Lecture 10
Accessible Multimodal Learning and Transfer Learning with PyKale

Haiping Lu

YouTube Playlist: https://www.youtube.com/c/HaipingLu/playlists

COM4059/6059: MLAI21@The University of Sheffield

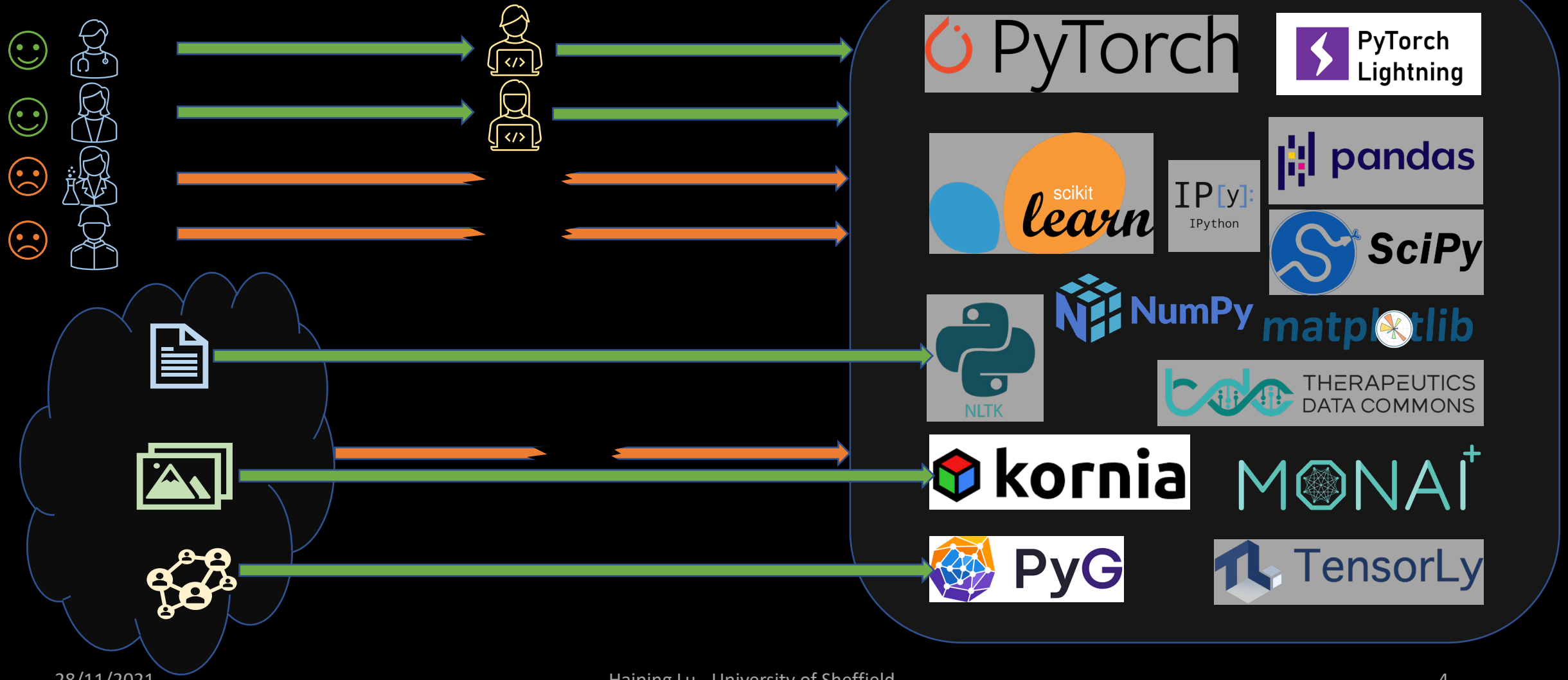
Week 10 Contents / Objectives

- Why PyKale
- How PyKale was Built
- What PyKale Looks Like
- Summary and Review

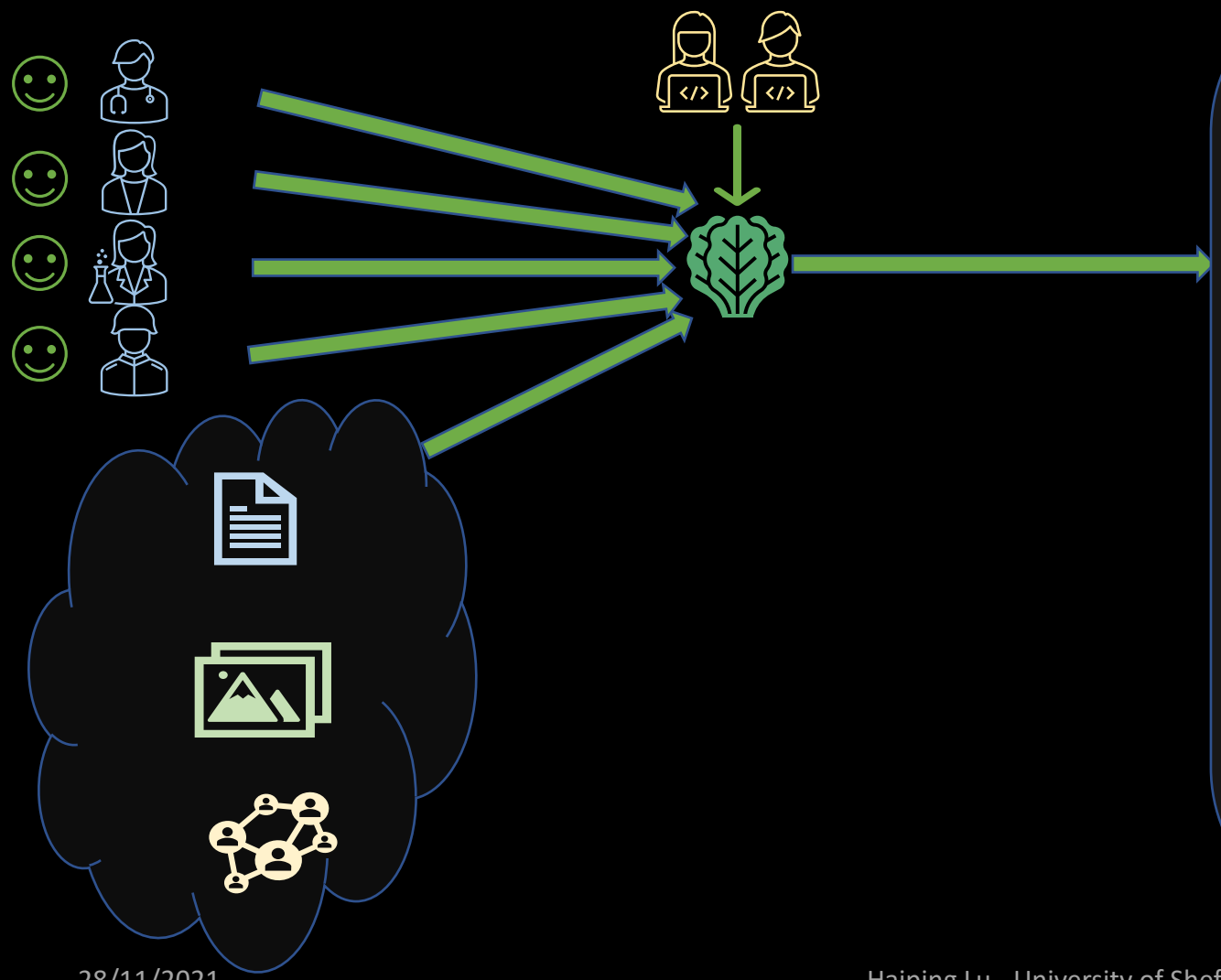
Week 10 Contents / Objectives

- **Why PyKale**
- How PyKale was Built
- What PyKale Looks Like
- Summary and Review

Accessibility issues in interdisciplinary research



Accessibility solution



PyTorch
PyTorch Lightning
scikit learn
IP[y]: IPython
pandas
SciPy
NumPy
matplotlib
NLTK
THERAPEUTICS DATA COMMONS
kornia
MONAI
PyG
TensorLy

Why build PyKale?

- Accessibility challenge in interdisciplinary research
- Abundant ML software vs unmet user demands

- Low-level ML software → focus on basics & fundamentals
- Single-domain ML software → tailor for single domains, varying APIs

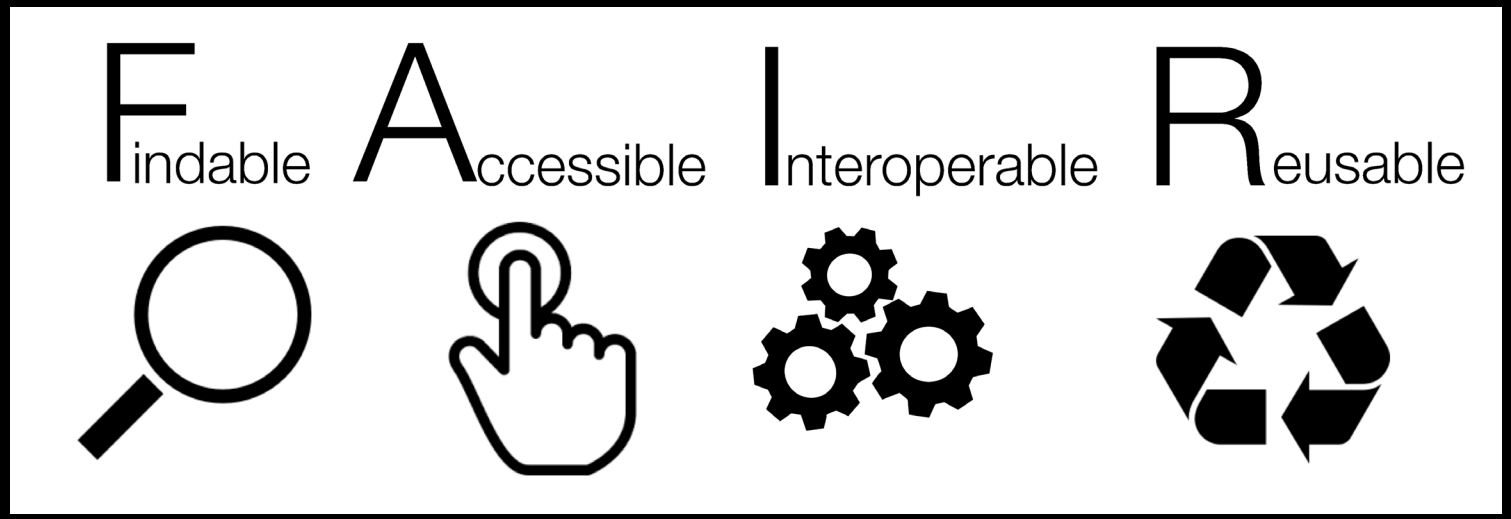
- Our software → high-level, multi-source, unified API, domain-traversing
- Bridge gaps between data, software and end users for accessible, scalable, and sustainable research in machine learning

Week 10 Contents / Objectives

- Why PyKale
- **How PyKale was Built**
- What PyKale Looks Like
- Summary and Review

Is it FAIR?

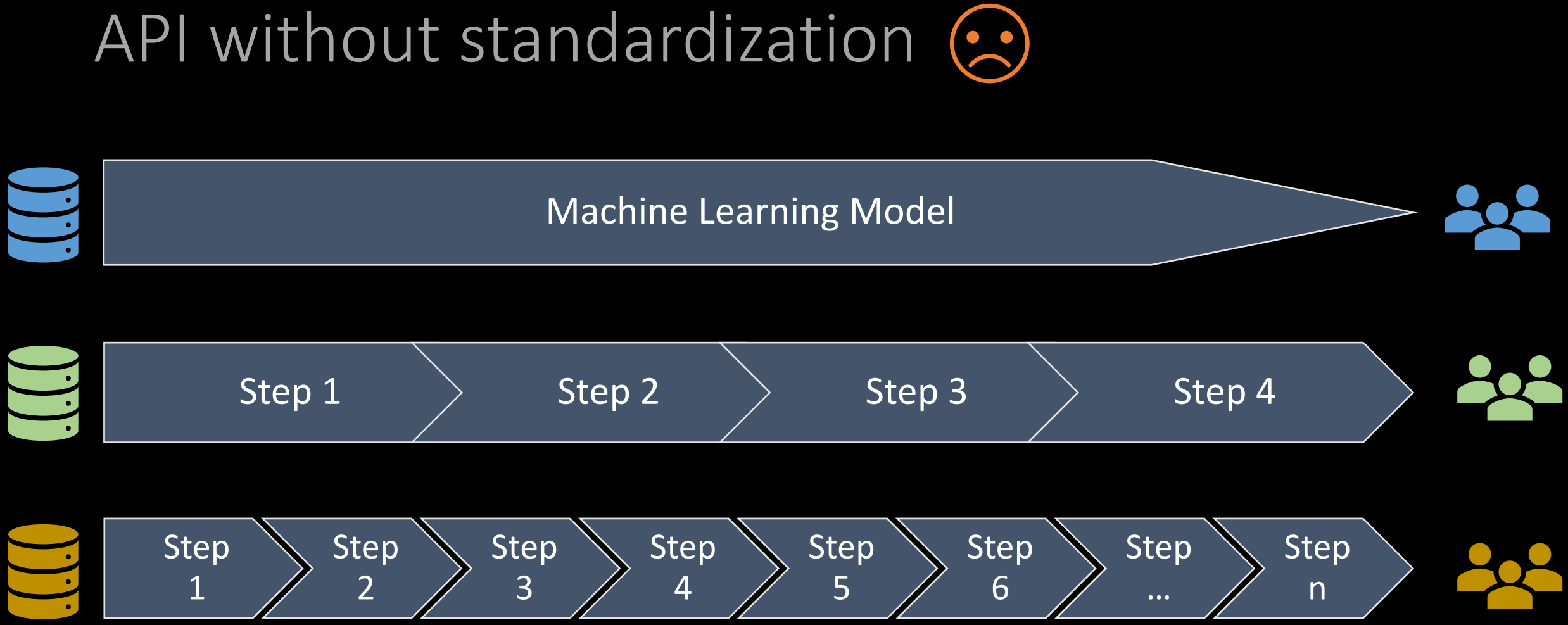

IT'S NOT FAIR
waaaaa-its-not-fair.jpg (560×415) (memegenerator.net)
memegenerator.net


Findable
Accessible
Interoperable
Reusable

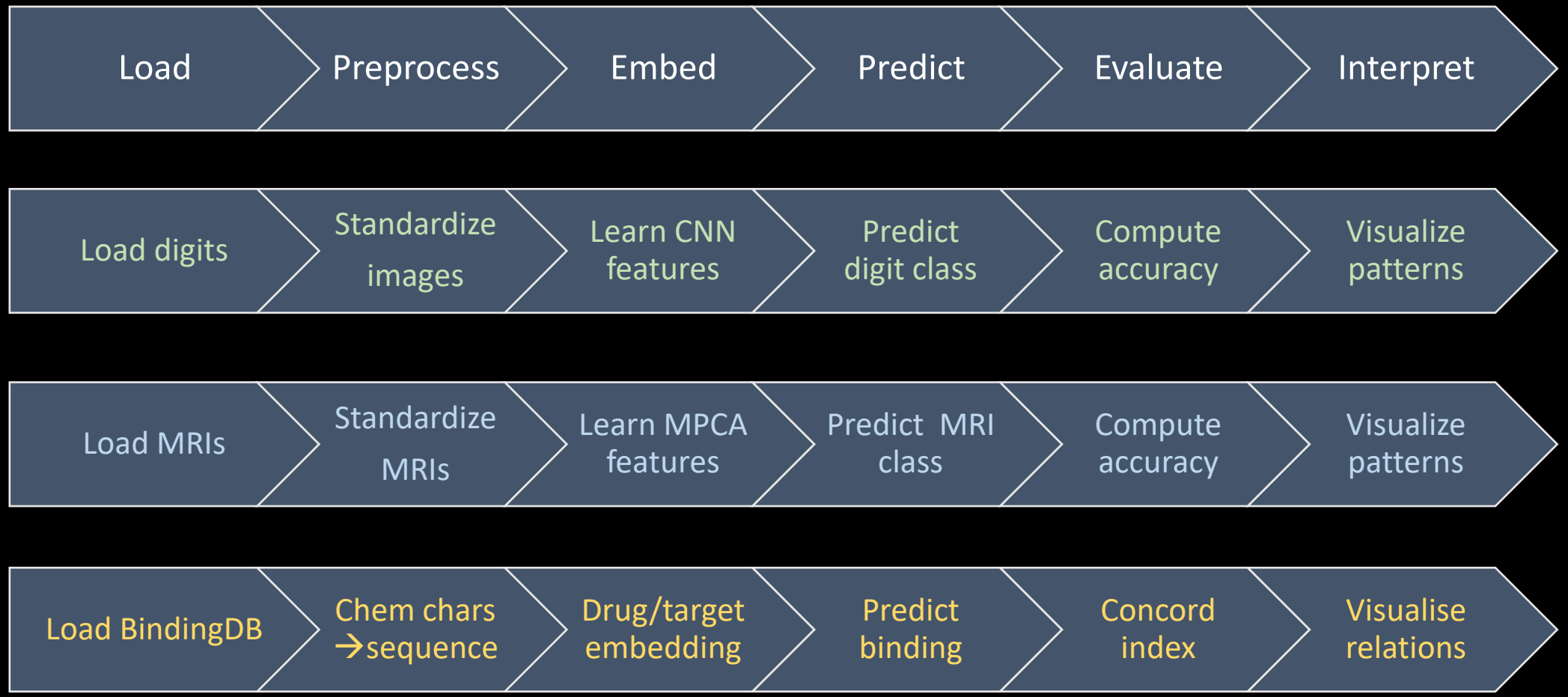
New principles: green machine learning

- Reduce repetition and redundancy
  - Refactor to standardize workflow and enforce styles
  - Identify and remove duplicated functionalities
- Reuse existing resources
  - Reuse the same machine learning pipeline for different data
  - Reuse existing software for available functionalities
- Recycle learning models across areas
  - Identify commonalities between applications
  - Recycle models for App A to App B

API without standardization ☹



Pipeline-based API ☺



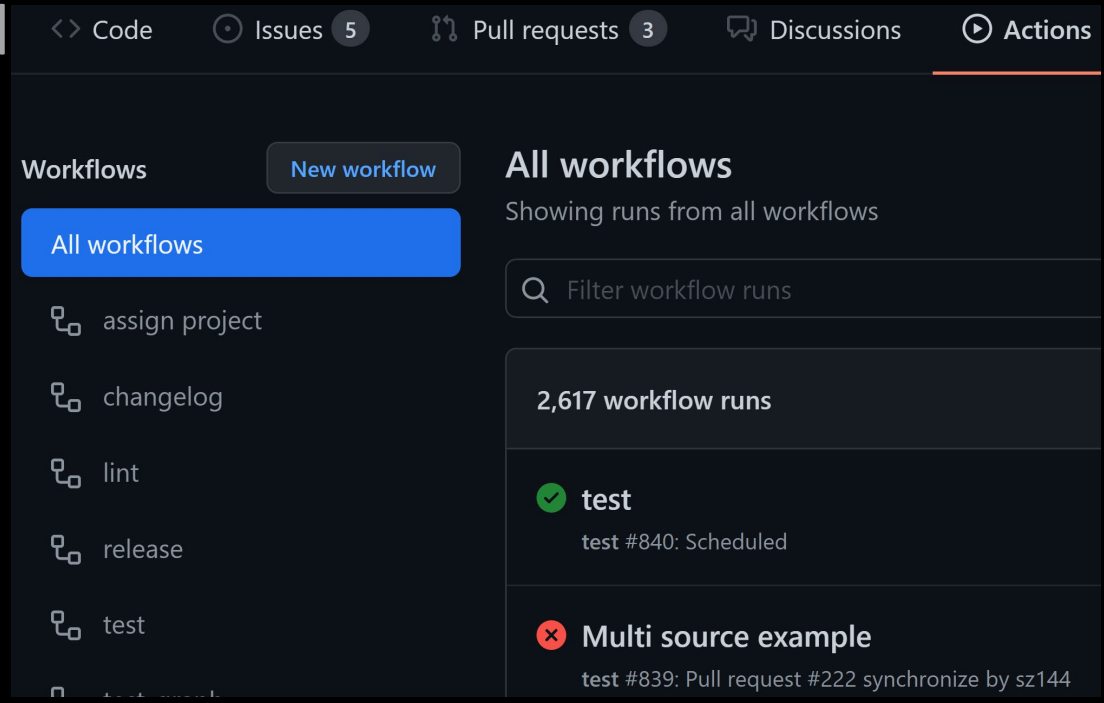
Accessible machine learning

- Bridge the gap: abundant ML software ⇦ ⇨ unmet user demands

- Separate code and configuration: pykale/examples
  - End users (non-experts in ML)
    - Configure via YAML, no need to write Python/ML → Lower entry barriers
  - ML experts
    - Manage config with min coding → Improve reproducibility/efficiency
    - Deliver accessible ML software → Scalable collaboration
      → Broad applications

Sustainable software development

- GitHub ecosystem → industrial standard
  - Automation & continuous integration
  - Read the Docs, PyPI release
  - Linting, Pre-commit, PyTest, Codecov
- Python/PyTorch ecosystem



Week 10 Contents / Objectives

- Why PyKale
- How PyKale was Built
- **What PyKale Looks Like**
- Summary and Review

Kale API snapshot: pipeline-based

main pykale / kale /

Xianyuan

- Load Data
- Preprocess Data
- Embed
- Predict
- Evaluate
- Interpret
- Pipeline
- Utilities

..

embed

evaluate

interpret

loaddata

pipeline

predict

prepdata

utils

main pykale / kale / loaddata /

sz144 fix digits_access

..

README.md

__init__.py

cifar_access.py

dataset_access.py

digits_access.py

get_dicom.py

mnistm.py

multi_domain.py

office_access.py

main pykale / kale / utils /

XianyuanLiu Delete csv_logger.py

..

README.md

__init__.py

download.py

logger.py

print.py

seed.py

main pykale / kale / pipeline /

haipinglu define acronyms

..

README.md

__init__.py

deep_dti.py

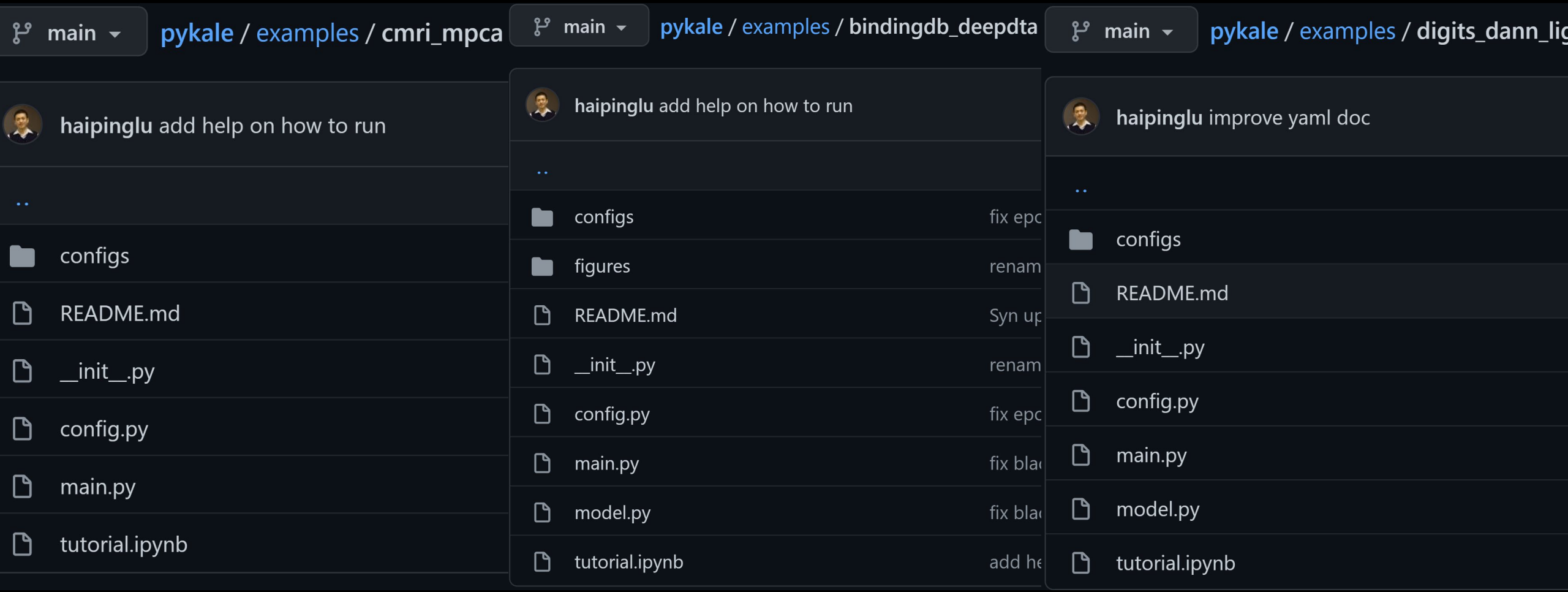
domain_adapter.py

mpca_trainer.py

multi_domain_adapter.py

video_domain_adapter.py

Standardized examples & code config separation



YAML configuration

5 lines (4 sloc) | 49 Bytes

```yaml
DAN:
  METHOD: "DANN"

DATASET:
  SOURCE: "svhn"
```

14 lines (11 sloc) | 155 Bytes

```yaml
DAN:
  METHOD: "CDAN"

DATASET:
  NUM_REPEAT: 1
  SOURCE: "svhn"
  VAL_SPLIT_RATIO: 0.5

SOLVER:
  MIN_EPOCHS: 0
  MAX_EPOCHS: 3

OUTPUT:
  PB_FRESH: None
```

```python
# ------------------------------------------------------------
# Dataset
# ------------------------------------------------------------
_C.DATASET = CN()
_C.DATASET.ROOT = "../data"  # Path to store data and results
_C.DATASET.NAME = "digits"  # Name of datasets
_C.DATASET.SOURCE = "mnist"  # The source dataset name
_C.DATASET.TARGET = "usps"  # The target dataset name
_C.DATASET.NUM_CLASSES = 10
_C.DATASET.NUM_REPEAT = 10
_C.DATASET.DIMENSION = 784
_C.DATASET.WEIGHT_TYPE = "natural"
_C.DATASET.SIZE_TYPE = "source"
_C.DATASET.VAL_SPLIT_RATIO = 0.1
# ------------------------------------------------------------
# Solver
# ------------------------------------------------------------
_C.SOLVER = CN()
_C.SOLVER.SEED = 2020
_C.SOLVER.BASE_LR = 0.001  # Initial learning rate
_C.SOLVER.MOMENTUM = 0.9
_C.SOLVER.WEIGHT_DECAY = 0.0005  # 1e-4
_C.SOLVER.NESTEROV = True

_C.SOLVER.TYPE = "SGD"
_C.SOLVER.MAX_EPOCHS = 120  # "nb_adapt_epochs": 100,
# _C.SOLVER.WARMUP = True
_C.SOLVER.MIN_EPOCHS = 20  # "nb_init_epochs": 20,
_C.SOLVER.TRAIN_BATCH_SIZE = 150  # 150
_C.SOLVER.TEST_BATCH_SIZE = 200  # No difference in ADA
```

Community engagement

https://github.com/pykale/pykale/blob/main/.github/CONTRIBUTING.md

189 lines (126 sloc) | 18.5 KB

Contributing to PyKale

Light involvements (viewers/users) | *Medium involvements (contributors)* | **Heavy involvements (maintainers)**

Ask questions | Report bugs | Suggest improvements | *Branch, fork & pull* | *Coding style* | *Test* | Review & merge | Release & management

PyKale

latest

Search docs

GETTING STARTED

Introduction

Installation

Tutorial

» PyKale Documentation

PyKale Documentation

Getting Started

- Introduction
- Installation

book Tutorials

n using YAML

Code | Issues 4 | Pull requests 1 | Discussions | Actions | Projects 2 | Security | Insights

v0.1.0

Updated 3 days ago

Filter cards | Add cards | Fullscree

2 Overview & backlog

Added by haipinglu

v0.2.0

Manage the progress to v0.2.0

#10 in pykale/pykale updated on 4 May

Roadmap: 0.1.0a1-->0.1.0b1-->0.1.0b3-->0.1.0rc1-->0.1.0rc3-->v0.1.0 to build

8 To do

Remove dependency on csv logger and maybe remove csv logger.

Added by haipinglu

Make pykale.github.io following https://github.com/Project-

Automated as To do | Manage

2 In progress

DA for action recognition TA3N

1 of 5 tasks

#217 opened by XianyuanLiu

work-in-progress

Review required

Automated as In progress

YouTube GB | Search

PyKale

Knowledge-aware machine learning from multiple sources in Python

The University Of Sheffield

0:01 / 11:25

Introduction to PyKale, a library for knowledge-aware machine learning from multi

643 views • Apr 17, 2021

20 | 0 | SHARE

Haiping Lu

74 subscribers

ANALYTICS

A short introduction to PyKale https://github.com/pykale/pykale, a PyTorch library for multimodal/transfer learning with deep learning and dimensionality reduction on graphs, images, texts, and videos. This video was its v0.1.0b. arXiv preprint for v0.1.0rc2 is at

SHOW MORE

arXiv.org > cs > arXiv:2106.09756

Search...

Help | Advance

Computer Science > Machine Learning

[Submitted on 17 Jun 2021]

PyKale: Knowledge-Aware Machine Learning from Multiple Sources in Python

Haiping Lu, Xianyuan Liu, Robert Turner, Peizhen Bai, Raivo E Koot, Shuo Zhou, Mustafa Chasmai, Lawrence Schobs

Code | Issues 4 | Pull requests 1 | Discussions | Actions | Projects 2 | Security | Insights

New | Top: All | Answered | Unanswered | Label | New discussion

Unique Selling Points

bobturneruk asked 11 days ago in General · Unanswered

Should we have tests for tutorial notebooks?

bobturneruk asked on 12 Aug in Q&A · Answered

Where should we publish on PyKale?

bobturneruk asked on 12 Aug in Q&A · Unanswered

https://pykale.readthedocs.io/en/latest/notebooks.html

PyKale
latest

Search docs

GETTING STARTED
Introduction
Installation
Tutorial
Jupyter Notebook Tutorials

» Jupyter Notebook Tutorials

Jupyter Notebook Tutorials

- **Image classification:** Digits domain adaptation
- **Drug discovery:** BindingDB drug-target interaction prediction
- **Medical image analysis:** Cardiac MRI for MPCA-based diagnosis

https://colab.research.google.com/github/pykale/pykale/blob/main/examples/digits_dann_lightn/tutorial.ipynb

tutorial.ipynb

File Edit View Insert Runtime Tools Help

+ Code + Text Copy to Drive

PyKale Tutorial: Domain Adaptation on Digits with Lightning

| Open in Colab (click `Runtime` → `Run all (Ctrl+F9)` | Launch Binder (click `Run` → `Run All Cells`) |

If using Google Colab, a free GPU can be enabled to save time via setting `Runtime` → `Change runtime type`

https://hub.gke2.mybinder.org/user/pykale-pykale-77qiut67/doc/tree/examples/digits_dann_lightn/tutorial.ipynb

tutorial.ipynb

File Edit View Run Kernel Tabs Settings Help

Download GitHub Binder Markdown Python 3 (ipykernel)

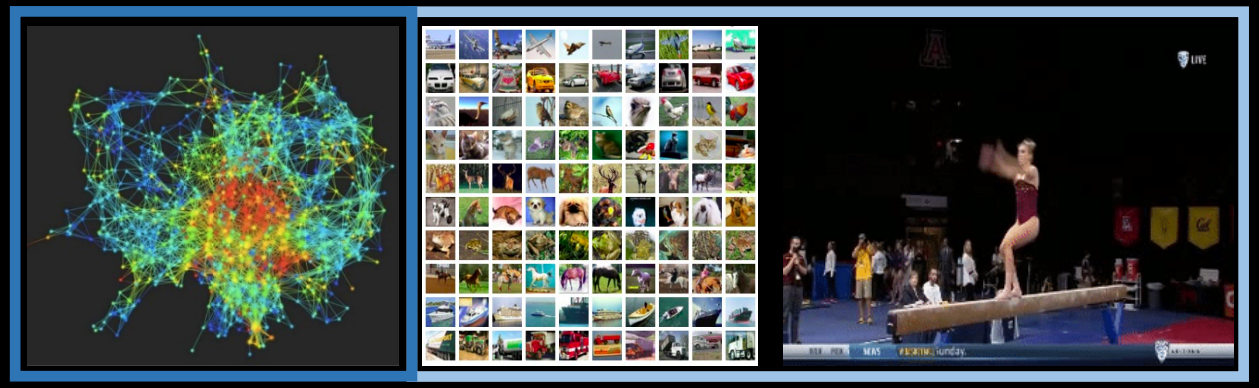
PyKale Tutorial: Domain Adaptation on Digits with Lightning ¶

| Open in Colab (click `Runtime` → `Run all (Ctrl+F9)` | Launch Binder (click `Run` → `Run All Cells`) |

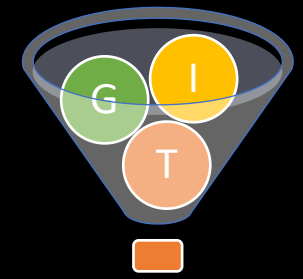
If using Google Colab, a free GPU can be enabled to save time via setting `Runtime` → `Change runtime type` → `Hardware accelerator: GPU`

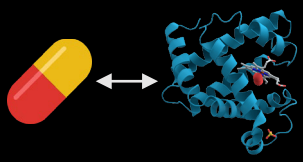
livedemo-compressor.jpg (1450×960) (posterno.com)

Data, model, & app



- Data: graph, image, text, video
- Machine learning models
  - Transfer learning: domain adaptation
  - Multimodal learning: integration of heterogeneous data
  - *Deep learning: CNN, GNN/GCN, transformers*
  - *Dimensionality reduction: tensor-based*
- Example applications
  - Image/video recognition: CIFAR, digits, office, action videos
  - Bioinformatics/graph analysis: BindingDB, knowledge graphs
  - Medical imaging: cardiac MRI, brain fMRI



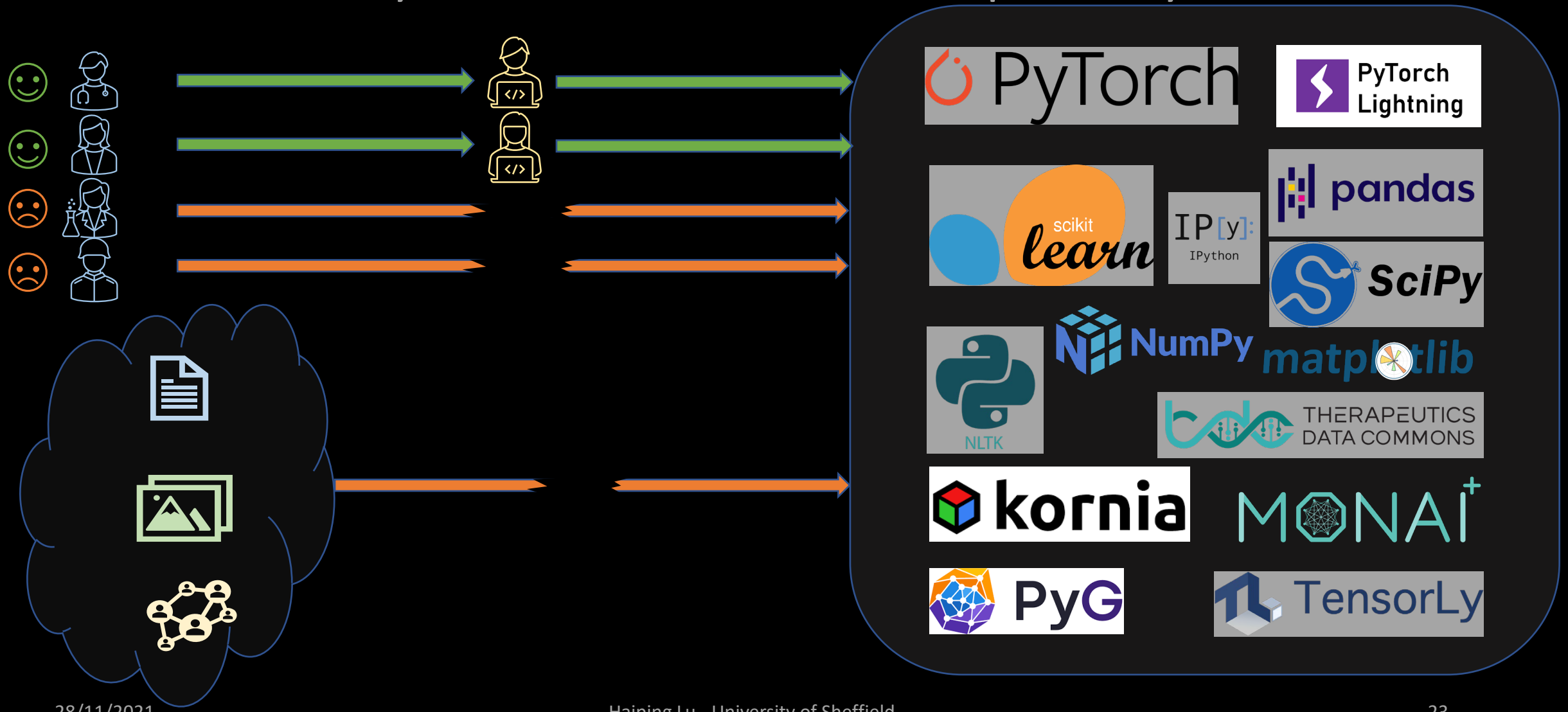


DICOM
Digital Imaging and Communications in Medicine

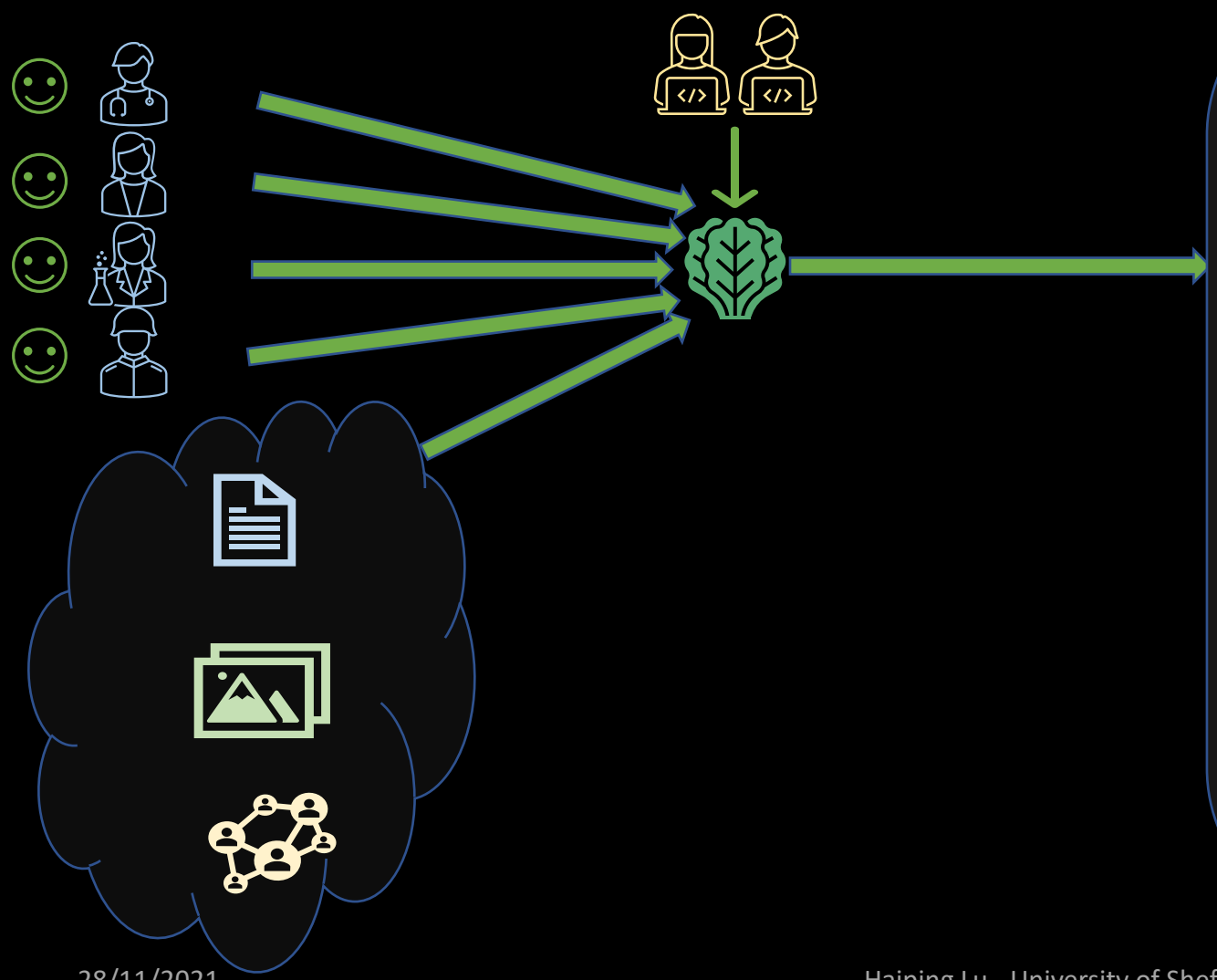
Week 10 Contents / Objectives

- Why PyKale
- How PyKale was Built
- What PyKale Looks Like
- **Summary and Review**

Accessibility issues in interdisciplinary research



PyKale fills the gaps

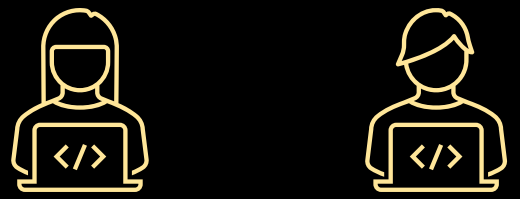


PyKale: accessible, scalable, sustainable

- Green ML principles: deliver FAIR ML software
- Pipeline-based API to unify workflow

- Industrial software engineering practices for sustainability
- Accessible and scalable with code-configuration separation
- Multimodal learning & transfer learning for graphs, images, videos, …

Targeted audience



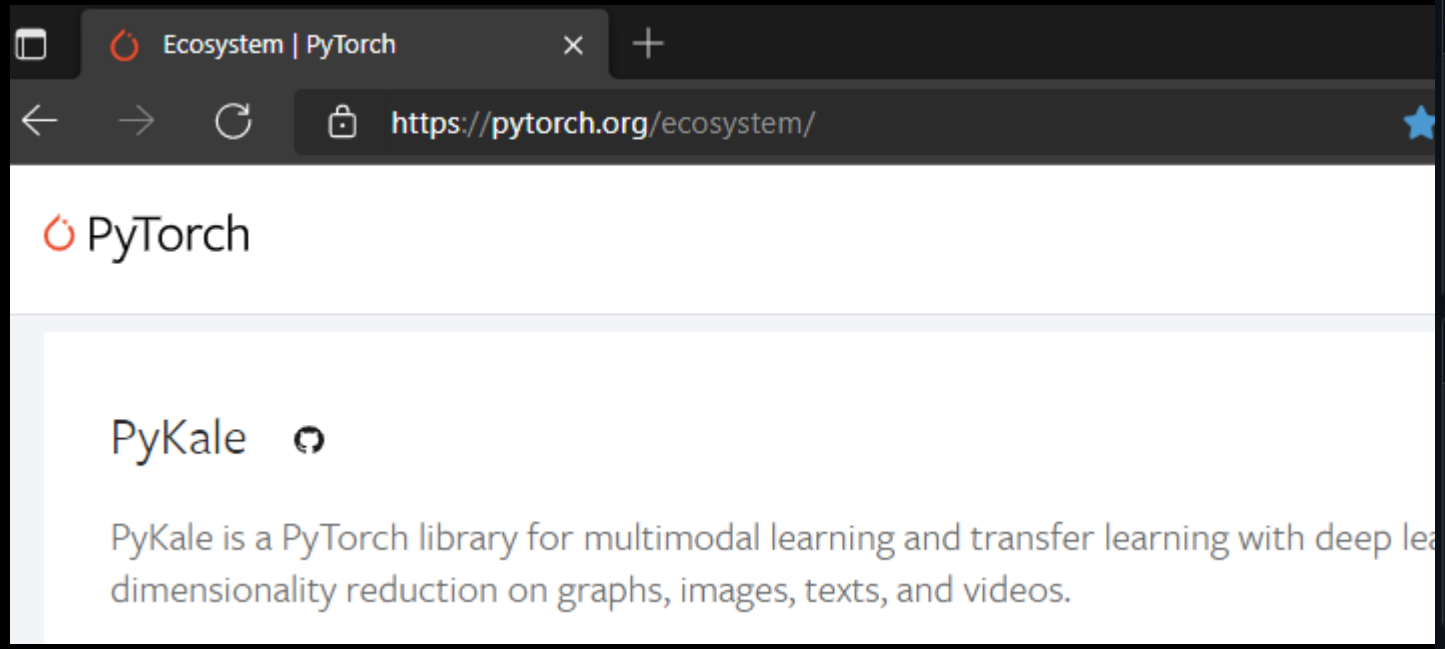
ML experts

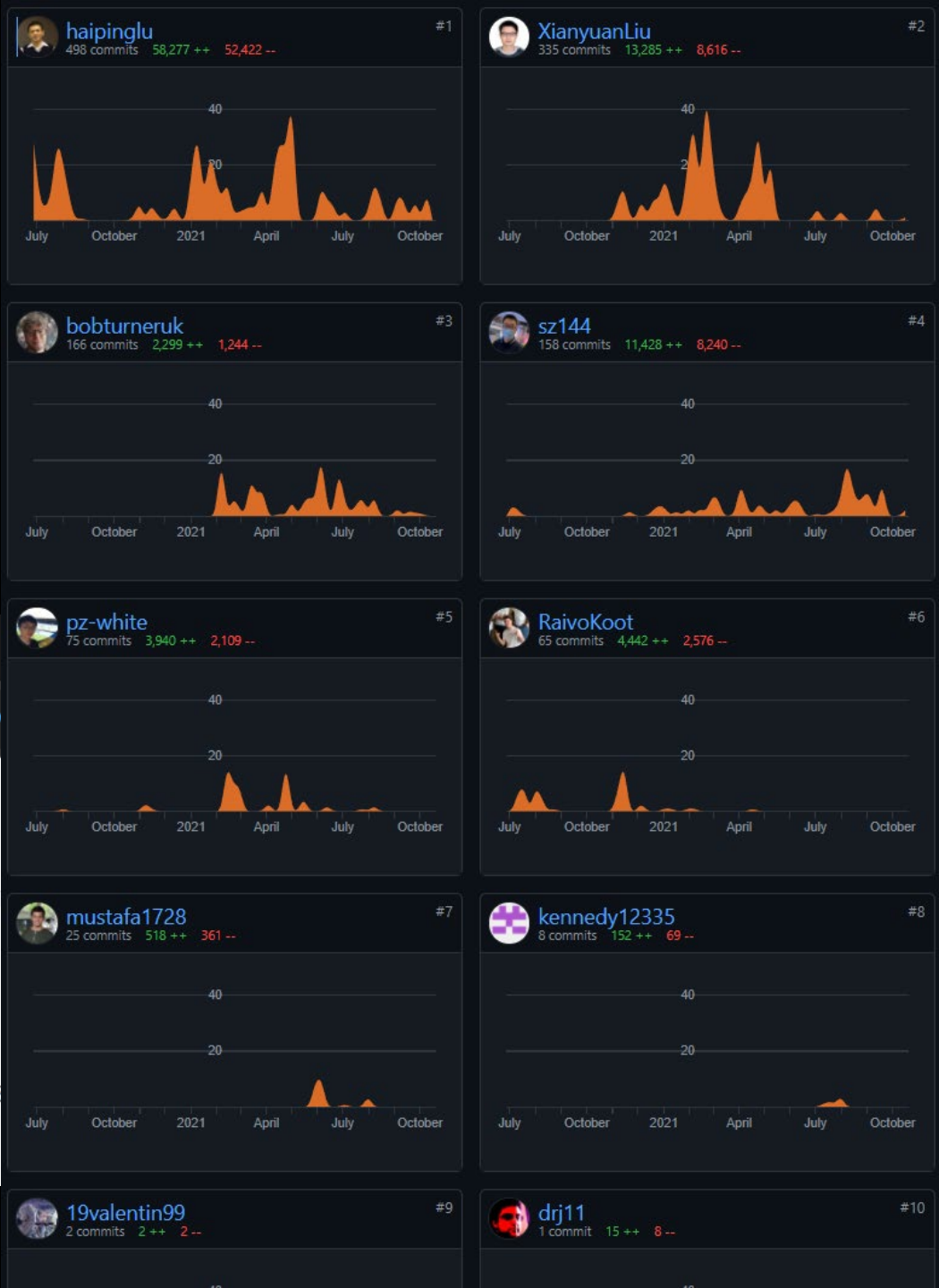
End users

- ☑ Reproducible research
- ☑ Scalable collaboration
- ☑ Cross-boundary models
- ☑ Off-the-shelf ML pipelines
- ☑ Friendly configuration editing
- ☑ Multi modalities under one roof
- ☑ Open with strong community engagement

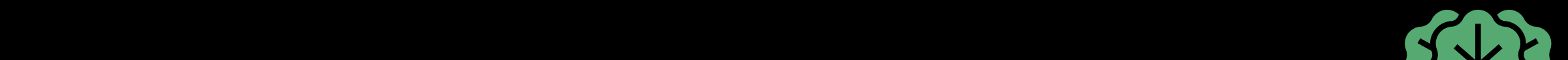
Open & welcome

- Open source on GitHub (MIT License)
  - https://github.com/pykale/pykale
- Welcome feedback/contribution!





Acknowledgement



- Shef groups + community

Machine Learning Sheffield

Research Software Engineering Sheffield

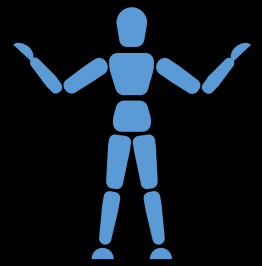
The University Of Sheffield.

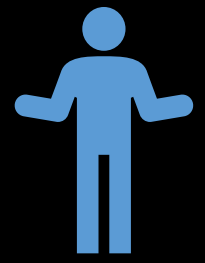
- Partially funded by Wellcome Trust via an Innovator Award

wellcome

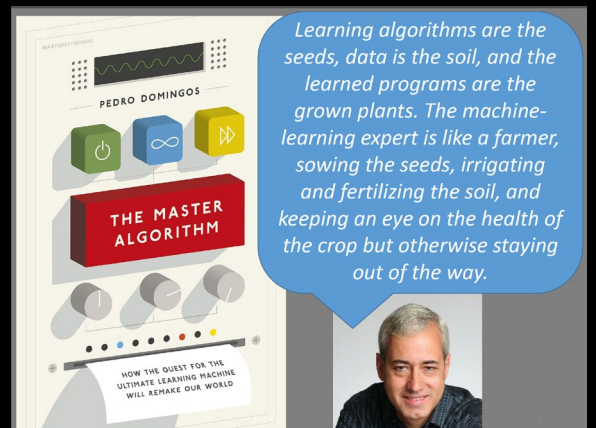
Review and Reflect on Machine Learning

- Training: model as well as people
  - You are a model
- Reflection and analogy
  - What you want to learn (objective function / goal)
  - How we teach you to learn (training / optimisation)
  - What you have learned (decision model)
- Key challenge: overfitting vs generalisation
- Post your questions
- Have fun





kisspng-fashion-show-computer-icons-man-icon-5acd9cfaa3d9f5.3374529915234245066712.jpg (900×900) (cleanpng.com)



Data, Model, Metric, Optimisation

1. Given training data:

$$\{\boldsymbol{x}_i, \boldsymbol{y}_i\}_{i=1}^{N}$$

2. Choose each of these:
   - Decision function/model

$$\hat{\boldsymbol{y}} = f_{\boldsymbol{\theta}}(\boldsymbol{x}_i)$$

   - Loss function/metric

$$\ell(\hat{\boldsymbol{y}}, \boldsymbol{y}_i) \in \mathbb{R}$$

3. Define goal:
   - Objective function

$$\boldsymbol{\theta}^* = \arg\min_{\boldsymbol{\theta}} \sum_{i=1}^{N} \ell(f_{\boldsymbol{\theta}}(\boldsymbol{x}_i), \boldsymbol{y}_i)$$

4. Train/optimize with SGD: (take small steps opposite the gradient)

$$\boldsymbol{\theta}^{(t+1)} = \boldsymbol{\theta}^{(t)} - \eta_t \nabla \ell(f_{\boldsymbol{\theta}}(\boldsymbol{x}_i), \boldsymbol{y}_i)$$