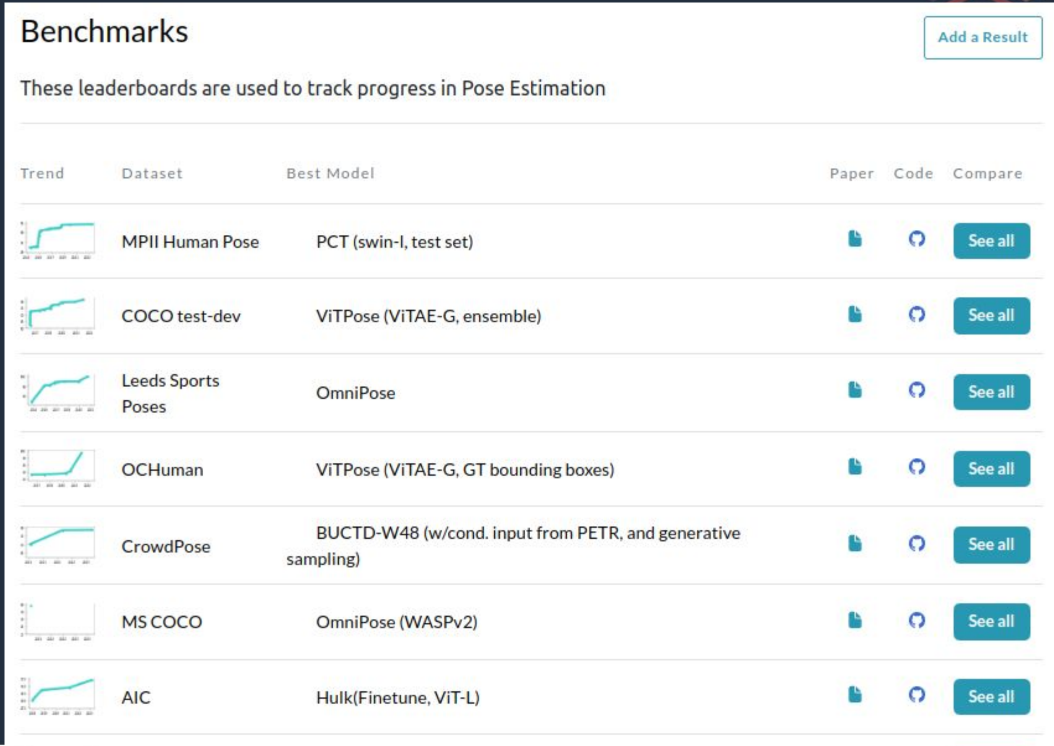
1. Learn something about the classes of models
2. Make a list of repos with public code/weights

Fact Finding

Two major classes of approaches: top-down vs. bottom-up

Lots of potential projects to try

Benchmark results are only a starting point



Benchmarks

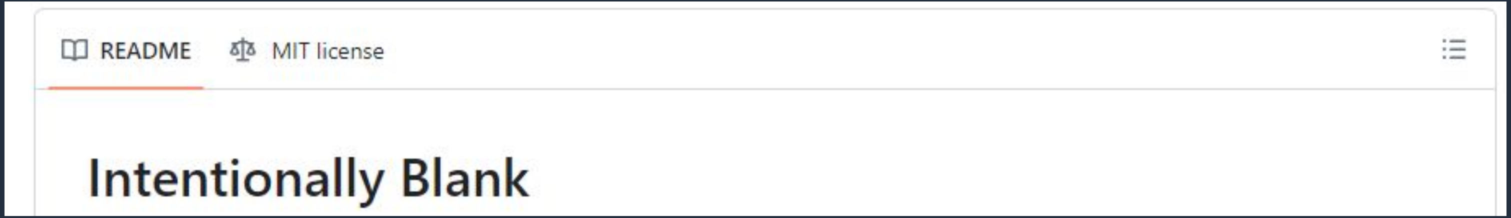
Add a Result

These leaderboards are used to track progress in Pose Estimation

Trend	Dataset	Best Model	Paper	Code	Compare
	MPII Human Pose	PCT (swin-l, test set)			See all
	COCO test-dev	ViTPose (ViTAE-G, ensemble)			See all
	Leeds Sports Poses	OmniPose			See all
	OCHuman	ViTPose (ViTAE-G, GT bounding boxes)			See all
	CrowdPose	BUCTD-W48 (w/cond. input from PETR, and generative sampling)			See all
	MS COCO	OmniPose (WASPv2)			See all
	AIC	Hulk(Finetune, ViT-L)			See all

Step #2: Bake Off

“Simple”: clone the repos, read the READMEs, and run the example scripts



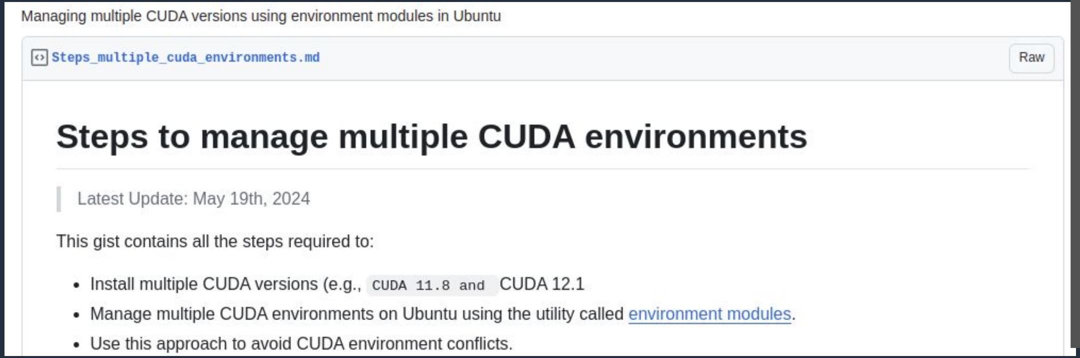
Intentionally Blank

Two main obstacles:

1. Managing CUDA versions
2. Bit rot

Managing CUDA Versions: Wrong Way

Main obstacle: CUDA



Managing multiple CUDA versions using environment modules in Ubuntu

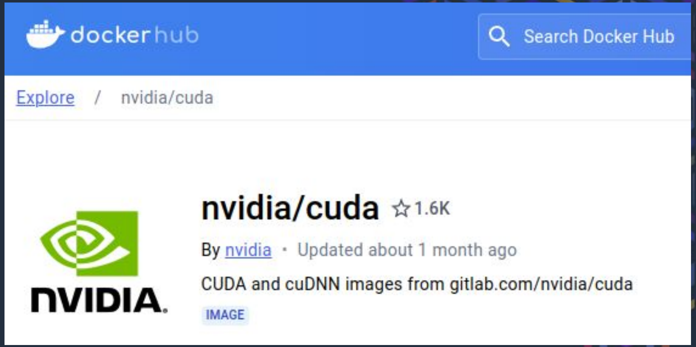
Steps to manage multiple CUDA environments

> Latest Update: May 19th, 2024

This gist contains all the steps required to:

- Install multiple CUDA versions (e.g., `CUDA 11.8 and` CUDA 12.1
- Manage multiple CUDA environments on Ubuntu using the utility called environment modules.
- Use this approach to avoid CUDA environment conflicts.

Solution: Docker



docker hub

Search Docker Hub

Explore / nvidia/cuda

nvidia/cuda ☆1.6K

By nvidia · Updated about 1 month ago

CUDA and cuDNN images from gitlab.com/nvidia/cuda

IMAGE

Every project gets a Dockerfile

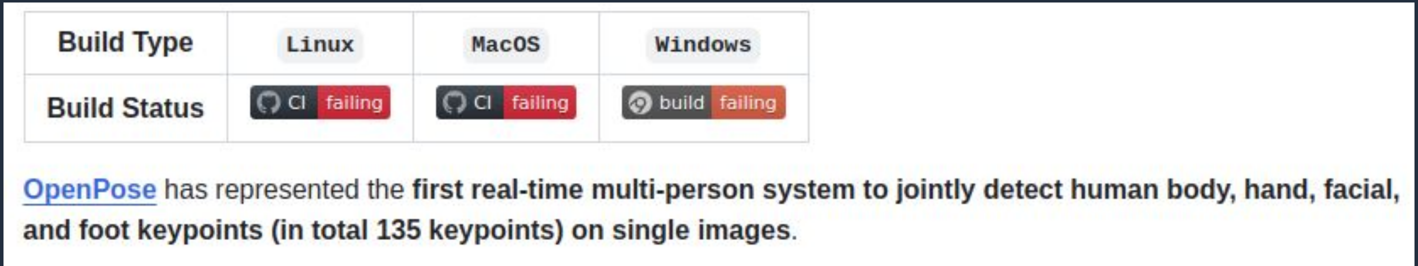
Install a recent version of CUDA on your machine

Pin it until you have a good reason to upgrade

Every ML project gets its own container isolated from your system

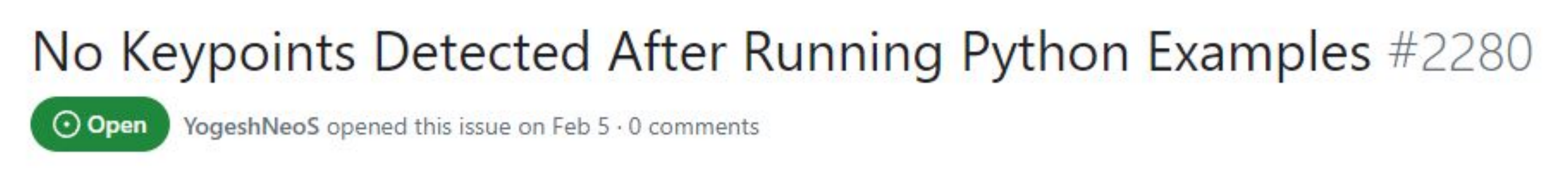
Bit Rot

Many academic repositories are not maintained



Build Type	Linux	MacOS	Windows
Build Status	CI failing	CI failing	build failing

[OpenPose](#) has represented the **first real-time multi-person system to jointly detect human body, hand, facial, and foot keypoints (in total 135 keypoints) on single images**.

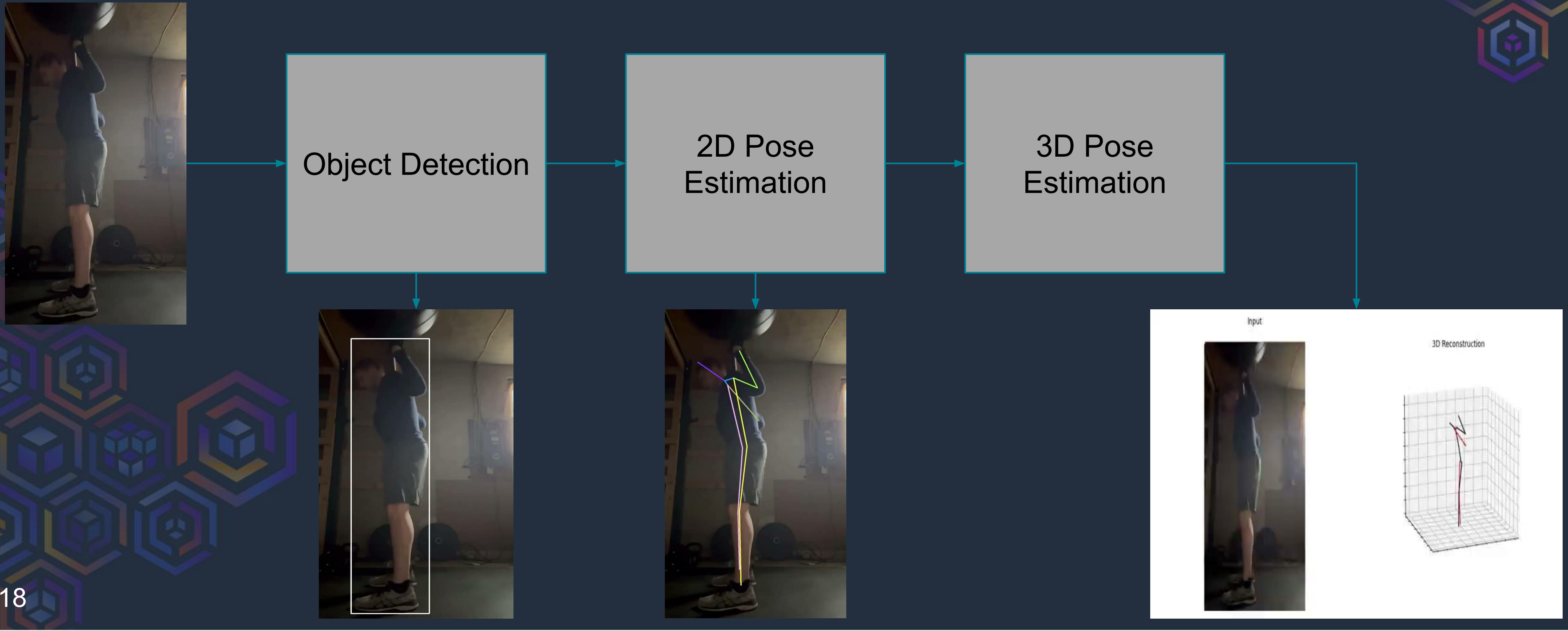


No Keypoints Detected After Running Python Examples #2280

Open YogeshNeoS opened this issue on Feb 5 · 0 comments

Finalizing Architecture

End goal: settle on the model pipeline



Object Detection

2D Pose Estimation

3D Pose Estimation

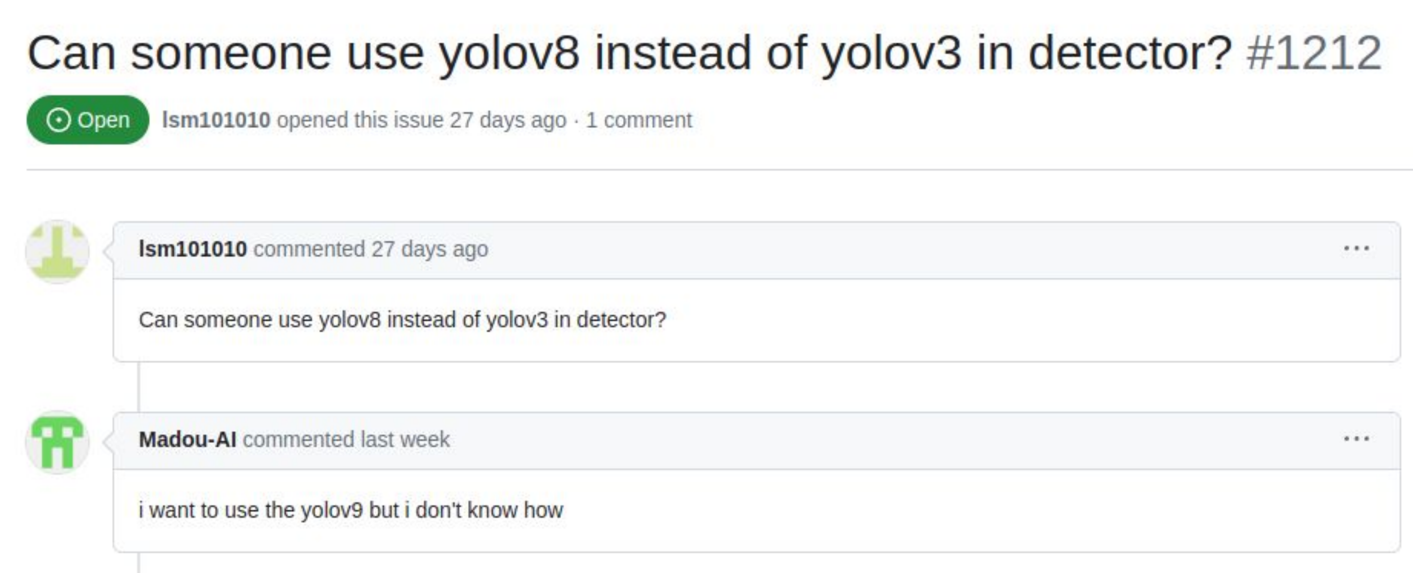
Step #3: Microservice Translation

General procedure:

1. Wrap model inference in APIs using Flask/fastAPI
2. Create a web server using gunicorn/uWSGI
3. Run NGINX reverse-proxy

Microservice Translation

Main obstacle: Monolith with tight coupling



Can someone use yolov8 instead of yolov3 in detector? #1212

lsm101010 opened this issue 27 days ago · 1 comment

lsm101010 commented 27 days ago

Can someone use yolov8 instead of yolov3 in detector?

Madou-AI commented last week

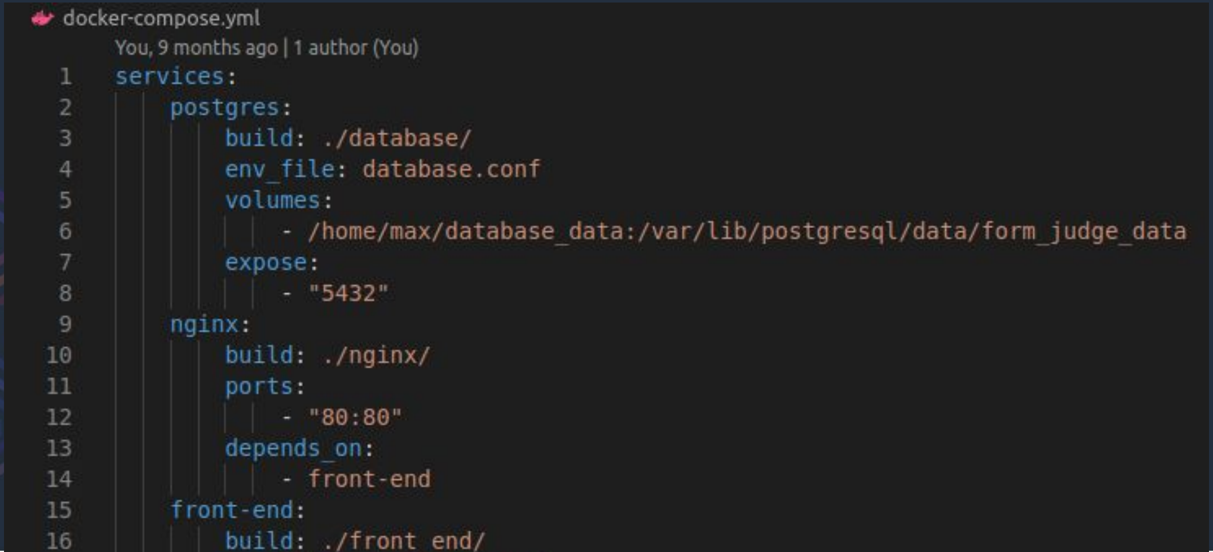
i want to use the yolov9 but i don't know how

Docker Compose

Other containers supporting the ML microservices

- Database, utilities, front end, etc.

Docker Compose allows running multi-container applications



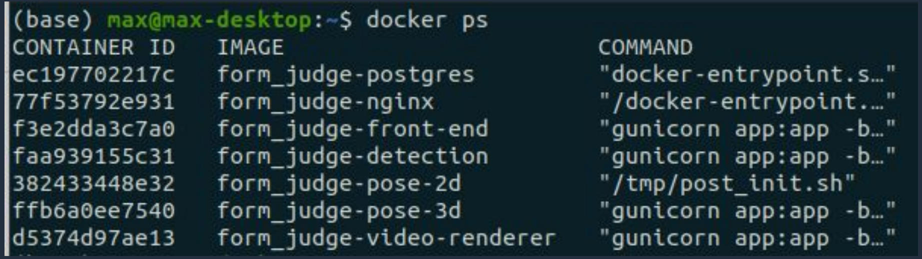
docker-compose.yml

You, 9 months ago | 1 author (You)

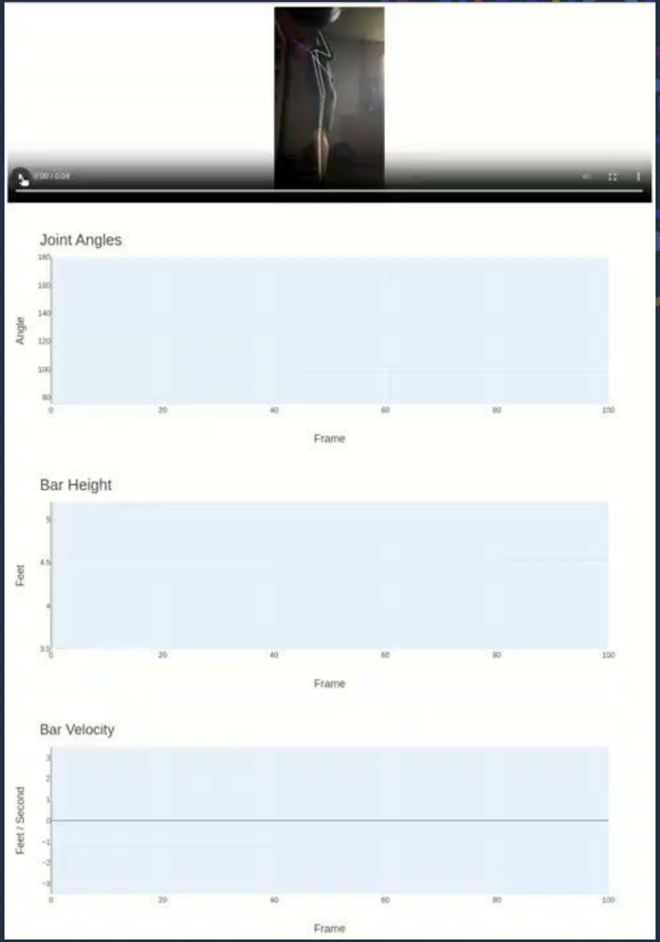
```yaml
services:
    postgres:
        build: ./database/
        env_file: database.conf
        volumes:
            - /home/max/database_data:/var/lib/postgresql/data/form_judge_data
        expose:
            - "5432"
    nginx:
        build: ./nginx/
        ports:
            - "80:80"
        depends_on:
            - front-end
    front-end:
        build: ./front_end/
```

Microservices End State

Local containerized service running end-to-end



```
(base) max@max-desktop:~$ docker ps
CONTAINER ID   IMAGE                          COMMAND
ec197702217c   form_judge-postgres            "docker-entrypoint.s…"
77f53792e931   form_judge-nginx               "/docker-entrypoint.…"
f3e2dda3c7a0   form_judge-front-end           "gunicorn app:app -b…"
faa939155c31   form_judge-detection           "gunicorn app:app -b…"
382433448e32   form_judge-pose-2d             "/tmp/post_init.sh"
ffb6a0ee7540   form_judge-pose-3d             "gunicorn app:app -b…"
d5374d97ae13   form_judge-video-renderer      "gunicorn app:app -b…"
```

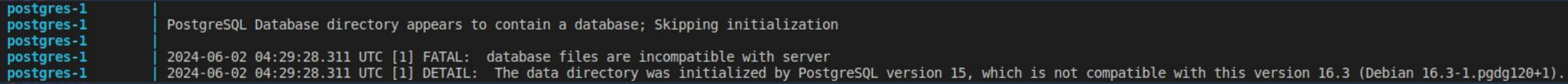


Step #4: Cloud Essentials

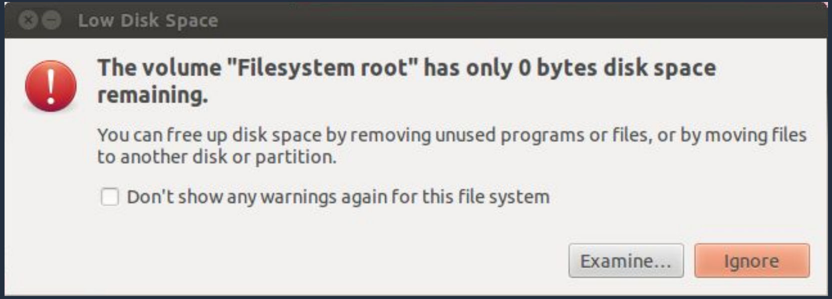
Still major design decisions before choosing a cloud architecture

Some common elements to all routes: database and object storage

- These are the first things we don’t want to manage



```
postgres-1          |
postgres-1          | PostgreSQL Database directory appears to contain a database; Skipping initialization
postgres-1          |
postgres-1          | 2024-06-02 04:29:28.311 UTC [1] FATAL:  database files are incompatible with server
postgres-1          | 2024-06-02 04:29:28.311 UTC [1] DETAIL:  The data directory was initialized by PostgreSQL version 15, which is not compatible with this version 16.3 (Debian 16.3-1.pgdg120+1).
```



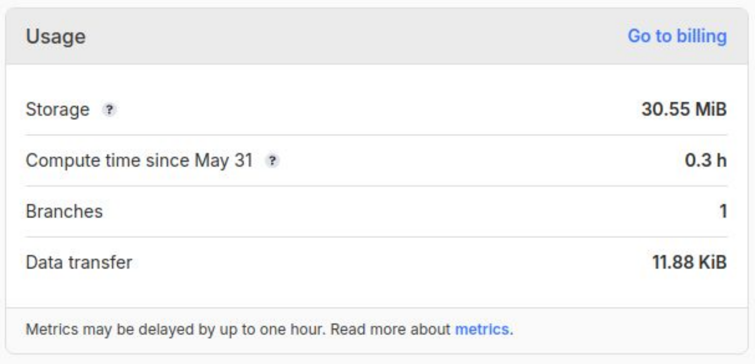
The volume "Filesystem root" has only 0 bytes disk space remaining.

You can free up disk space by removing unused programs or files, or by moving files to another disk or partition.

□ Don't show any warnings again for this file system

Database Choice

Hobby projects really benefit from a scale-to-zero database



Usage — Go to billing

Storage ?	30.55 MiB
Compute time since May 31 ?	0.3 h
Branches	1
Data transfer	11.88 KiB

Metrics may be delayed by up to one hour. Read more about metrics.

Minimum monthly cost of Aurora serverless: $43
Minimum monthly cost of Aurora on db.t4g.medium: $53

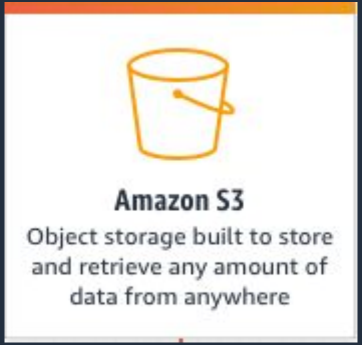
CockroachDB

NEON

Storage

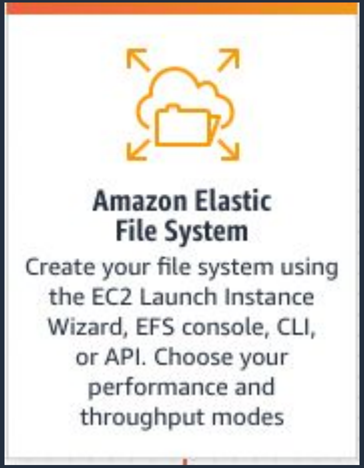
S3 always a good starting point

Later migrate to something EFS or similar



Amazon S3

Object storage built to store and retrieve any amount of data from anywhere



Amazon Elastic File System

Create your file system using the EC2 Launch Instance Wizard, EFS console, CLI, or API. Choose your performance and throughput modes

Step #5: Full Cloud Microservice Deployments

Two routes here:

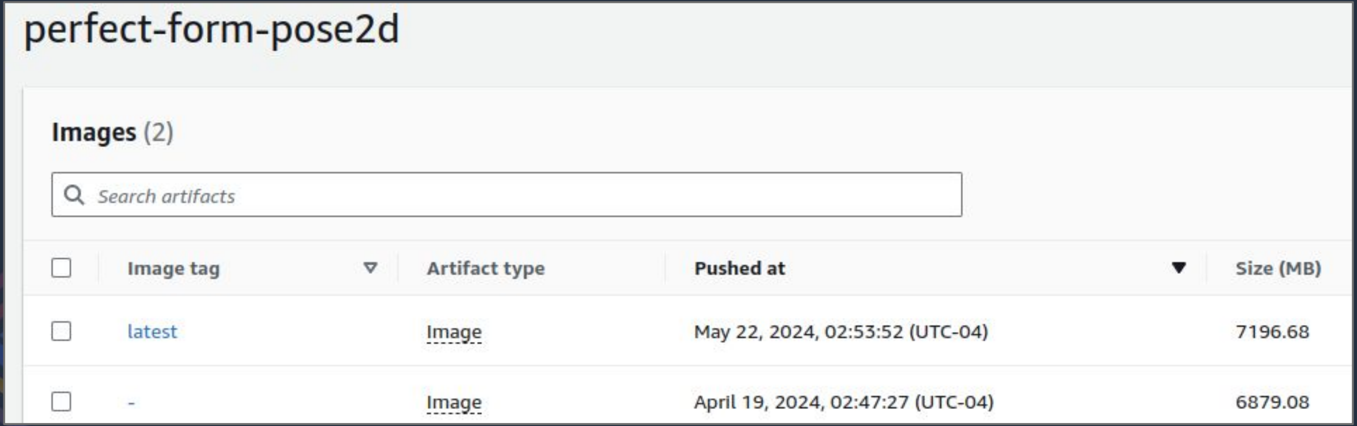
1. Run the microservices on Elastic Container Service
2. Run the microservices on Elastic Kubernetes Service

But first, let’s get ready for running containers on AWS hardware

Elastic Container Registry

Use ECR to store Docker container images

Large models produce large images



perfect-form-pose2d

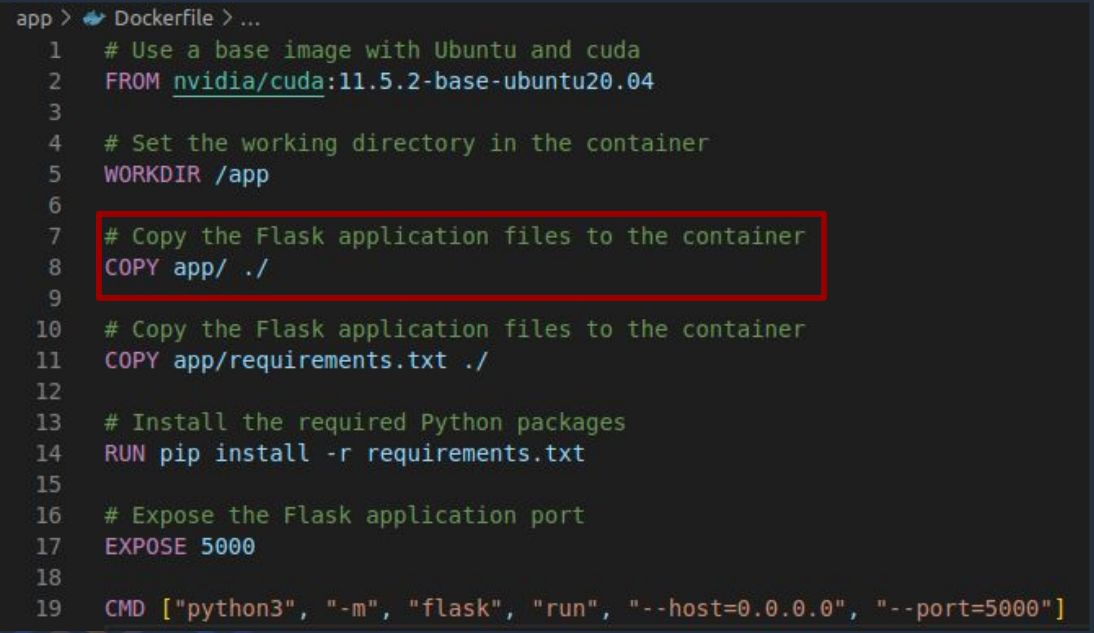
Images (2)

Search artifacts

□	Image tag ▽	Artifact type	Pushed at ▼	Size (MB)
□	latest	Image	May 22, 2024, 02:53:52 (UTC-04)	7196.68
□	-	Image	April 19, 2024, 02:47:27 (UTC-04)	6879.08

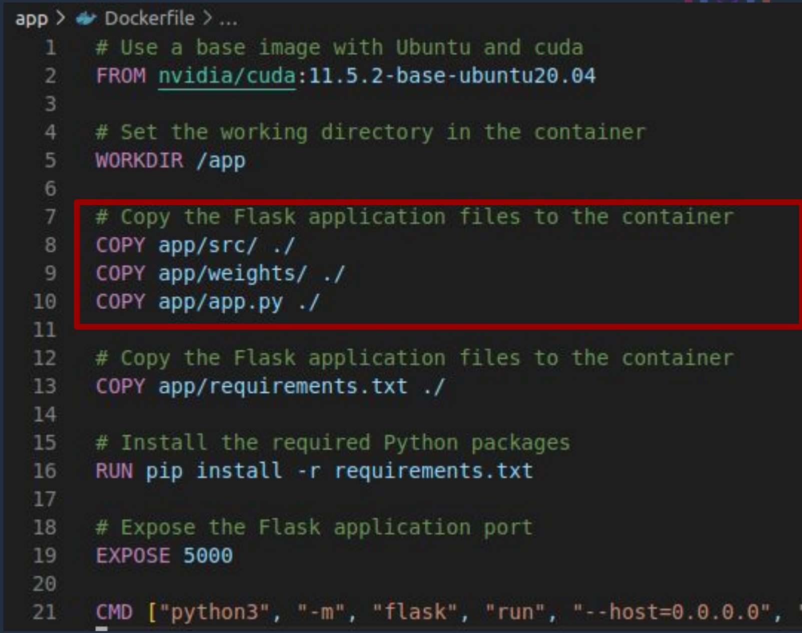
Optimizing Dockerfiles for ECR

Pushing to ECR is slow: think about layers



app > Dockerfile > ...

```
# Use a base image with Ubuntu and cuda
FROM nvidia/cuda:11.5.2-base-ubuntu20.04

# Set the working directory in the container
WORKDIR /app

# Copy the Flask application files to the container
COPY app/ ./

# Copy the Flask application files to the container
COPY app/requirements.txt ./

# Install the required Python packages
RUN pip install -r requirements.txt

# Expose the Flask application port
EXPOSE 5000

CMD ["python3", "-m", "flask", "run", "--host=0.0.0.0", "--port=5000"]
```



app > Dockerfile > ...

```
# Use a base image with Ubuntu and cuda
FROM nvidia/cuda:11.5.2-base-ubuntu20.04

# Set the working directory in the container
WORKDIR /app

# Copy the Flask application files to the container
COPY app/src/ ./
COPY app/weights/ ./
COPY app/app.py ./

# Copy the Flask application files to the container
COPY app/requirements.txt ./

# Install the required Python packages
RUN pip install -r requirements.txt

# Expose the Flask application port
EXPOSE 5000

CMD ["python3", "-m", "flask", "run", "--host=0.0.0.0", "
```

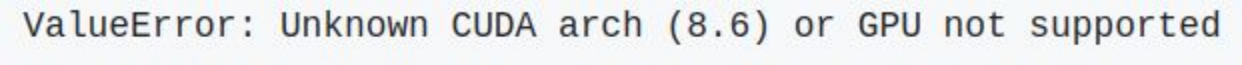
Precursor: EC2 with Docker

Spin up an EC2 instance with a GPU (p3.2xlarge)

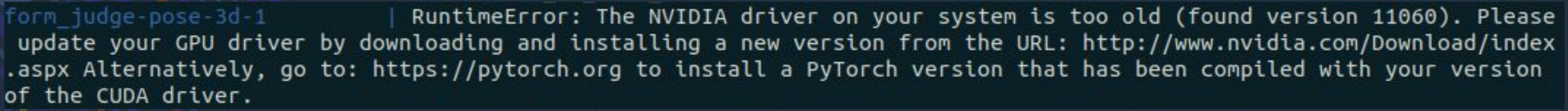
Recreate your local Docker Compose app pulling from ECR

CPU Dockerized applications behave better than GPU applications

If we have to debug something, let’s do it in easy mode



```
ValueError: Unknown CUDA arch (8.6) or GPU not supported
```



```
form_judge-pose-3d-1          | RuntimeError: The NVIDIA driver on your system is too old (found version 11060). Please
 update your GPU driver by downloading and installing a new version from the URL: http://www.nvidia.com/Download/index
.aspx Alternatively, go to: https://pytorch.org to install a PyTorch version that has been compiled with your version
of the CUDA driver.
```

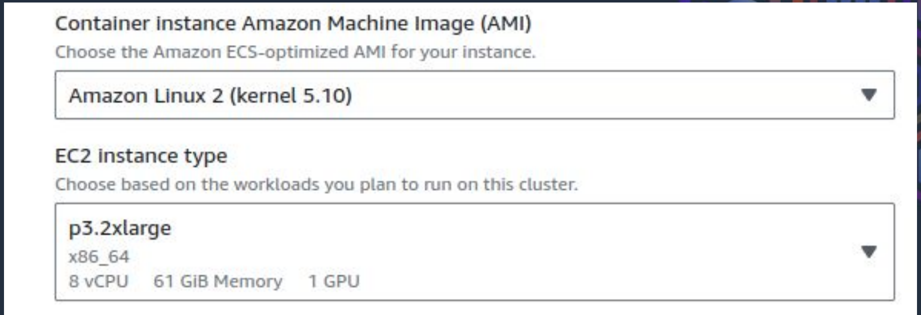
Choosing Cloud Direction

Elastic Container Service

- Less complexity
- Scale to zero

Elastic Kubernetes Service

- Best practice for full control
- Easier local development
- Multi-cloud solution



Container instance Amazon Machine Image (AMI)

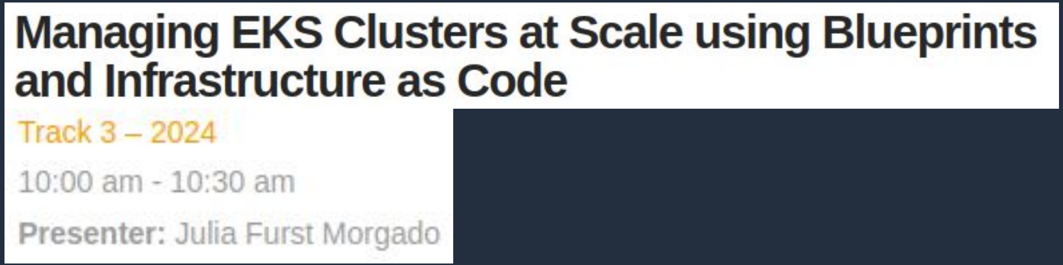
Choose the Amazon ECS-optimized AMI for your instance.

Amazon Linux 2 (kernel 5.10) ▼

EC2 instance type

Choose based on the workloads you plan to run on this cluster.

p3.2xlarge
x86_64
8 vCPU 61 GiB Memory 1 GPU ▼



Managing EKS Clusters at Scale using Blueprints and Infrastructure as Code

Track 3 – 2024

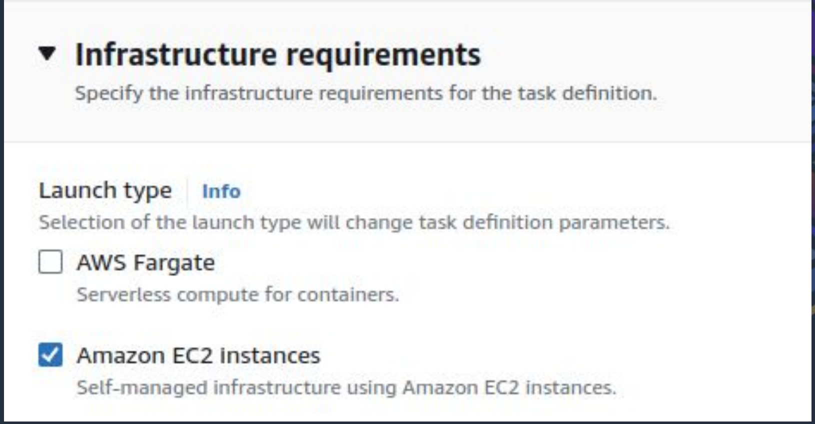
10:00 am - 10:30 am

Presenter: Julia Furst Morgado

Elastic Container Service

True container orchestration: scalable

Translate our Docker Compose YAML to a Task Definition



▼ **Infrastructure requirements**

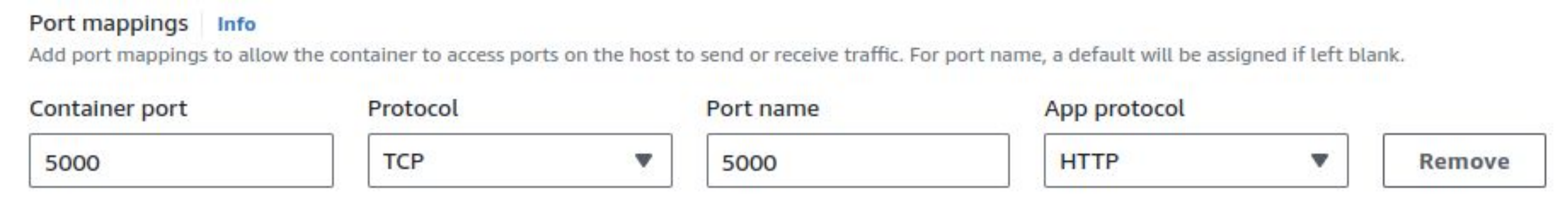
Specify the infrastructure requirements for the task definition.

Launch type | Info

Selection of the launch type will change task definition parameters.

☐ AWS Fargate
Serverless compute for containers.

☑ Amazon EC2 instances
Self-managed infrastructure using Amazon EC2 instances.



Port mappings | Info

Add port mappings to allow the container to access ports on the host to send or receive traffic. For port name, a default will be assigned if left blank.

Container port	Protocol	Port name	App protocol	
5000	TCP ▼	5000	HTTP ▼	Remove

To start, group all GPU containers into a single Task

Working with ECS

Pay attention to the Network Mode in Task Definitions

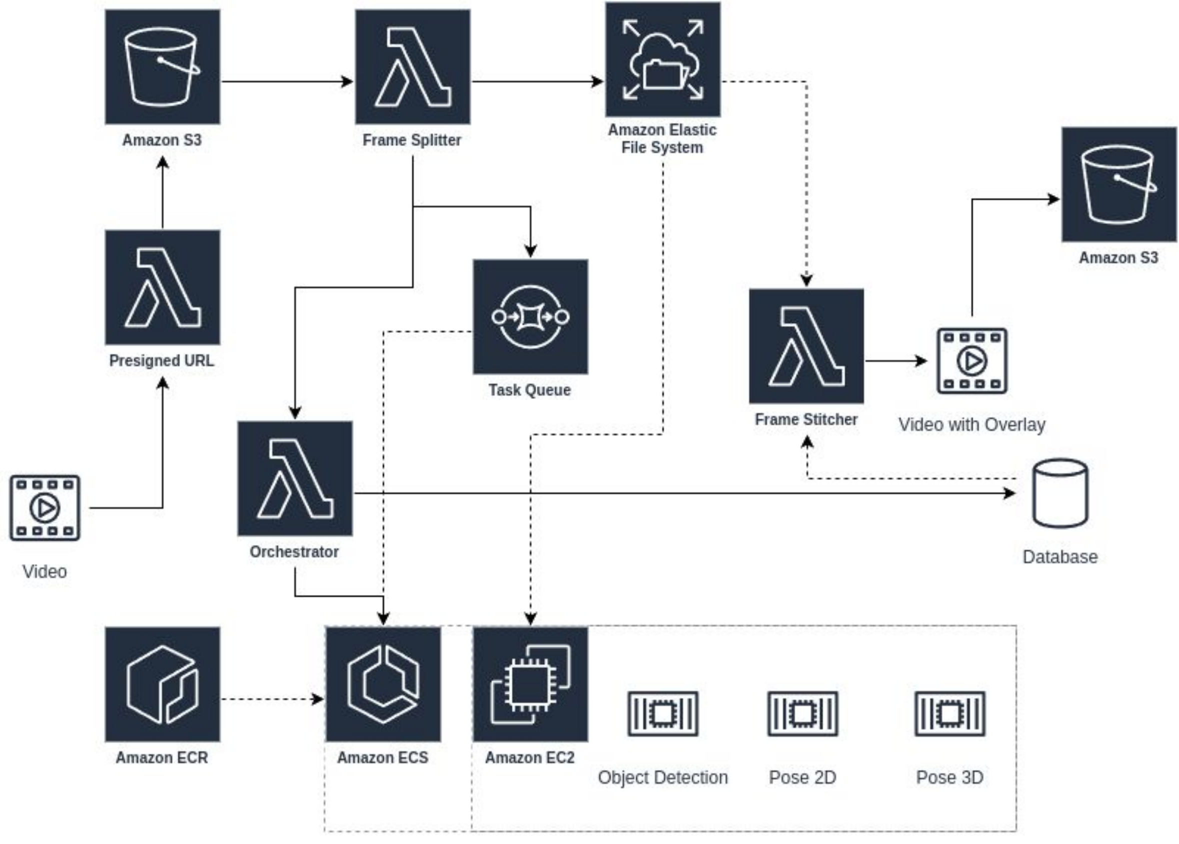
Can scale to 0 with some effort

Expose endpoints using Service Discovery/Service Connect

Don’t be afraid to re-architect system to better utilize AWS services

- API Gateway has 30 second timeout

Final Architecture



Recap

We started with a problem we wanted to solve

1. Found many potentially relevant repos with model weights
2. Determined the best model(s) for our use case
3. Created local microservice with Docker Compose
4. Moved storage to cloud
5. Migrated full microservice to AWS