CODE CARBONE

Un outil python pour mesurer l’empreinte carbone de votre code

AGENDA

- Introduction à l’outil CodeCarbon
- La méthodologie de calcul de l’empreinte carbone
- L’utilisation de l’outil
- Autres outils
- Bonnes pratiques

INTRODUCTION

OUTIL CODECARBON

Suivi des **émissions** de votre code Python, basées sur la **consommation d'énergie** de votre machine et **l'intensité de carbone** de votre localisation

Un **package Python** léger et facile à utiliser (à installer avec PyPi ou Conda)

Visualisation efficace des résultats dans un **dashboard** intégré

Open-source, **gratuit** et développé par une équipe de bénévoles

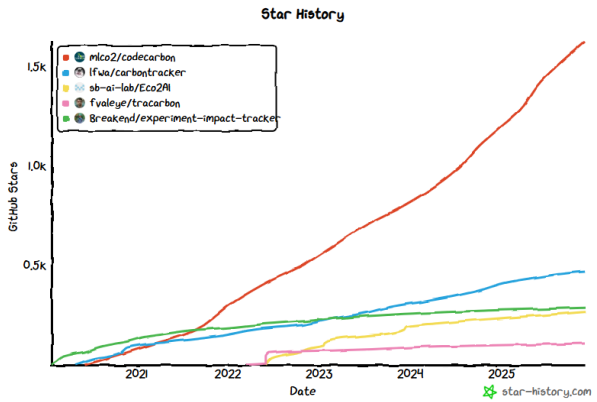
COMMENT J'EN SUIS ARRIVEE LA ?

Lancement de Code Carbon initié par Yoshua Bengio (prix Turing), puis développé par Data for Good et Mila

2020

Création d'une association à but non lucratif en France pour soutenir le projet

2023

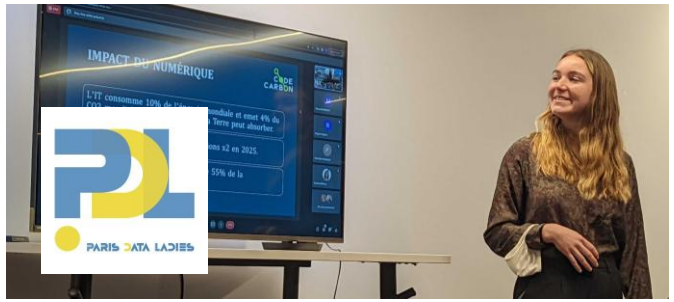


1,5 millions de téléchargements

2025

2021

Invitation pour présenté l'outil à un évènement *Paris Data Ladies*



2024

Cité dans *Le référentiel général pour l'IA frugale*

POURQUOI MESURER L'EMPREINTE CARBONE?

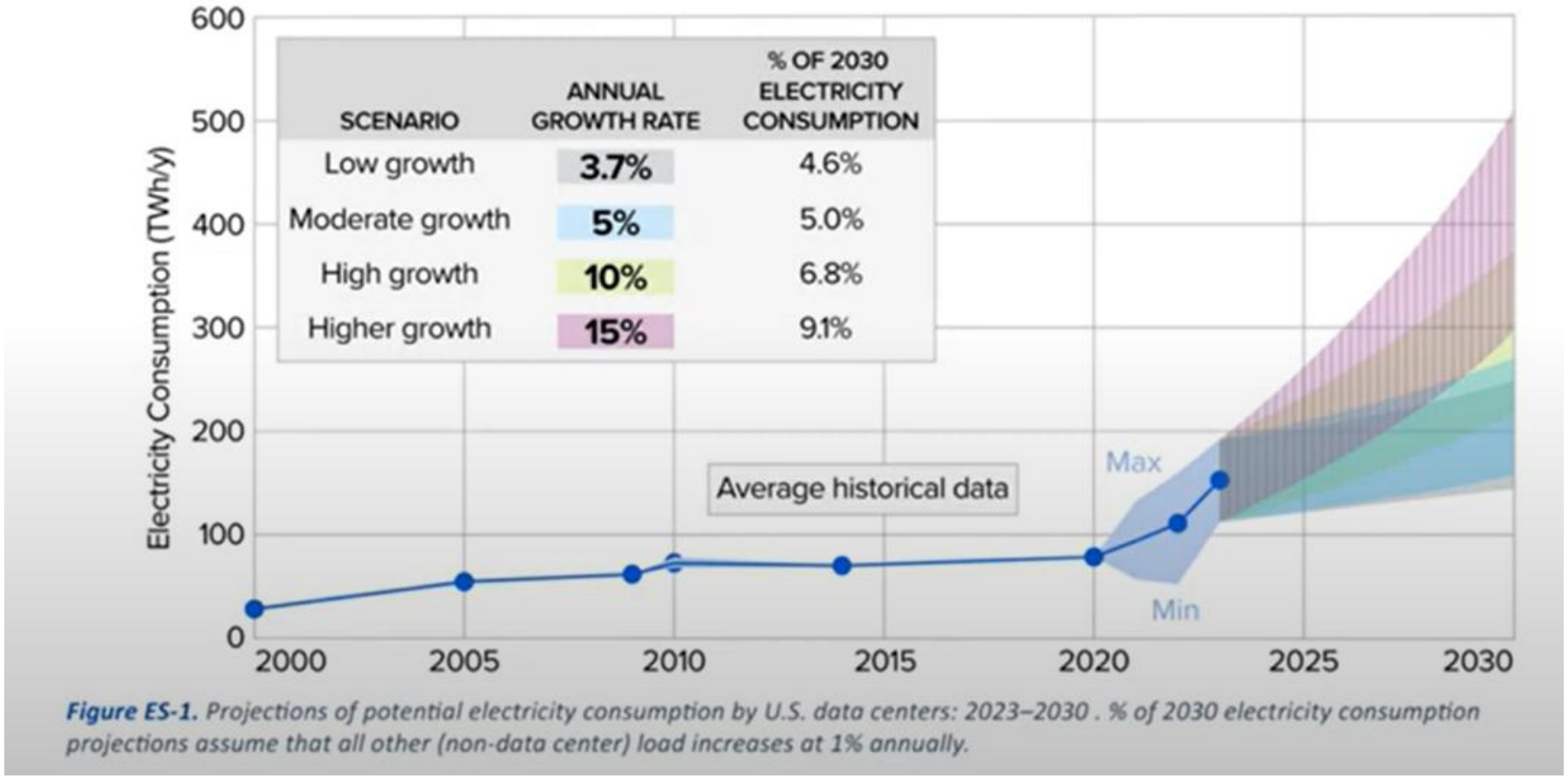
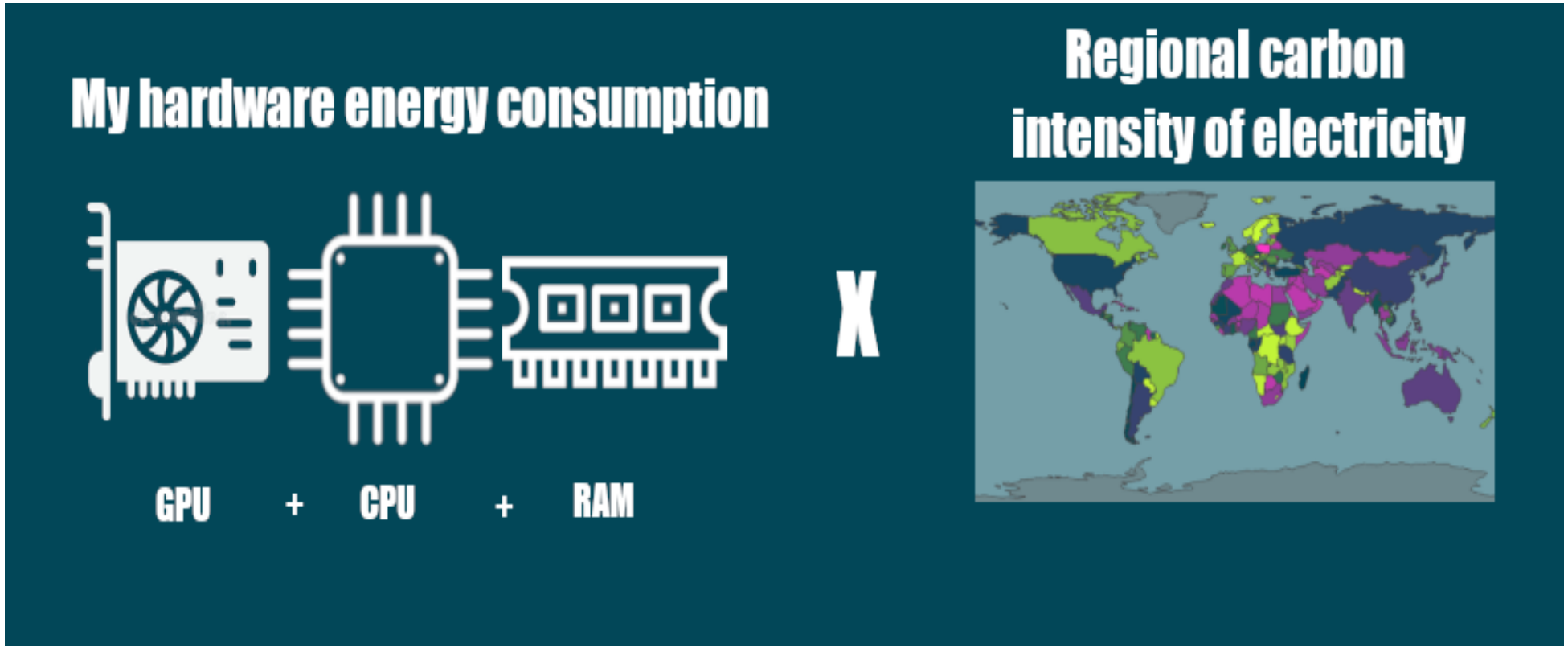


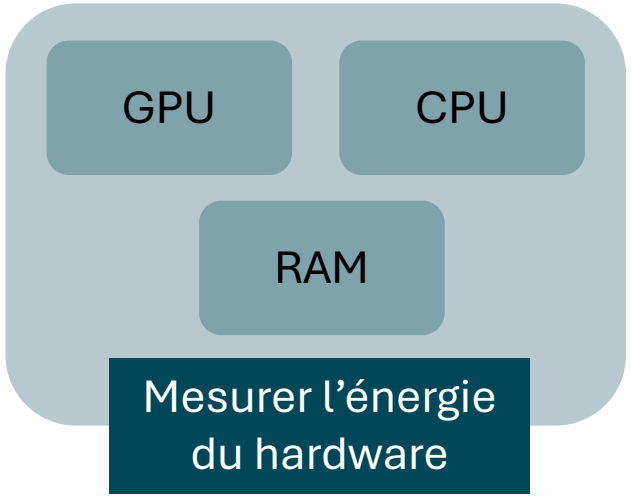
Figure ES-1. Projections of potential electricity consumption by U.S. data centers: 2023–2030 . % of 2030 electricity consumption projections assume that all other (non-data center) load increases at 1% annually.

METHODOLOGIE

COMMENT CA MARCHE ?



METHODOLOGIE - ENERGIE DU HARDWARE

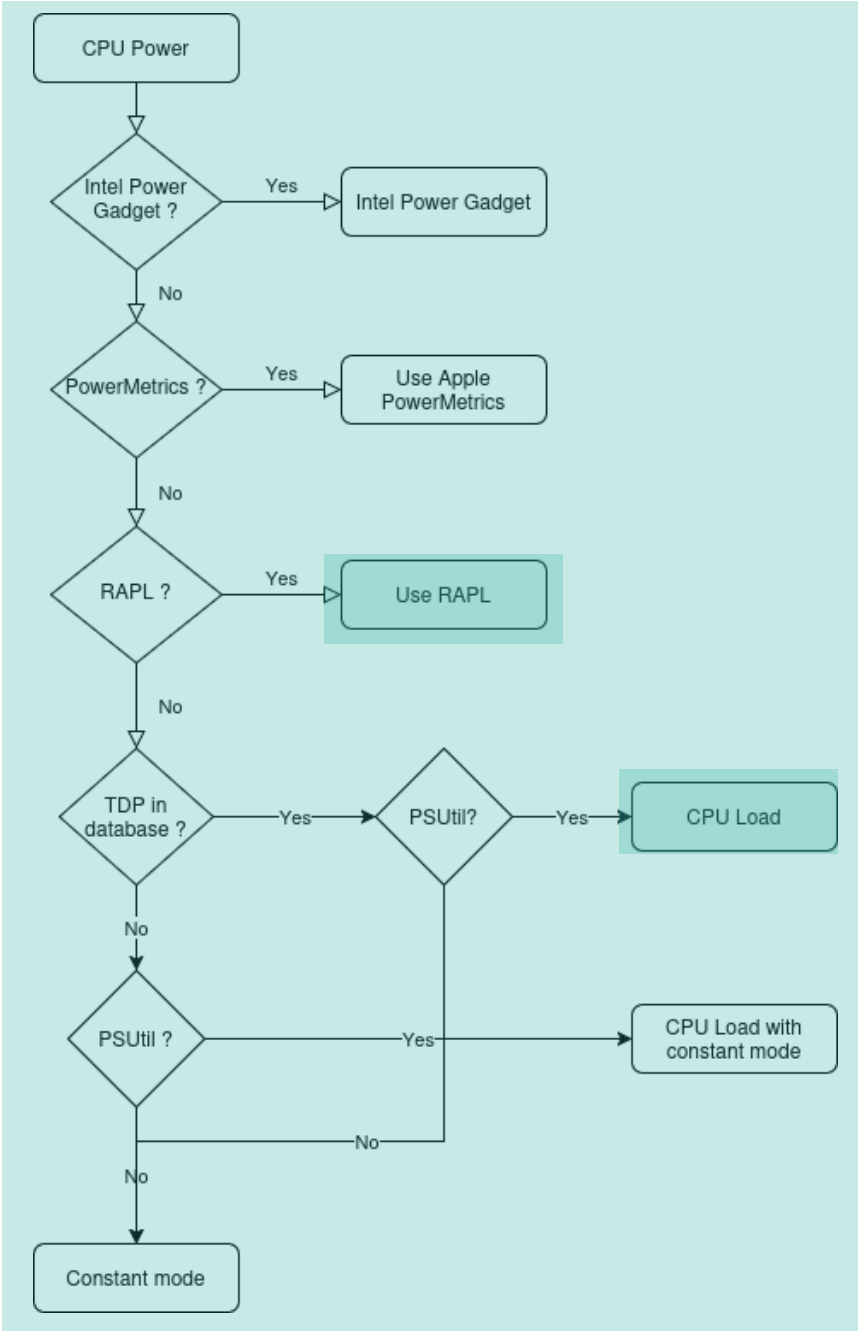


ENERGIE DU HARDWARE

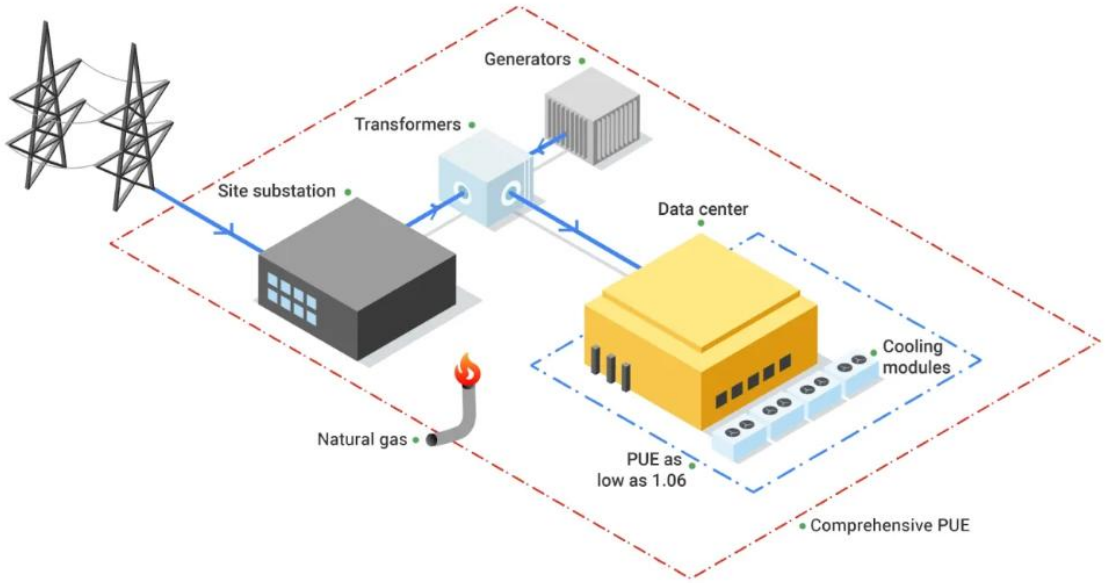
RAM: 5 Watts * Number of RAM slots used

- **CPU**: Intel and AMD via RAPL, Powermetrics, TDP database (Thermal Design Power)

- **GPU**: consommation d'énergie des GPU Nvidia à l'aide du package **nvidia-ml-py** (installée avec codecarbon).



PERTES D’ENERGIE

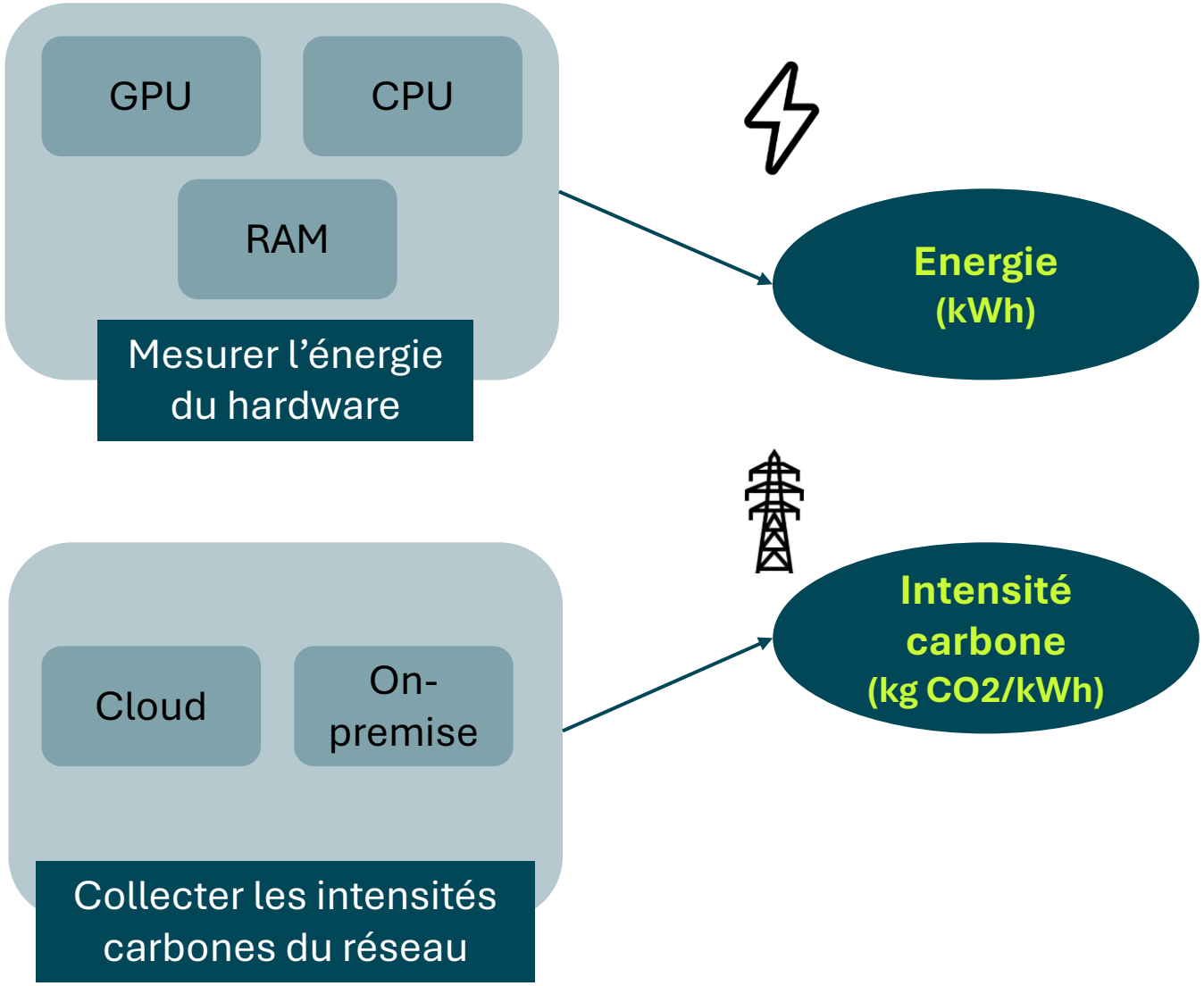


$$PUE = \frac{\text{E par le data center}}{\text{E serveurs informatiques}}$$

PUE moyen en France de **1.65**

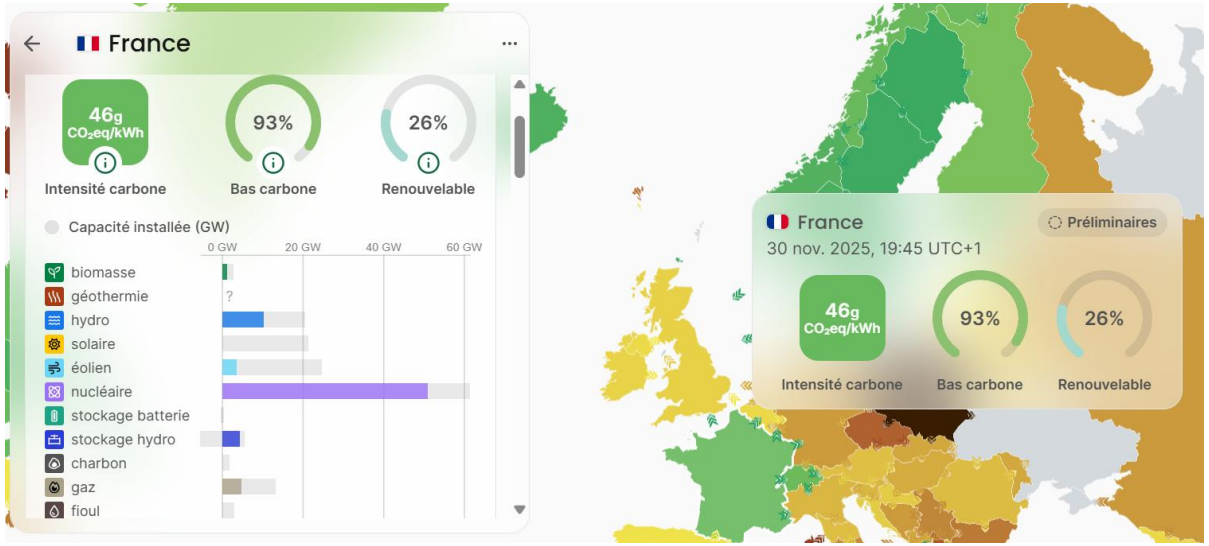
PUE moyen dans le monde de **1.58**

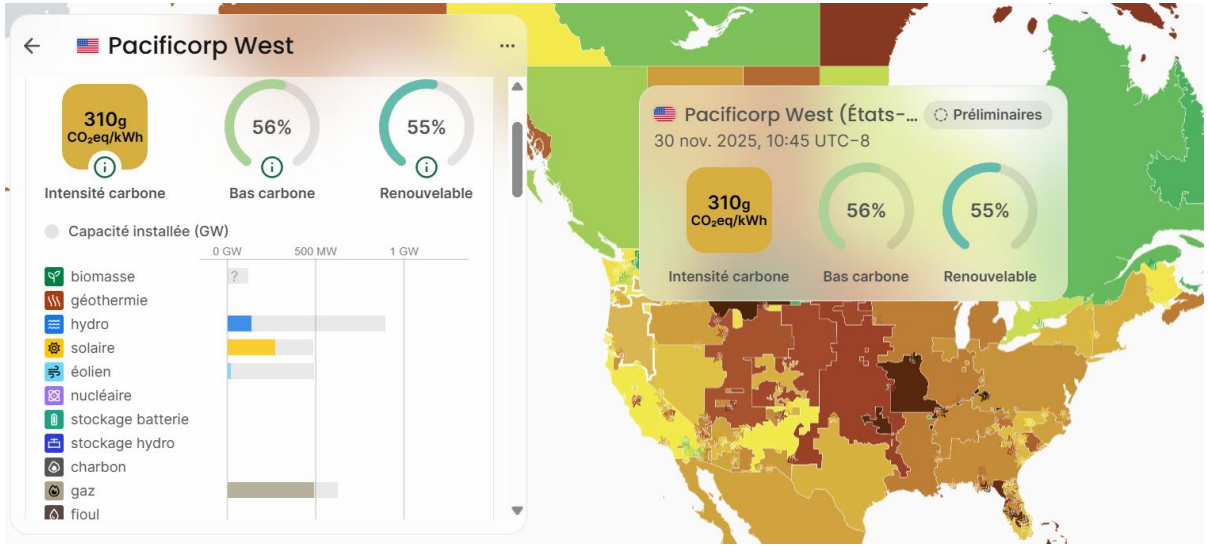
METHODOLOGIE - ENERGIE x INTENSITE CARBONE



INTENSITE CARBONE

Intensités carbones horaires en live (ou historique agrégées) dans (presque) tous les pays
Données fournies dans les paramètres ou collectées avec une API





Lien vers ELECTRICITY MAPS

METHODOLOGIE - EMISSIONS

GPU

CPU

RAM

Mesurer l'énergie du hardware

Energie (kWh)

Cloud

On-premise

Collecter les intensités carbones du réseau

Intensité carbone (kg CO2/kWh)

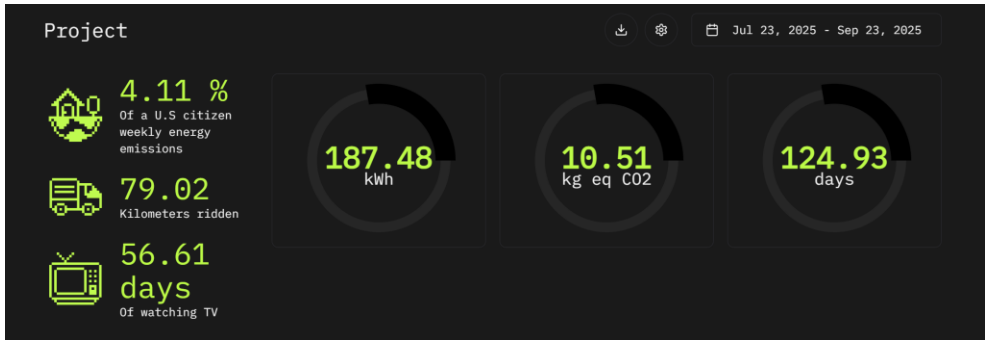
Emissions (kg CO2 eq)

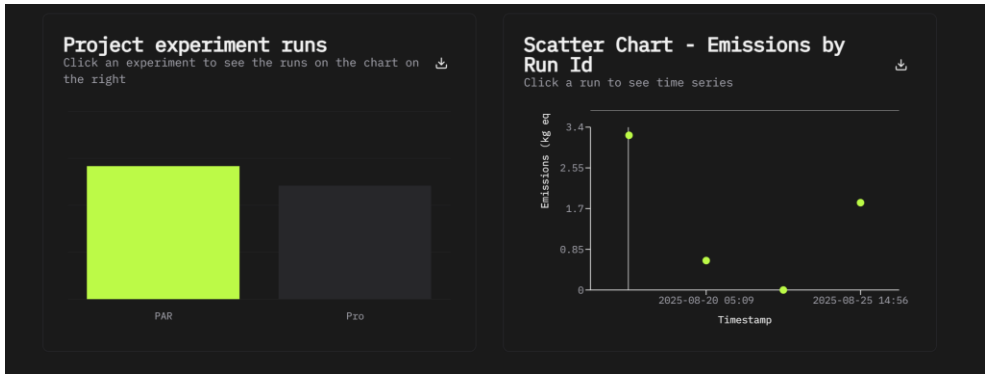
Convertir en émissions avec les intensités carbone en live

Centraliser les données utilisateurs avec API CodeCarbon et afficher sur dashboard

DASHBOARD

Mettre les émissions en perspective avec leurs équivalents dans le monde réel.





Comparer les émissions en fonction des infrastructures et de la consommation d'énergie.

Visualisez les émissions en direct pendant l'exécution de votre code.

UTILISATION

COMMENT UTILISER LE PACKAGE?

Interface de ligne de commande (pas de code Python) : suivre chaque ligne de code toutes les n secondes (15 sec par défaut)

-Créer un fichier de configuration : `codecarbon config`
-Lancer le monitoring : `codecarbon monitor`

Pour suivre des fonctions spécifiques:

- Appelez le traqueur d'émissions en définissant un **Object Explicit** , **Context manager** ou en tant que **décorateur** dans votre code Python.

- Utilisez le callback pour suivre les émissions à la fin de chaque époque, batch, etc.

IMPLEMENTATION EN QUELQUES LIGNES

Explicit Object

```python
from codecarbon import EmissionsTracker
tracker = EmissionsTracker()
tracker.start()
try:
    # Compute intensive code goes here
    _ = 1 + 1
finally:
    tracker.stop()
```

Decorator

```python
from codecarbon import track_emissions

@track_emissions
def training_loop():
    # Compute intensive training code goes here
```

Context Manager

```python
from codecarbon import EmissionsTracker

with EmissionsTracker() as tracker:
    # Compute intensive training code goes here
```

Measuring emissions per defined tasks

```python
try:
    tracker = EmissionsTracker(project_name="bert_inference", measure_power_secs=10)
    tracker.start_task("load dataset")
    dataset = load_dataset("imdb", split="test")
    imdb_emissions = tracker.stop_task()
    tracker.start_task("build model")
    model = build_model()
    model_emissions = tracker.stop_task()
finally:
    _ = tracker.stop()
```

Task 1

Task 2

COMMENT SAUVEGARDER LES EMISSIONS ?

Sauvegarder les données en format csv

	timestamp	run_id	duration	emissions_sum	energy_consumed	experiment_id	experiment_name	country_nar
0	2021-07-04T06:3(	9780b248-7	23	0.000266254	0.000627094	5b0fa12a-3dd7	Code Carbon user test	France
1	2021-07-04T06:4:	95ae3555-a:	122	0.002314277	0.005450693	5b0fa12a-3dd7	Code Carbon user test	France
2	2021-07-04T06:4!	95ae3555-a:	96	0.001747437	0.004115646	5b0fa12a-3dd7	Code Carbon user test	France
3	2021-07-04T06:4(	8f42a9e2-da	32	0.000553807	0.001304351	5b0fa12a-3dd7	Code Carbon user test	France
4	2021-07-04T06:4(	8f42a9e2-da	27	0.000565148	0.001331063	5b0fa12a-3dd7	Code Carbon user test	France
5	2021-07-04T06:4:	8f42a9e2-da	30	0.000552322	0.001300855	5b0fa12a-3dd7	Code Carbon user test	France
6	2021-07-04T06:4:	8f42a9e2-da	29	0.000573915	0.001351711	5b0fa12a-3dd7	Code Carbon user test	France
7	2021-07-04T06:4:	8f42a9e2-da	19	0.000256633	0.000604434	5b0fa12a-3dd7	Code Carbon user test	France
8	2021-07-06T19:2!	0dc75936-3	7	3.36E-05	7.91E-05	5b0fa12a-3dd7	Code Carbon user test	France
9	2021-07-06T19:2!	55ab25f7-c6	3	1.49E-05	3.51E-05	5b0fa12a-3dd7	Code Carbon user test	France
10	2021-07-06T19:3:	701df583-ae	4	8.58E-07	2.02E-06	5b0fa12a-3dd7	Code Carbon user test	France
11	2021-07-08T07:3:	252ccebc-ac	122	0.001412031	0.003325681	5b0fa12a-3dd7	Code Carbon user test	France
12	2021-07-08T07:3-	252ccebc-ac	100	0.001218871	0.002870743	5b0fa12a-3dd7	Code Carbon user test	France
13	2021-07-09T08:4:	7f596dc0-76	122	0.001412034	0.003325689	5b0fa12a-3dd7	Code Carbon user test	France
14	2021-07-09T08:4:	7f596dc0-76	78	0.000962114	0.002266017	5b0fa12a-3dd7	Code Carbon user test	France
15	2021-07-09T08:4:	8866c136-9	122	0.001415204	0.003333156	5b0fa12a-3dd7	Code Carbon user test	France
16	2021-07-09T08:4!	8866c136-9	107	0.00130631	0.003076683	5b0fa12a-3dd7	Code Carbon user test	France
17	2021-07-09T09:0!	f877e89d-31	20	0.000208158	0.000490263	5b0fa12a-3dd7	Code Carbon user test	France
18	2021-07-09T09:0:	63d5a122-f:	3	6.28E-07	1.48E-06	5b0fa12a-3dd7	Code Carbon user test	France

Ou autres: Webhook, LogFire, Prometheus

Sauvegarder les données avec l'API Code Carbon : en créant un compte pour l'organisation

Projects

- POST `/project` Add Project
- GET `/project/{project_id}` Read Project
- GET `/projects/team/{team_id}` Read Projects From Team

Experiments

- POST `/experiment` Add Experiment
- GET `/experiment/{experiment_id}` Read Experiment
- GET `/experiments/project/{project_id}` Read Experiment Experiments

Runs

- POST `/run` Add Run
- GET `/run/{run_id}` Read Run
- GET `/runs` List Runs
- GET `/runs/experiment/{experiment_id}` Read Runs From Experiment

Emissions

- POST `/emission` Add Emission
- GET `/emission/{emission_id}` Read Emission
- GET `/emissions/run/{run_id}` Get Emissions From Run

DOCUMENTATION DETAILLEE

La documentation

La configuration
https://mlco2.github.io/codecarbon/usage.html#configuration

Les paramètres de méthodologie
https://mlco2.github.io/codecarbon/parameters.html

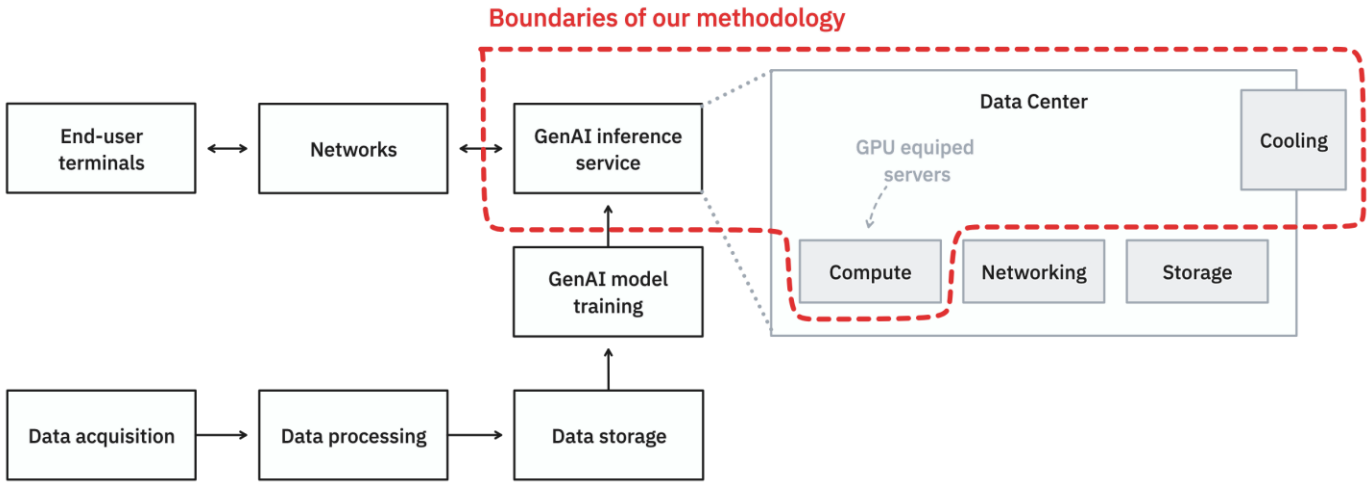
Le site web :
https://codecarbon.io/

AUTRES OUTILS

EMISSIONS DES INFERENCES DES LLMs

EcoLogits

Un outil pour mesurer la consommation d'énergie et les impacts environnementaux de l'utilisation de modèles d'IA générative par le biais d'API.
Il prend en charge les principaux fournisseurs de LLM tels que OpenAI, Gemini, Mistral AI et bien d'autres.



Émissions de carbone d'une requête (environ 400 tokens de sortie)

Tool	Emissions in grams of CO2
Google Search	**0,2**
Mistral AI 3b	**0,92**
Google Gemini Pro	**2,99**
OpenAI GPT4	**3,18**
Drive 1 km in fuel car	***140***

Mesurer d'autres modèles et d'autres tâches avec leur:
- Webapp
- package python

OUTILS D’OPTIMISATION DE MODELES

Pruna AI

Package de compression d’algorithmes

Pruners
Quantizers
Cachers
Compilers

Documentation

BONNES PRATIQUES

BONNES PRATIQUES - GESTION DE PROJET

- Considérer d'autres approches de performance: FRUGAL AI CHALLENGE

    Satisfaire un seuil minimum de performance de modèles

    Optimiser la consommation énergétique des algorithmes

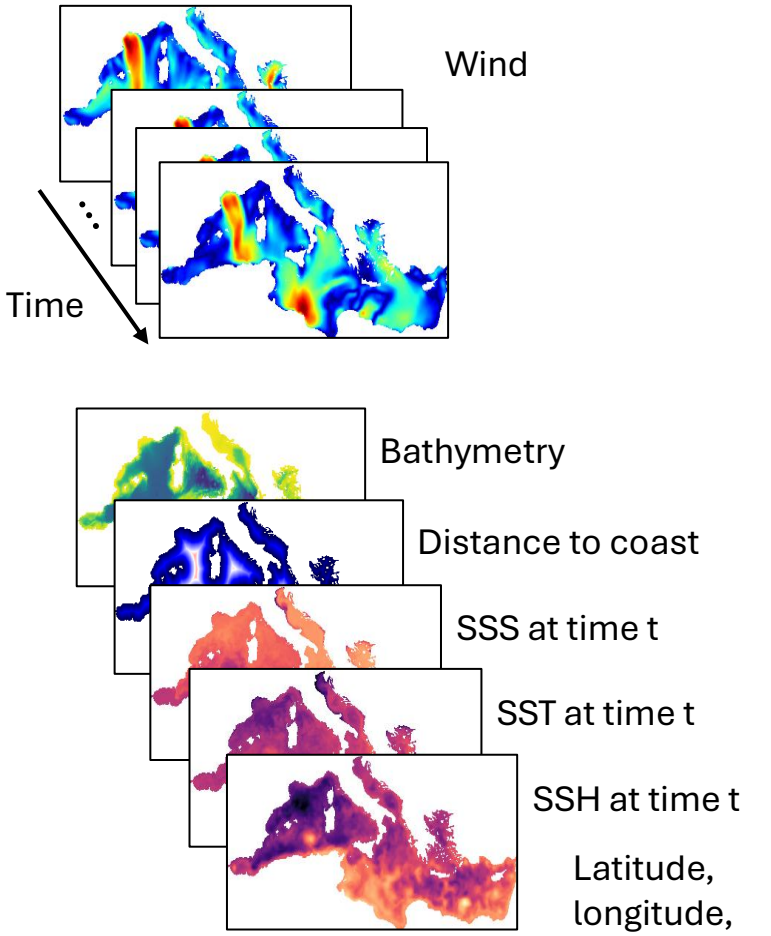
- Choix du hardware et leur durée de vie
- Si sur le cloud, choisir la région en fonction des mix électriques
- Demander et rester attentif à la transparence des GAFAMS
- Suivre: Dr. Sasha Luccioni (Head Climate à Hugging Face) très active sur Linkedin

BONNES PRATIQUES – CODEURS.EUSES

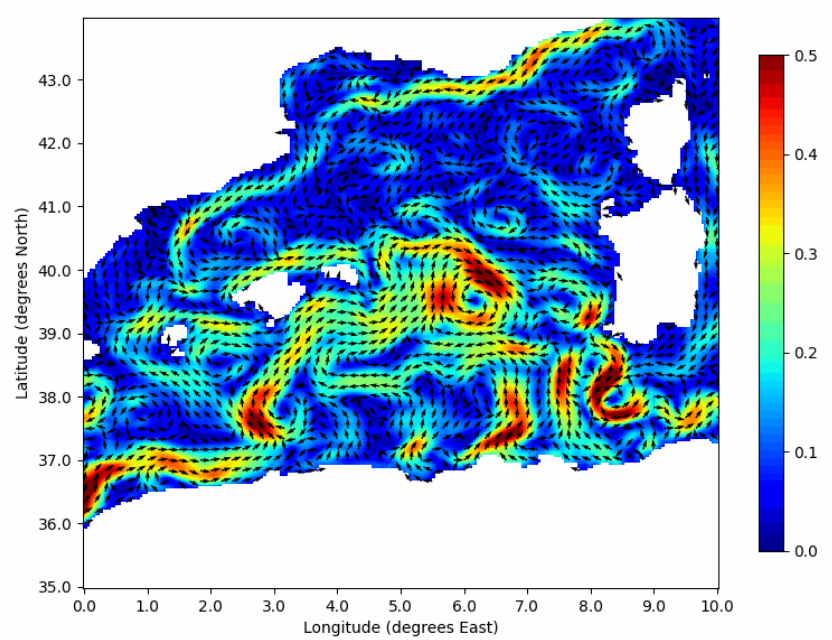
- For machine learning and deep learning training:
- Optimiser son code grâce à des fonctions timeit
- Fine-tunner des modèles
- Caching
- Mesurer l’entraînement et l’inférence des modèles
- Viser à satisfaire des requirements et non à développer des modèles état de l’art
- Utiliser GPU à la place de CPU
- Parameter tuning avec des recherches bayesian (plutôt que brute force)
- Model pruning pour réduire les coûts d’inférence

EXEMPLE DE PROJET PROFESSIONNEL

Reconstruction des courants de surface de la Méditerranée

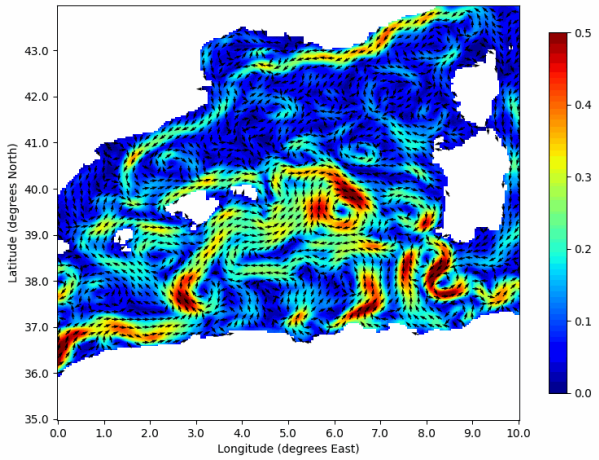


Traitement d'images
par Deep Learning

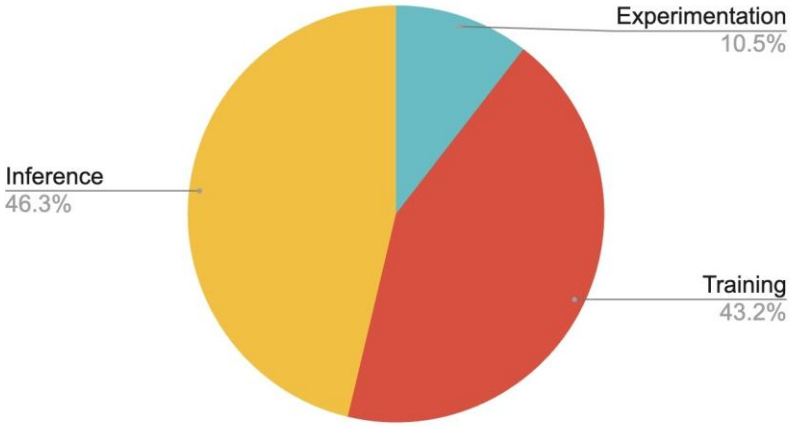


EXEMPLE DE PROJET PROFESSIONNEL

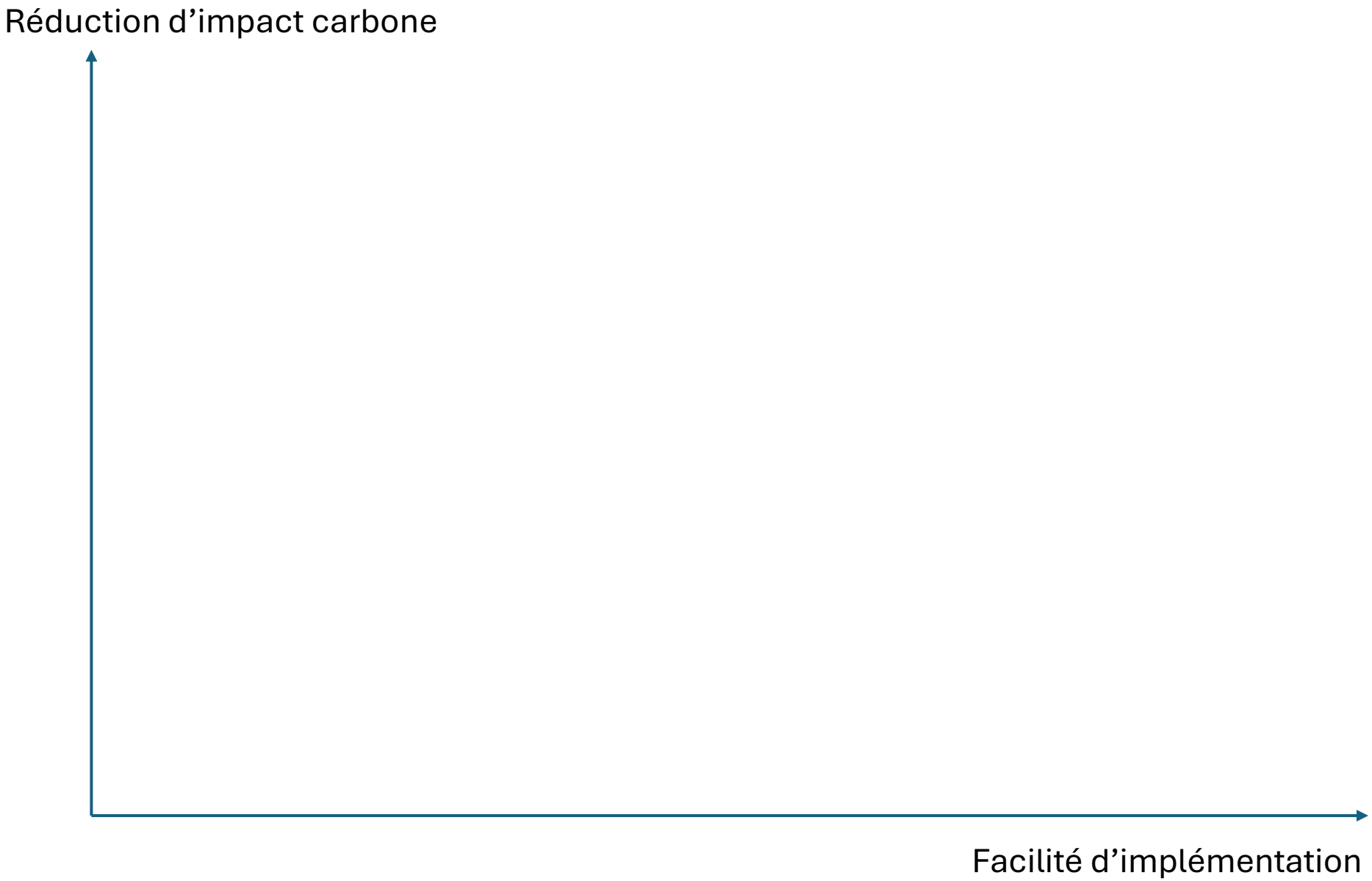
Réduction d’empreinte carbone dans la phase de développement des modèles:
Entraîner le modèle sur une résolution moins fine puis fine-tuner sur une résolution plus fine



Pour une mise en production avec des interférences fréquentes, un modèle simple comme un Unet entraîné « from scratch » peut être plus optimal qu’un modèle de type Foundation Model fine-tuné sur moins de données



BONNES PRATIQUES – CODEURS.EUSES



- Optimiser son code grâce à des fonctions timeit
- Model pruning pour réduire les coûts d'inférence
- Fine-tuner des modèles
- Caching
- Ne pas toujours viser à développer des modèles état de l'art
- Parameter tuning avec des recherches Bayesian (plutôt que brute force)
- Mesurer l'entraînement et l'inférence des modèles
- Utiliser GPU à la place de CPU

MERCI !

A VOUS DE JOUER