Student research project

Automated experiment manager for machine learning pipeline

Machine learning can be found at the heart of recent research in almost all fields, e.g. data analysis, advanced control, perception, etc. Machine learning experimentation often involves testing multiple model configuraions, hyperparameter combinations, and repeated runs, shown in Fig. 1. However, this process can be error-prone and, sometimes, intractable without proper automation. This project aims to build a modular and extensible automated experiment manager, primarily implemented in Python and JAX by utilizing existing open-source tools and libraries that integrate well with JAX and Equinox. It should support configuration handling, batch launching, logging, and checkpointing, making it easier for researchers to track and reproduce results. The primary use cases should be for learning-based control, e.g. differentiable predictive control (DPC) and reinforcement learning (RL), as well as system identification. Rather than building an entire framework from scratch, the goal is to assemble and script around existing libraries-such as Sacred, Weights & Biases (WandB), TensorBoard, MLflow, etc. to enable robust experimentation workflows with minimal overhead.

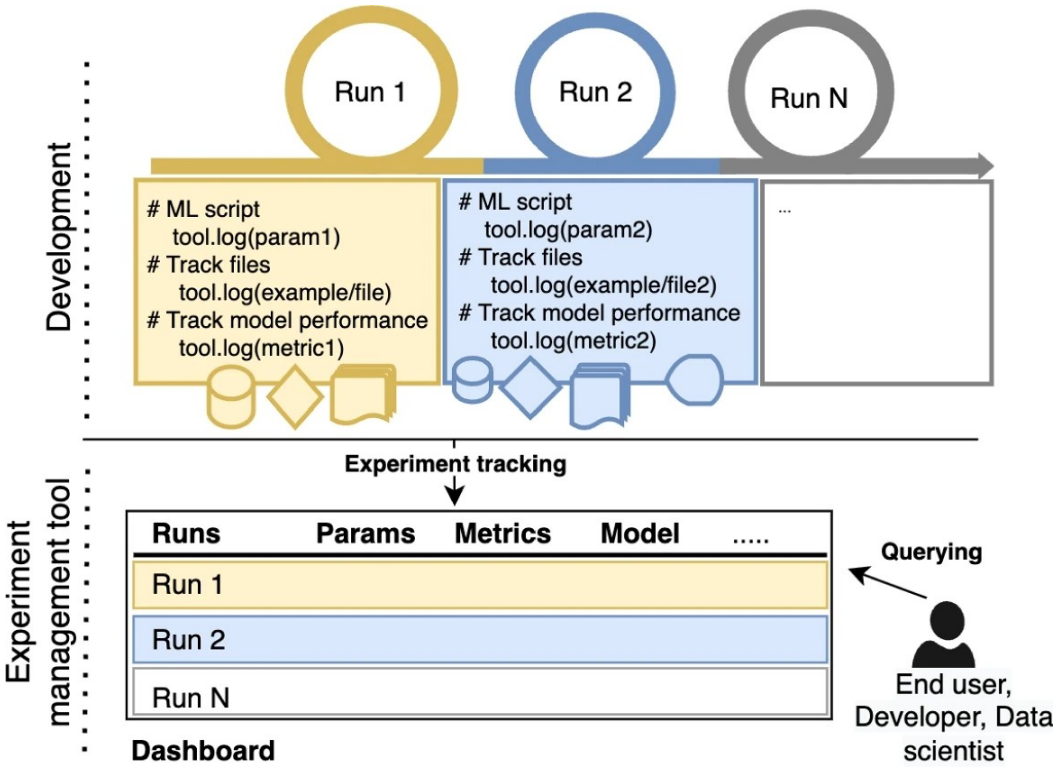

Figure 1: Experiments workflow utilizing experiment management tools. [1]

Necessary requirements:

- Proficiency in Python scripting and CLI tools
- Familiarity with configuration formats like YAML or JSON
- Basic knowledge in machine learning workflow
- Willingness to work with JAX and Equinox (prior experience helpful but not required)

WP 1: Exploration of existing tools and workflow planning **[3 weeks]**
Tools like Hydra, Sacred, WandB, MLflow, etc., will be explored and compared, focusing on their compatibility with JAX and Equinox. One will then define the scope and choose tools for configurarion management, logging, and experiments tracking. A minimum setup will be decided and documented.

WP 2: Implementation of configurable experiment runner **[2 weeks]**
This phase involves implementing a runner script that dynamically loads experiment configurarions (YAML/JSON), initializes training parameters, and launches experiments. The system should support JAX and Equinox implementation and be modular to plug in different use-cases (e.g. DPC and RL).

WP 3: Logging, output structuring, and results archiving **[2 weeks]**
Here, one integrates real-time logging for training/evaluation metrics such as losses, accuracies, rewards, etc., and organize them in structured folders. It should also support saving metadata about experiments.

WP 4: Checkpointing and experiment resumption **[2 weeks]**
This stage includes implementing experiment checkpointing and resumption using native JAX/Equinox saving mechanisms. For that, the model state, optimizer state, and configuration will be saved such that they can be restored for future use or fine-tuning.

WP 5: Final integration, testing, and documentation **[3 weeks]**
One will finalize the project by integrating all components into a single workflow, testing with a use-case, and writing comprehensive documentation/tutorials for future users. A presentation will be prepared to demonstrate usage.

Gantt chart

The planned timetable is shown in the Gantt diagram below.

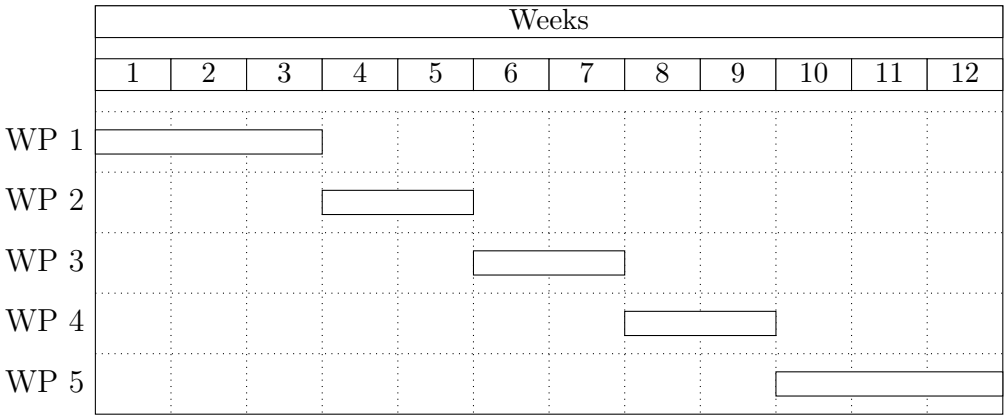

Figure 2: Gantt chart for the project.

References

[1] S. Idowu, O. Osman, D. Strüber, and T. Berger, "Machine learning experiment management tools: A mixed-methods empirical study," *Empirical Software Engineering*, vol. 29, no. 4, p. 74, 2024.