Beam ML Containers

Tracking issue: #35487

Danny McCormick (dannymccormick@google.com)

Overview

Today, we publish a set of base beam containers with standard and distroless variants. Users often also want fat containers with a specific set of dependencies, with the most common request seeming to be ML-specific containers. This proposal recommends adding 2 sets of python-based ML-containers: ML containers for CPU-based workloads and ML containers for GPU-based workloads.

Dependencies Included

To begin with, all ML containers will include the following set of dependency extras:

- Tensorflow
- Pytorch
- Transformers

This is mostly based on experience working with various users and which containers are most popular. Over time, we could expand this list or add new containers to satisfy different dependency needs (e.g. an XgBoost container).

GPU Dependencies

To make GPU dependencies work well, we will install the appropriate Nvidia drivers, following the path outlined here: https://cloud.google.com/dataflow/docs/gpu/use-gpus#install-python. GPU images will also have a vLLM extra installed (alongside the other ML dependencies mentioned above).

Because Apache licensing does not allow publishing containers with dependencies necessary to publish GPUs, we will rely on Google-published containers with the appropriate dependencies and will clearly document that these do not have the same licensing guarantees.

Container Naming

Following the naming convention of distroless containers, we will create new repositories for each new ML container (as opposed to using tagging to specify ML variants). The containers added will have the following naming conventions:

- apache/beam_python3.X_sdk_ml
- apache/beam_python3.X_sdk_gpu

Specifying ML containers

To specify an ml container, users will be able to specify `--sdk_container_image=apache/beam_python3.X_sdk_ml`. However, to simplify the UX, we will also support specifying `--sdk_container_image=ml` or `--sdk_container_image=gpu`, and Beam will automatically resolve the container image to the corresponding image. We will also extend a similar capability for distroless containers (`--sdk_container_image=distroless`).