Proposal for GSoC 2024

Push-based Metrics Collection for Katib

About Me

- Name: Shao Wang
- Email: shaowang@sjtu.edu.cn
- Phone: (+86) 13505873867
- Github: Electronic-Waste
- Education:
    - 2020.09 ~ 2024.06: B.E. Software Engineering, Shanhai Jiao Tong University
    - 2024.09 ~ 2027.03: M.Sc. Computer Science, Shanghai Jiao Tong University
- Professional Experience:
    - 2024.01 ~ now: Three-month experience developing a scalable, high performance and cloud native storage system to support financial data analysis business, as an infrastructure engineer intern in BondiTech Ltd.
    - 2023.06 ~ 2023.12: Accomplish project “Large-scale Heterogeneous Network Emulation Based on Kubernetes” individually, as an RA in Shanghai Jiao Tong University Software Architecture and Infrastructure Lab.
    - 2023.04 ~ 2023.06: Develop a naive k8s-like container orchestration framework with classmates.
- Open Source Contribution
    - tensorchord/envd
    - kubeflow/katib
- Know more about me in my resume.

About Project

- Project Name: Push-based Metrics Collection for Katib
- Project Description: Push-based Metrics Collection for Katib

Proposal

Background

Katib is a Kubernetes-native project for automated machine learning (AutoML). It can not only tune hyperparameters of applications written in any language and natively supports many ML frameworks, but also supports features like early stopping and neural architecture search.

In the procedure of tuning hyperparameters, Metrics Collector, which is implemented as a sidecar container attached to each training container, will collect training logs from Trials once the training is complete. Then, the Metrics Collector will parse training logs to get appropriate metrics like accuracy or loss and pass the evaluation results to the HyperParameter tuning algorithm.

However, current implementation of Metrics Collector is pull-based, raising some design problems such as determining the frequency we scrape the metrics, performance issues like the overhead caused by too many sidecar containers, and restrictions on developing environments which must support sidecar containers. Thus, we should implement a new API for Katib Python SDK to offer users a push-based way to store metrics directly into the Kaitb DB and resolve those issues raised by pull-based metrics collection.

Goals and Non-Goals

The goals of the project during official coding period:

- An interface in Python SDK for users to push metrics to katib-db directly.
- Several demos of push-based metrics collection.
- E2e tests for push-based metrics collection.

The non-goals include:

- Support pushing data to different types of storage system(prometheus, self-defined interface etc.)
- Optimize Katib CI for push-based metrics collection.

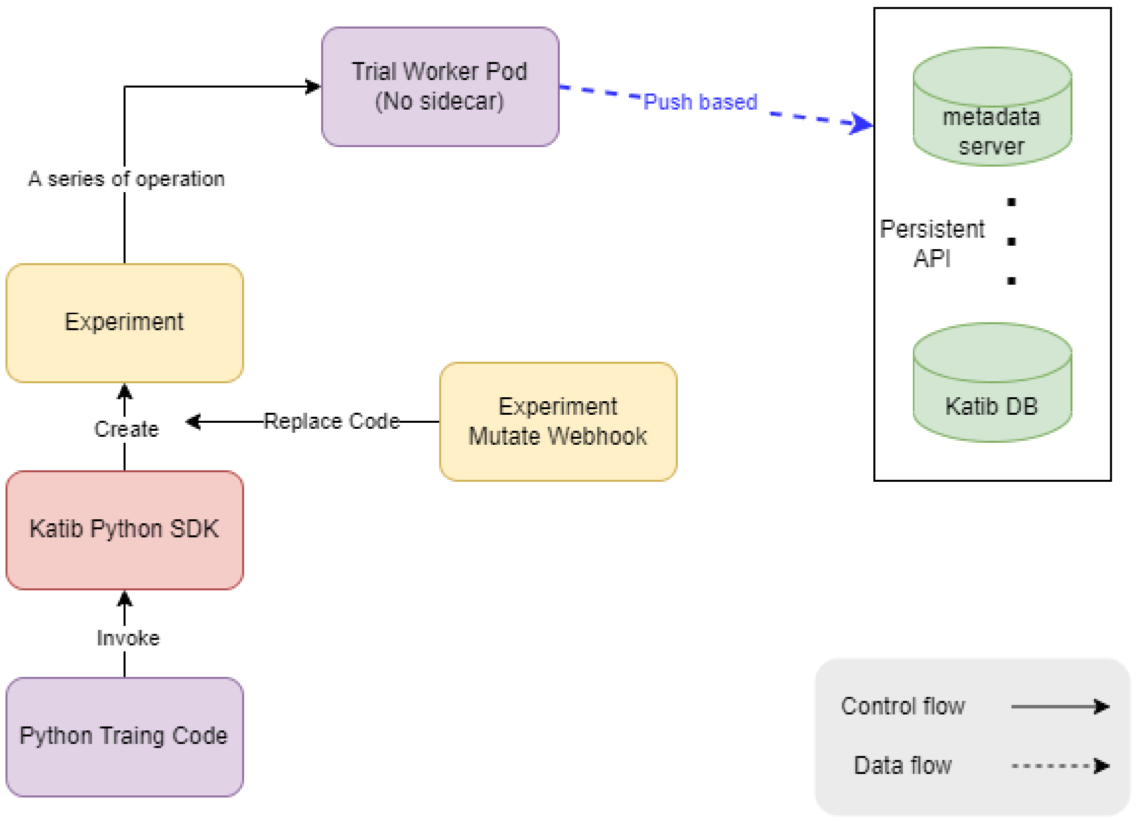
Design

Synopsis

The project aims at providing an Python SDK API interface for users to push metrics to Katib DB directly. In the meantime, the new version of Python SDK should be compatible with code written in the former version. As a result, we plan to add a string parameter `metric_collection_mechanism` (choosing value from `pull` or `push`, default `pull`) to function `tune` in order to decide the usage of sidecar containers.

The workflow of push-based metrics collection

In the common workflow based on Katib SDK, we initialize `KatibClient` and invoke `tune` or `create_experiement` function to create experiments and deploy them to a Kubernetes cluster. After the api server receives an Experiment object, the mutating webhook of Experiment will be invoked to replace the metrics output code with the push-based metrics collection code (i.e. make a grpc call to the Persistent API to store metrics). Then, the Experiment will be created, thus applying Trial Worker Pods without metrics collection sidecar containers to the Kubernetes cluster. When the training task is completed, the training script executed in the Trial Worker Pod, of which metrics output code was replaced with push-based metrics code, will push appropriate metrics into the Katib DB.



Implementation

Add new parameter in `tune`

As is mentioned above, we decided to add `metrics_collection_mechanism` to the `tune` function in Python SDK. Also, we have some changes to be made:

1. Disable injection: set `katib.kubeflow.org/metrics-collector-injection` to `disabled` when the push-based way of metrics collection is adopted so as to disable the injection of the metrics collection sidecar container.
2. Configure the way of metrics collection: set the configuration specifying the way of metrics collection `spec.metricsCollectionSpec.collector.kind` to `NoneCollector`.

Code Injection in Webhook

We decided to implement a code replacing function in Experiment Mutating Webhook. The code replacing function will recognize the metrics output lines (e.g. print, log.Info, e.t.c.) and replace them with push-based metrics collection code which will be discussed in the next section. It's a better decision compared with offering users a `katib_client.push`-like interface, for that users can't use a yaml file to define this operation.

Push-based metrics collection code

The push-based metrics collection code is a function making a grpc call to the persistent API to store training metrics. It will be injected to container args in the Experiment Mutating Webhook and then be called inside the Trial Worker Pod to push metrics to Katib DB.

Schedule

Date	Work
May 1st - 26th	Learn about Kubeflow community in the Community Bonding, dive into the architecture and the workflow of Katib, set up a development environment for coding and debugging.
May 27th - June 2nd	Implement "Add new parameter in `tune`" part
June 3rd - 9th	Implement "Code injection in Webhook" part
June 10th - 16th	Implement "Code injection in Webhook" part
June 17th - 23rd	Implement "Code injection in Webhook" part
June 24th - 30th	Implement "Code injection in Webhook" part
July 1st - 7th	Test, debug and review / Write a document of evaluation
July 8th - 14th	Implement "Push-based metrics collection code" part
July 15th - 21st	Implement "Push-based metrics collection code" part
July 22th - 28th	Implement "Push-based metrics collection code" part
July 29th - August 4th	Implement several demos of push-based metrics collection / Write a document of evaluation
August 5th - 11th	E2e tests for push-based metrics collection
August 12th - 18th	Test, debug and review
August 19th - 26th	Write a summary throughout the project and a document of evaluation

Extra Information

Working Time

I'll be based in Shanghai, China during the summer. Therefore, I'll work in the GMT +8 timezone. During the official coding period, I'll spend at least 25 hours a week on the work.

Reasons for Participation

As a fanatical computer system fan, I have great enthusiasm for fields like cloud infrastructure and AI infrastructure. I got to know about Kubeflow with the introduction of Ce Gao, a.k.a gaocegege in github, also my senior in the school and lab, who told me that AI Infrastructure is the future.

After having dipped into the Kubeflow community, I'm thrilled with the idea of AutoML and cloud native platform for AI workload. And I hold a firm belief that it's the just thing I want to do in my future career. What's more, anticipating a well-known open source community and making some contributions are of great benefits to my coding ability and engineering ability, which may also promote and flourish the community.

As a result, regardless of getting chosen or not, I'm willing to keep contributing to Kaitb and other projects under Kubeflow.