AMLD

A Machine Learning Collaboration with Neuroscience: Opportunities & Challenges

Francesco Casalegno
Blue Brain Project – EPFL

Lausanne, 2022-03-30

Machine Learning at the Blue Brain Project

The Blue Brain Project (BBP)

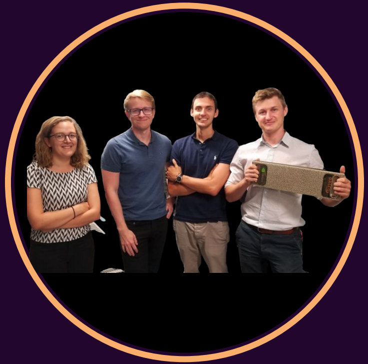
Reconstruct and simulate the mouse brain.

Mouse Brain: > 70 M neurons + > 100 B synapses!

Diverse teams: 60 scientists + 70 IT professionals

Iterative workflow:
… → get data → build model → simulate → refine → …

Data is key: experiments, databases, inference, …



Machine Learning Team

Support BBP scientists by creating ML-based tools.

Design + implement solutions: initiation → deployment.

How can ML support neuro-scientists?

1. accelerate workflows
2. reproducibility & consistency
3. automated end-to-end scientific pipelines

🔥 "Atlas Alignment" – Merging Brain Atlases

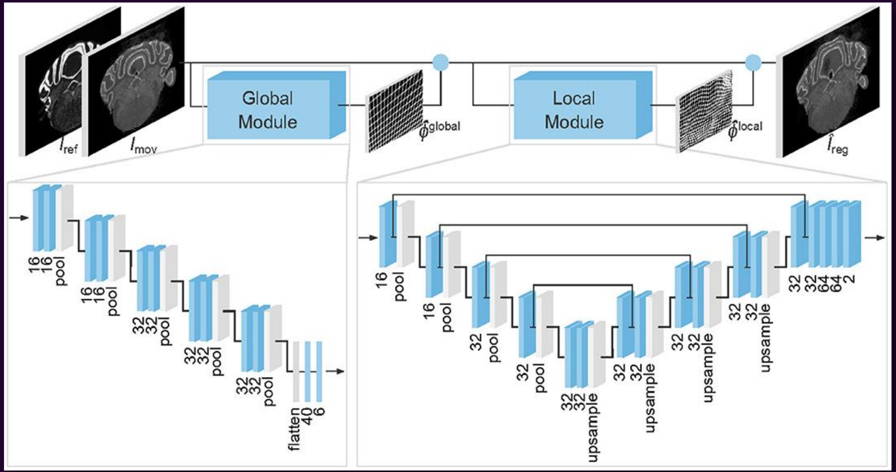
The problem

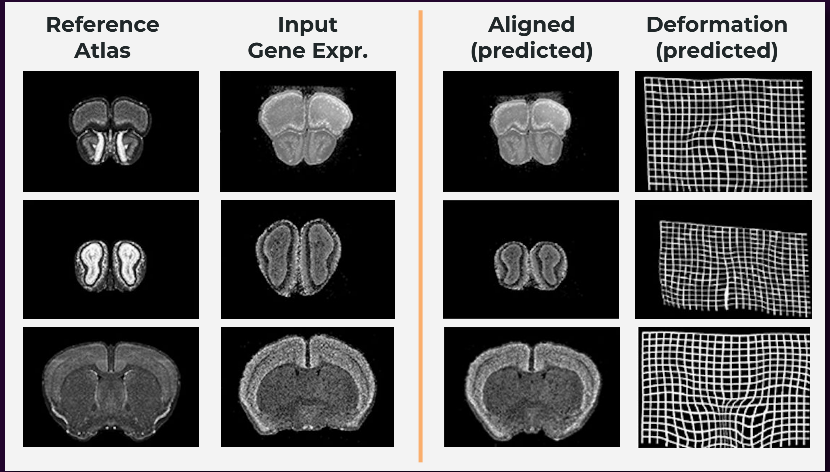
"Reference Brain Atlas" — Nissl staining, 1 specimen
"Gene Expressions" — In-situ hybridization, > 20,000 genes!

→ **Gene Expression slices are not aligned w.r.t. Reference Atlas + modalities are different!**

Our approach

→ Train ML model that, given a pair Nissl and GE slices, **predicts a deformation that aligns the GE onto the Nissl**.





Krepl J. et al. "**Supervised Learning With Perceptual Similarity for Multimodal Gene Expression Registration of a Mouse Brain Atlas.**", Front. Neuroinform., 2021
BlueBrain / atlas-alignment

🔥 "Blue Brain Search" – Literature Search & Mining

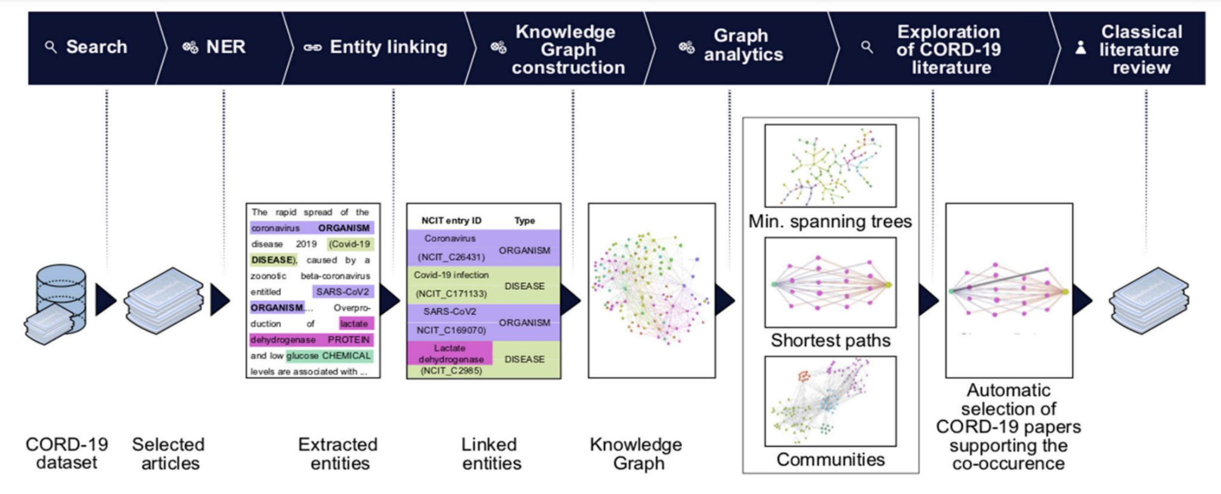
The problem

Massive open literature databases: CORD-19 had 600,000 articles (incl. 250,000 full-texts)!
Too much text for a human.

→ **We need tools to automatically search and mine information from a literature database!**

Our approach

→ Train ML models for **search** (~BioBERT **embeddings**) and **mining** (**NER**), then create **knowledge graph** with results.



Logette E. et al. **"A Machine-Generated View of the Role of Blood Glucose Levels in the Severity of COVID-19."**, Front. Neuroinform., 2021.
BlueBrain / Search

Challenges

💎 1. Gold Standard vs. Ground Truth

Supervised ML tasks = train + eval a model on `(X, y)` — `y` is usually "**ground truth**".

But in many scientific cases, the "ground truth" is not available, and `y` is just a "**gold standard**" (= human annotation).

As such, annotations may be **noisy** (= human errors) — or at least **subjective**, as there's no "objective" truth

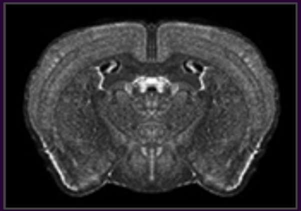
1. Gold Standard vs. Ground Truth

Supervised ML tasks = train + eval a model on `(X, y)` — `y` is usually "**ground truth**".

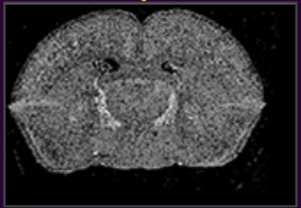
But in many scientific cases, the "ground truth" is not available, and `y` is just a "**gold standard**" (= human annotation).

As such, annotations may be **noisy** (= human errors) — or at least **subjective**, as there's no "objective" truth

Reference



Input



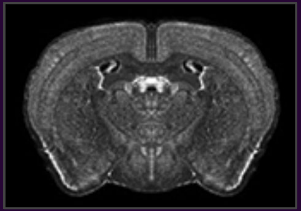
1. Gold Standard vs. Ground Truth

Supervised ML tasks = train + eval a model on `(X, y)` — `y` is usually "**ground truth**".

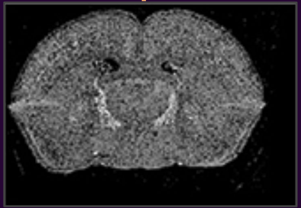
But in many scientific cases, the "ground truth" is not available, and y is just a "**gold standard**" (= human annotation).

As such, annotations may be **noisy** (= human errors) — or at least **subjective**, as there's no "objective" truth

Reference



Input



Ground Truth
labels

→ **Coronal section number.** 37

→ **Gene staining used on Input.** NPY

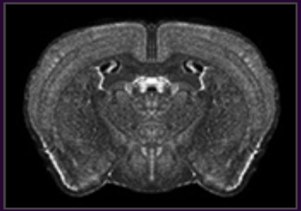
1. Gold Standard vs. Ground Truth

Supervised ML tasks = train + eval a model on (X, y) — y is usually "**ground truth**".

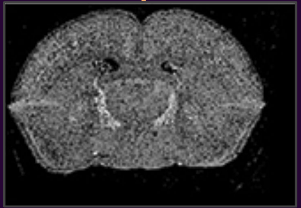
But in many scientific cases, the "ground truth" is not available, and y is just a "**gold standard**" (= human annotation).

As such, annotations may be **noisy** (= human errors) — or at least **subjective**, as there's no "objective" truth

Reference



Input



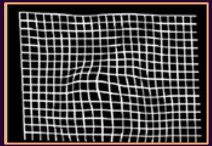
Ground Truth labels

→ **Coronal section number.** 37

→ **Gene staining used on Input.** NPY

Gold Standard labels

→ **Deformation aligning Input w. Reference.**



→ **Brain sub-region segmentation.**

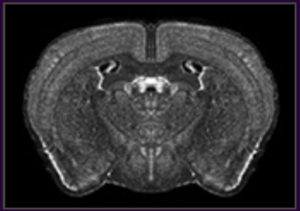
1. Gold Standard vs. Ground Truth

Supervised ML tasks = train + eval a model on `(X, y)` — `y` is usually "**ground truth**".

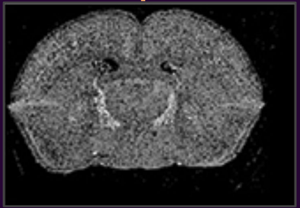
But in many scientific cases, the "ground truth" is not available, and `y` is just a "**gold standard**" (= human annotation).

As such, annotations may be **noisy** (= human errors) — or at least **subjective**, as there's no "objective" truth

Reference



Input



→

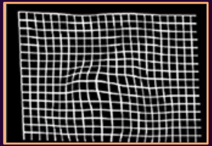
Ground Truth
labels

→ **Coronal section number.** — 37

→ **Gene staining used on Input.** — NPY

Gold Standard
labels

→ **Deformation aligning Input w. Reference.**



→ **Brain sub-region segmentation.**

Why does it matter?

→ **Garbage-in-garbage-out** – training models on noisy annotations `y_true` will produce an under-performing model

→ **Biased evaluation** – comparing vs. noisy `y_true` makes you jump to wrong conclusions (model selection/validation)

🤔 2. Ask for a Second Opinion

Even w/o real mistakes (e.g. no objective truth), experts may **disagree** with each other → lack of **inter-rater agreement**.

2. Ask for a Second Opinion

Even w/o real mistakes (e.g. no objective truth), experts may **disagree** with each other → lack of **inter-rater agreement**.

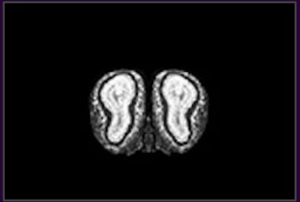
Reference

Input

2. Ask for a Second Opinion

Even w/o real mistakes (e.g. no objective truth), experts may **disagree** with each other → lack of **inter-rater agreement**.

Reference

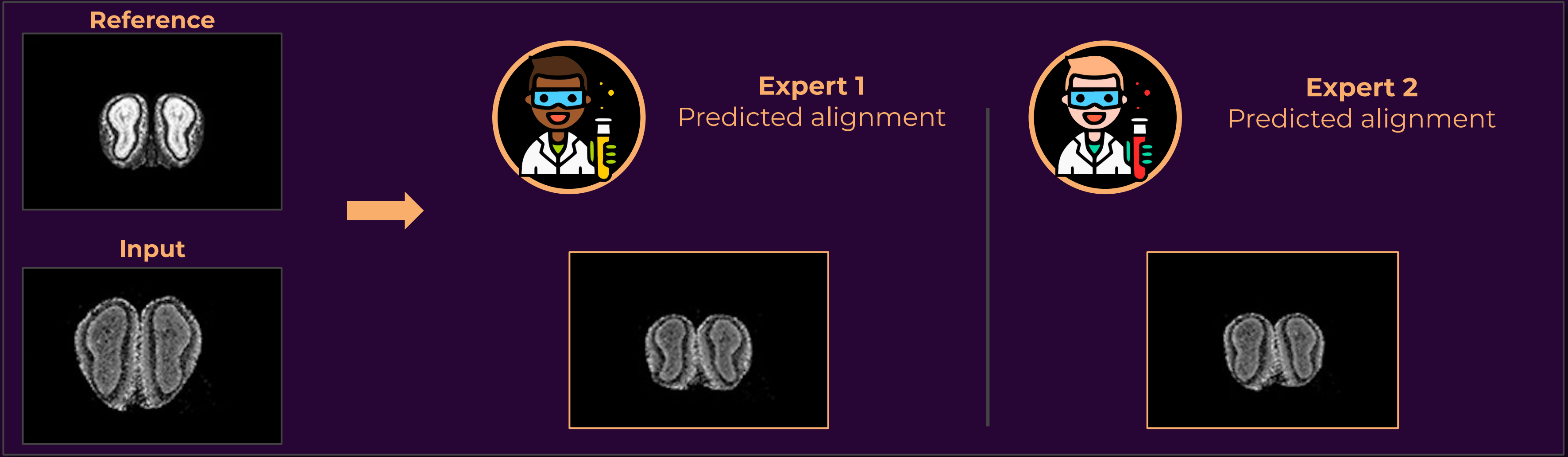


Input

Expert 1
Predicted alignment

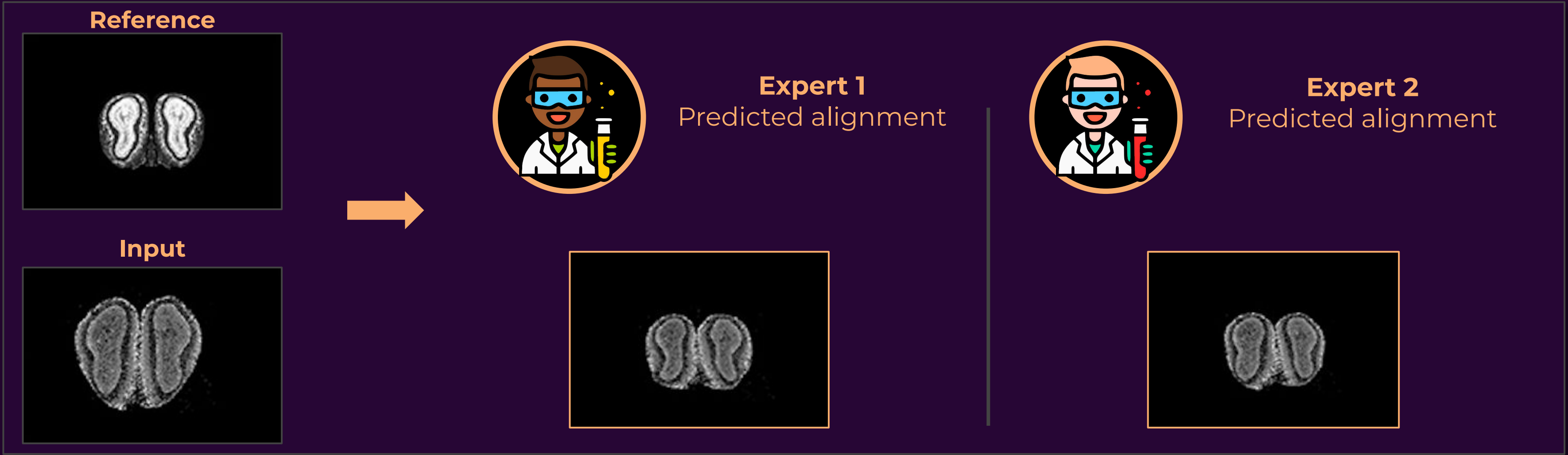
2. Ask for a Second Opinion

Even w/o real mistakes (e.g. no objective truth), experts may **disagree** with each other → lack of **inter-rater agreement**.



🤔 2. Ask for a Second Opinion

Even w/o real mistakes (e.g. no objective truth), experts may **disagree** with each other → lack of **inter-rater agreement**.



Why does it matter?

→ **Setting expectations** – should we aim at reaching 100% accuracy? with respect to what? what does it even mean?

→ **Definition-of-Done** – until when should we invest time and resources in an effort to "improve results"?

→ **Design training/evaluation** – should we train/evaluate using the `y_true` of Expert 1? Or the `y_true` of Expert 2?

3. Annotations are Expensive

Annotating samples is often a **time-consuming process** – e.g. aligning GE onto Nissl took a whole PhD's summer!

We want to make sure that we choose carefully how many and which samples x to annotate.

3. Annotations are Expensive

Annotating samples is often a **time-consuming process** – e.g. aligning GE onto Nissl took a whole PhD's summer!

We want to make sure that we choose carefully how many and which samples x to annotate.

200
annotated samples

80%
accuracy

3. Annotations are Expensive

Annotating samples is often a **time-consuming process** – e.g. aligning GE onto Nissl took a whole PhD's summer!

We want to make sure that we choose carefully how many and which samples x to annotate.

→ **200** annotated samples

↓

80% accuracy

→ **600** annotated samples

↓

??? accuracy

3. Annotations are Expensive

Annotating samples is often a **time-consuming process** – e.g. aligning GE onto Nissl took a whole PhD's summer!

We want to make sure that we choose carefully how many and which samples x to annotate.

200
annotated samples

80%
accuracy

600
annotated samples

???
accuracy

"How many annotations do you need for your model?"
"Which samples should I annotate?"

3. Annotations are Expensive

Annotating samples is often a **time-consuming process** – e.g. aligning GE onto Nissl took a whole PhD's summer!

We want to make sure that we choose carefully how many and which samples x to annotate.

→ **200** annotated samples ↓ **80%** accuracy

→ **600** annotated samples ↓ **???** accuracy

"How many annotations do you need for your model?"
"Which samples should I annotate?"

Why does it matter?

→ **Cost and Time Optimization** – The time of a human expert is typically expensive and/or limited.
We want to choose the optimal number and type of samples x to provide to the human expert for annotation.

♻️ 4. Tracked & Reproducible science

During exploration/research, we try out various models, hyperparameters, libraries, ... on your machine.

How do we **track and share our work** (both training and inference) in a **reproducible way**?

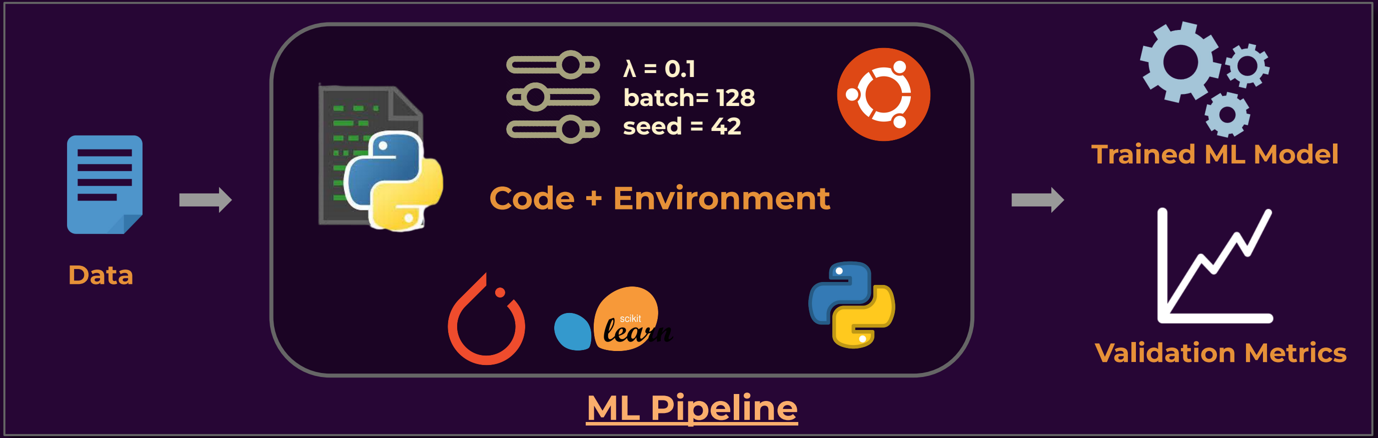
Even more complex if we don't just `model.fit(X, y)`, but we have a **whole pipeline** (data prep, train-valid split, ...)

4. Tracked & Reproducible science

During exploration/research, we try out various models, hyperparameters, libraries, … on your machine.

How do we **track and share our work** (both training and inference) in a **reproducible way**?

Even more complex if we don't just `model.fit(X, y)`, but we have a **whole pipeline** (data prep, train-valid split, …)

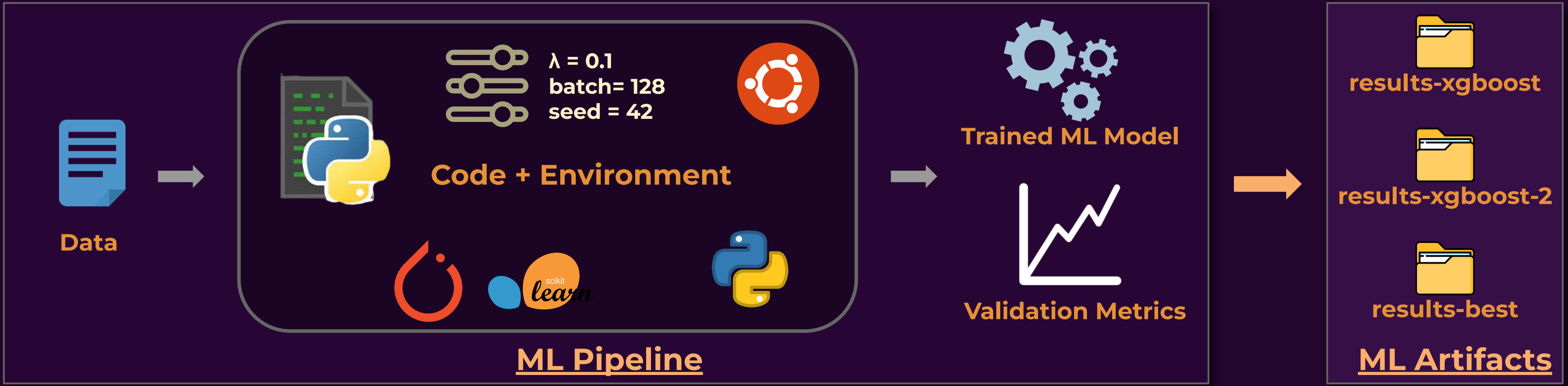


4. Tracked & Reproducible science

During exploration/research, we try out various models, hyperparameters, libraries, … on your machine.

How do we **track and share our work** (both training and inference) in a **reproducible way**?

Even more complex if we don't just `model.fit(X, y)`, but we have a **whole pipeline** (data prep, train-valid split, …)

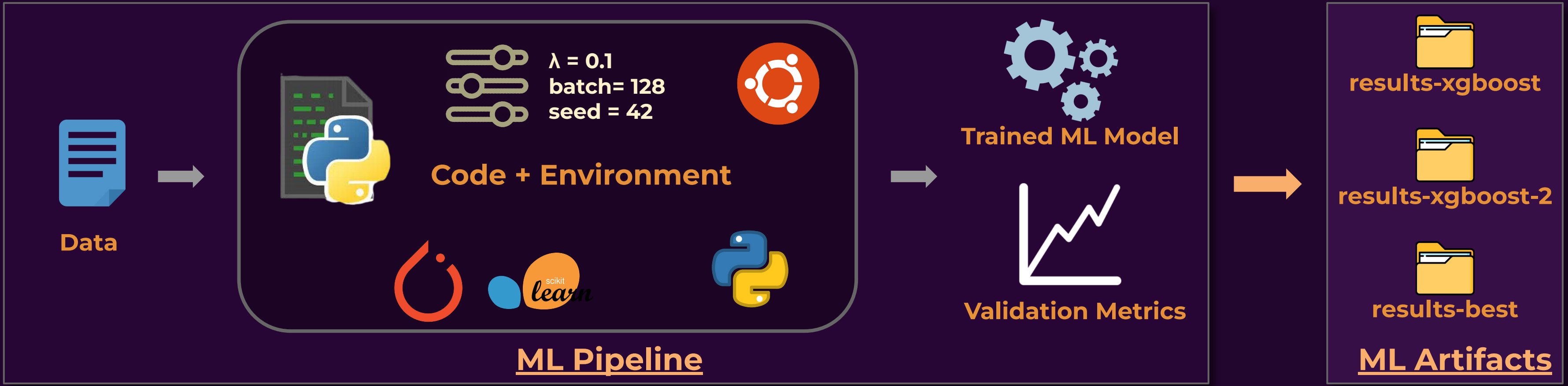


4. Tracked & Reproducible science

During exploration/research, we try out various models, hyperparameters, libraries, … on your machine.

How do we **track and share our work** (both training and inference) in a **reproducible way**?

Even more complex if we don't just `model.fit(X, y)`, but we have a **whole pipeline** (data prep, train-valid split, …)



Why does it matter?

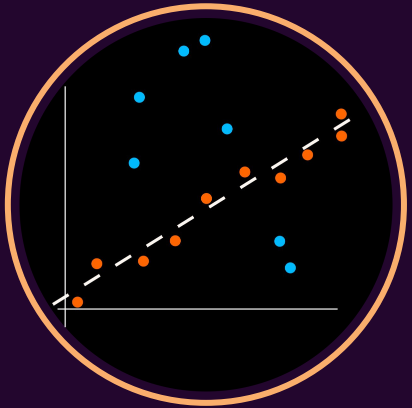
→ **Peer Review** – Paper readers may want to re-run and verify experiment result and analyze the workflow.

→ **Scientific Method** – Experiment reproducibility is the basis of the scientific method.

→ **Deployment** – We want to ensure that results will be consistent for future users.

Solutions

💎 1. Gold Standard vs. Ground Truth



RANSAC – RANdom SAmple Consensus

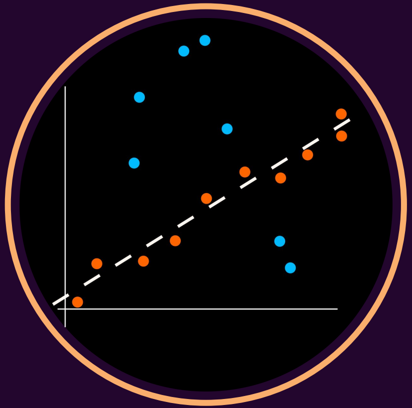
`from sklearn.linear_model import RANSACRegressor`

Iterative method for robust fitting of linear and non-linear regression models.

1. select random subset of `X, y` and fit model on those points
2. compute residuals w.r.t. model prediction → flag "outlier" if residual > threshold
3. choose as "best" model the one minimizing number of "outlier"
4. best model is fitted only on "inliers".

Outliers (= samples with noisy labels `y_true`) have no impact!

1. Gold Standard vs. Ground Truth



RANSAC – RANdom SAmple Consensus

`from sklearn.linear_model import RANSACRegressor`

Iterative method for robust fitting of linear and non-linear regression models.

1. select random subset of `X, y` and fit model on those points
2. compute residuals w.r.t. model prediction → flag "outlier" if residual > threshold
3. choose as "best" model the one minimizing number of "outlier"
4. best model is fitted only on "inliers".

Outliers (= samples with noisy labels `y_true`) have no impact!

CleanLab

`pip install cleanlab`

"**Confident Learning**" – Robust classifier fitting using exact noise estimation.

Key ideas and assumptions

1. we can't access **ground truth** labels `y`, but only **noisy labels** `s`.
2. noisy and true labels relation is captured by **noise matrix** `Q(s, y)` ≈ p(s | y)
3. estimate `Q(s, y)` with out-of-sample pred. + "confident join" (~ conf. matrix)
4. **prune samples** (= likely wrong labels) based on `Q(s, y)`.

2. Ask for a Second Opinion

$$\kappa \equiv \frac{p_o - p_e}{1 - p_e}$$

Inter-rater agreement

Inter-rater reliability is important when there is **no "objective" ground truth**.

How to compute it?

→ specific test statistics – Cohen's K, Pearson r, Spearman's ρ, Kendall's τ, ...

→ typically include chance correction!

→ (symmetric) eval. metrics – accuracy, intersection-over-union,

How to use it?

→ specific statistics – interpret inter-rater reliability level ("$K > 0.75$ is excellent")

→ symmetric eval. metrics – define baseline for model eval. score

🤔 2. Ask for a Second Opinion

$$\kappa \equiv \frac{p_o - p_e}{1 - p_e}$$

Inter-rater agreement

Inter-rater reliability is important when there is **no "objective" ground truth**.

How to compute it?

→ specific test statistics – Cohen's K, Pearson r, Spearman's ρ, Kendall's τ, ...
  → typically include chance correction!
→ (symmetric) eval. metrics – accuracy, intersection-over-union,

How to use it?

→ specific statistics – interpret inter-rater reliability level ("K > 0.75 is excellent")
→ symmetric eval. metrics – define baseline for model eval. score

Train on soft labels + Evaluate model against baselines

1. Train on "**soft labels"**: for each sample, compute per-class expert consensus.

2. Evaluate the validation score of our models against baselines.

– **Inter-rater baseline**: "higher accuracy" (w.r.t. what?) doesn't make sense!
– **Non-ML**: added value of ML model is only shown when comparing to non-ML.
– **Naive ML**: simple ML model (linear regression, ...) for quick benchmark..

Inter-rater agreement can be used as **Definition-of-Done**.

3. Annotations are Expensive

prodigy

Radically efficient machine teaching.

EXPLOSION

Transfer Learning + Active Learning (+ Indirect Feedback)

Unlabeled data → cheap and abundant → **self-supervised pre-training**
Labeled data → scarce and expensive → **fine-tuning** on task
E.g. pre-train BERT on "masked language model", then fine-tune on STS-NLI.

Active Learning: Ask to annotate samples where model is least certain.

Indirect Feedback: Propose model prediction to expert, who says if it is correct.
→ Faster annotations, but can introduce some bias + less info (just Yes/No).

3. Annotations are Expensive

Transfer Learning + Active Learning (+ Indirect Feedback)

Unlabeled data → cheap and abundant → **self-supervised pre-training**
Labeled data → scarce and expensive → **fine-tuning** on task
E.g. pre-train BERT on "masked language model", then fine-tune on STS-NLI.

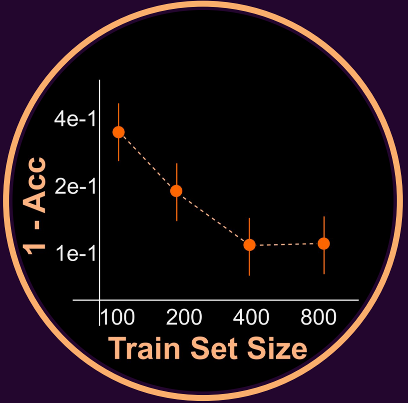
Active Learning: Ask to annotate samples where model is least certain.

Indirect Feedback: Propose model prediction to expert, who says if it is correct.
→ Faster annotations, but can introduce some bias + less info (just Yes/No).

Accuracy vs. Train Set Size curve

How many more samples do we need to improve accuracy by X%?
→ Train model with different fractions of dataset and look at validation accuracy.

Typically, **power law (~ linear in log-log)** until one of the following happens:
- model power saturates (→ look for more complex model?)
- inter-rater agreement level is reached (→ intrinsic noise, we can't do more)
- notice the **diminishing returns**



4. Tracked & Reproducible science

Docker

"But I promise that yesterday it worked on my laptop! "

Package ML app with its dependencies as a portable container image:

– operating system (Ubuntu 21.10, ...)
– libraries and binaries (cuda 10.2, python 3.8, ...)
– Python packages (sklearn 1.0.2, ...)

Share and run the ML application:

→ consistent and isolated environment
→ anyone can run application anywhere, as easy as: `docker run ml-app:1.0`

4. Tracked & Reproducible science

Docker

"But I promise that yesterday it worked on my laptop! "

Package ML app with its dependencies as a portable container image:
- operating system (Ubuntu 21.10, ...)
- libraries and binaries (cuda 10.2, python 3.8, ...)
- Python packages (sklearn 1.0.2, ...)

Share and run the ML application:
→ consistent and isolated environment
→ anyone can run application anywhere, as easy as: `docker run ml-app:1.0`

DVC – Data Version Control

`pip install dvc`

Version models and data like we version code.
→ fully integrated with and same interface as Git: `dvc add` / `git add`

Manage and version ML pipelines and artifacts
→ typically: load data, prepare it, and produce artifacts (trained model, ...)
→ DVC tracks these pipelines with a `dvc.yaml` file, same as a `Makefile`

See also → MLflow

Data Version Control

Thank you for your attention!

scan me !

Francesco Casalegno
Machine Learning Section Lead
Blue Brain Project - EPFL

FrancescoCasalegno

tds @francesco.casalegno