Distributed learning on sensitive health data

2020/01/27

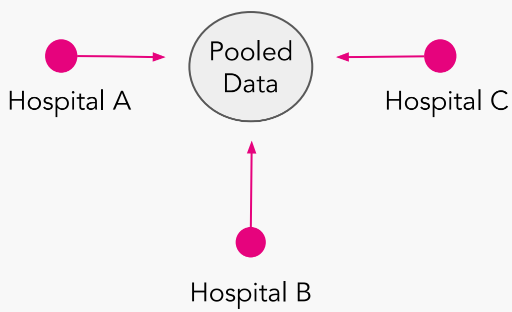
Camille Marini

OWKIN

OWKIN

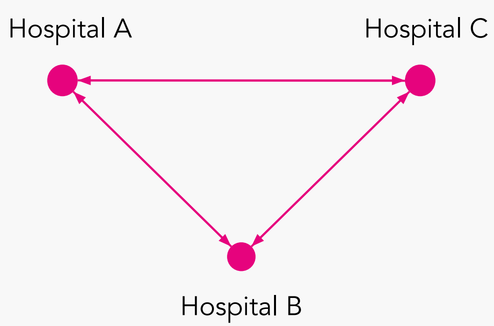
FEDERATED LEARNING

Owkin develops federated learning to train machine learning models on distributed data at scale



STANDARD APPROACH

Requires data pooling and central coordination



OWKIN FED-AI

Only neural network weights travel, while data stays onsite

Data pooling implies a reduced control and governance of data owners

No transparency on how the algorithm is trained and how the data are used

OUR PRODUCT: OWKIN CONNECT

OWKIN CONNECT

Owkin Connect helps build state-of-the-art models from heterogeneous multicentric data, with high privacy and traceability standards.

Owkin Connect connects data managers to our data scientists

Privacy

Traceability

Real World Deployments

FED-AI REAL WORLD DEPLOYMENTS

Pioneering FED-AI to connect distributed data at scale

- Won €10M BPI-funded Healthchain grant
- Consortium of 7 leading academic medical centers, 2 research centers and 2 start-ups
- Initial collaborations to develop predictive models of treatment outcomes in Breast Cancer & Melanoma
- Project coordination done by Owkin
- Live working experiment on real data

« This project is supported by Bpifrance as part of the "Healthchain" project, which resulted from the "Digital Investments Program for the major challenges of the future" RFP. As part of the "Healthchain" project, a consortium coordinated by Owkin (a private company) has been established, including the Substra association, Apricity (a private company), the Assistance Publique des Hôpitaux de Paris, the University Hospital Center of Nantes, the Léon Bérard Center, the French National Center for Scientific Research, the École Polytechnique, the Institut Curie and the University of Paris Descartes ».

FED-AI REAL WORLD DEPLOYMENTS

Pioneering FED-AI to connect distributed data at scale

Project website

MACHINE LEARNING LEDGER
ORCHESTRATION FOR DRUG DISCOVERY

JUNE 2019 – MAY 2022

MELLODDY

powered by aws

PHARMA PARTNERS

AMGEN
astellas
AstraZeneca
BAYER
Boehringer Ingelheim
gsk
janssen
MERCK
NOVARTIS
SERVIER

PUBLIC PARTNERS

IKTOS
KU LEUVEN
loodse
NVIDIA
OWKIN
SUBSTRA

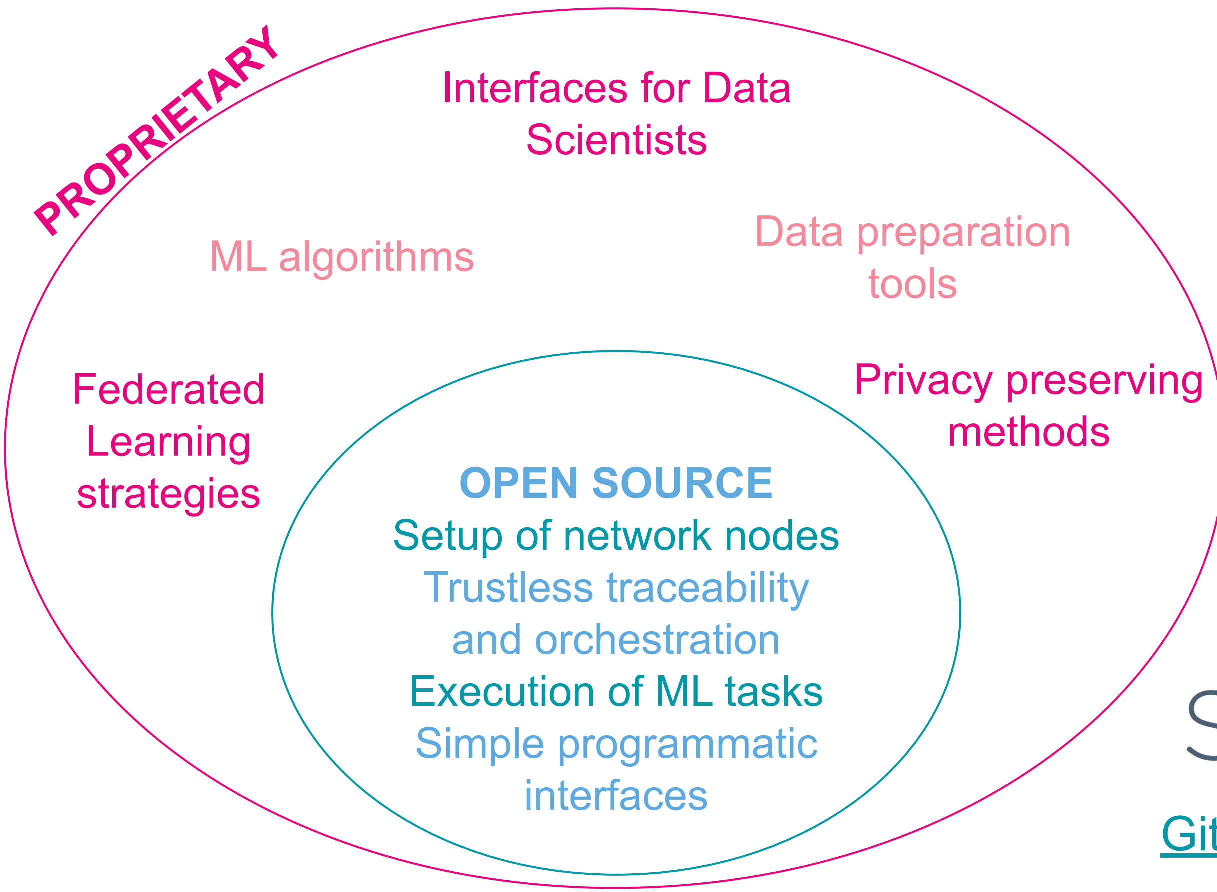
This project has received funding from the Innovative Medicines Initiative 2 Joint Undertaking under grant agreement N° 831472. This Joint Undertaking receives support from the European Union's Horizon 2020 research and innovation programme and EFPIA

imi
efpia

Technical description

OWKIN CONNECT - OPEN CORE APPROACH

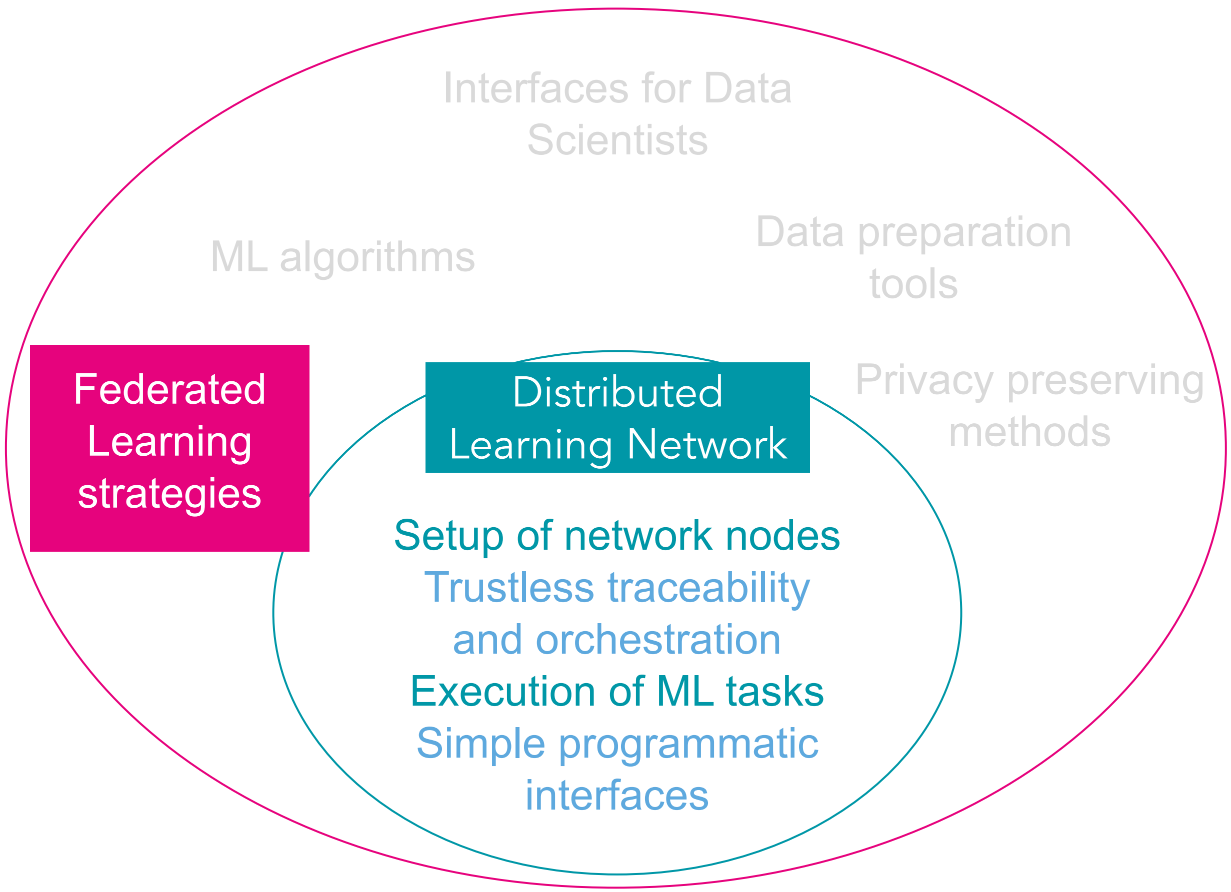
Powered by the Open-Source Framework Substra



SUBSTRA

[GitHub SubstraFoundation]

OWKIN CONNECT



OWKIN

OWKIN CONNECT

Framework for ML orchestration on decentralized sensitive data

Data privacy **Traceability**

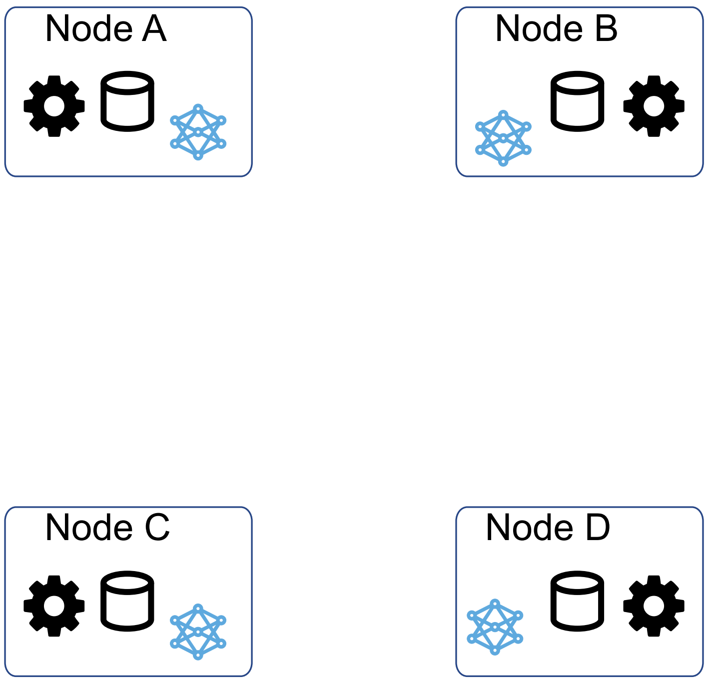
Data agnostic

Algorithm agnostic

ML framework agnostic

OWKIN

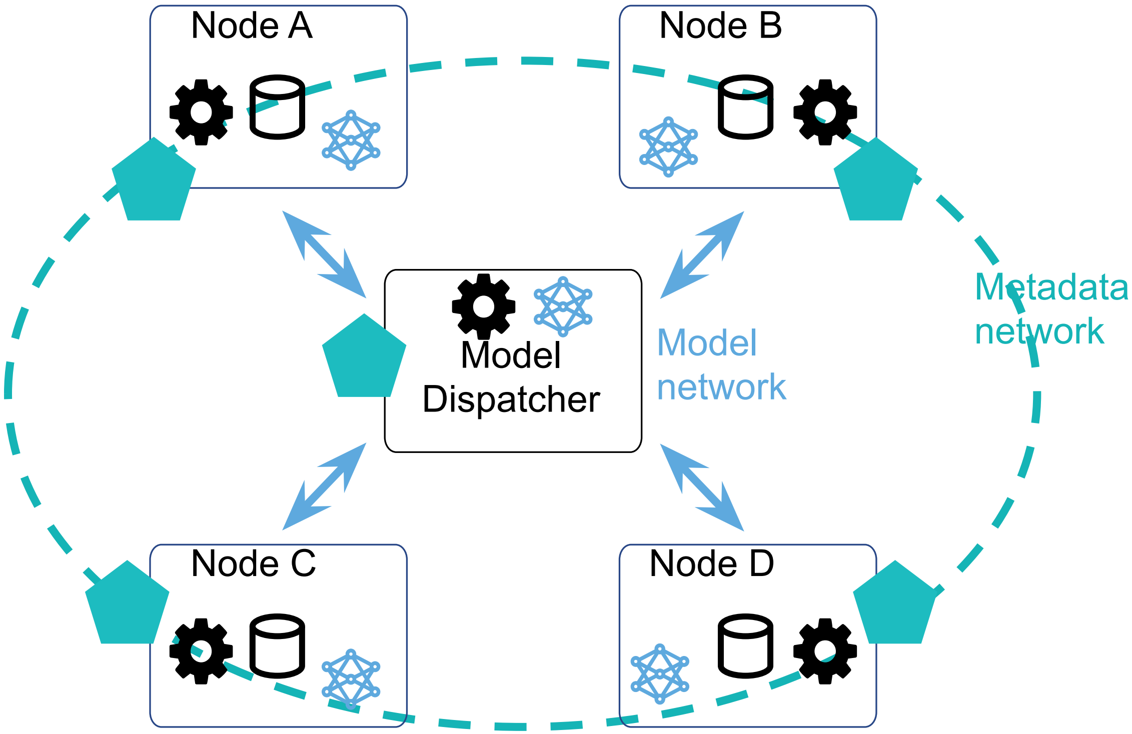
ASSETS



- Objective
  - scientific question
  - evaluation metrics
  - test dataset
- Dataset
  - set of data samples
  - opener (script to read data samples)
- Algo
  - ML algo and its dependencies
- Models
  - learnt parameters
  - training tasks specification
  - testing tasks specification
  - aggregation tasks specification

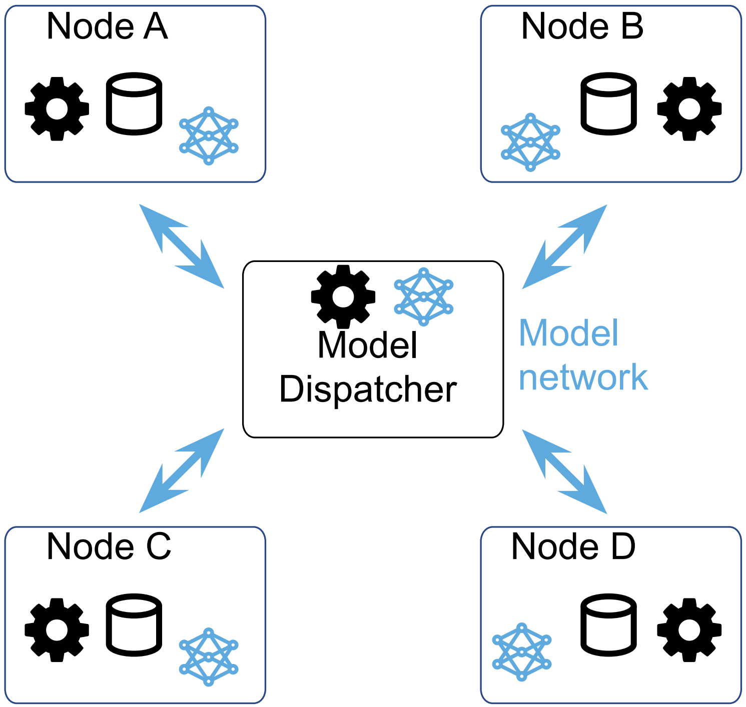
OWKIN

THE TWO NETWORKS



OWKIN

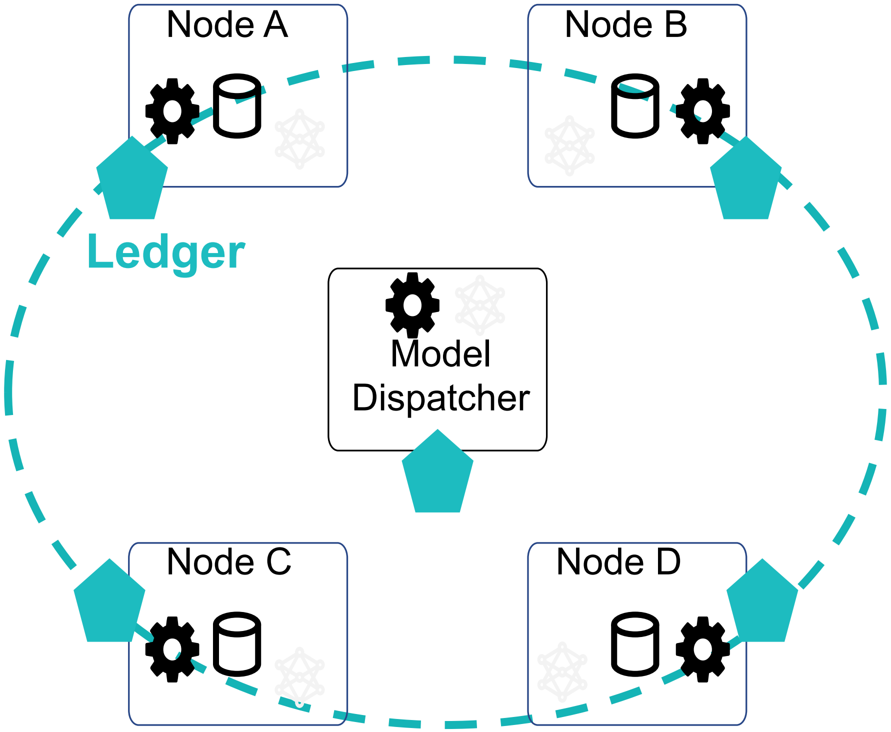
THE MODEL NETWORK



Exchange of model (updates) through a REST API.

OWKIN

THE METADATA NETWORK



Distributed Ledger Technology:

Trustless traceability

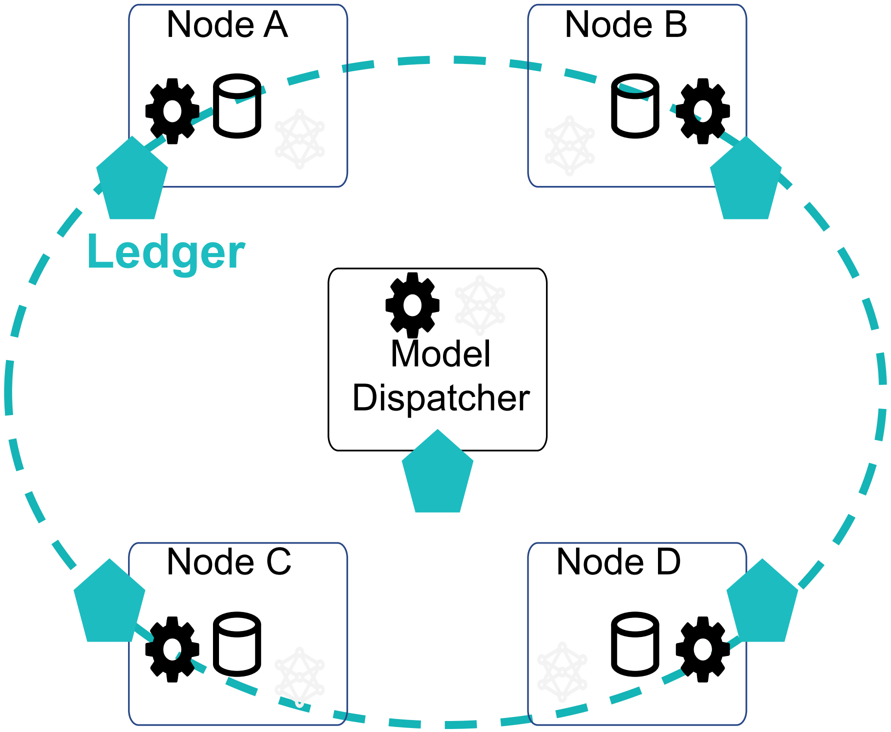
Learning orchestration

Permissions management

HYPERLEDGER FABRIC

OWKIN

THE METADATA NETWORK

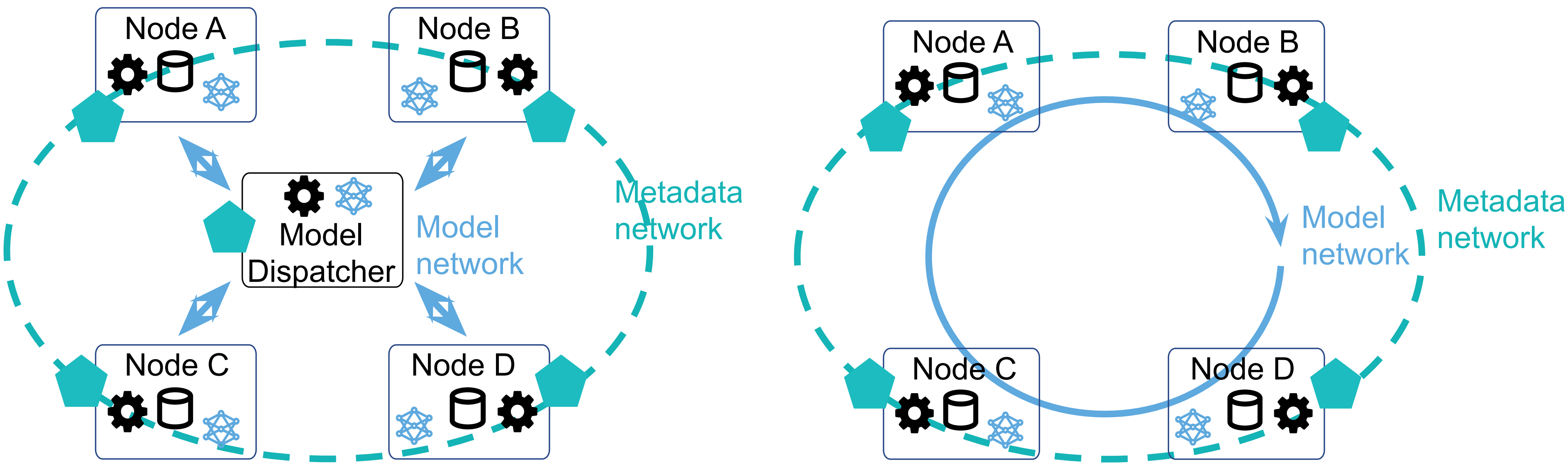


Metadata of the 4 assets

- Objective
  - Hash of the metrics and url of to request it
  - Id and hash of test data; node that stores them
  - Permissions
- Dataset
  - Hash of the opener and url of to request it
  - Id and hash of data; node that stores them
  - Permissions
- Algo
  - Hash of the algo and url of to request it
  - Node that created it
  - Permissions
- Models
  - Specification of training tasks
  - Specification of aggregation tasks
  - Specification of evaluation tasks

OWKIN

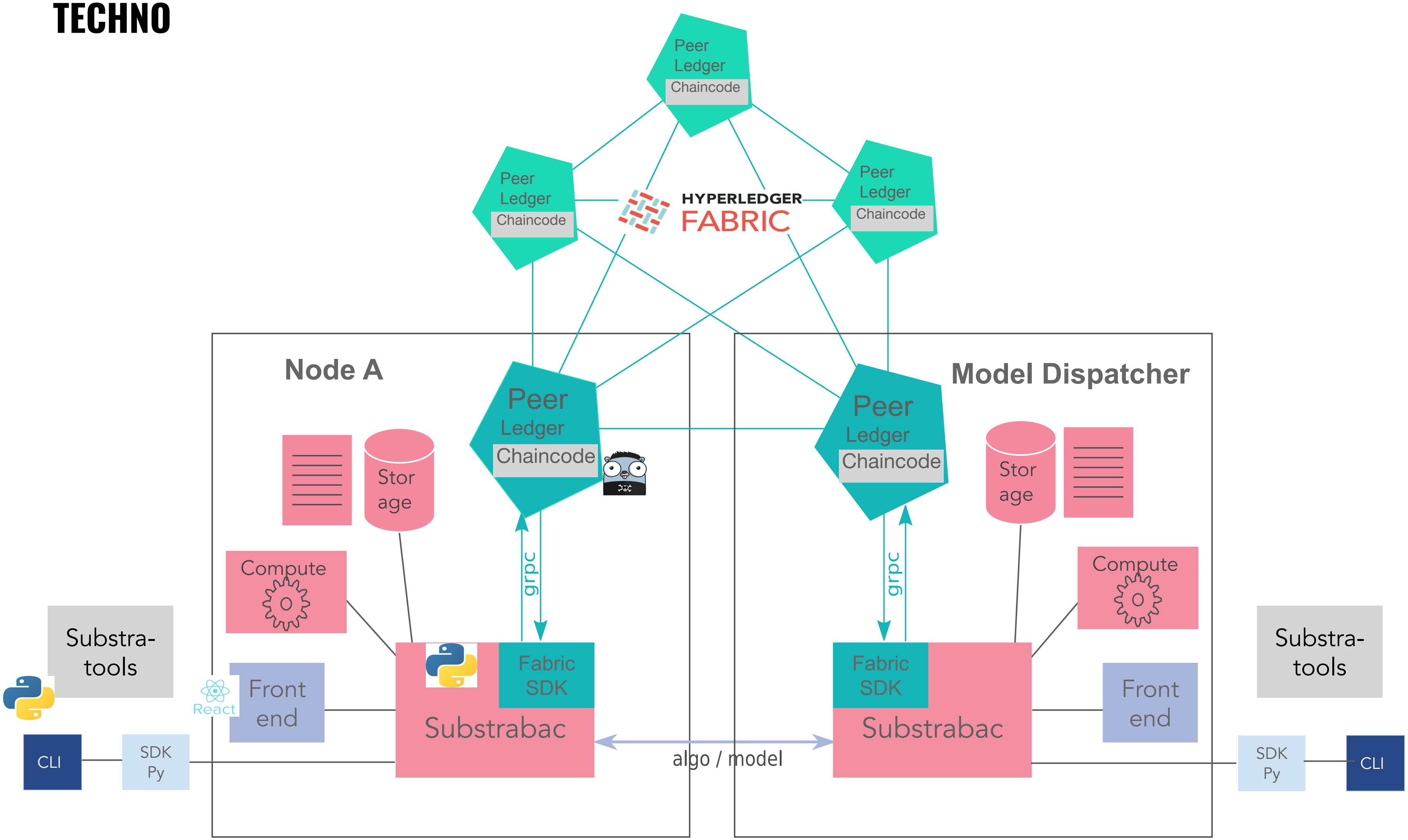
NETWORK TOPOLOGIES



OWKIN

ARCHITECTURE OVERVIEW

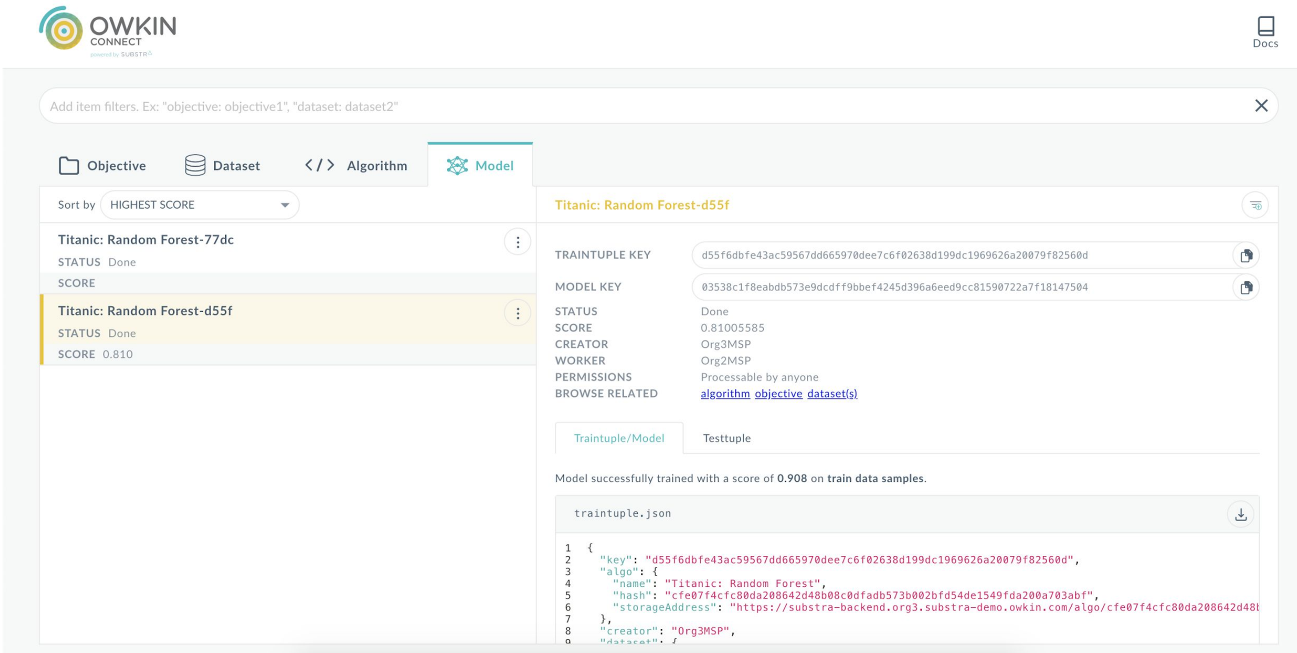
TECHNO



OWKIN

OWKIN CONNECT INTERFACES

Frontend
(traceability/perf)

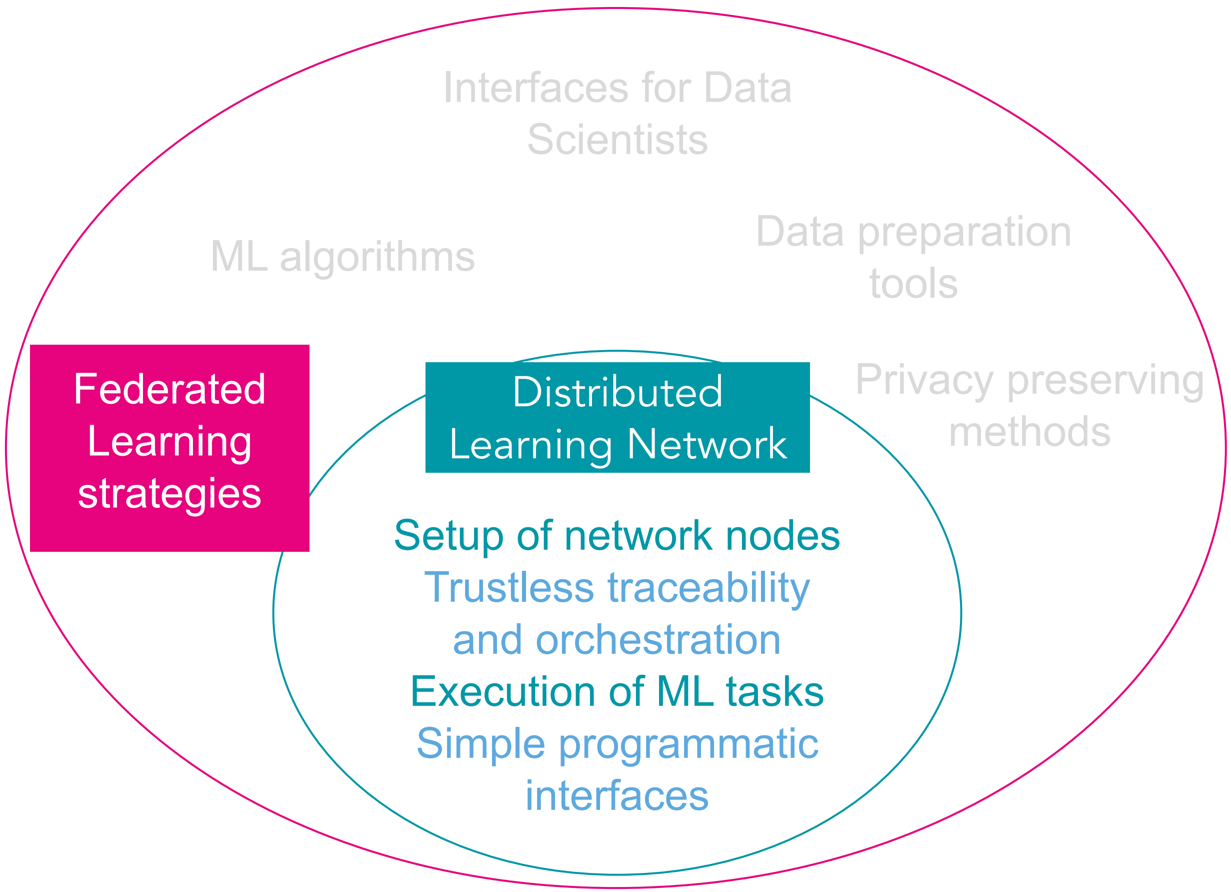


Command Line Interface

Python SDK

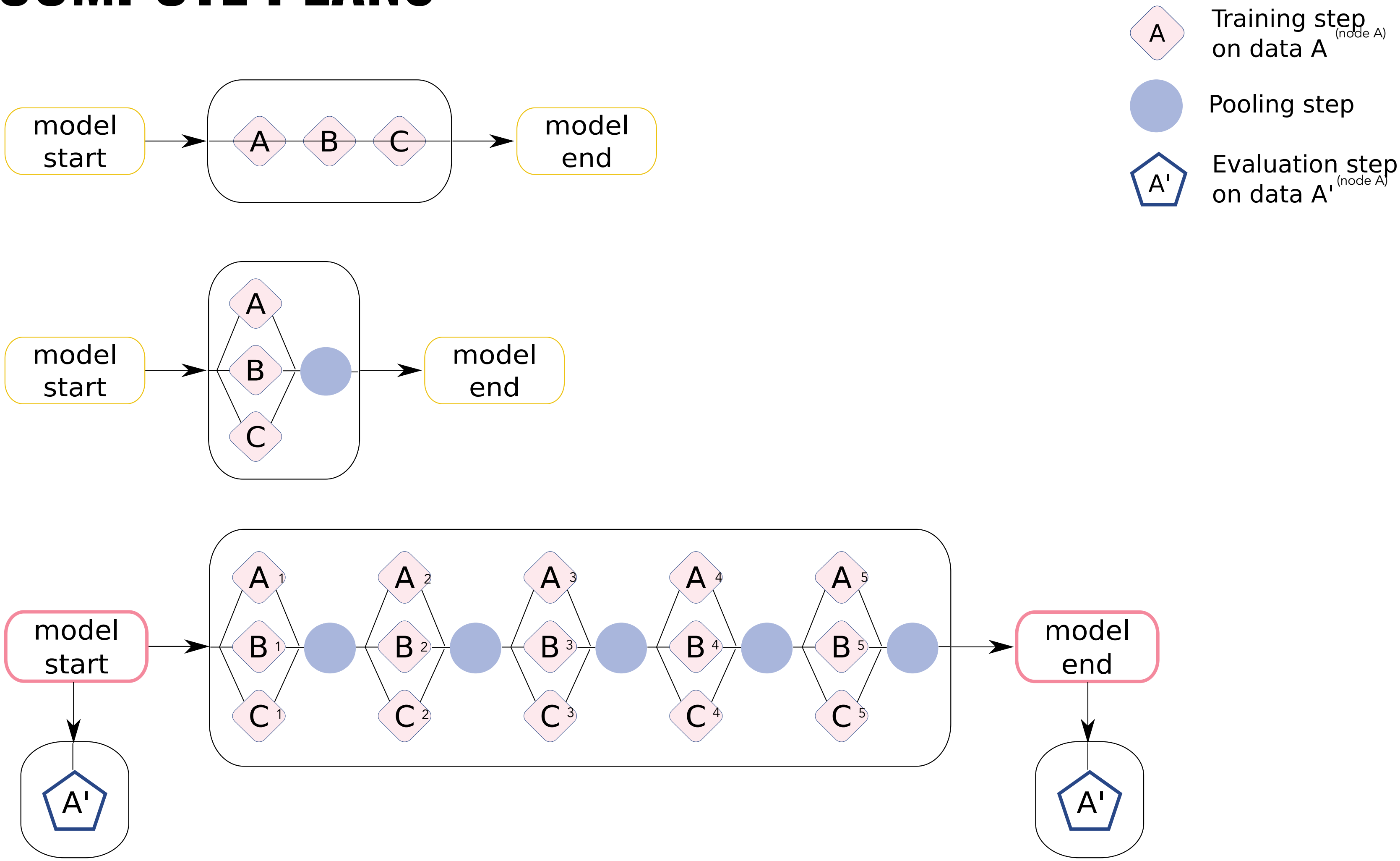
```
print('Adding dataset...')
dataset_key = client.add_dataset(DATASET, exist_ok=True)['pkhash']
assert dataset_key, 'Missing data manager key'

train_data_sample_keys = []

print('Adding train data samples...')
with progress_bar(len(TRAIN_DATA_SAMPLES_PATHS)) as progress:
    for path in TRAIN_DATA_SAMPLES_PATHS:
        data_sample = client.add_data_sample({
            'data_manager_keys': [dataset_key],
            'test_only': False,
            'path': path,
            'permissions': {'public': True},
        }, local=True, exist_ok=True)
        data_sample_key = data_sample['pkhash']
        train_data_sample_keys.append(data_sample_key)
        progress.update()
```

OWKIN CONNECT

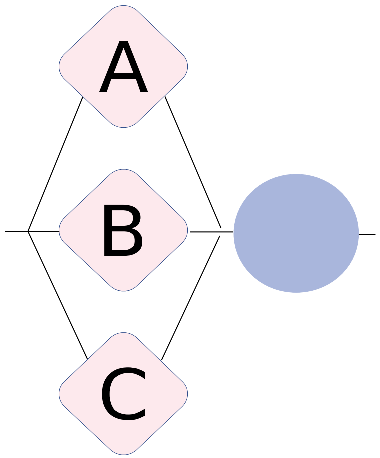


OWKIN

COMPUTE PLANS



FL LEARNING ALGORITHMS



Different ways to handle a training step and the aggregation of models.

Algorithm 1 FedSGD

Initialize the parameters with some value $\mathbf{w}_0$
C: proportion of workers chosen at each iteration ($0 < C \leq 1$)
T: number of aggregation rounds
$\{\eta_t\}_{t\geq 1}$: sequence of step sizes
for $t = 1, \ldots, T$ **do**
  Choose uniformly at random $\lfloor CK \rfloor$ distinct edge participants, stored in a set $H = \{k_1, \ldots k_{\lfloor CK \rfloor}\}$.
  for $k \in H$ **in parallel do**
    Edge participant k receives the previous value $\mathbf{w}_{t-1}$ from the central participant
    Edge participant k computes the gradient of $\mathcal{L}_k$ with respect to $\mathbf{w}$:

$$\mathbf{g}_{t,k} = \frac{1}{N_k} \sum_{i=1}^{N_k} \nabla_{\mathbf{w}} \mathcal{L}_k(\mathbf{w}_{t-1}) \tag{3}$$

    Edge participant k returns local gradient $\mathbf{g}_{t,k}$ to the central server
  end for
  Central computation server aggregates the different gradients:

$$\mathbf{g}_t = \sum_{k\in H} \frac{N_k}{N} \mathbf{g}_{t,k} \tag{4}$$

  Central computation server performs a gradient descent step:

$$\mathbf{w}_t = \mathbf{w}_{t-1} - \eta_t \mathbf{g}_t \tag{5}$$

end for
return $\mathbf{w}_T$

Algorithm 2 FedAvg

Initialize the parameters with some value $\mathbf{w}_0$
C: proportion of workers chosen at each iteration ($0 < C \leq 1$)
T: number of aggregation rounds
B: local batch size
E_l: number of local epochs
$\{\eta_t\}_{t\geq 1}$: sequence of step sizes
for $t = 1, \ldots, T$ **do**
  Choose uniformly at random $\lfloor CK \rfloor$ distinct edge participants, stored in a set $H = \{k_1, \ldots k_{\lfloor CK \rfloor}\}$.
  for $k \in H$ **in parallel do**
    Edge participant k receives the previous value $\mathbf{w}_{t-1}$ from the central participant: $\mathbf{w}^0_{t,k} \triangleq \mathbf{w}_{t-1}$
    for $e = 1, \ldots, E_l$ **do**
      Initialize value of the parameters for epoch e: $\mathbf{w}^e_{t,k} \leftarrow \mathbf{w}^{e-1}_{t,k}$
      for $b = 1, \ldots, {}^{N_k}/_B$ **do**
        Choose B samples uniformly at random (with or without replacement) from $\mathcal{D}_k$: $s_{i_1}, \ldots, s_{i_B}$
        Perform a gradient descent step over the batch

$$\mathbf{w}^e_{t,k} \leftarrow \mathbf{w}^e_{t,k} - \frac{\eta_t}{B} \sum_{j=1}^{B} \nabla_{\mathbf{w}} \ell_{i_j}(\mathbf{w}^e_{t,k}) \tag{6}$$

      end for
    end for
    Edge participant k returns the weight updates for the round:

$$\Delta \mathbf{w}_{t,k} = \mathbf{w}^{E_l}_{t,k} - \mathbf{w}_{t-1} \tag{7}$$

  end for
  Central computation server aggregates the local updates

$$\Delta \mathbf{w}_t = \sum_{k\in H} \frac{N_k}{N} \Delta \mathbf{w}_{t,k} \tag{8}$$

  Central computation server performs a gradient descent step:

$$\mathbf{w}_t = \mathbf{w}_{t-1} + \Delta \mathbf{w}_t \tag{9}$$

end for
return $\mathbf{w}_T$

McMahan et al, 2017

Thank you !

Contact us to learn more:

https://owkin.com/

@marini_camille @OWKINscience @Substra_org

Owkin