Background

- Inspired by the “Machine Learning Landscape of Top Taggers Paper”
  - They compiled and compared multiple top tagging models
- A significant amount of time is required to organize and evaluate large benchmarks
- Goal of the ROB is to provide a platform that **automates the collection, execution and comparison of submissions** of participants in the benchmark
How the ROB works

1. A workflow is defined by the coordinator of the benchmark along with input data
2. The users provide code that satisfy the steps in the workflow
3. The ROB back-end processes the workflow using the code provided by the user, and evaluates the metrics set by the coordinator
4. The Front end displays the results
Example Workflow: Top Tagging

- Coordinator provides: input data
- Coordinator provides 2 steps for submission:
  - Preprocessing
  - Tagger
- Coordinator Provides: Result evaluation task, which is displayed in the front end

User provides code to preprocess the input data, and a top tagging model which is used to evaluate the data
Current Problem:

- ROB currently requires participants to package their submissions into docker containers
  - Steep learning curve for people unfamiliar
Solution

- In order to increase ease of use, we implement support for the commonly used Jupyter Notebooks
  - We allow users to submit their implementations directly as Jupyter Notebooks
How to run a Jupyter Notebook?

- Jupyter Notebook executions are cell by cell
- Cannot be run like a python script

Issues we needed to solve:

- How to define parameters within a jupyter notebook
- How to run the same jupyter notebook with new parameters without needing to open and edit the notebook itself
- How to run the Jupyter Notebook within the ROB without the need to open the python notebook using the Jupyter Notebook Editor
Papermill!

- Papermill is a convenient way to run Jupyter Notebooks
  - It is a tool that parametrizes, and executes notebooks in its entirety
How Papermill Works

● Users are required to define one cell in their Jupyter Notebook with the tag “parameters”

```python
In [1]:
parameters
# This cell is tagged `parameters`
1 alpha = 0.1
2 ratio = 0.1
```

● Users need to ensure that the entire notebook runs with no errors (just like any python script)
● Ensure that all parameters that can or might be changed in the future is included in the parameters cell
How Papermill Works Continued.

- Users can then execute the notebook using the papermill “execute notebook function”
  - Inputs are:
    - Path to the input
    - Path to the output notebook
    - Dictionary of parameters
- Will execute your notebook similar to how scripts can be executed
- Will generate an output notebook at the path of the output notebook, with an output that is exactly like if you pressed “run all cells” in the Jupyter Notebooks
- Parameter values in the notebook will be replaced by the values defined in the function
Implementing Papermill Into ROB

● Create a new **Step Type** that uses papermill to execute the notebook
  ○ Current Step Types: ContainerStep, FunctionStep
  ○ Added Step Type: NotebookStep
  ○ This is the function in the ROB that takes defined workflow inputs and user inputs, and runs the functions
  ○ Done by implementing a subclass of the WorkflowStep Class

● Create a new “worker” that executes steps of the given context
  ○ Done by implementing a subclass of the Worker Class in ROB
  ○ A function that calls the step and executes the function

● Create another worker that is able to run Jupyter Notebooks contained within docker containers
Hello World Demo

Inputs:

- Data: list of names
- Jupyter Notebook that reads the list of names and writes greetings to an output file

Output: List of greetings

- List of greetings is then analyzed by counting average characters per line and max output length, which is then put in a json file as the final “output”
How to run demo:

- Create ROB User
- Create Hello World Workflow
- Start the Hello World Demo Run
- Submit python notebook and data in the command line
- Retrieve Results

Detailed functions/explanations at:
https://github.com/scailfin/rob-demo-hello-world
Jet Flavor Demo

- Implementation of the Jet Flavor Dataset into the ROB
- Inspired by the “Jet Flavor Classification in High-Energy Physics with Deep Neural Networks” paper
- Lots of combinations of features that need to be compared
- Hard to compile all the models and train all of them individually
- Implement into the ROB as a workflow to make comparison and training on the same validation set easier and more organized
Workflow

User inputs:
- Model architectures
- Preprocessing script
- Training script

Workflow Coordinator inputs:
- Training and validation data
- Validation script
Summary

- Implemented jupyter notebooks into the ROB as a form of input
- Created demos using Jupyter Notebooks for use of the ROB for
  - Quickdraw dataset
  - Jet Flavor Dataset
  - Top Tagging dataset
  - Mnist Dataset

Future Work:
- Presenting at the ML4Jets conference on 6/7
- Fix flask web server for browser input into ROB
Questions?