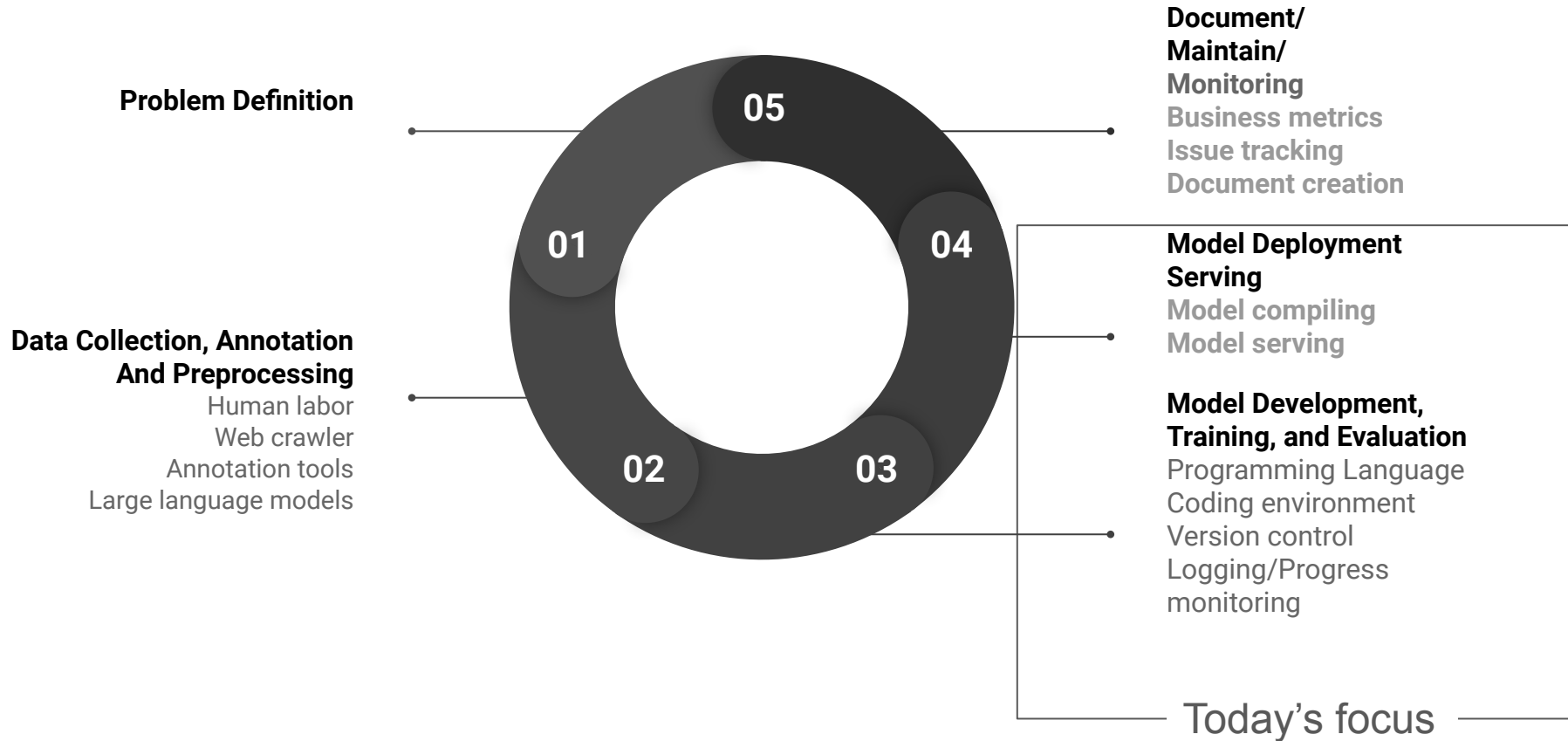


Tools For Machine Learning

An overview

Yujia Yan
ECE 208/408

Typical Machine Learning System Life Cycle



Using the command line interface

- Instead of using Graphical interface, one often resort to command line interfaces when needed
 - Operating systems provide a set of tools that is more powerful than GUI tools
 - For development, CLI interfaces are easier to be programmed and maintained
- Shell language: POSIX compatible systems (Linux, MacOS, Linux, etc) include their own implementation of shell languages that are similar
- Cheat sheet: <https://github.com/RehanSaeed/Bash-Cheat-Sheet>
- Basic usage: `command argument1 argument2`
 - `find . -type f`
- Pipeline: `command1 | command2 | command3...`
 - `pgrep -f "train.py" | xargs kill`

```
find . -type f
```

```
./maestro-v1.0.0-midi/maestro-v1.0.0/2013/ORIG-MIDI 02 7 7 13 Group MID--AUDIO 19 R1 2013 wav--1.midi
./maestro-v1.0.0-midi/maestro-v1.0.0/2013/ORIG-MIDI 02 7 10 13 Group MID--AUDIO 11 R3 2013 wav--3.midi
./maestro-v1.0.0-midi/maestro-v1.0.0/2013/ORIG-MIDI 01 7 10 13 Group MID--AUDIO 08 R3 2013 wav--3.midi
./maestro-v1.0.0-midi/maestro-v1.0.0/2013/ORIG-MIDI 03 7 10 13 Group MID--AUDIO 18 R3 2013 wav--3.midi
./maestro-v1.0.0-midi/maestro-v1.0.0/2013/ORIG-MIDI 02 7 6 13 Group MID--AUDIO 06 R1 2013 wav--3.midi
./maestro-v1.0.0-midi/maestro-v1.0.0/2013/ORIG-MIDI 03 7 6 13 Group MID--AUDIO 10 R1 2013 wav--4.midi
./maestro-v1.0.0-midi/maestro-v1.0.0/2013/ORIG-MIDI 01 7 8 13 Group MID--AUDIO 07 R2 2013 wav--1.midi
./maestro-v1.0.0-midi/maestro-v1.0.0/2013/ORIG-MIDI 01 7 6 13 Group MID--AUDIO 04 R1 2013 wav--3.midi
./maestro-v1.0.0-midi/maestro-v1.0.0/2013/ORIG-MIDI 02 7 10 13 Group MID--AUDIO 12 R3 2013 wav--4.midi
./maestro-v1.0.0-midi/maestro-v1.0.0/2013/ORIG-MIDI 02 7 10 13 Group MID--AUDIO 11 R3 2013 wav--1.midi
./maestro-v1.0.0-midi/maestro-v1.0.0/2013/ORIG-MIDI 03 7 8 13 Group MID--AUDIO 19 R2 2013 wav--4.midi
./maestro-v1.0.0-midi/maestro-v1.0.0/2013/ORIG-MIDI 01 7 7 13 Group MID--AUDIO 11 R1 2013 wav--4.midi
./maestro-v1.0.0-midi/maestro-v1.0.0/2013/ORIG-MIDI 01 7 6 13 Group MID--AUDIO 02 R1 2013 wav--2.midi
./maestro-v1.0.0-midi/maestro-v1.0.0/2013/ORIG-MIDI 01 7 8 13 Group MID--AUDIO 02 R2 2013 wav--5.midi
./maestro-v1.0.0-midi/maestro-v1.0.0/2013/ORIG-MIDI 03 7 6 13 Group MID--AUDIO 09 R1 2013 wav--1.midi
./maestro-v1.0.0-midi/maestro-v1.0.0/2013/ORIG-MIDI 02 7 7 13 Group MID--AUDIO 17 R1 2013 wav--1.midi
./maestro-v1.0.0-midi/maestro-v1.0.0/2013/ORIG-MIDI 02 7 6 13 Group MID--AUDIO 08 R1 2013 wav--3.midi
./maestro-v1.0.0-midi/maestro-v1.0.0/2013/ORIG-MIDI 02 7 8 13 Group MID--AUDIO 08 R2 2013 wav--2.midi
./maestro-v1.0.0-midi/maestro-v1.0.0/2013/ORIG-MIDI 01 7 6 13 Group MID--AUDIO 03 R1 2013 wav--4.midi
./maestro-v1.0.0-midi/maestro-v1.0.0/2013/ORIG-MIDI 01 7 7 13 Group MID--AUDIO 12 R1 2013 wav--4.midi
./maestro-v1.0.0-midi/maestro-v1.0.0/2013/ORIG-MIDI 02 7 6 13 Group MID--AUDIO 05 R1 2013 wav--3.midi
./maestro-v1.0.0-midi/maestro-v1.0.0/2013/ORIG-MIDI 02 7 6 13 Group MID--AUDIO 08 R1 2013 wav--2.midi
./maestro-v1.0.0-midi/maestro-v1.0.0/2013/ORIG-MIDI 03 7 8 13 Group MID--AUDIO 19 R2 2013 wav--3.midi
./maestro-v1.0.0-midi/maestro-v1.0.0/2013/ORIG-MIDI 02 7 7 13 Group MID--AUDIO 15 R1 2013 wav--1.midi
./maestro-v1.0.0-midi/maestro-v1.0.0/2013/ORIG-MIDI 02 7 7 13 Group MID--AUDIO 18 R1 2013 wav--5.midi
./maestro-v1.0.0-midi/maestro-v1.0.0/2013/ORIG-MIDI 02 7 7 13 Group MID--AUDIO 17 R1 2013 wav--4.midi
./maestro-v1.0.0-midi/maestro-v1.0.0/2013/ORIG-MIDI 02 7 6 13 Group MID--AUDIO 08 R1 2013 wav--1.midi
./maestro-v1.0.0-midi/maestro-v1.0.0/2013/ORIG-MIDI 01 7 7 13 Group MID--AUDIO 11 R1 2013 wav--3.midi
./maestro-v1.0.0-midi/maestro-v1.0.0/2013/ORIG-MIDI 02 7 7 13 Group MID--AUDIO 17 R1 2013 wav--2.midi
```

Find all pids for process matches keyword "train.py" and then kill them all:

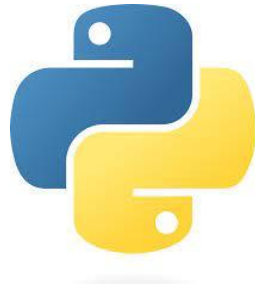
```
pgrep -f "train.py" | xargs kill
```

(?) How about

```
Find . -type f -name '*.py' | xargs rm
```

(Don't try if you do not understand)

Intro to Python



Python programming language

- The current dominant programming language for machine learning
- Dynamically typed and garbage collected
- Advantage:
 - Usually used as a **glue language** for calling various libraries written in more “low level” language, e.g., C++.
 - Flexible and suitable for fast experimenting
 - Large community base
 - Easy to install and manage packages via the built-in package manager, i.e., PIP
- Disadvantages
 - Slow performance
 - Currently, threads can not be executed on multiple cores in parallel: still more than 5 years to go to remove the global interpreter lock (GIL).
 - Weak and optional typing system makes it prone to bugs easily avoidable by languages like C++.
- The official tutorial: <https://docs.python.org/3/tutorial/index.html>

Basic Python Code Structure

Package Import

```
1 import math
2
3 def compute_circle_area(r):
4     """
5     Returns the area of a cycle
6
7     Parameter r: the radius of a circle
8     """
9     return math.pi* r**2
10
11
12 # for loop over 0,1,2,3,4
13 for i in range(5):
14     print(compute_circle_area(i))
15
```

Function declaration

Indentation to indicate
the code block level

recommended to use 4
spaces (PEP 8) or 2
spaces(GOOGLE),
equivalent of {} in
C/C++ like language

Multi-line comment

Single line comment

Iterator Style For loop:
Iterate over elements in
a iterable object

range(5) is generator

Basic Python Code Structure: Classes

Reference to the
invoking instance
of the class

Constructor

Names like
“__init__” are
special names
provided by
python.

```
18 class Rectagular:
19     def __init__(self, x, y, width, height):
20         self.x = x
21         self.y = y
22         self.width = width
23         self.height = height
24
25     def area(self):
26         return self.width*self.height
27
28 <
29 rect = Rectagular(0,0, 3,4)
30 print("the area is ", rect.area())
```

‘__init__’ called

Python Control flows: if

```
>>> x = int(input("Please enter an integer: "))
Please enter an integer: 42
>>> if x < 0:
...     x = 0
...     print('Negative changed to zero')
... elif x == 0:
...     print('Zero')
... elif x == 1:
...     print('Single')
... else:
...     print('More')
...
More
```

<https://docs.python.org/3.10/tutorial/controlflow.html>

Python Control flows: for loop

```
>>> # Measure some strings:
... words = ['cat', 'window', 'defenestrate']
>>> for w in words:
...     print(w, len(w))
...
cat 3
window 6
defenestrate 12
```

<https://docs.python.org/3.10/tutorial/controlflow.html>

Python Control flows: while loop

```
>>> # Fibonacci series:  
... # the sum of two elements defines the next  
... a, b = 0, 1  
>>> while a < 10:  
...     print(a)  
...     a, b = b, a+b  
...  
0  
1  
1  
2  
3  
5  
8
```

<https://docs.python.org/3.10/tutorial/introduction.html#first-steps-towards-programming>

Python: built-in types

- Basic types: bool (`True`, `False`), int (`0`, `1`, `2`), float (`3.5`, `4.6`), complex (`5+4j`), string(`"hi"`)
- List: `l = [1,2,3,4,5]`:
 - `len(l)` # 5
 - `l[2]`
 - `l.append(7)`
 - `l[:2]+ l[-2:]` # `[1,2,4,5]`
- Set: `s = ['apple', 'pear', 'orange']`
 - `len(s)` # 3
 - `'apple' in s` # `True`
 - `s.add('grape')`
- Dict: `d={'apple': 1, 'pear': 2, 'orange': 3}`
 - `d['pear']` # 2
 - `d['grape'] = 7` # `['apple': 1, 'pear': 2, 'orange': 3, 'grape':7]`
- Other types: <https://docs.python.org/3/library/stdtypes.html>

Python: typing hint

Type annotations can be added **to enable static type checker**, (more types are in package *typing*).

```
type Vector = list[float]

def scale(scalar: float, vector: Vector) -> Vector:
    return [scalar * num for num in vector]

# passes type checking; a list of floats qualifies as a Vector.
new_vector = scale(2.0, [1.0, -4.2, 5.4])
```

No error when executed:

```
1 a: list[float] = []
2 a.append("hi")
```

But with a static checker, e.g. mypy:

```
yujia@CCat:~/Programming/Untitled Folder$ mypy typeExample.py
typeExample.py:2: error: Argument 1 to "append" of "list" has incompatible type "str"; expected "float" [arg-type]
Found 1 error in 1 file (checked 1 source file)
```

Resources for learning python

- The official tutorial:
 - <https://docs.python.org/3/tutorial/index.html>
- W3C learning by examples:
 - <https://www.w3schools.com/python/>
- Coursera python courses
 - <https://www.coursera.org/search?query=python&>
- The Hitchhiker's Guide to Python
 - <https://docs.python-guide.org/>
- Style Guide
 - PEP 8: <https://peps.python.org/pep-0008/>
 - GOOGLE: <https://google.github.io/styleguide/pyguide.html>

Development Environment

Package manager

- Linux/OSX package managers
 - Depending on specific linux distributions: APT/DPKG (Debian, Ubuntu, Mint), YUM/RPM(Redhat/Fedora), Pacman(Arch), Zypper(openSUSE), Portage (Gentoo)
 - Homebrew(MacOS)
- PIP
 - Python package management tool
 - Packages can be built into *wheels* and published on <https://pypi.org/> and then the package can be directly installed via
 - `pip3 install PACKAGENAME`
 - See <https://packaging.python.org/en/latest/flow/> on how to package and distribute your python project.
- Anaconda
 - A proprietary package management tool that provides an easy-to-use integration of virtual environment and package management.
 - One can create several environment installed with different set of packages under different versions.
 - Not only for python packages
 - Can be used together with PIP in the conda virtual environment
 - Cheatsheet:
https://docs.conda.io/projects/conda/en/4.6.0/_downloads/52a95608c49671267e40c689e0bc00ca/conda-cheatsheet.pdf

When you need to install something, first try to install via your package manager.

Development environments

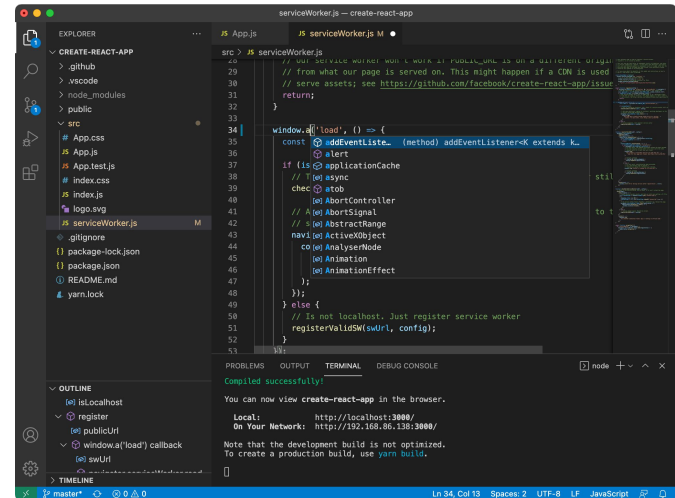
- VIM

- Simple, fast and reliable
- Available anywhere for linux based system
- Powerful plugin system
- High learning curve

- Visual studio code

- Powerful plugin system
- Integrated feature for performing development on a remote machine
- Integrated version controlling

```
1 import os
2 import sys
3
4 import torch
5 import torch.nn as nn
6 import torch.nn.functional as F
7
8 import CopyDataGenerator
9 import RNN
10
11 os.
12 abc          m module abc
13 abort        f def abort()
14 nS access    f def access(path: _FdOrAnyPath, mode: int, dir_fd: Optional[int]=
15 nT add_dll_directory f def add_dll_directory(path: str)
16 altsep       s altsep: Optional[str]()
17 nS chdir     f def chdir(path: _FdOrAnyPath)
18 em chflags   f def chflags(path: AnyPath, flags: int, follow_symlinks: bool=...
19 hi chmod     f def chmod(path: _FdOrAnyPath, mode: int, dir_fd: Optional[int]=.
20 chown        f def chown(path: _FdOrAnyPath, uid: int, gid: int, dir_fd: Option
copy l        f def chroot(path: AnyPath)
-- INSERT --
```



Jupyter Notebook

- Web based Interactive notebook
- Good for simple experiments and documenting stuffs that require some coding example/ outputs from running.
- Run locally or remote from a server
- Frequently seen in code repos as the format of examples

[3]: `print("hello")`

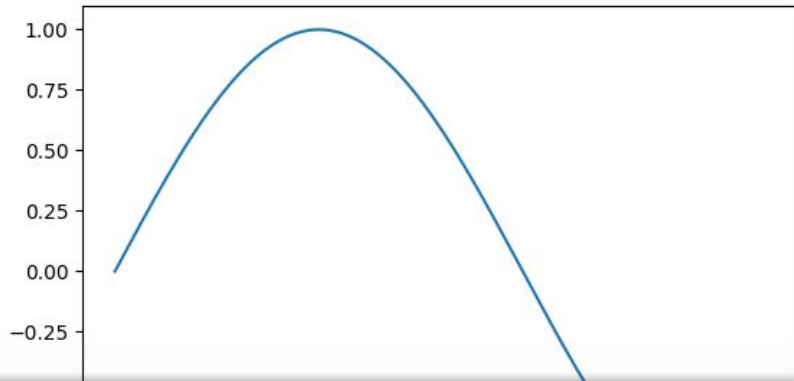
hello

Let's plot

$$y = \sin(x)$$

[2]: `import matplotlib.pyplot as plt`
`import torch`
`plt.plot(torch.sin(torch.linspace(0, 5, 1000)))`

[2]: [`matplotlib.lines.Line2D at 0x7f0d0a6dbd00`>]



[]:

Version Control

- Keep track of changes of your codes
- A necessary for doing real programming
- A lot of options: **Git**/Mercurial/SVN
- Git: the most popular among individuals,
 - Good resource to learn
 - Tutorial videos: <https://git-scm.com/videos>
 - Visualized learning platform: <https://learngitbranching.js.org/>
 - Code hosting
 - github, bitbucket, gitlab, etc.

LLM based Tools

- E.g., ChatGPT/Copilot
- Example usage
 - Help with boring tasks
 - Generate Unit tests
 - Help write Commit Messages/ documents/Reports
- Use with caution
 - Fake information
 - Misunderstanding
 - Potential (?) plagiarism

 You

Can you generate a chirp signal with sampling rate F_s that spans T seconds using `pytorch`?

 You


Show me how to use `rsync` to synchronize my local folder with a remote folder



ChatGPT

Sure! `rsync` is a powerful command-line utility for synchronizing files and directories between two locations. To use `rsync` to synchronize your local folder with a remote folder, you typically use it with the `ssh` command for secure remote synchronization. Here's a basic example:

```
bash
```

 Copy code

```
rsync -avz -e ssh /path/to/local/folder/ username@remote-server:/path/t
```

```
return t, chirp_signal
```

Python Package and Frameworks

Scientific computing libraries

Numpy: numerical computation library

Scipy: scientific computing library built on numpy, which covers python counterparts to Matlab functionalities: integration, optimization, fft, signal processing etc.

Scikit-learn: A (traditional) Machine learning library which allows you to perform non-deep-learning based regression, classification, etc.

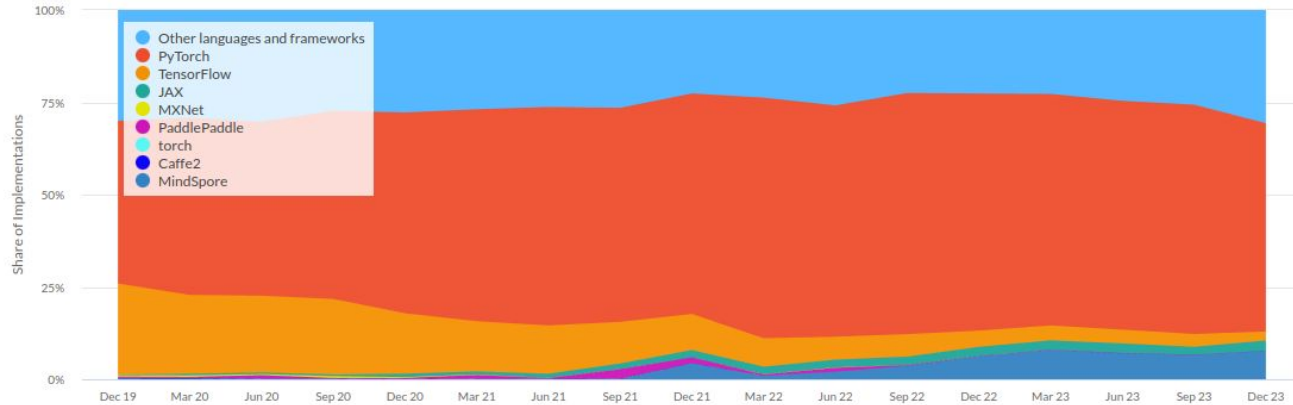
Matplotlib/Seaborn: data visualization

You will get familiar with these tools throughout the progress of programming assignment

Deep learning frameworks

- What they are doing?
 - Numerical Algebra
 - automatic differentiation (aka. backpropagation)
 - Modules for implementing commonly used deep learning practices
- Current dominance in research: **Pytorch**

Paper Implementations grouped by framework



From: <https://paperswithcode.com/trends>

Repository Creation Date

Popular Deep learning frameworks

- Pytorch (Meta/ Pytorch Foundation)
 - Dynamic Computational Graph
 - Largest amount of Machine Learning project code repos
 - Just-in-time compiler
 - TorchScript Language for higher performance and generic deployment
- JAX (Google)
 - Similar to Pytorch
 - Created by original authors of the opensource *Autograd* package for numpy, but this time it is sponsored and maintained by Google
 - Most advanced Automatic differentiation features (fast high-order derivatives, automatic vectorization based on function transforms)
 - Have better support on Google's TPU as a Google product
- Tensorflow (Google)
 - One of the early frameworks
 - Static computational graph: declare the whole computational process and then feed in the data
 - Current version supports both static and dynamic computational graph, e.g., GradientTape
 - Have better support on Google's TPU as a Google product
- MindSpore (Huawei)
 - More similar to tensorflow: support both static/dynamic computational graphs
 - Out of box training utilities, Automatic Parallelized Training

 PyTorch



TensorFlow



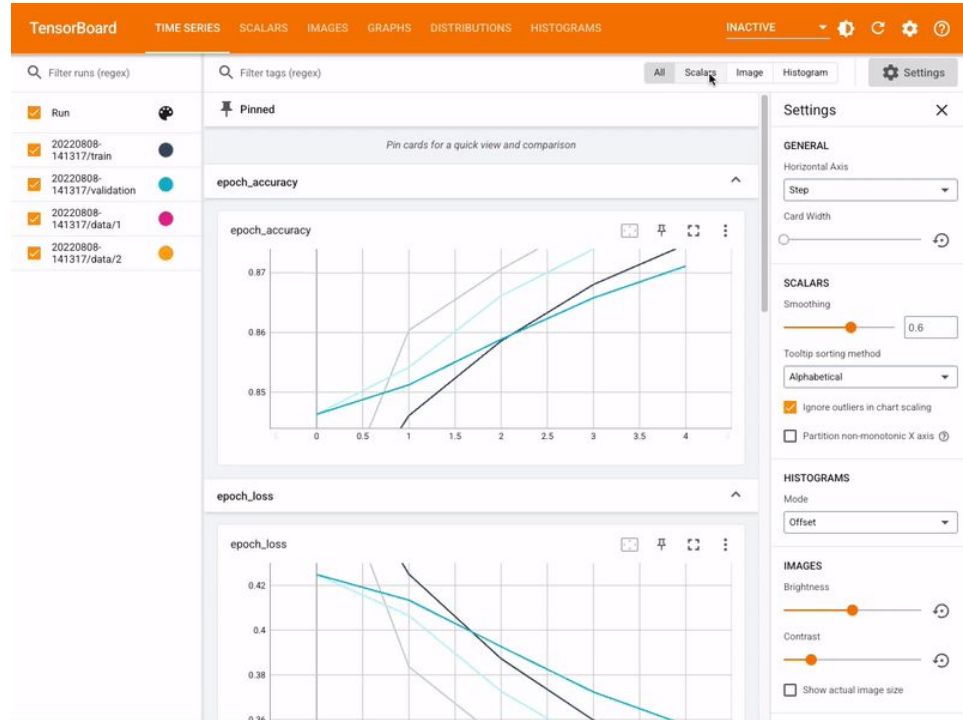
MindSpore

Higher level training frameworks for DL frameworks

- Deep learning frameworks typically require you to write the actual logics for training, which typically include:
 - data loading, model initialization/restoration, looping over the dataset, backpropagating and updating the model at certain point, parallelization, and logging some metrics during training.
- Instead of writing training logics from scratch, one can use some frameworks that directly provides certain abstractions:
 - **Pytorch Lightning for pytorch**
 - **Keras for tensorflow**

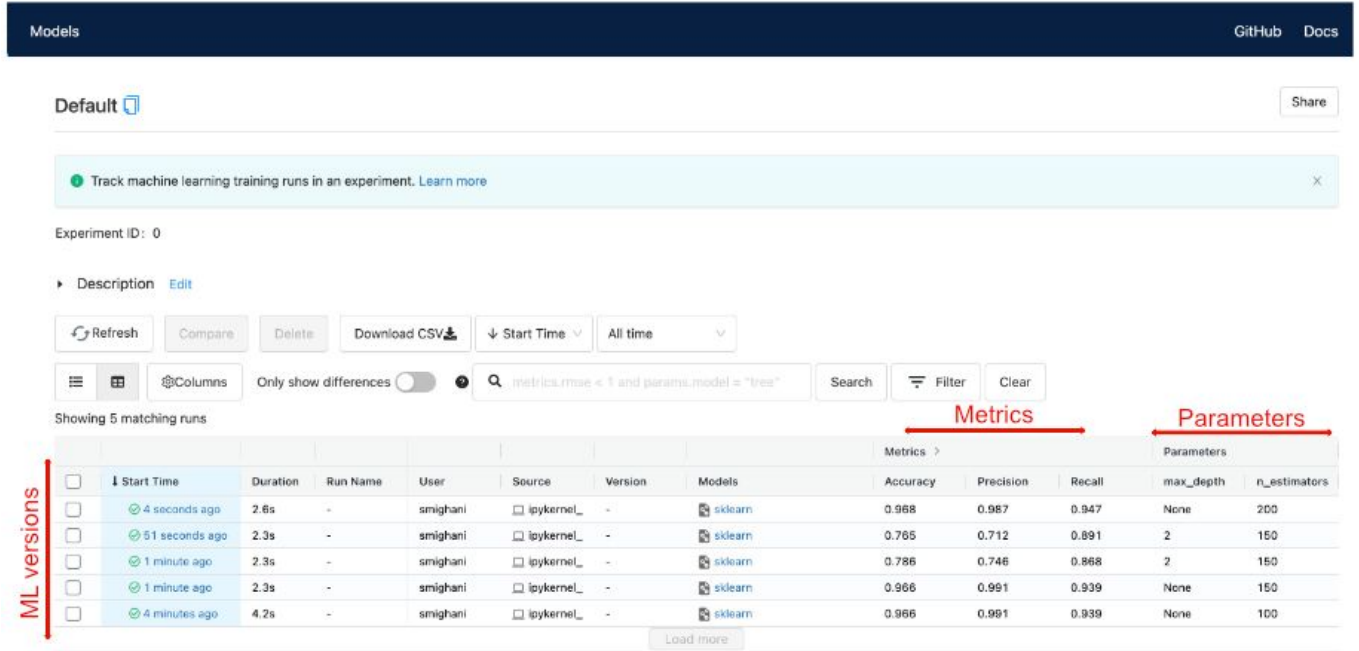
Training Logging

- Keep track of any metrics (loss, accuracy, etc.) during training
- E.g. Tensorboard



Model Versioning

- Keep track of your codes together with trained models
- E.g., MLFlow



The screenshot displays the MLFlow Models web interface. At the top, there's a dark blue header with 'Models' on the left and 'GitHub Docs' on the right. Below the header, there's a 'Default' view selector and a 'Share' button. A light blue banner contains the text 'Track machine learning training runs in an experiment. Learn more'. The main content area shows 'Experiment ID: 0' and a 'Description' section with an 'Edit' link. Below this are several action buttons: 'Refresh', 'Compare', 'Delete', 'Download CSV', 'Start Time' (dropdown), and 'All time' (dropdown). A search bar contains the query 'metrics.rmse < 1 and params.model = "tree"'. To the right of the search bar are 'Filter' and 'Clear' buttons. The table below shows 'Showing 5 matching runs'. The table has columns for 'Start Time', 'Duration', 'Run Name', 'User', 'Source', 'Version', 'Models', 'Accuracy', 'Precision', 'Recall', 'max_depth', and 'n_estimators'. A red vertical label 'ML versions' is on the left side of the table. Red arrows point to the 'Metrics' and 'Parameters' columns. A 'Load more' button is at the bottom right of the table.

	Start Time	Duration	Run Name	User	Source	Version	Models	Accuracy	Precision	Recall	max_depth	n_estimators
<input type="checkbox"/>	4 seconds ago	2.6s	-	smighani	ipykernel_	-	sklearn	0.968	0.967	0.947	None	200
<input type="checkbox"/>	51 seconds ago	2.3s	-	smighani	ipykernel_	-	sklearn	0.765	0.712	0.891	2	150
<input type="checkbox"/>	1 minute ago	2.3s	-	smighani	ipykernel_	-	sklearn	0.786	0.746	0.868	2	150
<input type="checkbox"/>	1 minute ago	2.3s	-	smighani	ipykernel_	-	sklearn	0.966	0.991	0.939	None	150
<input type="checkbox"/>	4 minutes ago	4.2s	-	smighani	ipykernel_	-	sklearn	0.966	0.991	0.939	None	100

<https://towardsdatascience.com/mlflow-a-primer-6dfe6be48353>

Model Deployment

- Generic Deployment: let your model run on different devices
 - **ONNX**: Open Neural Network Exchange
 - The open standard for representing the machine learning model
 - Can be exported from a variety of Deep learning frameworks
 - Generic tools for deployment, targeting e.g., mobile/DSP board/MicroController

Summary of supported Execution Providers

CPU	GPU	IoT/Edge/Mobile	Other
Default CPU	NVIDIA CUDA	Intel OpenVINO	Rockchip NPU (preview)
Intel DNNL	NVIDIA TensorRT	ARM Compute Library (preview)	Xilinx Vitis-AI (preview)
TVM (preview)	DirectML	Android Neural Networks API	Huawei CANN (preview)
Intel OpenVINO	AMD MIGraphX	ARM-NN (preview)	AZURE (preview)
XNNPACK	Intel OpenVINO	CoreML (preview)	
	AMD ROCm	TVM (preview)	
	TVM (preview)	Qualcomm QNN	
		XNNPACK	

Model deployment

- Model Serving: let you create API endpoints for calling your model from remote locations
 - TorchServe
 - TensorFlow serving
 - Huggingface
 - And others

```
curl http://127.0.0.1:8080/predictions/densenet161 -T kitten_small.jpg
```

Which will return the following JSON object

```
[
  {
    "tiger_cat": 0.46933549642562866
  },
  {
    "tabby": 0.4633878469467163
  },
  {
    "Egyptian_cat": 0.06456148624420166
  },
  {
    "lynx": 0.0012828214094042778
  },
  {
    "plastic_bag": 0.00023323034110944718
  }
]
```