

# **Accelerated Data Science: A Paradigm Shift for Data Wrangling, Analysis and Machine Learning**


William Hill  
Sr. Developer Advocate



# Speaker Intro Who am I?

William Hill

[linkedin.com/in/accelerated-ai/](https://www.linkedin.com/in/accelerated-ai/)



## NVIDIA Developer

@NVIDIADeveloper · 185K subscribers · 1.1K videos

Welcome to the NVIDIA Developer YouTube Channel ...more

[developer.nvidia.com/developer-program](https://developer.nvidia.com/developer-program) and 1 more link

Customize channel Manage videos

Home Videos Shorts Live Playlists Posts



Faster pandas on  
Kaggle

3.9K views



Here's How One Line Of  
Code Can Accelerate ...

3.2K views



Here's How Rescale Is  
Using CUDA For Jet ...

4.4K views



## Higher Education Research

[Industries](#) / [Higher Education Research](#) / [Academic Grant Program](#)

# Accelerating Innovation in Academia

Submit your research proposal to the NVIDIA Academic Grant Program.

 Feedback

[Benefits](#) [Program Requirements](#) [Calls for Proposal](#) [Use Cases](#) [FAQ](#)

[Apply Now](#)

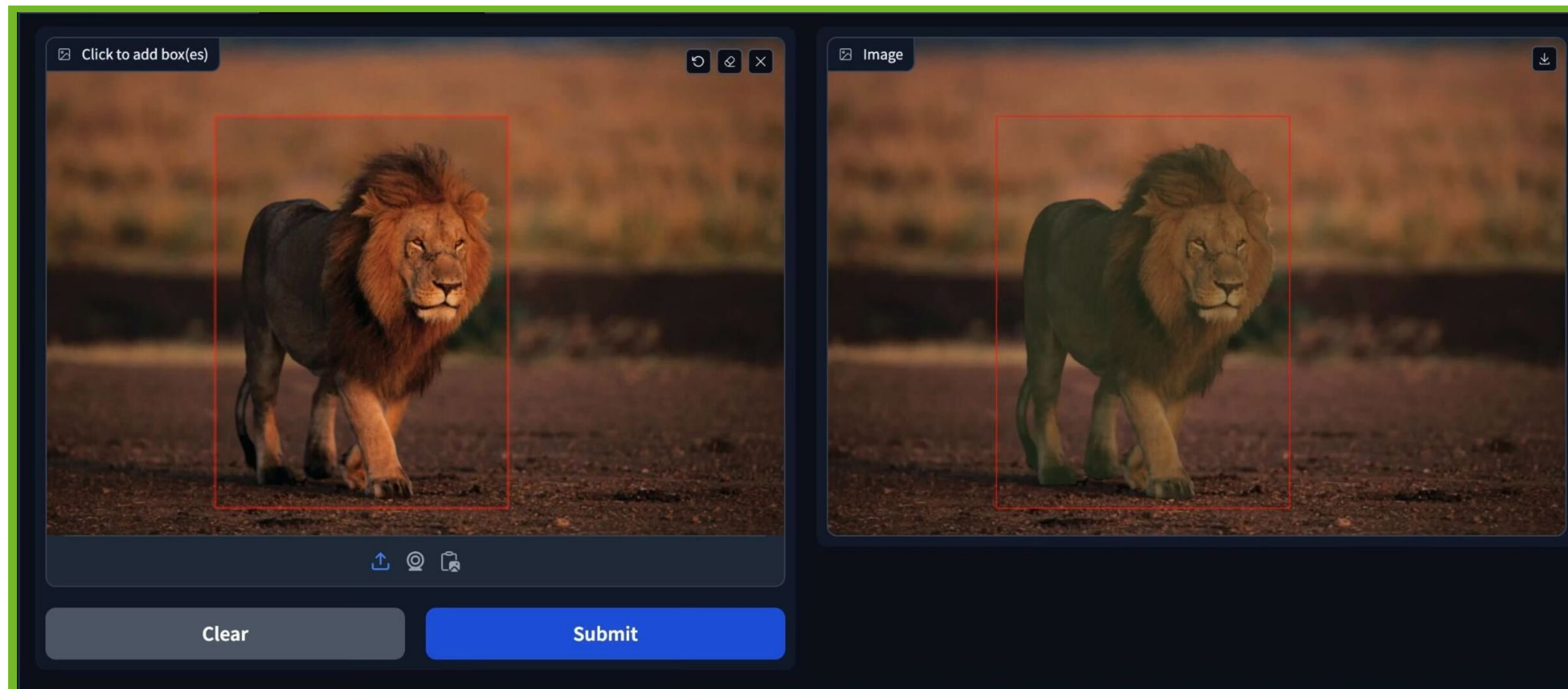
## Advancing Academic Research

The NVIDIA Academic Grant Program advances academic research by providing world-class computing access and resources to researchers.



# Don't Perish! Publish & *Promote*

Public & Internal Promotion



## EfficientViT-SAM: Accelerated Segment Anything Model Without Performance Loss

Zhuoyang Zhang<sup>1,3\*</sup>, Han Cai<sup>2,3\*</sup>, Song Han<sup>2,3</sup>

<sup>1</sup>Tsinghua University, <sup>2</sup>MIT, <sup>3</sup>NVIDIA

<https://github.com/mit-han-lab/efficientvit>

### Getting Started

We start by installing the recommended dependancies from the repository README

```
!wget -L https://raw.githubusercontent.com/mit-han-lab/efficientvit/master/requirements.txt
!pip install openmpi
!pip install -r requirements.txt
```

```
!git clone https://github.com/NVIDIA-AI-IOT/torch2trt
!cd torch2trt && python setup.py install
```

```
!cd torch2trt && cmake -B build . && cmake --build build --target install && ldconfig
```



# AI-Enabled Data Compression

The O'Donnell Institute scientists are developing several AI-enabled data compression algorithms and testing them on SMU's HPC platforms. Data compression is a key research area focused on reducing the size of digital data, such as text, images, video, or audio, to enable faster transmission and more efficient storage. Unlike traditional rule-based techniques that depend on fixed statistical models, modern AI/ML methods use neural network and deep learning architectures including Convolutional Neural Networks (CNNs), Autoencoders, Transformers, and Generative Adversarial Networks (GANs) to learn complex patterns directly from large datasets. These approaches achieve higher compression ratios and improved perceptual quality compared to classical algorithms by capturing both local and global dependencies. AI-enabled data compression has gained momentum in recent years due to its impact on optimizing communication, lowering infrastructure costs, and enabling data-intensive applications in fields such as medical imaging, satellite communications, and multimedia streaming.



# AREAD Initiative

The O'Donnell Institute recently launched the AREAD (***A**dvancing **R**esource-**E**fficient **A**I and Smart **D**ata Management*) initiative for energy-efficient AI solutions. As AI models and datasets continue to grow, optimizing computational efficiency and data utilization is essential for sustainability and scalability. Resource-efficiency may be achieved by leveraging lightweight AI models, adaptive processing techniques, smart data management, and resource-aware algorithms. The AREAD initiative supports projects in AI model optimization—including model compression, quantization, and lightweight models—as well as intelligent data storage, retrieval, and processing. Targeted application areas include but are not limited to AI-powered IoT, autonomous systems, AR/VR, and AI-driven digital twins.



# Federated Learning

Federated Learning (FL) is a decentralized machine learning paradigm that enables multiple clients to collaboratively train a global model while preserving data privacy. Unlike traditional centralized learning, FL ensures that raw data remains local, with only model updates being shared, thereby mitigating privacy risks and reducing communication overhead. FL has gained significant traction in research for its applications in privacy-preserving AI and efficiency in distributed systems, making it particularly valuable in domains such as healthcare, finance, and edge computing.




Unveiling Seismotectonics of the Subduction Zones: Leveraging  
Machine Learning Analysis of Ocean Bottom Seismometers  
(Professor [Heather De Shon](#))



# Fairness-aware Multimodal AI for Healthcare Data (Professors Eric Larson and Mehak Gupta)

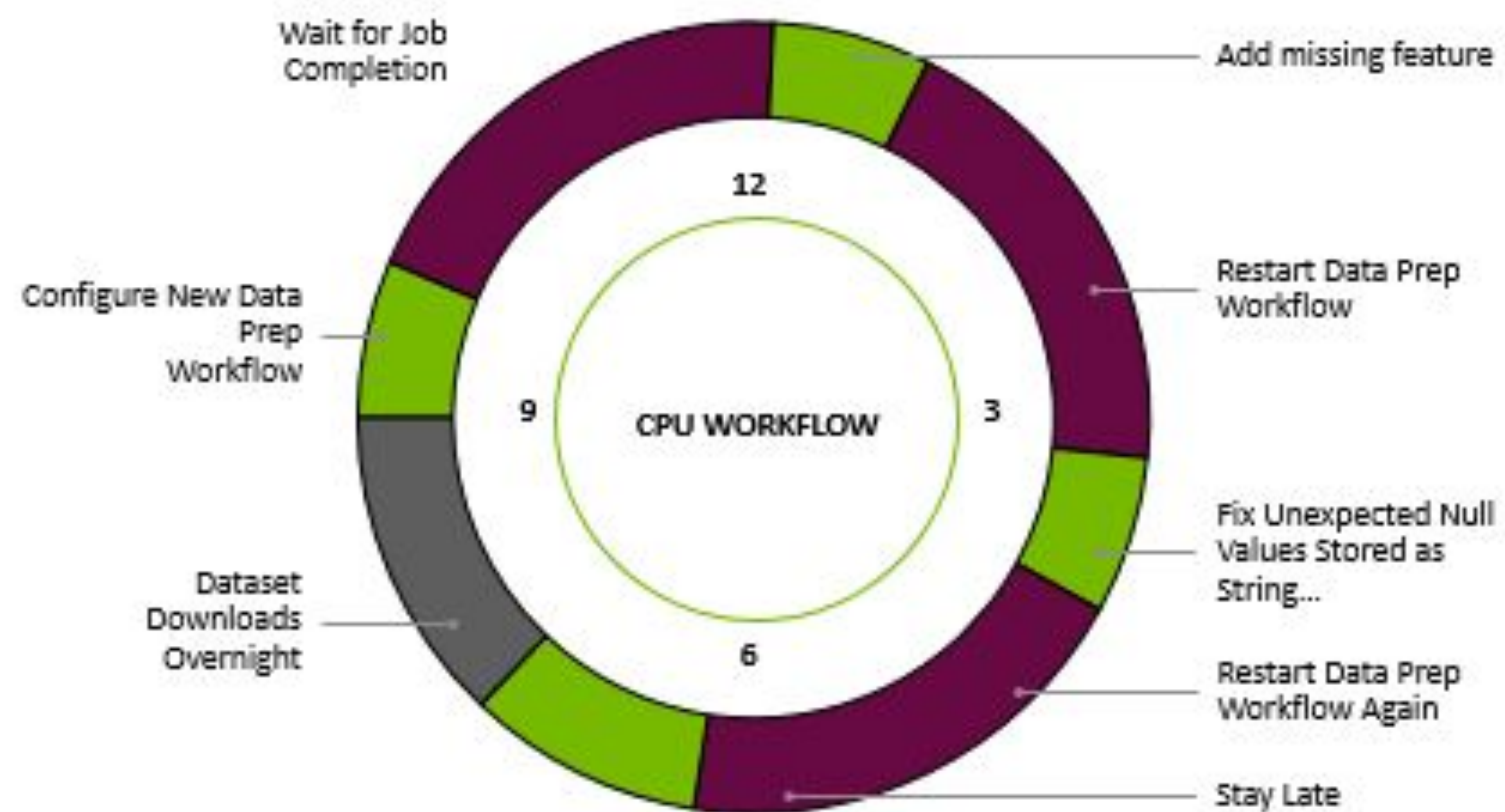




**In the before times...*of 2015***



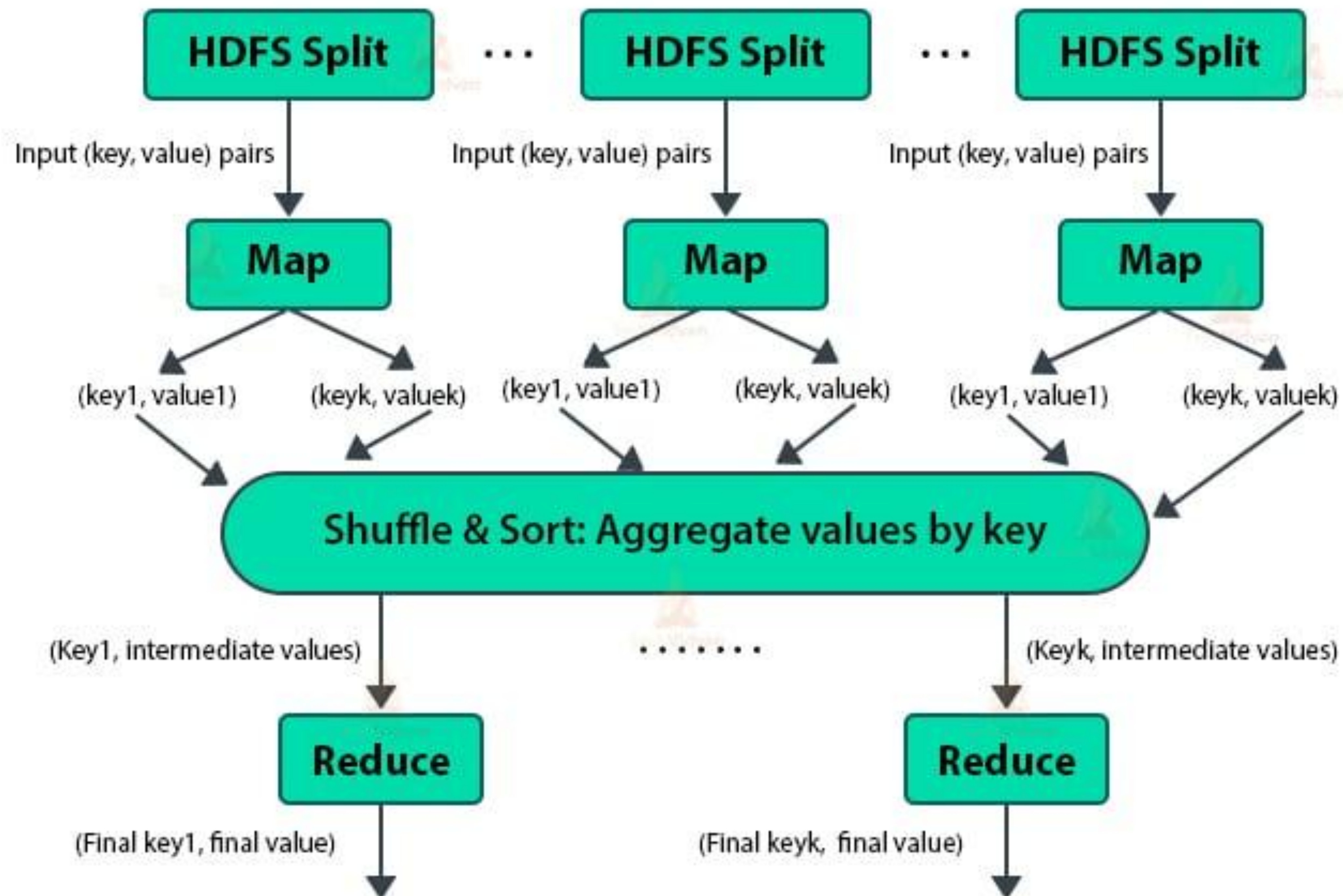
# Data Processing on CPUs Limits Iterations





# The Age of Big Data

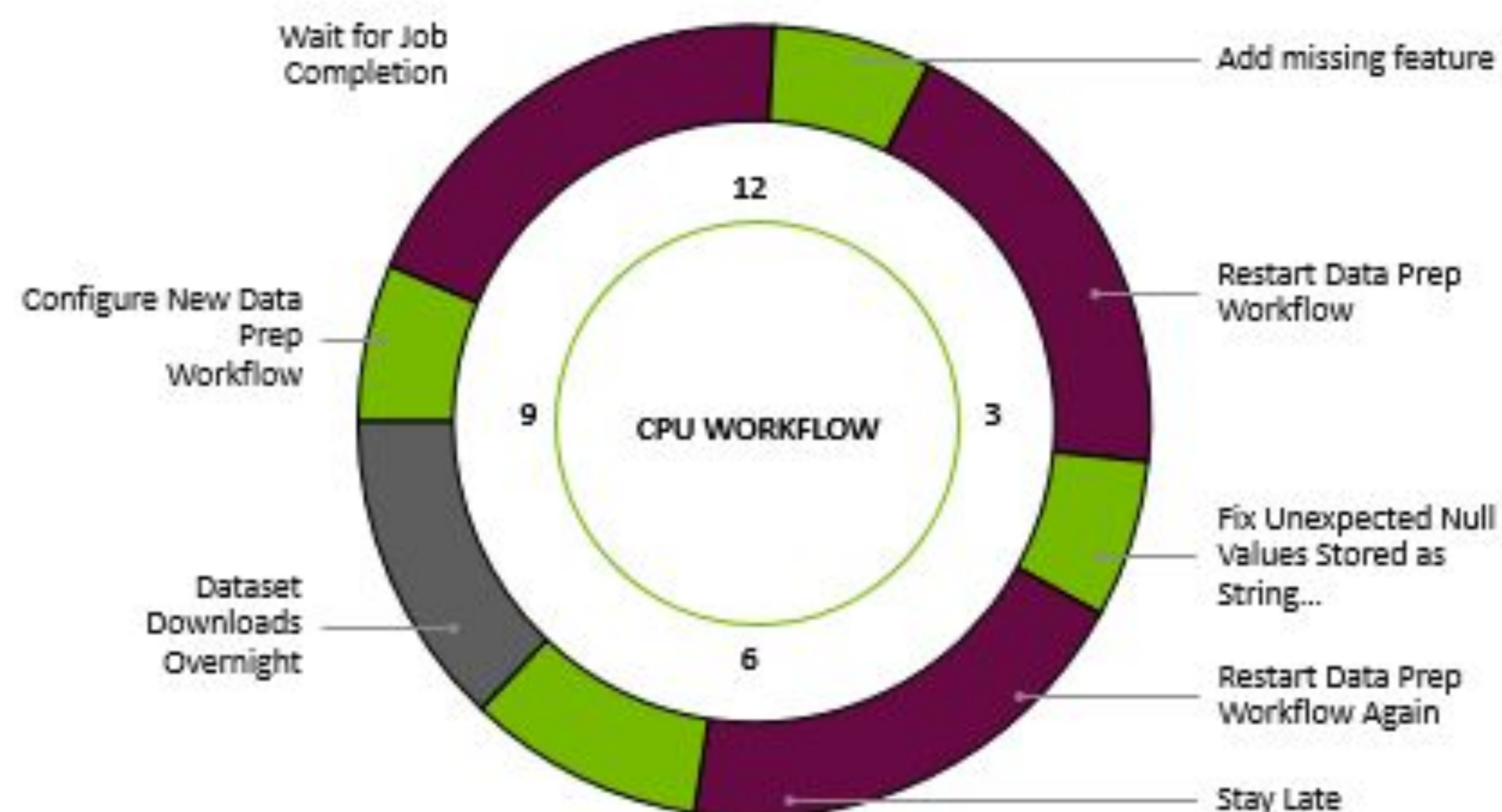
## Hadoop MapReduce



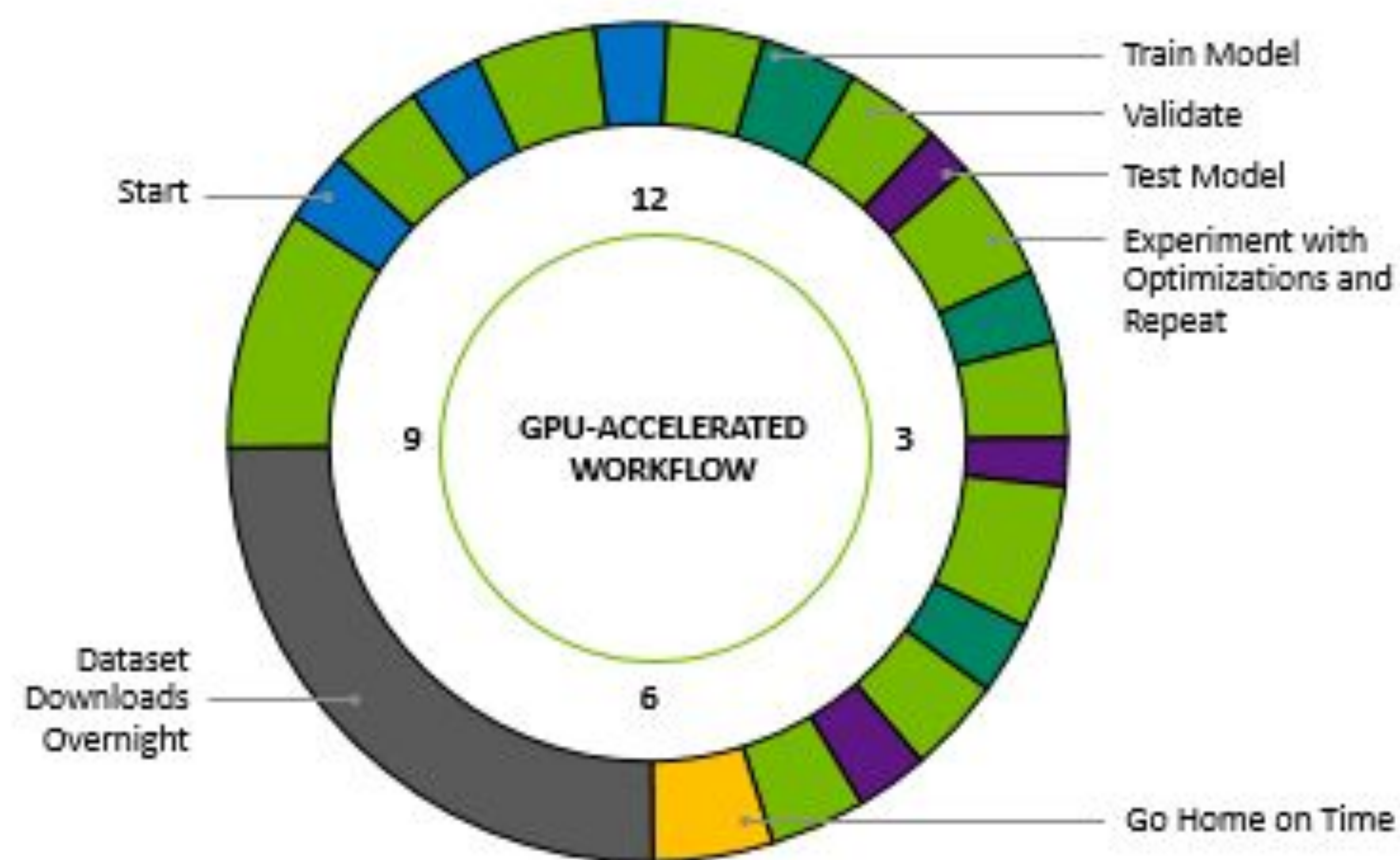


# Accelerated Computing Empowers Modern Data Teams

Faster Processing | No code change | More Iterations



Before

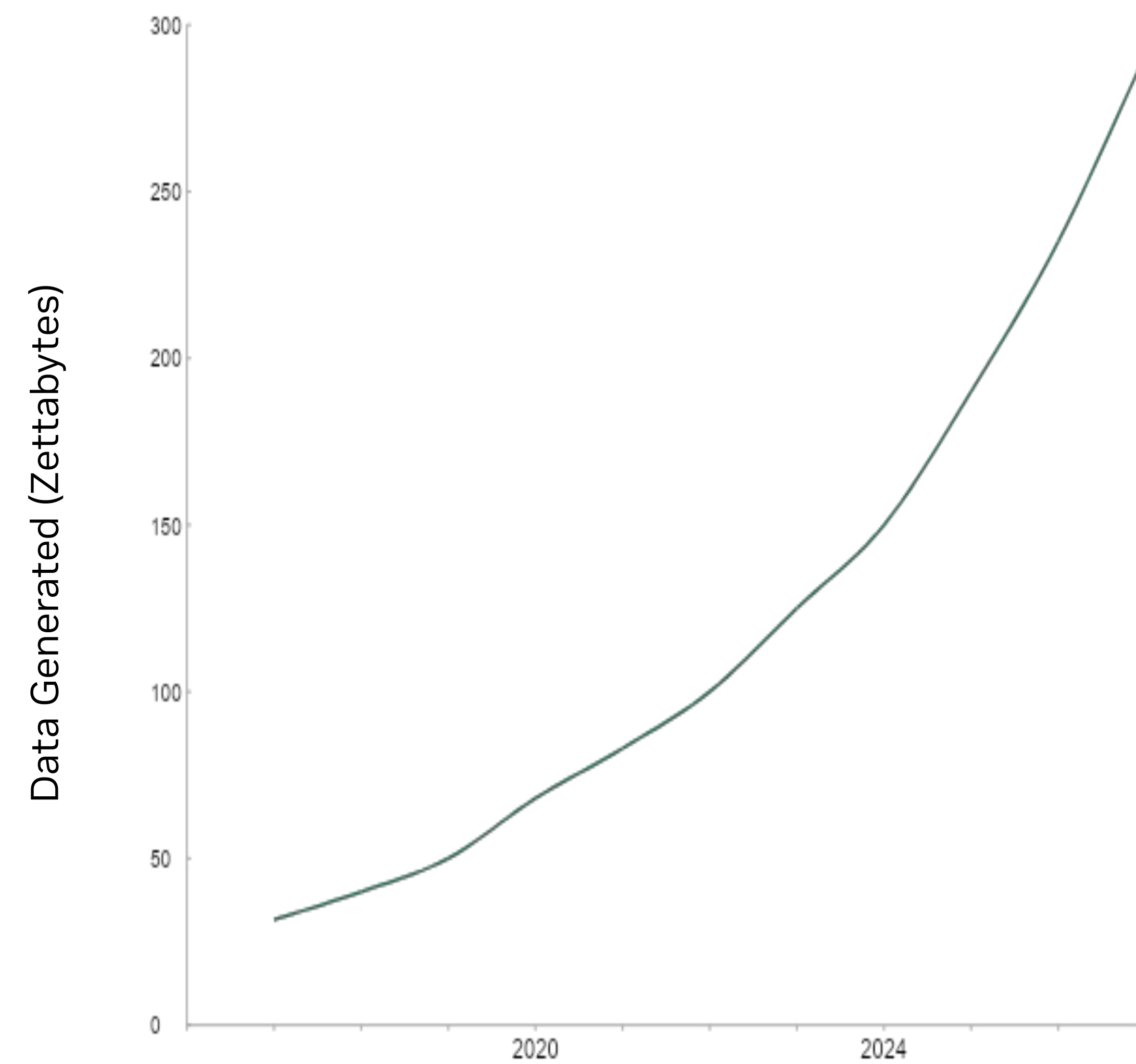


After

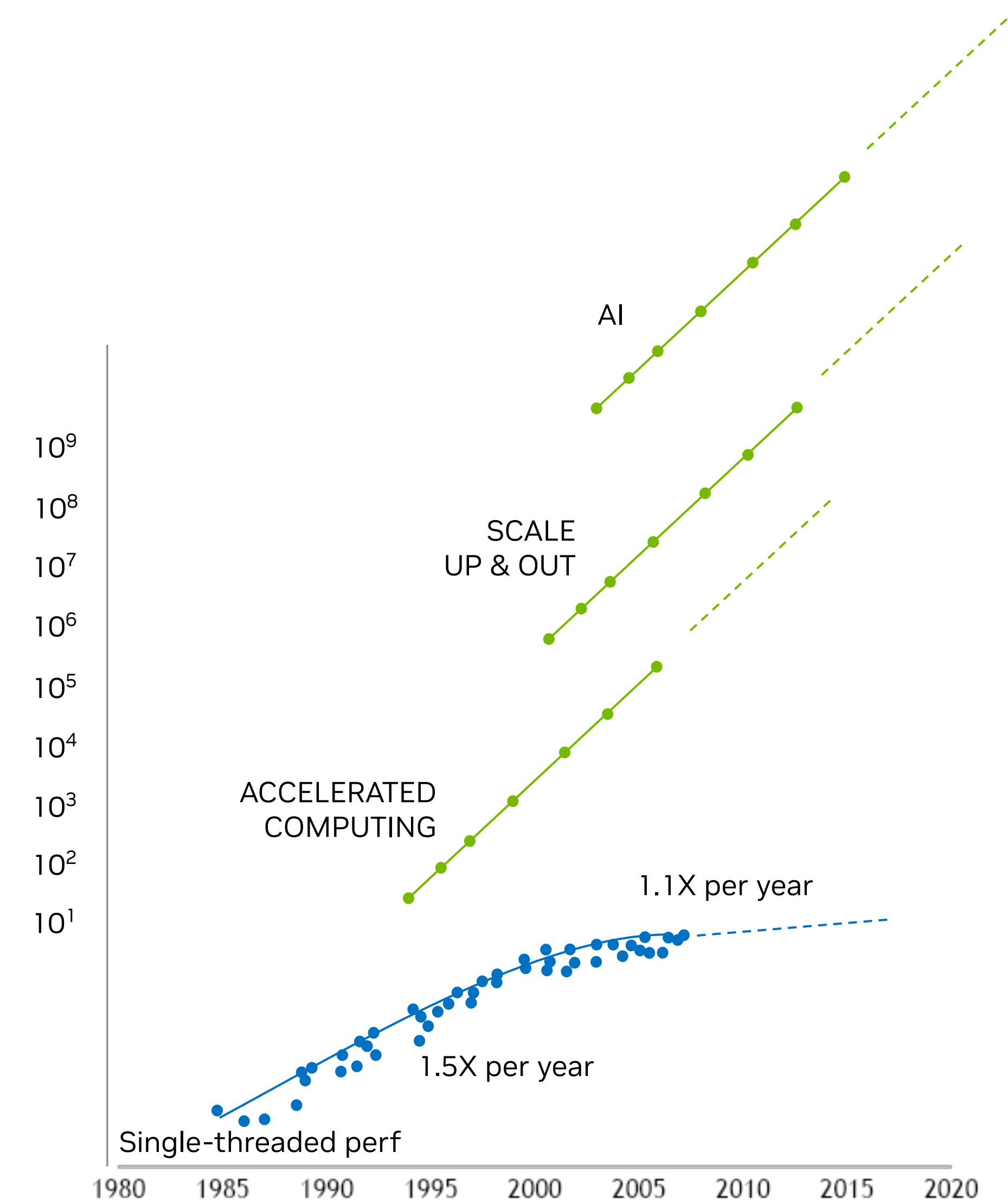


# Data Analytics Needs Accelerated Computing

Petabytes of Data Generated Yearly



Moore's Law Has Ended





the formula...

$$\text{value} = \text{utility} - \text{friction}$$

*results,  
novelty,  
publishable*

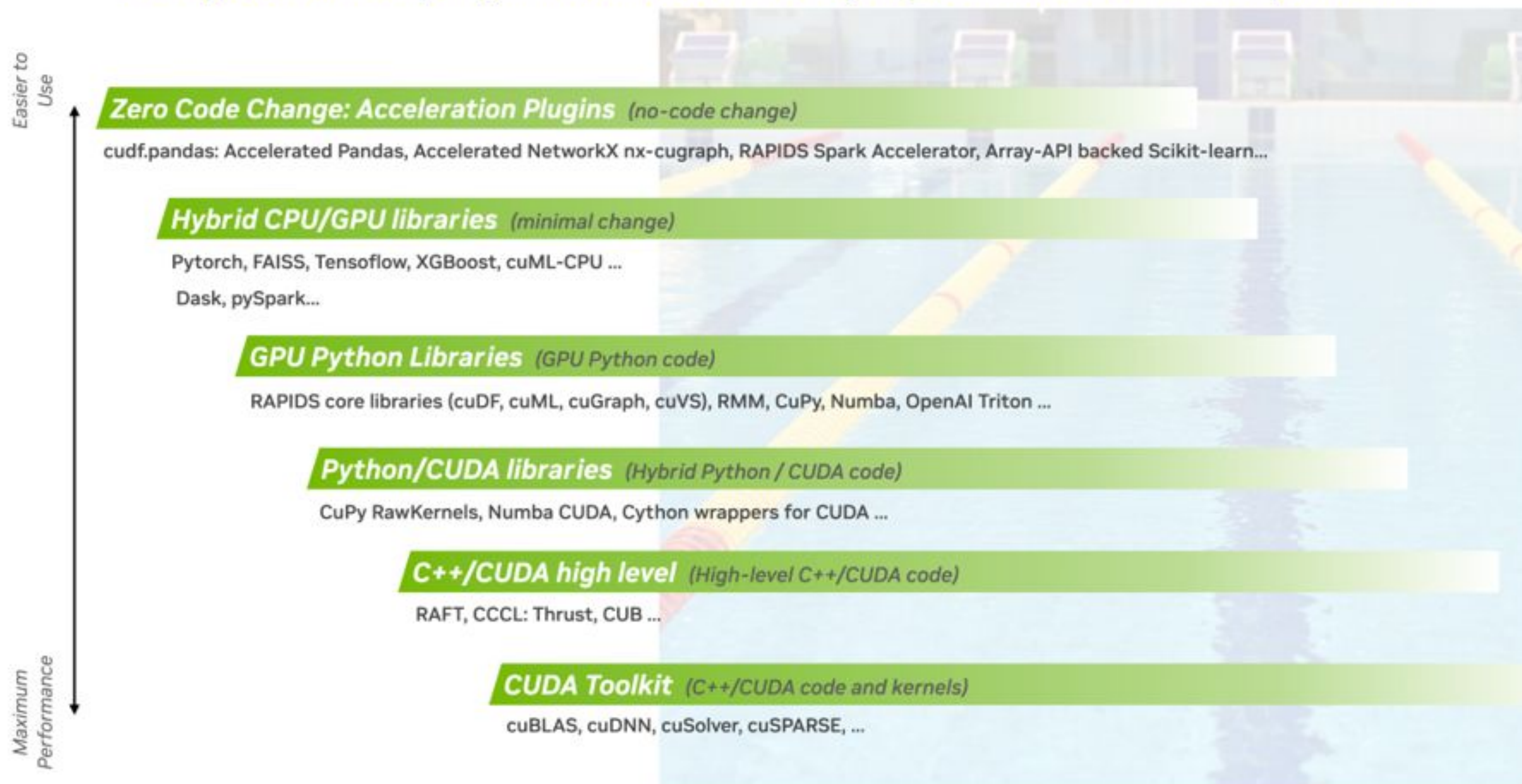
*execution  
speed*

*more code  
new API*



# Accelerated Computing Swim Lanes

Making accelerated computing more seamless while enabling deep customization for maximum performance



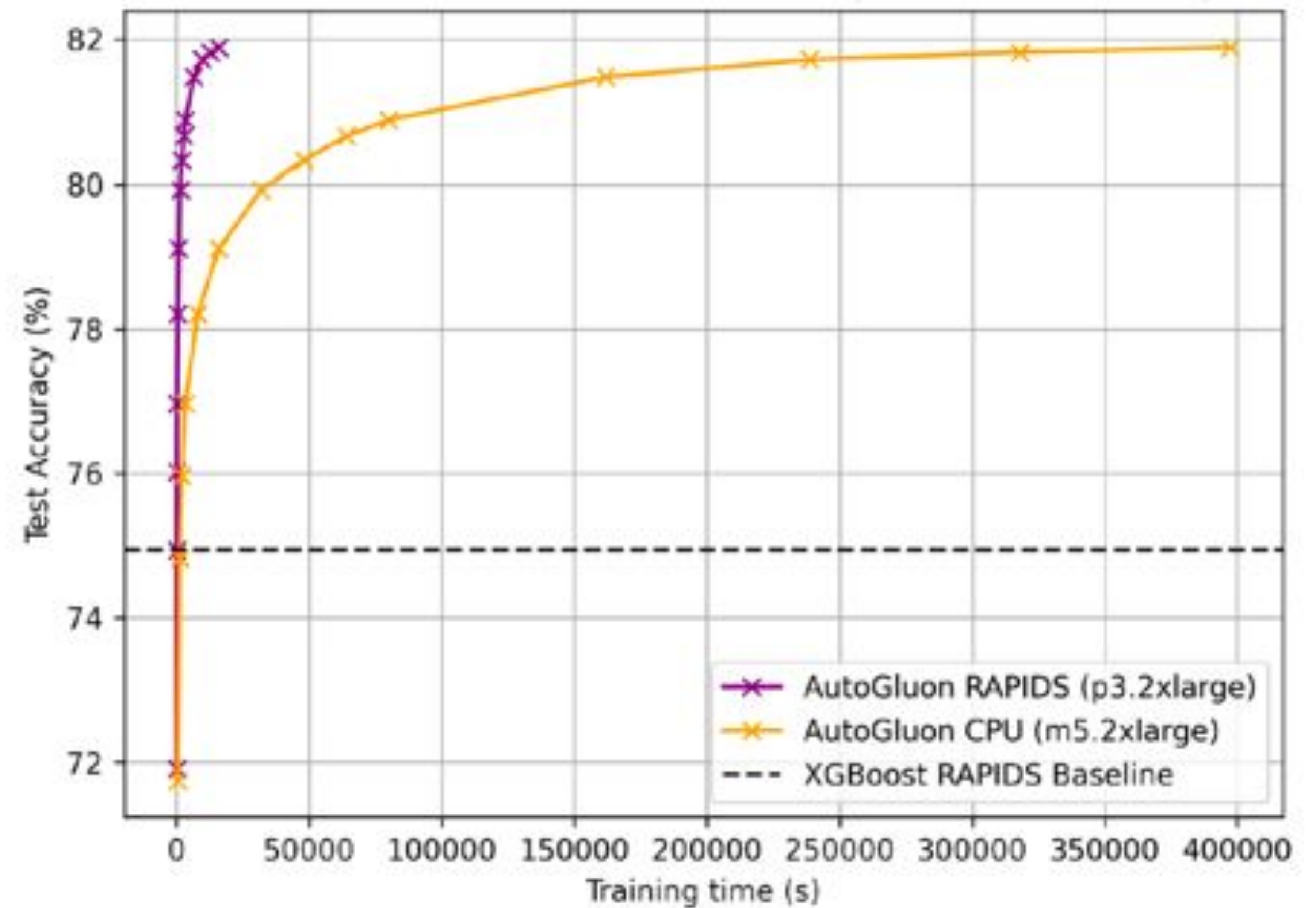


# Accelerated Computing Supercharges Hyperparameter Optimization

Run more experiments, faster and cheaper

- Building the best models requires finding the right combination of data and model
- Experimentation is compute-intensive, requiring both iterative data processing and model optimization
- Accelerated computing enables you to run more experiments, leading to better models in practice

**Amazon Case Study: GPU-accelerated experimentation reached an optimal model in 4 hours vs. 4.5 days on CPUs (at ¼ the cost)**






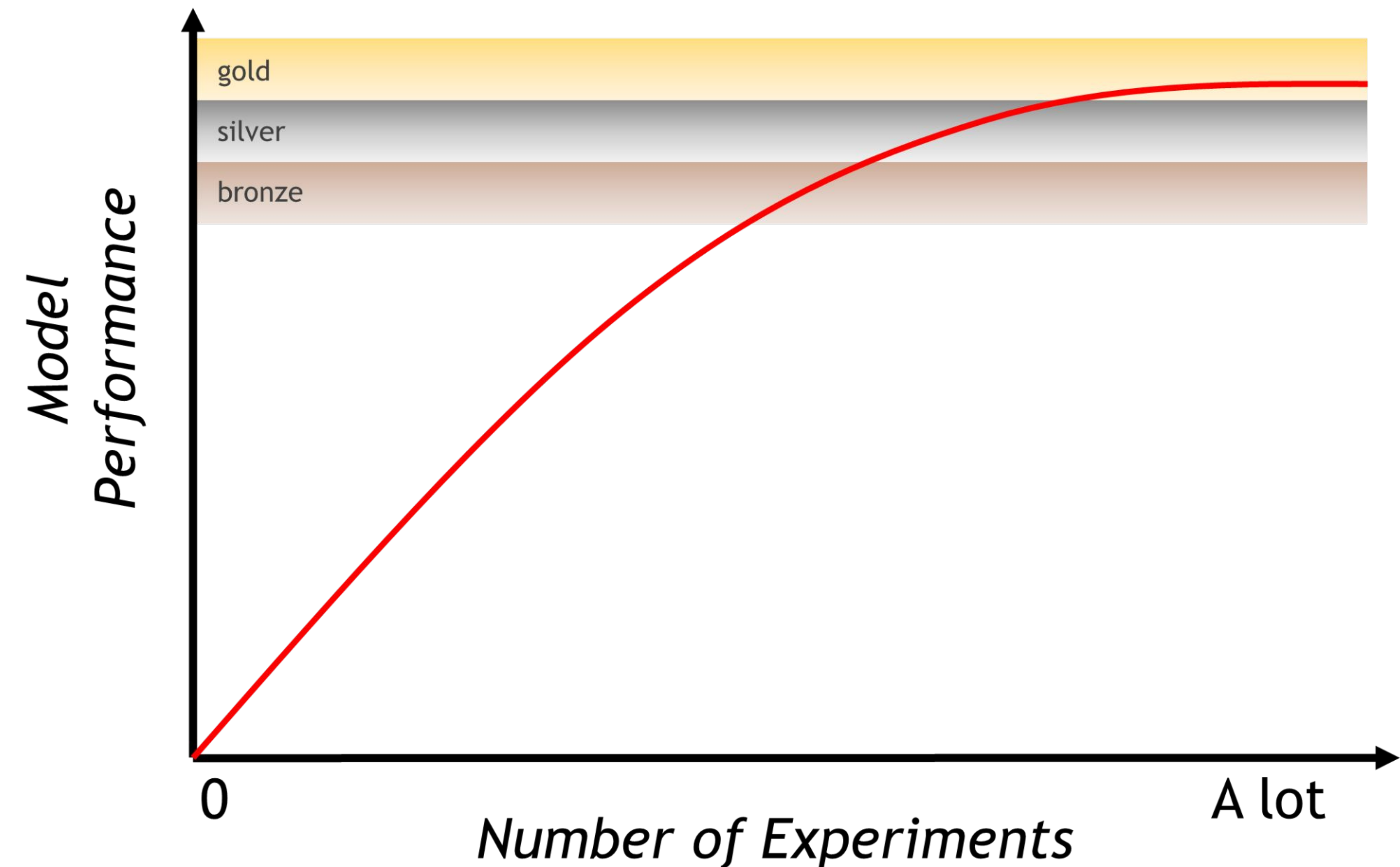
# Fast Experimentation

## Why Speed Matters

- The more you experiment ...
  - The better your models
  - The more you understand the data
  - The higher your chances of finding a winning idea
- ... But experimenting takes time :
  - Optimize your pipeline to be able to experiment more

### → Leverage GPU acceleration

- RAPIDS CuDF for Analytics
  - Pandas, but faster !
  -  Polars is also faster on GPU !
- RAPIDS CuML for Machine Learning



**RAPIDS**

Accelerated with







# Highly Structured, Tabular Data

aka

## Data Frames

Unveiling Seismotectonics of the Subduction Zones: Leveraging Machine Learning Analysis of Ocean Bottom Seismometers  
(Professor [Heather De Shon](#))

Fairness-aware Multimodal AI for Healthcare Data (Professors [Eric Larson](#) and [Mehak Gupta](#))





# Traditional Machine Learning

O'DONNELL DATA SCIENCE AND RESEARCH COMPUTING INSTITUTE · RESEARCH · FEDERATED LEARNING

# Federated Learning

Federated Learning (FL) is a decentralized machine learning paradigm that enables multiple clients to collaboratively train a global model while preserving data privacy. Unlike traditional centralized learning, FL ensures that raw data remains local, with only model updates being shared, thereby mitigating privacy risks and reducing communication overhead. FL has gained significant



```
from obspy.clients.fdsn import Client
from obspy import UTCDateTime

import pandas as pd
import numpy as np

import time
```

## Unveiling Seismotectonics of the Subduction Zones: Leveraging Machine Learning Analysis of Ocean Bottom Seismometers

(Professor [Heather De Shon](#))

```
# Fetch seismic data from multiple stations
client = Client("IRIS")
starttime = UTCDateTime("2024-01-01")
endtime = starttime + 7 * 86400 # 7 days of data
```

```
# Get data from multiple stations
stations = ["ANMO", "HRV", "CCM", "DWPF", "COR"]
network = "IU"
```

```
print("Fetching IRIS seismic data (7 days, 5 stations)...")
streams = []
for station in stations:
    try:
        st = client.get_waveforms(network, station, "00", "BHZ",
                                   starttime, endtime)

        streams.append(st)
        print(f" ✓ {station}: {len(st[0].data):,} samples")
    except Exception as e:
        print(f" ✗ {station}: {e}")
```





## 1. NHANES (via Python package)

python

```
import pandas as pd
from nhanesdata import nhanes
```

```
# Load NHANES data directly
df = nhanes.load_NHANES_data()
```

```
# Or specific years and tables
```

```
demo = nhanes.load_NHANES_data(year=2017, data_type='DEMO')
biopro = nhanes.load_NHANES_data(year=2017, data_type='BIOPRO')
```

Fairness-aware Multimodal AI for Healthcare Data (Professors  
[Eric Larson](#) and [Mehak Gupta](#))

## 2. MEPS (Medical Expenditure Panel Survey)

python

```
import pandas as pd
```

```
# Direct URL loading - Full Year Consolidated Data
```

```
url = 'https://meps.ahrq.gov/mepsweb/data_files/pufs/h233/h233csv.zip'
```

```
meps_df = pd.read_csv(url, compression='zip')
```

```
# Or use pymeps package
```

```
# pip install pymeps
```

```
import pymeps
```

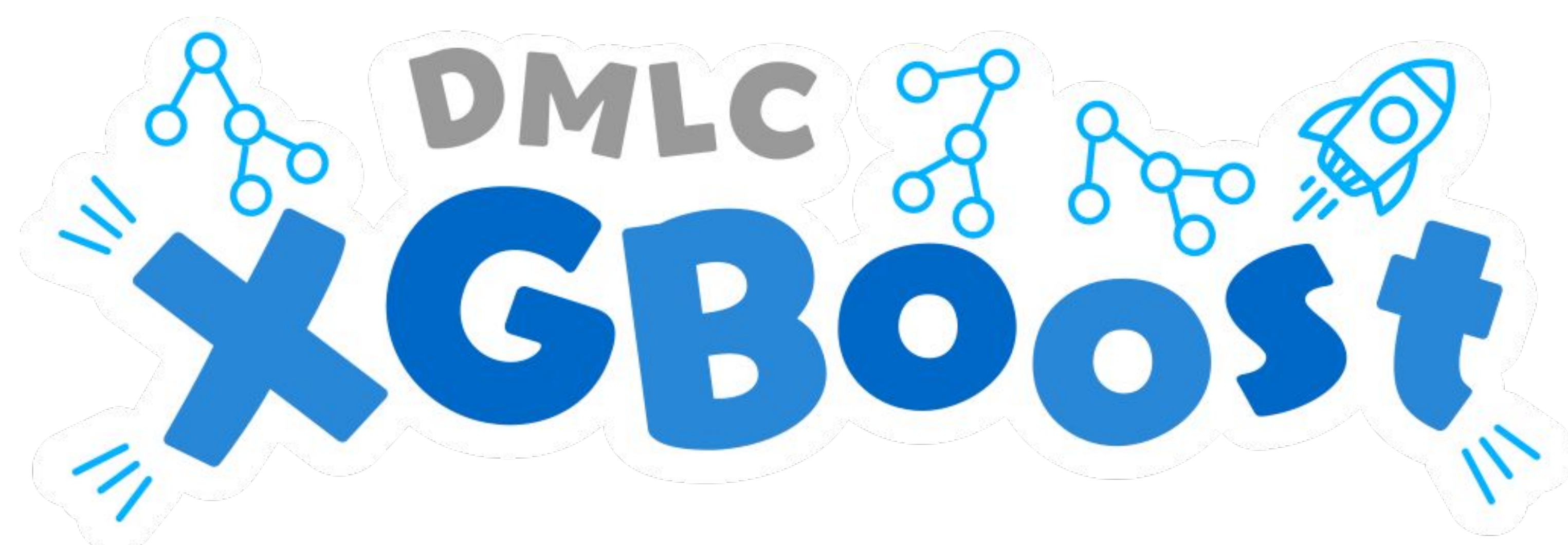
```
df = pymeps.load_data(year=2021, type='FYU')
```



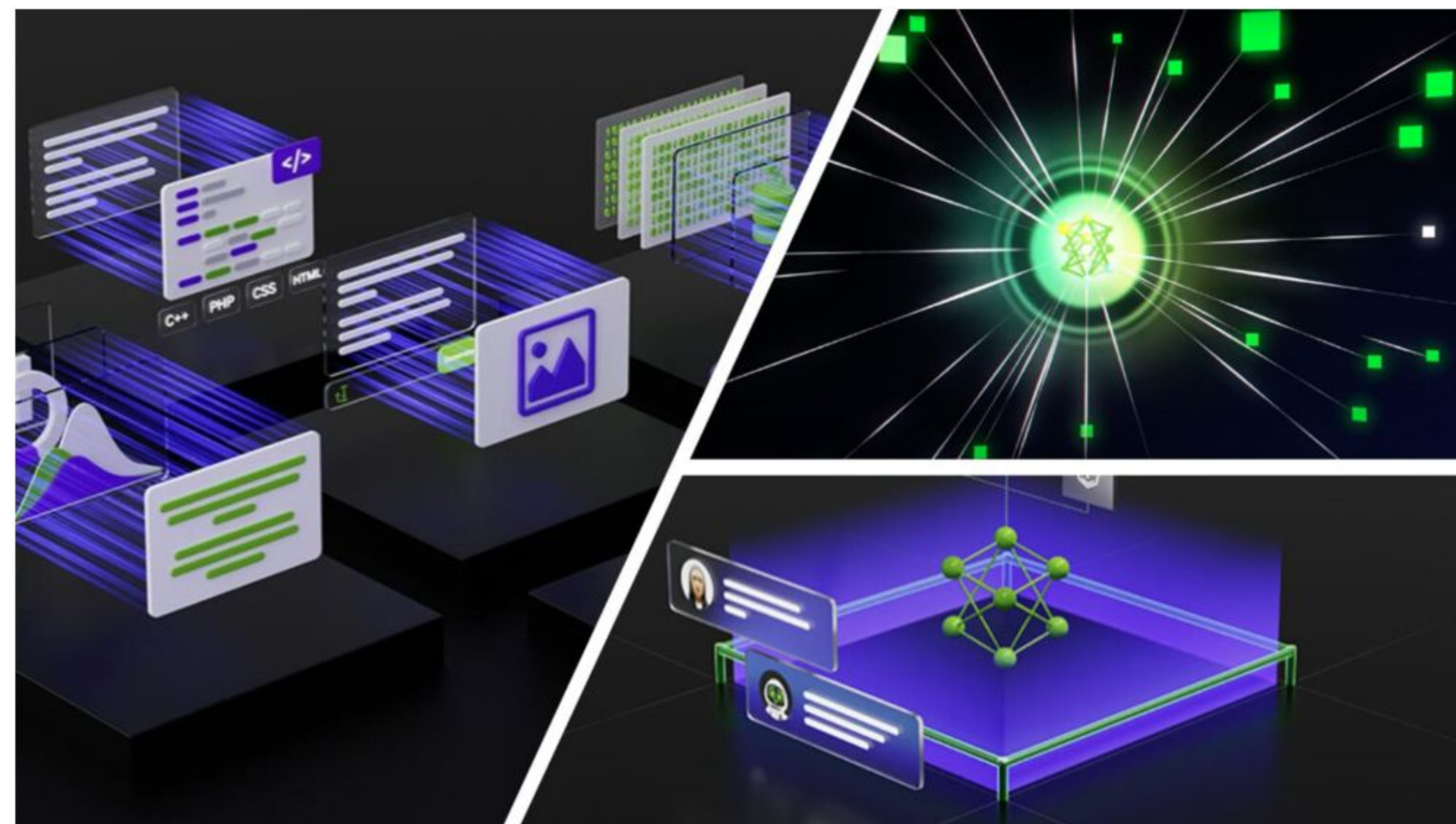


# Federated Learning

Federated Learning (FL) is a decentralized machine learning paradigm that enables multiple clients to collaboratively train a global model while preserving data privacy. Unlike traditional centralized learning, FL ensures that raw data remains local, with only model updates being shared, thereby mitigating privacy concerns and reducing communication overhead. FL has gained significant traction in research for its applications in privacy-preserving machine learning, efficiency in distributed systems, making it particularly valuable in domains such as healthcare, finance, and edge computing.



## Federated XGBoost Made Practical and Productive with NVIDIA FLARE



Jun 28, 2024

By [Yuan-Ting Hsieh](#) and [Yan Cheng](#)

+2 Like Discuss (1)

- **Decentralized tracking:** Each client manages its own metrics and experiment tracking server locally, maintaining training metric privacy. However, this setup limits the ability to compare data across different sites.
- **Centralized tracking:** All metrics are streamed to a central FL server, which then pushes the data to a selected tracking system. This setup supports effective cross-site metric comparisons

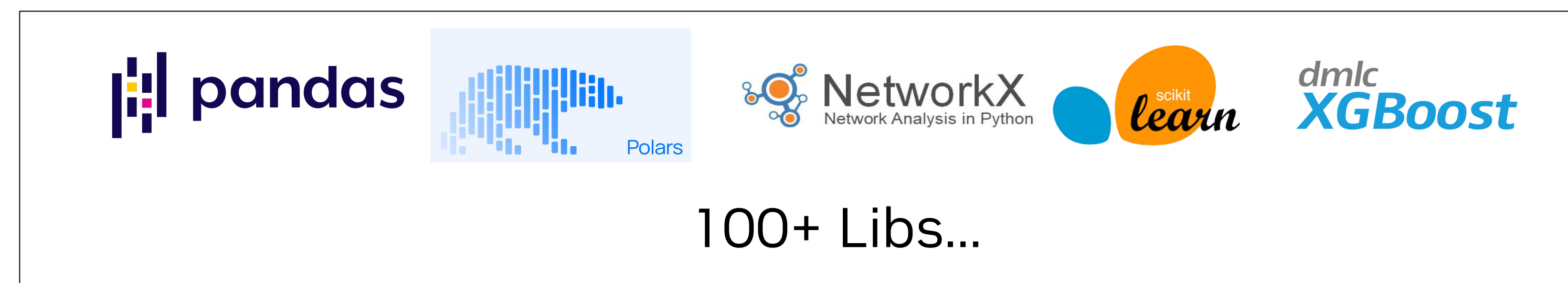


# The Scientific Python Stack





# Challenges with Traditional Data Processing



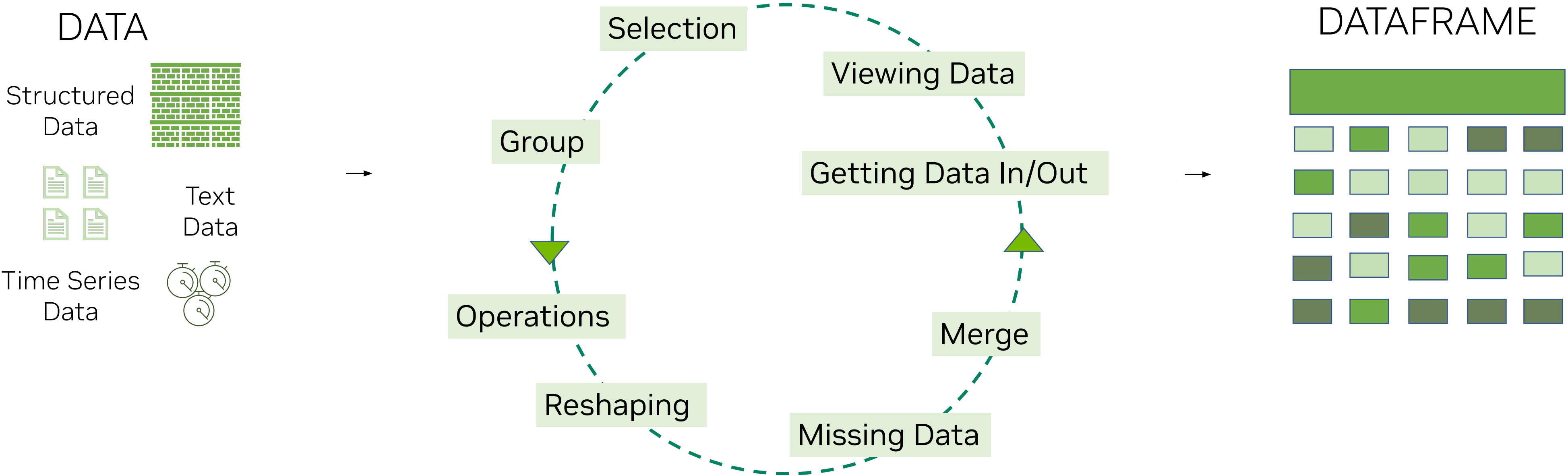
- **Performance slows** as data grows
- Need to learn **bespoke APIs** and specialized data formats
- Require **significant code changes** to run on new infrastructure





# GPU Accelerated Data Analytics with cuDF-pandas

*With accelerated compute, data analytics can speed-up 40x<sup>4</sup>*



***Faster processing***





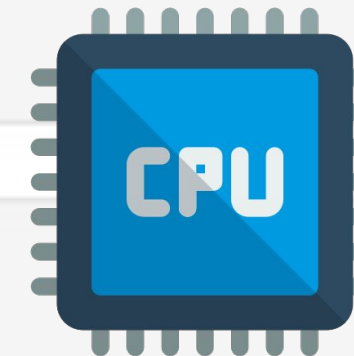
# NVIDIA cuDF

Load NYC Taxi Data into pandas withOUT GPU.

```
[1]: import pandas as pd  
import glob
```

```
[2]: %%time  
# load NYC Data into pandas withOUT GPU  
df = pd.concat([pd.read_parquet(f) for f in glob.glob("nyc_taxi_data/*.parquet")], ignore_index=True)
```

CPU times: user 25.1 s, sys: 10.6 s, total: 35.7 s  
Wall time: 5.57 s



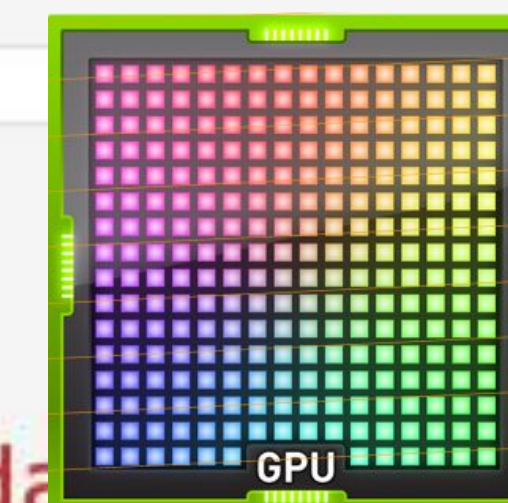
Load NYC Taxi Data into pandas WITH GPU

```
[1]: %load_ext cudf.pandas
```

```
[2]: import pandas as pd  
import glob
```

```
[3]: %%time  
# load NYC Data into pandas withOUT GPU  
df = pd.concat([pd.read_parquet(f) for f in glob.glob("nyc_taxi_data/*.parquet")], ignore_index=True)
```

CPU times: user 1.64 s, sys: 520 ms, total: 2.16 s  
Wall time: 1.53 s



```
[4]: df.shape
```

```
[4]: (115939623, 20)
```

NVIDIA cuDF

116M Rows

x3.6  
Speed  
up!



```
[1]: %load_ext cudf.pandas
# > python -m cudf.pandas your_code.py
```

```
[2]: import pandas as pd
import glob

pd
```

```
[2]: <module 'pandas' (ModuleAccelerator(fast=cudf, slow=pandas))>
```

```
[3]: %%time
# load NYC Data into pandas withOUT GPU
df = pd.concat([pd.read_parquet(f) for f in glob.glob("nyc_taxi_data/*.parquet")], ignore_index=True)

CPU times: user 1.67 s, sys: 477 ms, total: 2.15 s
Wall time: 1.52 s
```

```
[4]: %%time
df = (
    df[['VendorID', 'passenger_count', 'store_and_fwd_flag', 'PULocationID', 'tip_amount', 'fare_amount', 'total_amount']]
    .sort_values(['tip_amount', 'fare_amount', 'total_amount'], ascending=[True, False, True])
    .groupby(['VendorID', 'passenger_count', 'store_and_fwd_flag', 'PULocationID'])
    .mean()
    .sort_values(['tip_amount', 'fare_amount', 'total_amount'], ascending=[False, True, False])
)

CPU times: user 707 ms, sys: 68.8 ms, total: 776 ms
Wall time: 755 ms
```

x37  
Speedup!



```
[1]: import pandas as pd
import glob

pd
```

```
[1]: <module 'pandas' from '/home/will/git/smu/venv/lib/python3.12/site-packages/pandas/__init__.py'>
```

```
[2]: %%time
# load NYC Data into pandas withOUT GPU
df = pd.concat([pd.read_parquet(f) for f in glob.glob("nyc_taxi_data/*.parquet")], ignore_index=True)

CPU times: user 24.3 s, sys: 9.74 s, total: 34 s
Wall time: 5.25 s
```

```
[3]: %%time
df = (
    df[['VendorID', 'passenger_count', 'store_and_fwd_flag', 'PULocationID', 'tip_amount', 'fare_amount', 'total_amount']]
    .sort_values(['tip_amount', 'fare_amount', 'total_amount'], ascending=[True, False, True])
    .groupby(['VendorID', 'passenger_count', 'store_and_fwd_flag', 'PULocationID'])
    .mean()
    .sort_values(['tip_amount', 'fare_amount', 'total_amount'], ascending=[False, True, False])
)

CPU times: user 25 s, sys: 3.15 s, total: 28.2 s
Wall time: 28.2 s
```



```
[1]: import sklearn
      from sklearn.datasets import make_classification
      from sklearn.model_selection import train_test_split
      from sklearn.ensemble import RandomForestClassifier

[2]: X, y = make_classification(n_samples=500_000, n_features=100)
      X_train, X_test, y_train, y_test = train_test_split(X, y, test_size=0.2)

[3]: %%time
      rf = RandomForestClassifier(n_estimators=1_000, n_jobs=-1)
      rf.fit(X_train, y_train)

CPU times: user 6h 9min 35s, sys: 2 s, total: 6h 9min 35s
Wall time: 11min 44s
```

11:44

```
[1]: %load_ext cuml.accel

[2]: import sklearn
      from sklearn.datasets import make_classification
      from sklearn.model_selection import train_test_split
      from sklearn.ensemble import RandomForestClassifier

[3]: X, y = make_classification(n_samples=500_000, n_features=100)
      X_train, X_test, y_train, y_test = train_test_split(X, y, test_size=0.2)

[4]: %%time
      rf = RandomForestClassifier(n_estimators=1_000, n_jobs=-1)
      rf.fit(X_train, y_train)

CPU times: user 22.3 s, sys: 11 s, total: 33.8 s
Wall time: 12.3 s
```

NVIDIA cuML

12.3 s

x57  
Speedup!





```
cuML.ipynb +
Python 3 (ipykernel)

[1]: import sklearn
    from sklearn.datasets import make_classification
    from sklearn.model_selection import train_test_split
    import xgboost as xgb

[2]: X, y = make_classification(n_samples=500_000, n_features=100, random_state=42)
    X_train, X_test, y_train, y_test = train_test_split(X, y, test_size=0.2, random_state=0)

[3]: %%time
    model = xgb.XGBClassifier(n_estimators=500,
                             max_depth=15,
                             learning_rate=0.01,
                             device="cpu"
    )

    model = model.fit(X_train, y_train)

    CPU times: user 54min 14s, sys: 1.4 s, total: 54min 5s
    Wall time: 1min 47s

[4]: %%time
    model = xgb.XGBClassifier(n_estimators=500,
                             max_depth=15,
                             learning_rate=0.01,
                             device="cuda"
    )

    model = model.fit(X_train, y_train)

    CPU times: user 39.8 s, sys: 303 ms, total: 40.1 s
    Wall time: 32.2 s

[6]: 107 / 32.2

[6]: 3.322981366459627
```

“cpu”

1:47

“cuda”

32.2 s

x3.6  
Speed  
up!

*dmlc*  
**XGBoost**

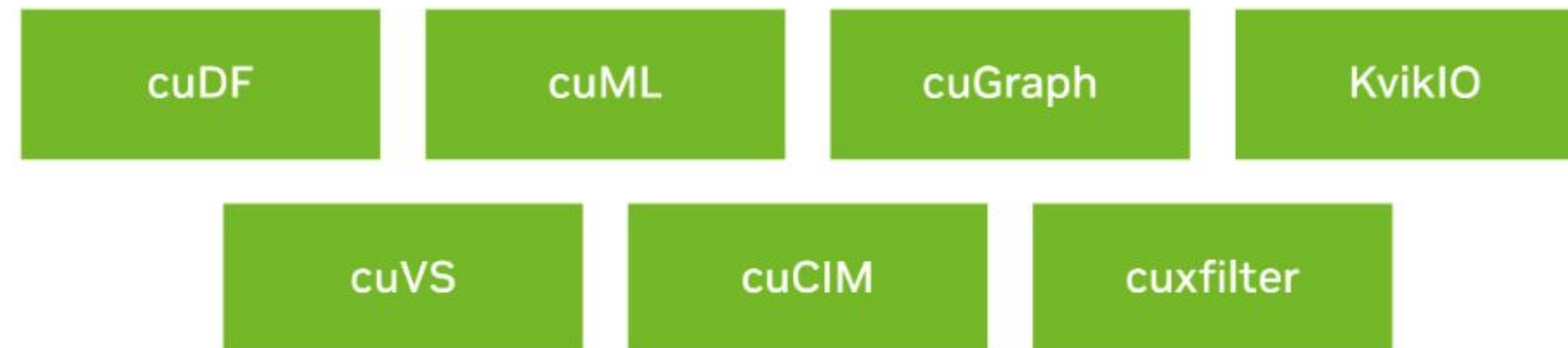


# Data Science and AI Applications

## Data Science and Data Processing Ecosystem



## CUDA-X



## CUDA

## Compute Infrastructure





# RAPIDS cuML

Accelerated Machine Learning with a Scikit-Learn API

## 50+ GPU-Accelerated Algorithms & Growing

CPU

Scikit-learn

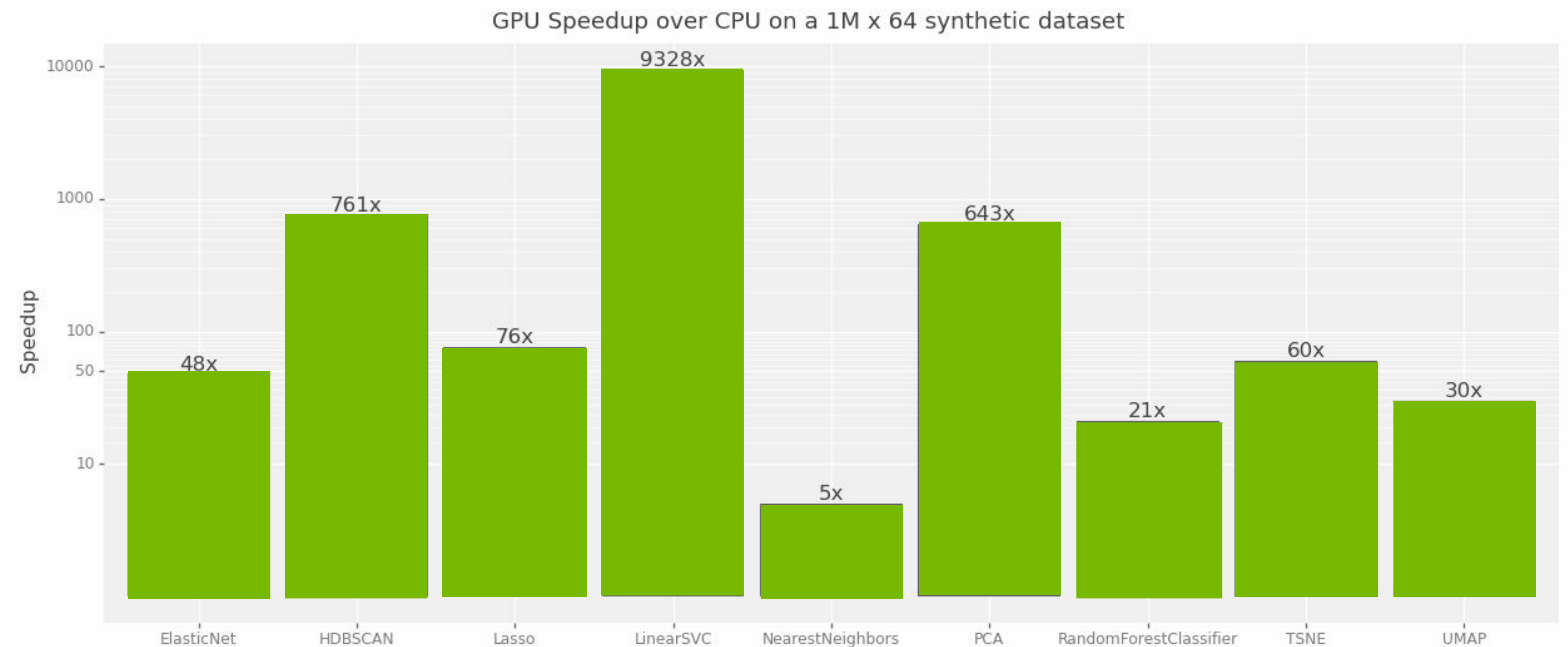
```
>>> from sklearn.ensemble import  
RandomForestClassifier  
>>> clf = RandomForestClassifier()  
>>> clf.fit(x, y)
```



cuML

GPU

```
>>> from cuml.ensemble import  
RandomForestClassifier  
>>> clf = RandomForestClassifier()  
>>> clf.fit(x, y)
```



Time Series

Classification

Regression

Clustering

Preprocessing

Cross Validation

Tree Models

Dimensionality  
Reduction

Explainability

A100 GPU vs. AMD EPYC 7642 (96 logical cores)  
cuML 23.04, scikit-learn 1.2.2, umap-learn 0.5.3



# Accelerating Pandas with NVIDIA cuDF

```
File Edit View Run Kernel Tabs Settings Help

regular-pandas.ipynb X +

[ ]: %%time

import pandas as pd

# Read the data
df = pd.read_parquet("nyc_parking_violations_2022.parquet")

[ ]: %%time

# Which parking violation is most commonly
# committed by vehicles from various U.S states?
(df[["Registration State", "Violation Description"]]
 .value_counts()
 .groupby("Registration State")
 .head(1)
 .sort_index()
 .reset_index()
 )

[ ]: %%time

# Which vehicle body types are most frequently
# involved in parking violations?
(df
 .groupby(["Vehicle Body Type"])
 .agg({"Summons Number": "count"})
 .rename(columns={"Summons Number": "Count"})
 .sort_values(["Count"], ascending=False)
 )

[ ]: %%time
```

```
cudf-pandas-accelerator-mo X +
[ ]: %load_ext cudf.pandas

[ ]: %%time

import pandas as pd

# Read the data
df = pd.read_parquet("nyc_parking_violations_2022.parquet")

[ ]: %%time

# Which parking violation is most commonly
# committed by vehicles from various U.S states?
(df[["Registration State", "Violation Description"]]
 .value_counts()
 .groupby("Registration State")
 .head(1)
 .sort_index()
 .reset_index()
 )

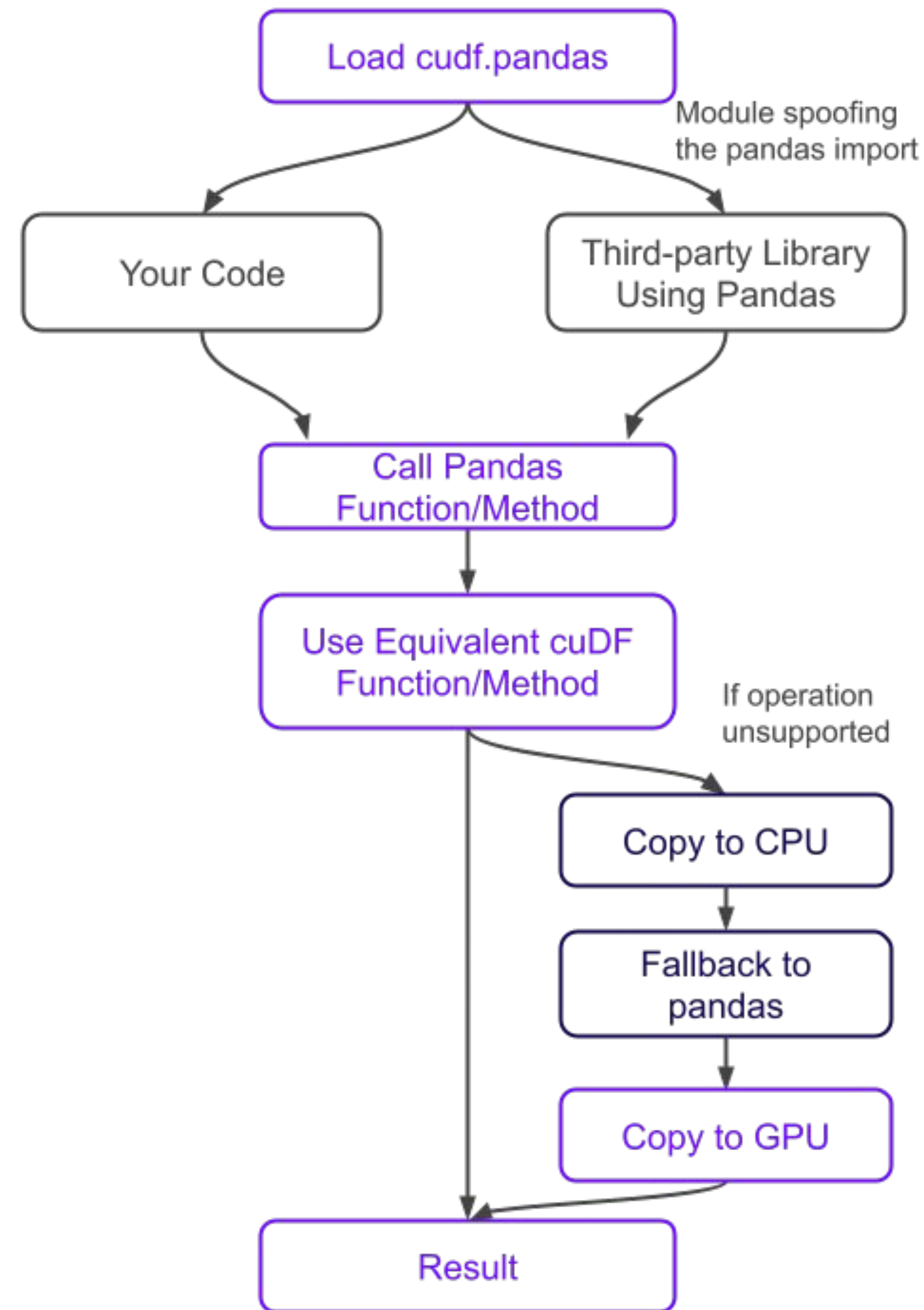
[ ]: %%time

# Which vehicle body types are most frequently
# involved in parking violations?
(df
 .groupby(["Vehicle Body Type"])
 .agg({"Summons Number": "count"})
 .rename(columns={"Summons Number": "Count"})
 .sort_values(["Count"], ascending=False)
 )
```



# Accelerating Pandas with NVIDIA cuDF

Up to 50x faster with zero code change

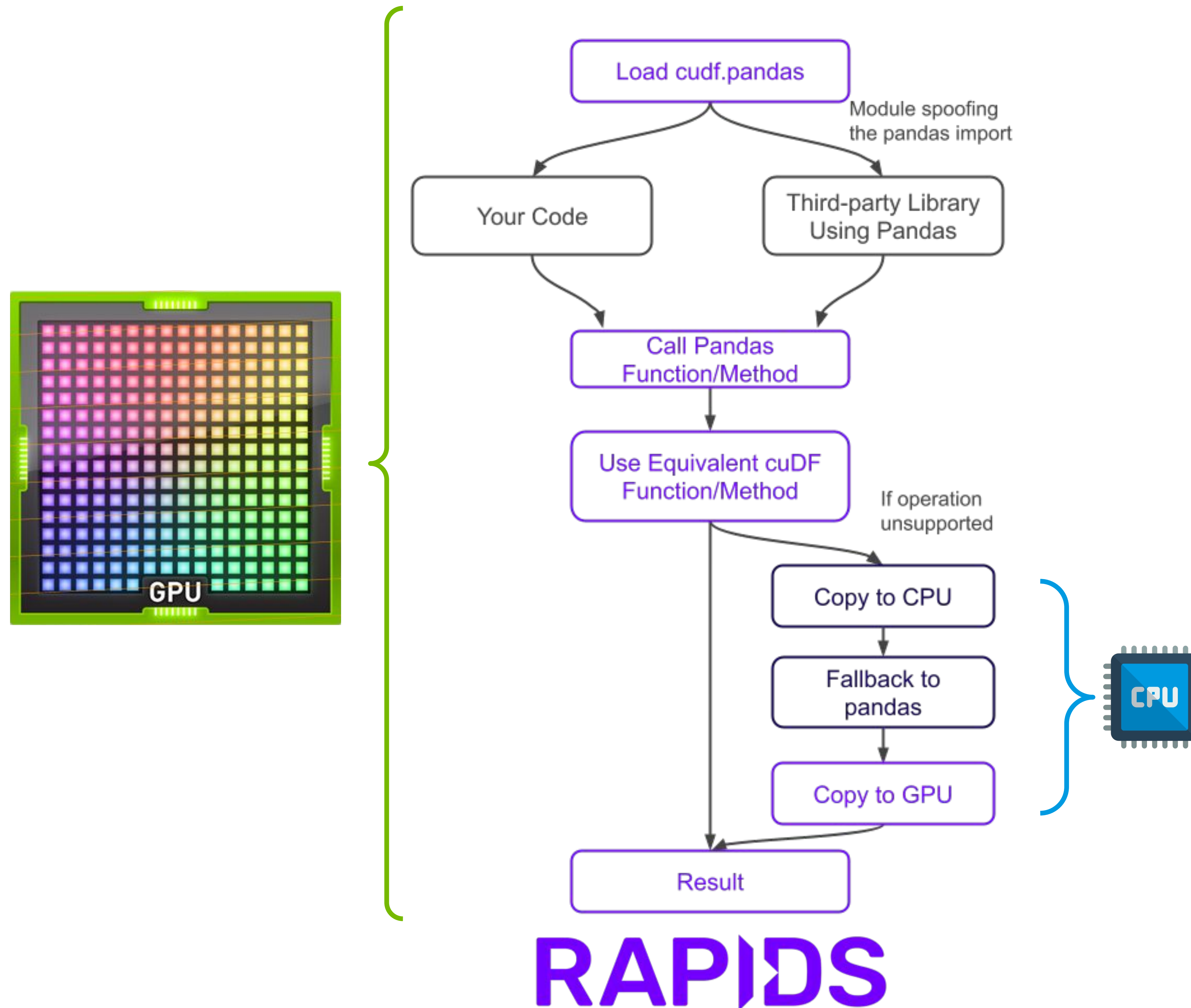


# RAPIDS



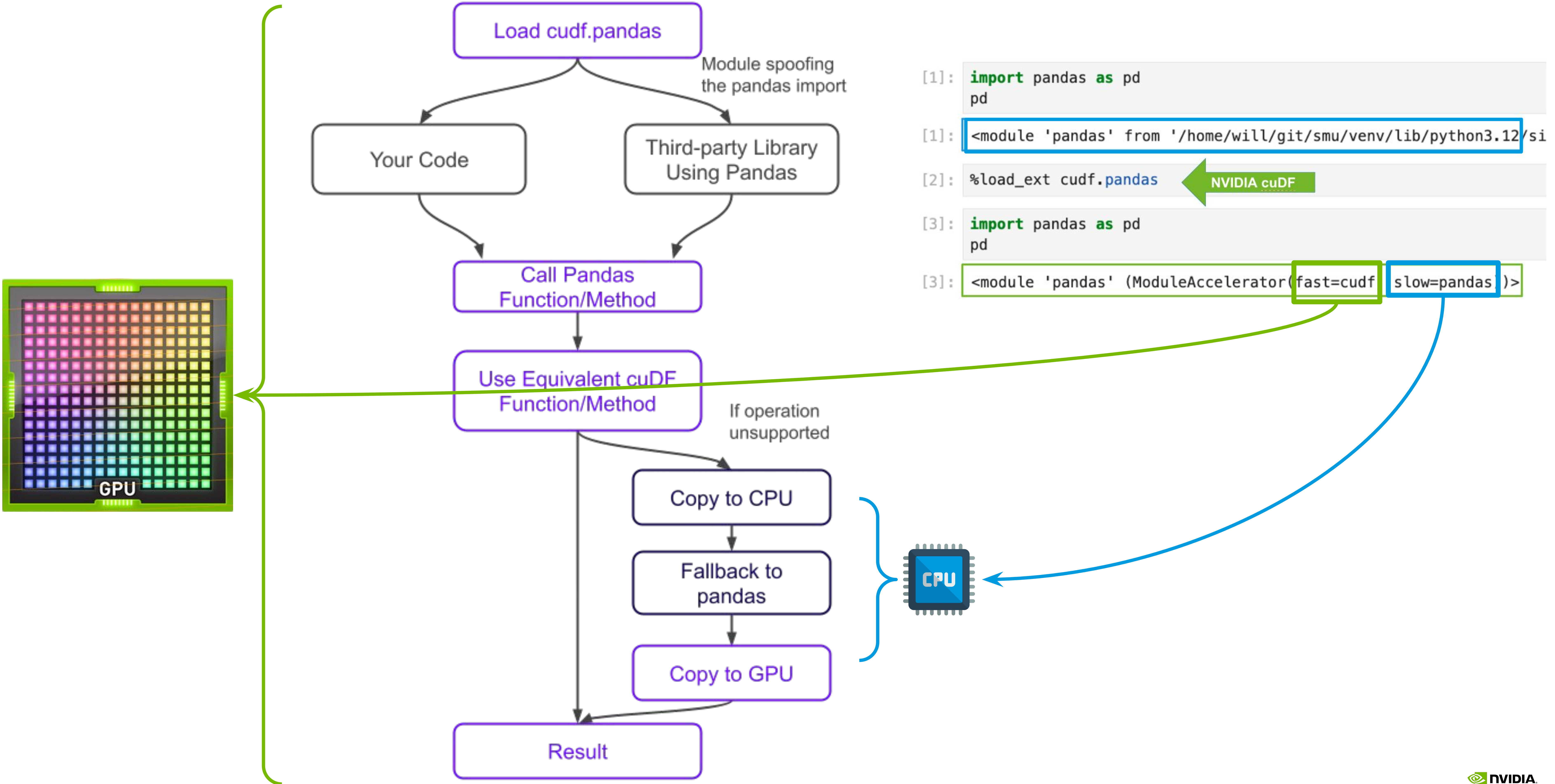
# Accelerating Pandas with NVIDIA cuDF

Up to 50x faster with zero code change





# cuDF + pandas Workflow







# Neural Nets | Transformers | LLMs | Gen AI

O'DONNELL DATA SCIENCE AND RESEARCH COMPUTING INSTITUTE · RESEARCH · AI-ENABLED DATA COMPRESSION

## AI-Enabled Data Compression

The O'Donnell Institute scientists are developing several AI-enabled data compression algorithms and testing them on SMU's HPC platforms. Data compression is a key research area focused on reducing the size of digital data, such as text, images, video, or audio, to enable faster transmission and more efficient storage. Unlike traditional methods that depend on fixed statistical models, modern AI/ML methods use neural network and deep learning models like Convolutional Neural Networks (CNNs), Autoencoders, Transformers, and Generative Adversarial Networks (GANs) to learn complex patterns directly from large datasets. These approaches achieve higher compression ratios and better quality compared to classical algorithms by capturing both local and global dependencies. AI-enabled data compression has gained momentum in recent years due to its impact on optimizing communication, lowering storage costs, and enabling data-intensive applications in fields such as medical imaging, satellite communications, and autonomous systems.

O'DONNELL DATA SCIENCE AND RESEARCH COMPUTING INSTITUTE · RESEARCH · AREAD INITIATIVE

## AREAD Initiative

The O'Donnell Institute recently launched the AREAD (**A**dvancing **R**esource-**E**fficient **A**I and **S**mart **D**ata Management) initiative for energy-efficient AI solutions. As AI models and datasets continue to grow, optimizing computational efficiency and data utilization is essential for sustainability and scalability. Resource-efficiency may be achieved by leveraging lightweight AI models, adaptive processing techniques, smart data management, and resource-aware algorithms. The AREAD initiative supports projects in AI model optimization—including model compression, quantization, and lightweight models—as well as intelligent data storage, retrieval, and processing. Targeted application areas include but are not limited to AI-powered IoT, autonomous systems, AR/VR, and AI-driven digital twins.



# AI-Enabled Data Compression

The O'Donnell Institute scientists are developing several AI-enabled data compression algorithms and testing them on SMU's HPC platforms. Data compression is a key research area focused on reducing the size of digital data, such as text, images, video, or audio, to enable faster transmission and more efficient storage. Unlike traditional rule-based techniques that depend on fixed statistical models, modern AI/ML methods use neural network and deep learning architectures including Convolutional Neural Networks (CNNs), Autoencoders, Transformers, and Generative Adversarial Networks (GANs) to learn complex patterns directly from large datasets. These approaches achieve higher compression ratios and improved perceptual quality compared to classical algorithms by capturing both local and global dependencies. AI-enabled data compression has gained momentum in recent years due to its impact on optimizing communication, lowering infrastructure costs, and enabling data-intensive applications in fields such as medical imaging, satellite communications, and multimedia streaming.

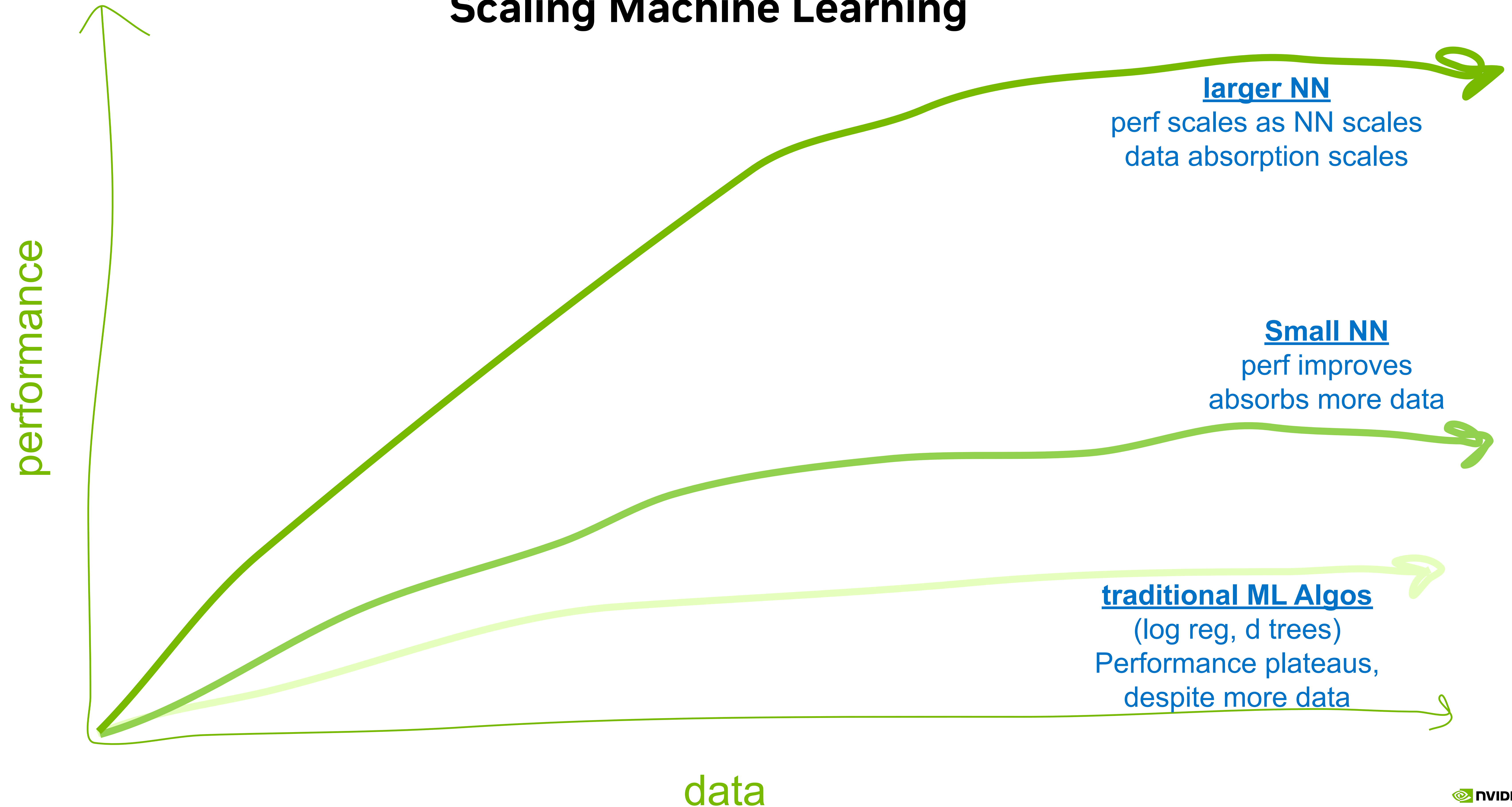


# AREAD Initiative

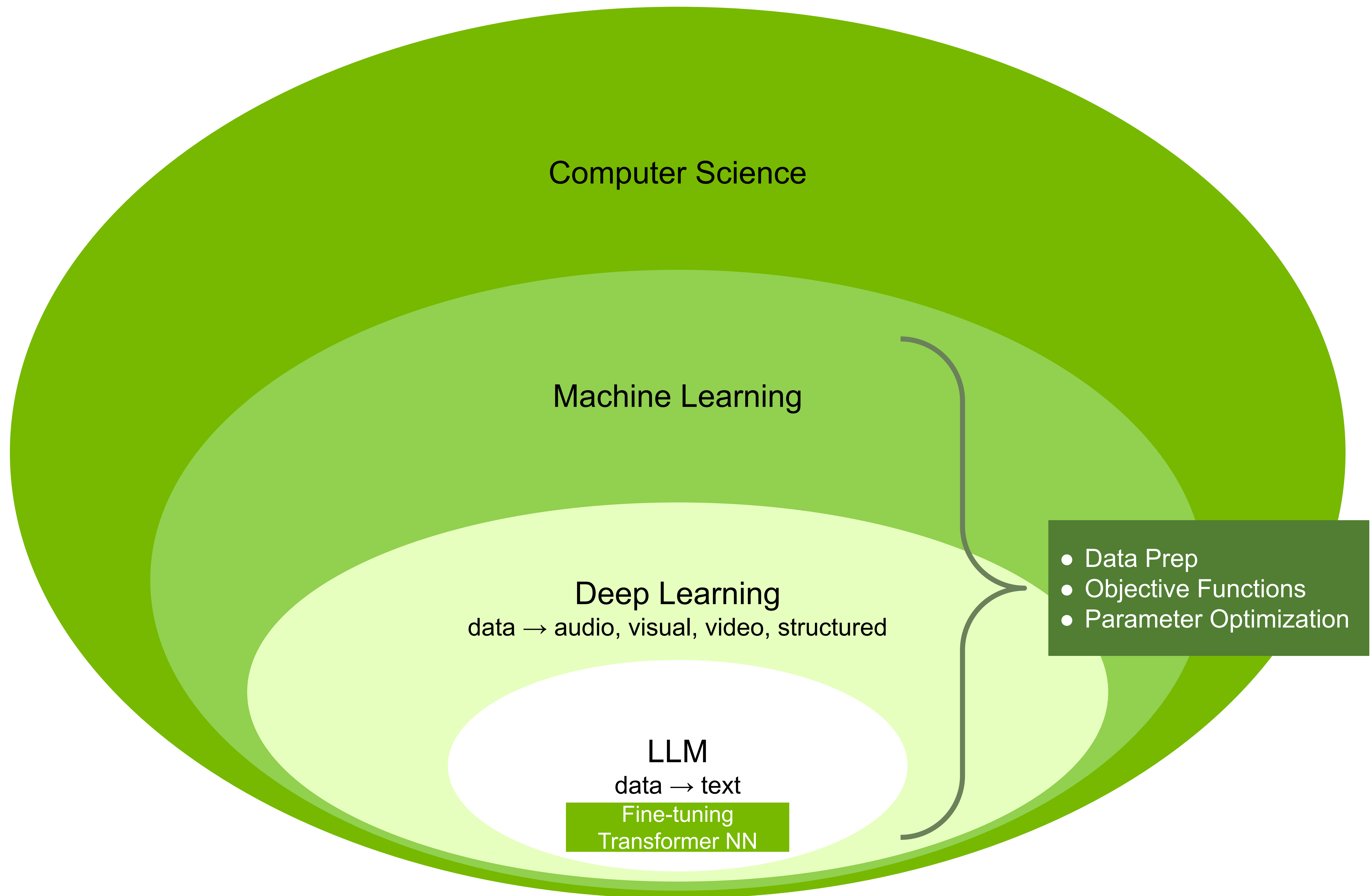
The O'Donnell Institute recently launched the AREAD (***A**dvancing **R**esource-**E**fficient **A**I and Smart **D**ata Management*) initiative for energy-efficient AI solutions. As AI models and datasets continue to grow, optimizing computational efficiency and data utilization is essential for sustainability and scalability. Resource-efficiency may be achieved by leveraging lightweight AI models, adaptive processing techniques, smart data management, and resource-aware algorithms. The AREAD initiative supports projects in AI model optimization—including model compression, quantization, and lightweight models—as well as intelligent data storage, retrieval, and processing. Targeted application areas include but are not limited to AI-powered IoT, autonomous systems, AR/VR, and AI-driven digital twins.



# Scaling Machine Learning











**In the before times...**





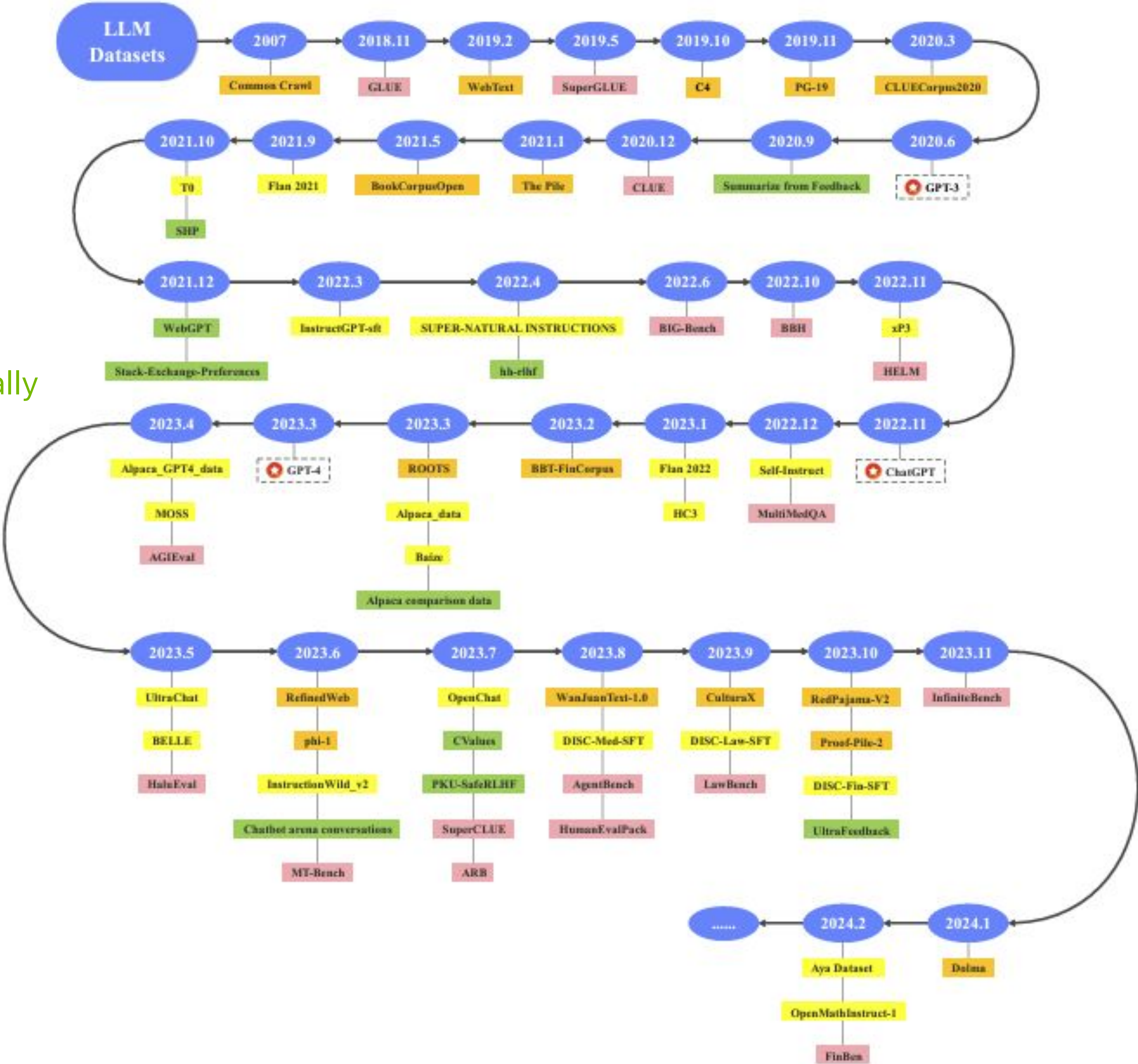
**In the before times...*of 2021***




# LLMs Are Trained on Internet Scale Data

Data Generated is Growing Exponentially

- Training corpa
- Instruct fine tuning
- Preference dataset
- Evaluation dataset



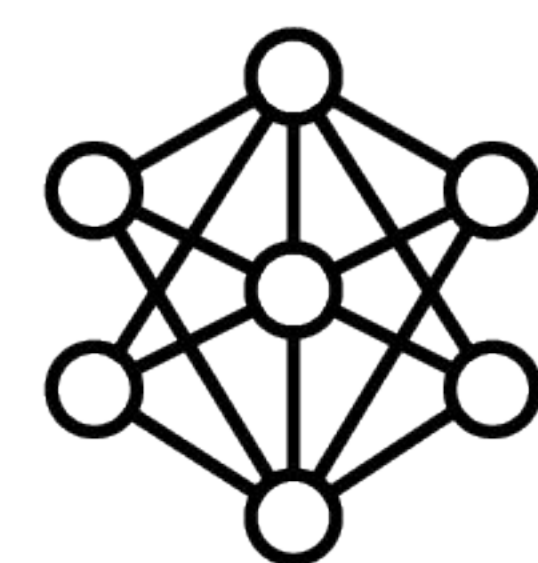




**In the before times...*of 2022***



# Data Processing for Different LLM Needs



Training Foundation Model



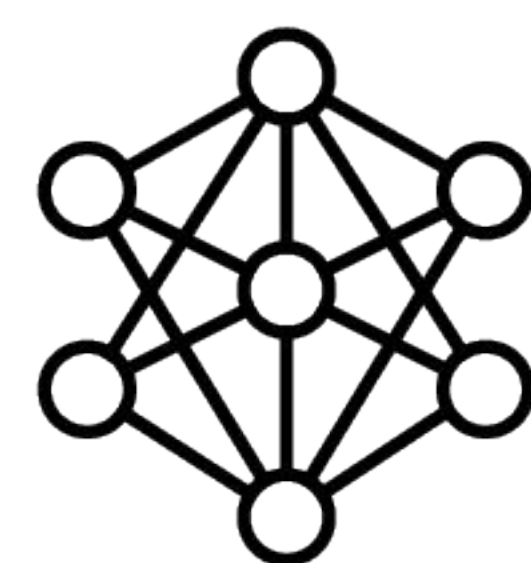
Fine-Tuning Foundation Model

Retrieval Augmented Generation (RAG)

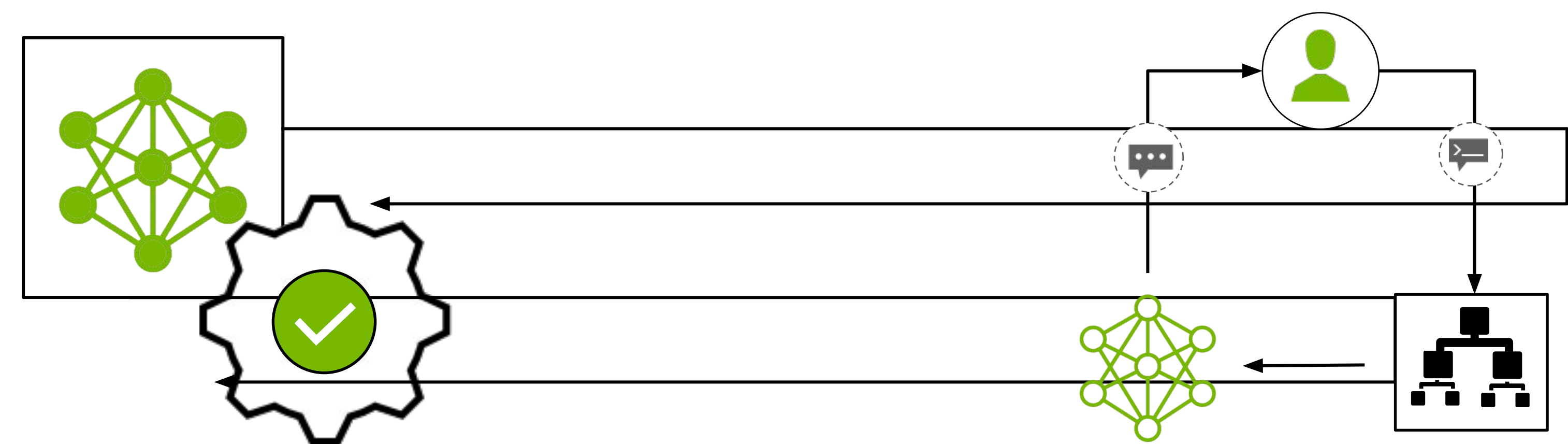
Data Size	TB and PB		GBs	GBs
Compute Scale	Supercomputer		Single-node	Single GPU
Frequency	One-time		Iterative	Iterative & Continuous



# Data Processing for Different LLM Needs



Training Foundation Model



Fine-Tuning Foundation Model

Retrieval Augmented Generation (RAG)


Data Size	TB and PB	GBs	GBs
Compute Scale	Supercomputer	Single-node	Single GPU
Frequency	One-time	Iterative	Iterative & Continuous





# **Data Processing for LLMs w/ NeMo Curator**





**In the before times...*of 2024***



# Shrinking the LLMs






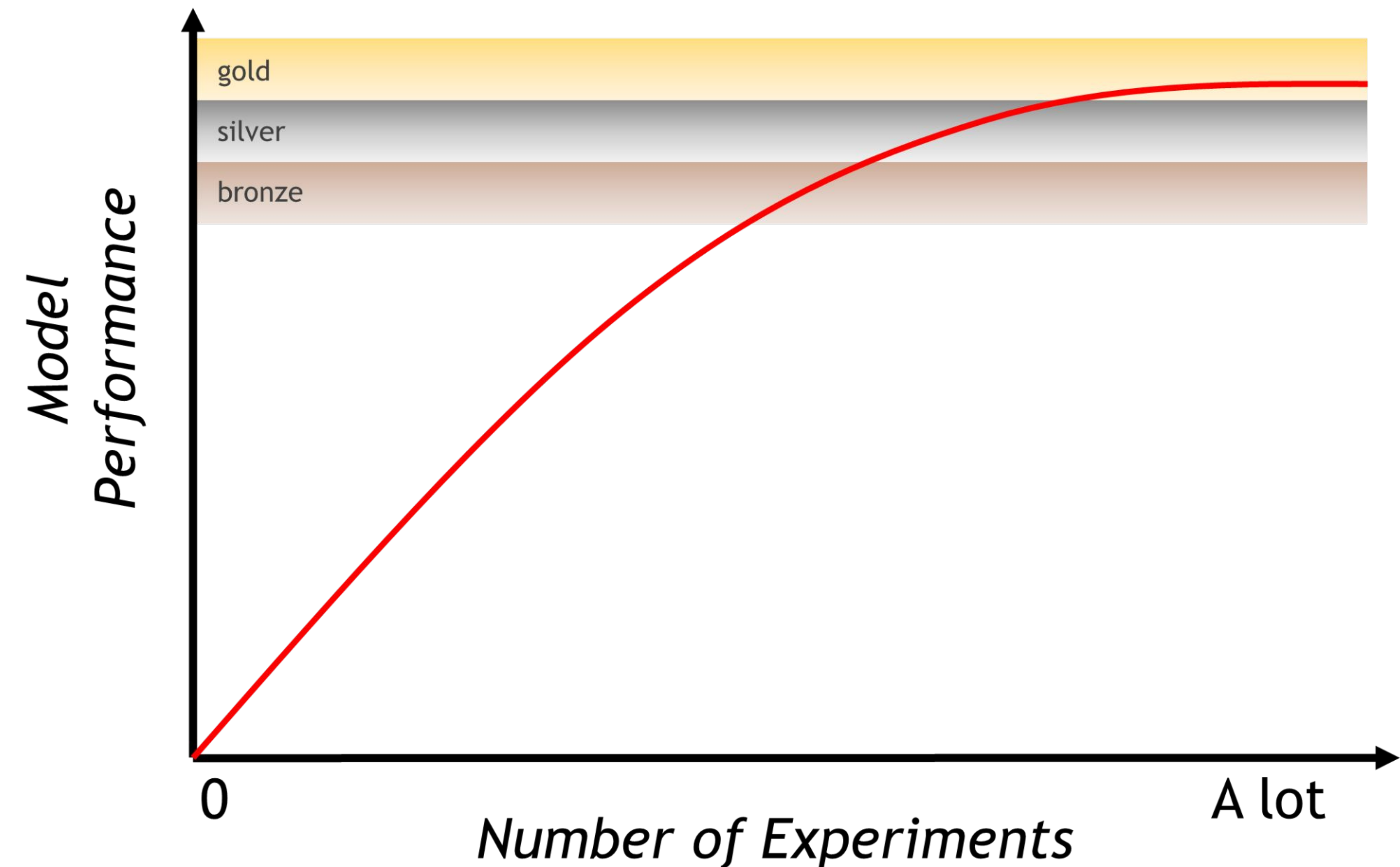
# Fast Experimentation

## Why Speed Matters

- The more you experiment ...
  - The better your models
  - The more you understand the data
  - The higher your chances of finding a winning idea
- ... But experimenting takes time :
  - Optimize your pipeline to be able to experiment more

### → Leverage GPU acceleration

- RAPIDS CuDF for Analytics
  - Pandas, but faster !
  -  Polars is also faster on GPU !
- RAPIDS CuML for Machine Learning




**RAPIDS**

Accelerated with





## Technical Blog

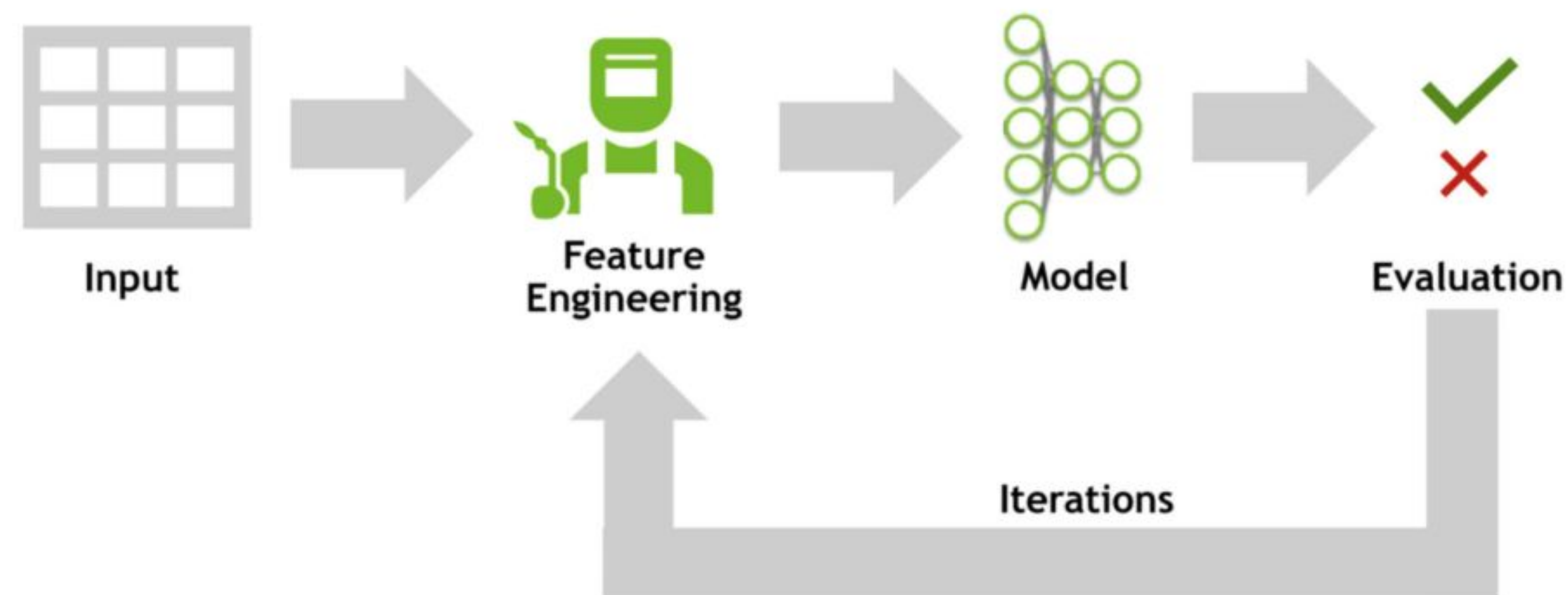
 Search blog

[Subscribe](#) >

Data Science

English ▾

# Grandmaster Pro Tip: Winning First Place in Kaggle Competition with Feature Engineering Using cuDF pandas



Apr 17, 2025

By [Chris Deotte](#)

 +14 Like  Discuss (0)



AI-Generated Summary



### Related posts





Data Science

English [v](#)

# Grandmaster Pro Tip: Winning First Place in a Kaggle Competition with Stacking Using cuML





## Groupby(COL1)[COL2].agg(STAT)

The most powerful feature engineering technique is groupby aggregations. Namely, we execute the code `groupby(COL1)[COL2].agg(STAT)`. This is where we group by COL1 column and aggregate (i.e. compute) a statistic STAT over another column COL2. We use the speed of NVIDIA cuDF-Pandas to explore thousands of COL1, COL2, STAT combinations. We try statistics (STAT) like “mean”, “std”, “count”, “min”, “max”, “nunique”, “skew” etc etc. We choose COL1 grouping columns. When COL2 is the target column, then we use nested cross-validation computation. When COL2 is the target, this operation is called Target Encoding.

## Groupby(COL1)['Price'].agg(QUANTILES)

We can groupby and compute the quantiles for `QUANTILES = [5,10,40,45,55,60,90,95]` and return the eight values to create eight new columns.

```
for k in QUANTILES:
    result = X_train2.groupby('Weight Capacity (kg)').\
        agg({'Price': lambda x: x.quantile(k/100)})
```

## All NaNs as Single Base-2 Column

We can create a new column from all the NaNs over multiple columns. This is a powerful column which we can subsequently use for groupby aggregations or combinations with other columns.

```
train["NaNs"] = np.float32(0)
for i,c in enumerate(CATS):
    train["NaNs"] += train[c].isna()*2**i
```

## Put Numerical Column into Bins

The most powerful (predictive) column in this competition is Weight Capacity. We can create more powerful column based on this column by binning this column with rounding.

```
for k in range(7,10):
    n = f"round{k}"
    train[n] = train["Weight Capacity (kg)"].round(k)
```

## Groupby(COL1)['Price'].agg(HISTOGRAM BINS)

When we groupby(COL1)[COL2] we have a distribution (set) of numbers for each group. Instead of computing a single statistic (and making one new column), we can compute any collection of numbers that describe this distribution of numbers and make many new columns together.

Below we display a histogram for the group `Weight Capacity = 21.067673`. We can count the number of elements in each (equally spaced) bucket and create a new engineered feature for each bucket count to return to the groupby operation! Below we display seven buckets, but we can treat the number of buckets as a hyperparameter.

```
result = X_train2.groupby("WC")["Price"].apply(make_histogram)
X_valid2 = X_valid2.merge(result, on="WC", how="left")
```

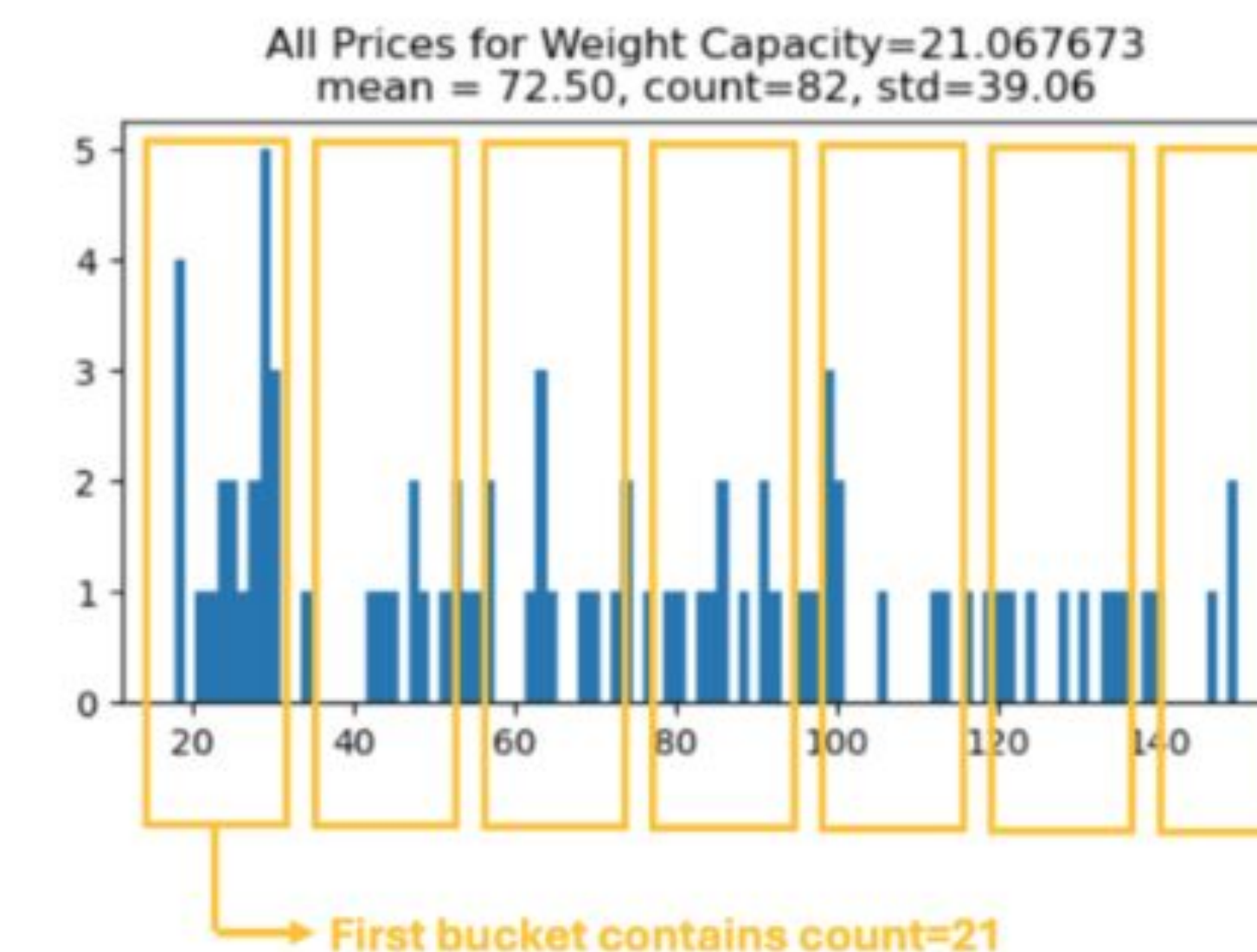


Figure 1. Histogram of price values when weight capacity equals 21.067673



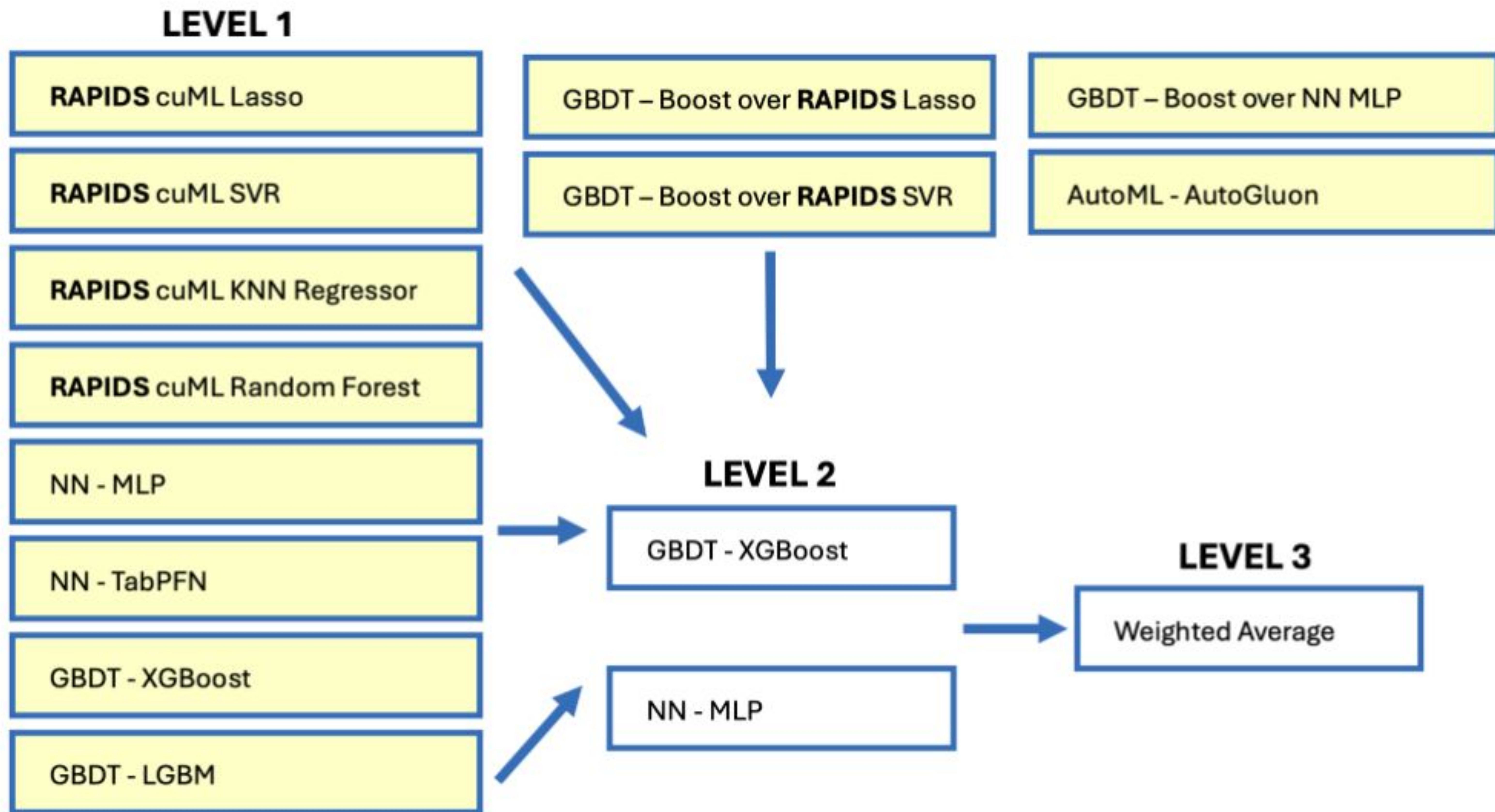


Figure 1. The winning entry in the Kaggle April 2025 Playground competition used stacking with three levels of models, with the results of each level used in subsequent levels




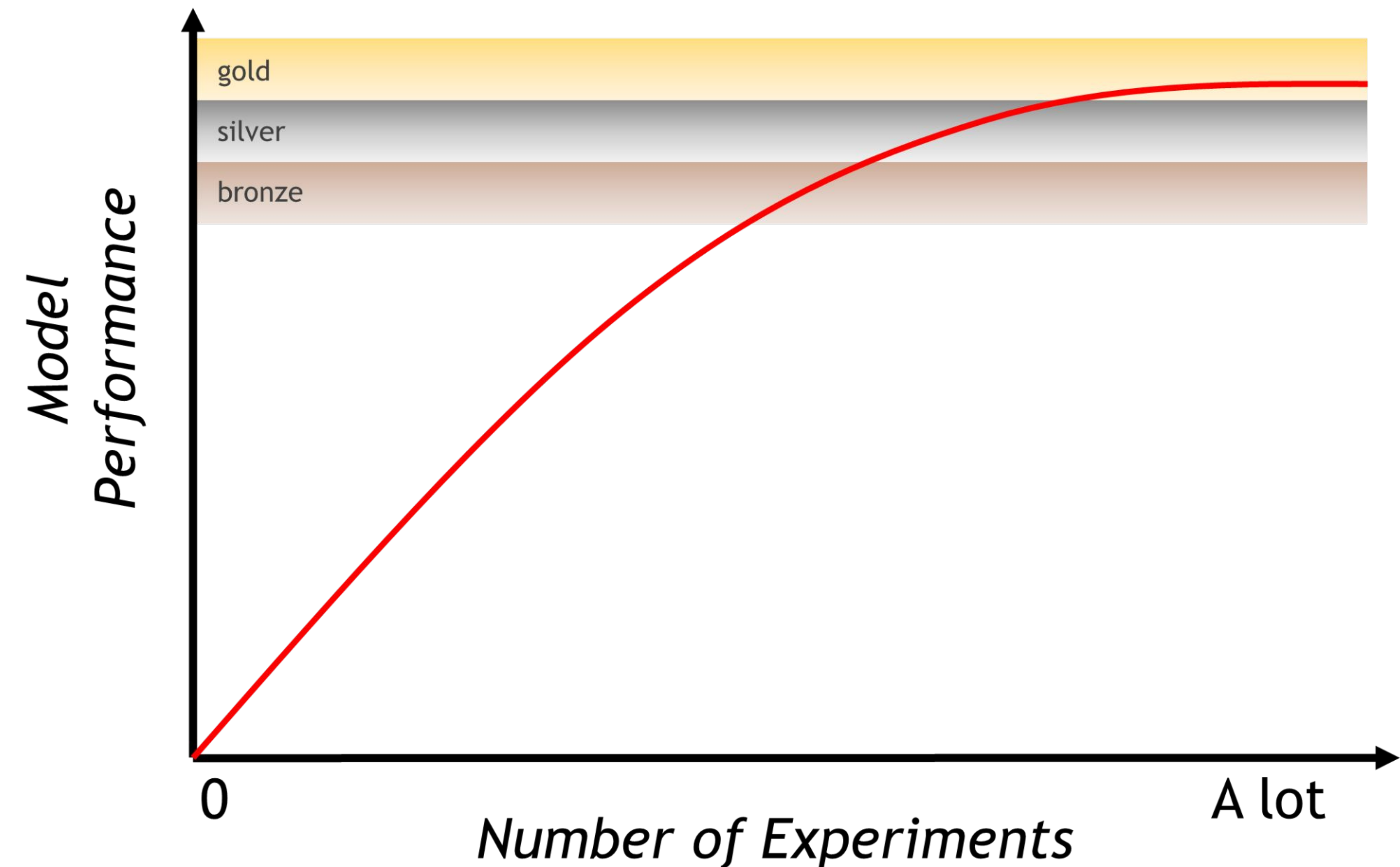
# Fast Experimentation

## Why Speed Matters

- The more you experiment ...
  - The better your models
  - The more you understand the data
  - The higher your chances of finding a winning idea
- ... But experimenting takes time :
  - Optimize your pipeline to be able to experiment more

### → Leverage GPU acceleration

- RAPIDS CuDF for Analytics
  - Pandas, but faster !
  -  Polars is also faster on GPU !
- RAPIDS CuML for Machine Learning



**RAPIDS**

Accelerated with





# Shrinking the LLMs

- Self-Instruct
- Model Distillation
- Instruction Tuning
- Knowledge Distillation
- Synthetic Instruction Data Generation





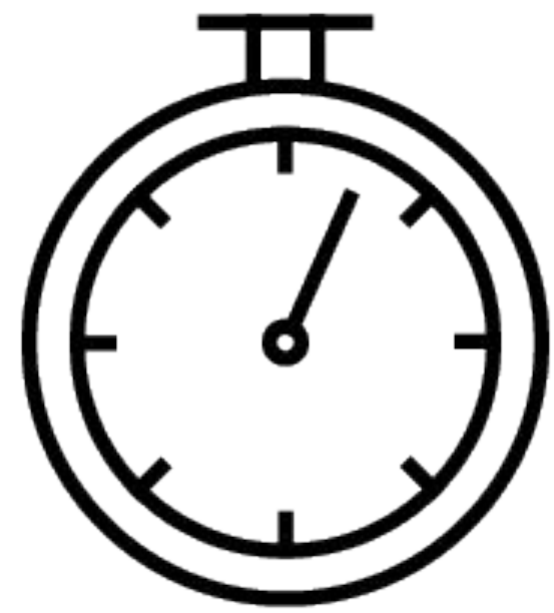


# **Text Processing**



# Challenges with Existing Solutions for Training Foundation Models

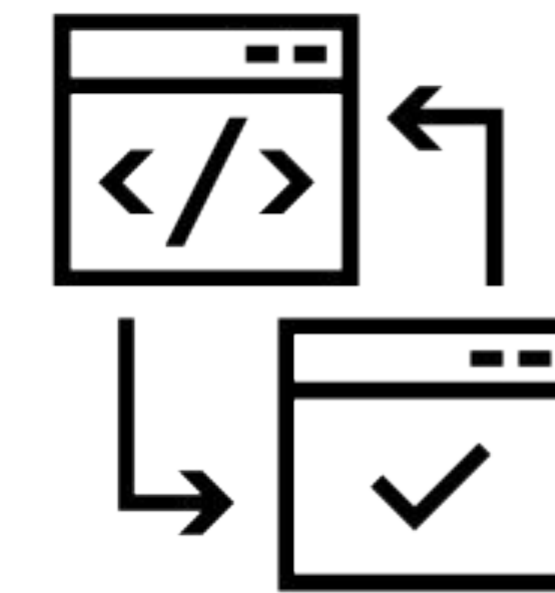
Inefficiencies lead to higher TCO and slower time to market



Longer Processing Time



Un-optimized Models



Un-optimized Pipelines



Knowledge & Expertise



# Why is Data Curation Important?

State-of-the-art data curation is essential for developing state-of-the-art models across all modalities

Higher Accuracy



Properly curated data leads to more improved accuracy across tasks

TCO Savings



Decreases both training and inference costs significantly

Faster Training



Reduces compute requirements by orders of magnitude, enabling faster iterations

Task Specialization



Enables optimization for specific tasks such as reasoning rather than general-purpose solutions

[GitHub](#)

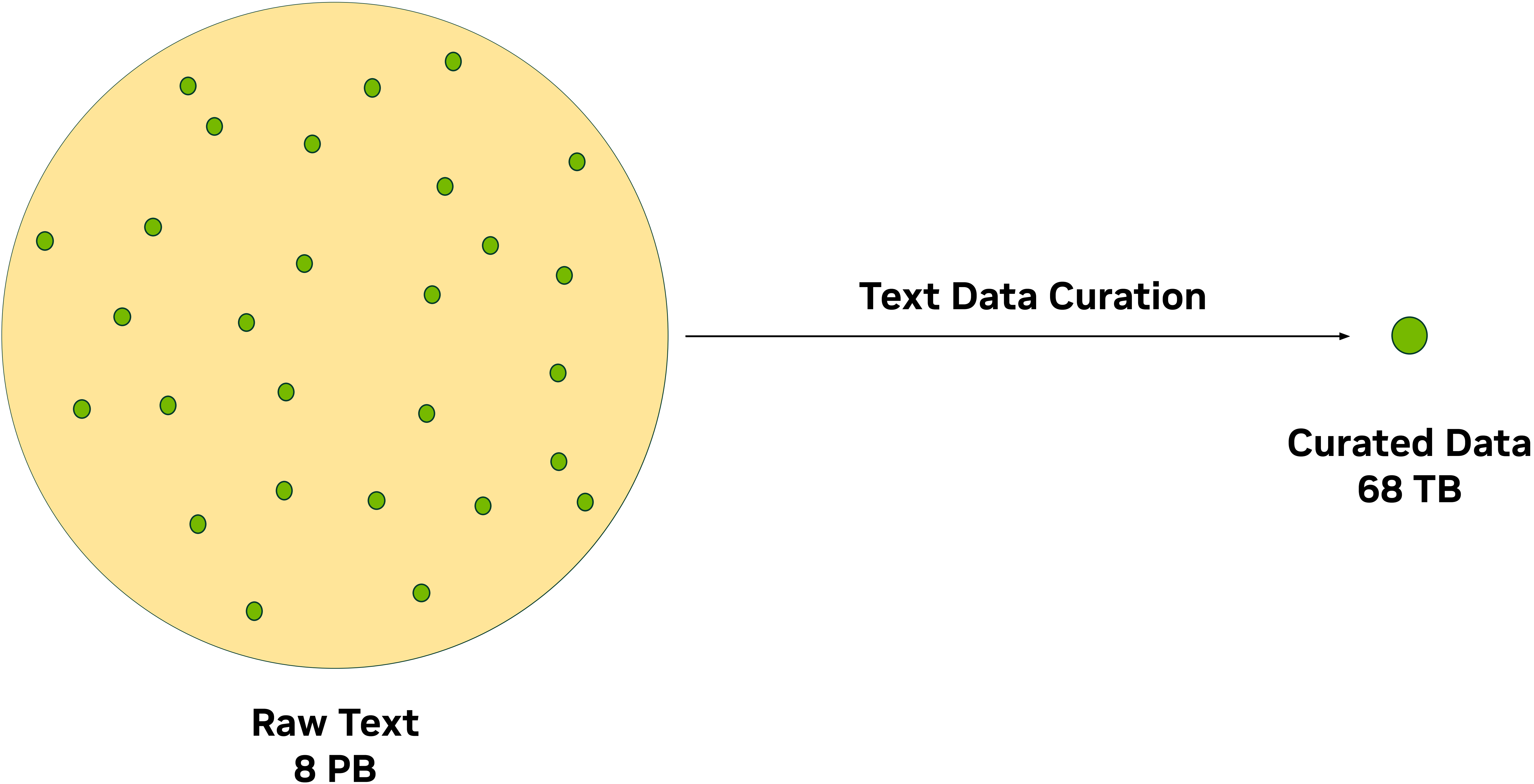
[NeMo framework container](#)

[PyPI](#)

Microservice  
(coming soon)



# Only 1% of Raw Text Data is Curated to Train Foundation Models

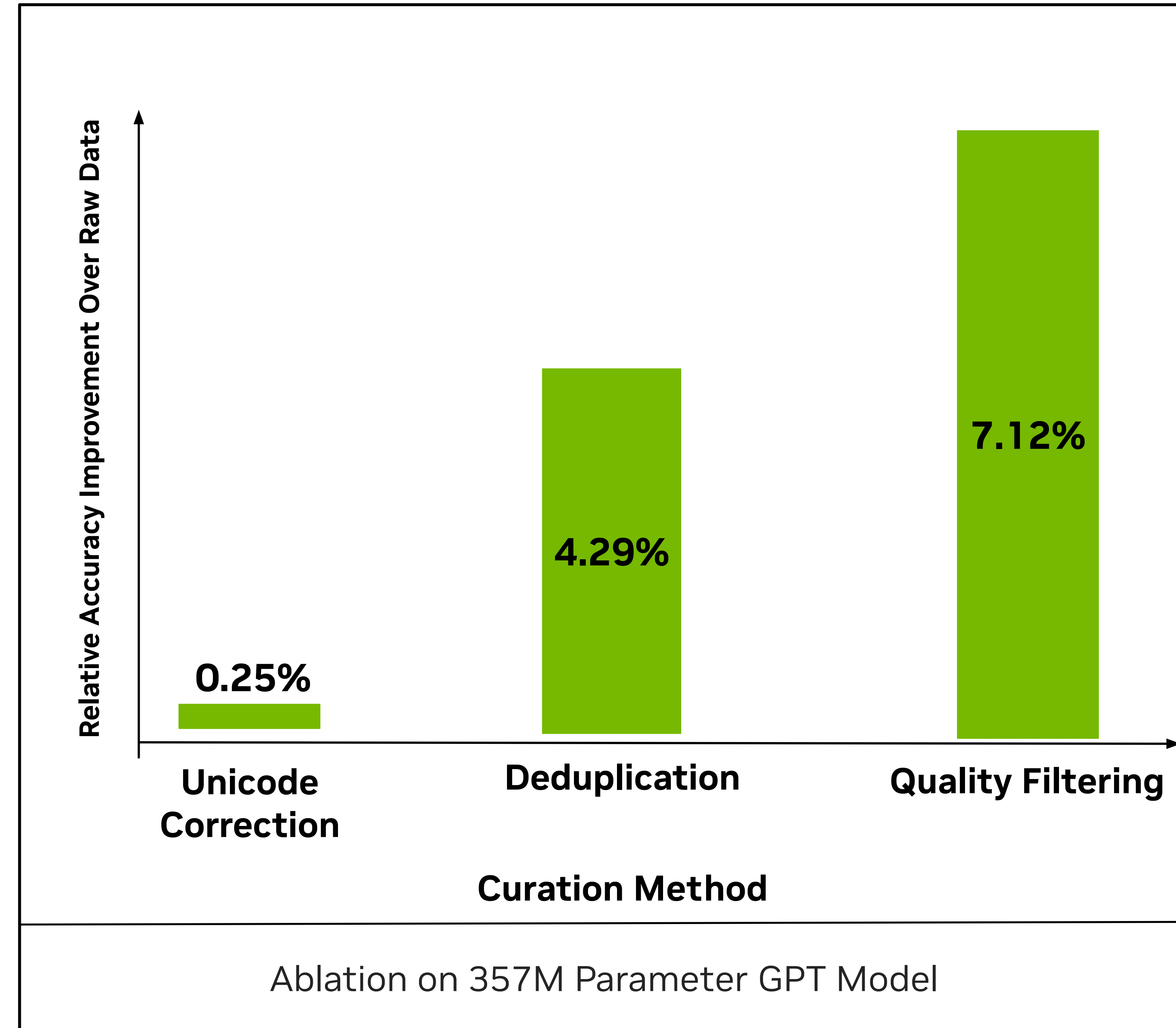




# High Quality Data Processing Maximizes Model Performance

Data Curation helps build SOTA Models

LLM Accuracy Improvement on Curated Data





# NeMo Curator: Features to Train Foundation Models

Achieve higher accuracy with a variety of GPU-accelerated features



## Synthetic Data Generation

- ❑ **Pre-built pipelines** - for tasks like prompt generation, dialogue generation, and entity classification
- ❑ **Modular** - Easily integrate NeMo Curator's features into your existing pipelines
- ❑ **OpenAI API compatible** - Integrate custom Instruct and Reward models



## Deduplication & Classification

- ❑ **Lexical Deduplication** – Identical (Exact) or near identical (Fuzzy)
- ❑ **Semantic Deduplication** – focuses on the meaning rather than the exact text
- ❑ **Classifier Models** - State-of-the-art open models to either enrich or filter your data.



## GPU Acceleration with RAPIDS

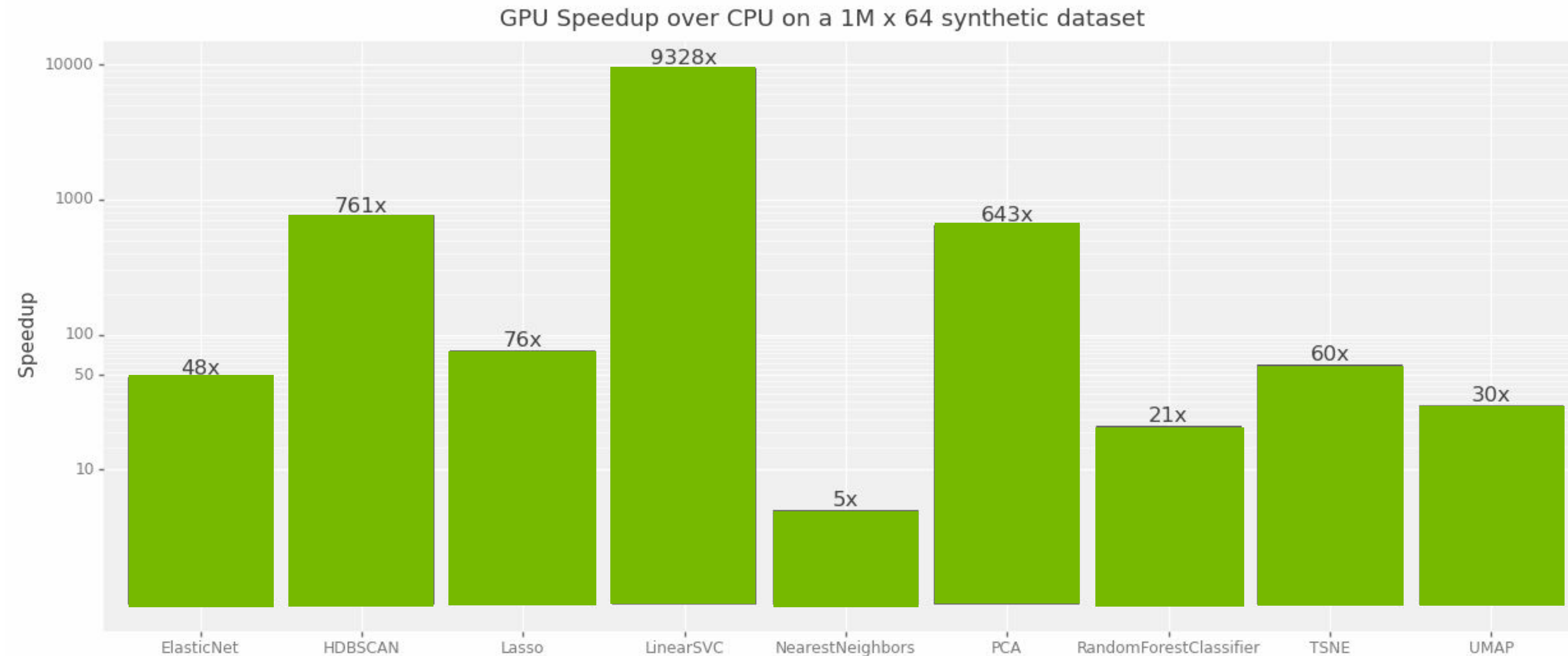
- ❑ **cuDF** - for deduplication & classifier models
- ❑ **cuML** - for K-means clustering in semantic deduplication
- ❑ **cuGraph** – for fuzzy deduplication



# RAPIDS cuML

## 50+ GPU-Accelerated Algorithms & Growing

Accelerated Machine Learning with a Scikit-Learn API



Time Series

Classification

Regression

Clustering

Preprocessing

Cross Validation

Tree Models

Dimensionality  
Reduction

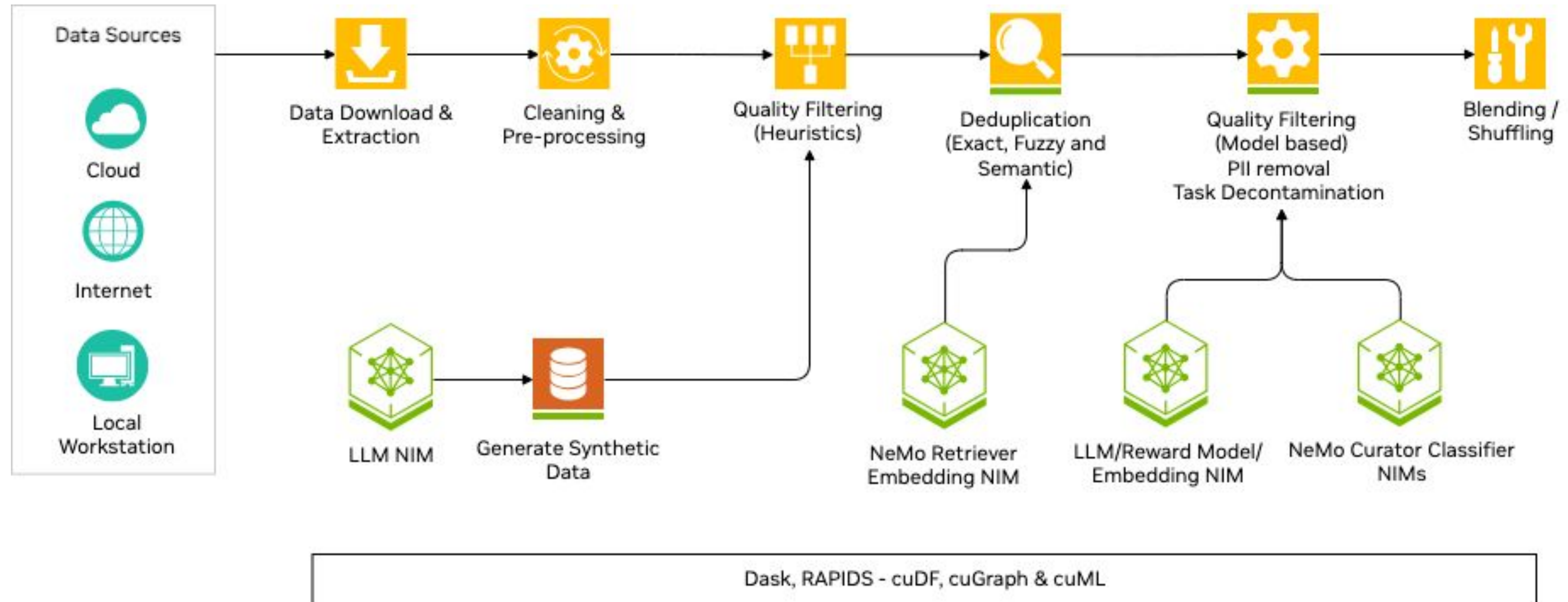
Explainability

A100 GPU vs. AMD EPYC 7642 (96 logical cores)  
cuML 23.04, scikit-learn 1.2.2, umap-learn 0.5.3



# NeMo Curator for Text Processing

Easily integrate different features into your existing pipelines with Python APIs



GPU accelerated

[GitHub](#)

[NeMo framework container](#)

[PyPI](#)



# NeMo Curator: Example

```
# Download your dataset
dataset = download_common_crawl("/datasets/common_crawl/", "2021-04", "2021-10", url_limit=10)
# Build your pipeline
curation_pipeline = Sequential([
    # Fix unicode
    Modify(UnicodeReformatter()),
    # Discard short records
    ScoreFilter(WordCountFilter(min_words=80)),
    # Discard low-quality records
    ScoreFilter(FastTextQualityFilter(model_path="model.bin")),
    # Discard records from the evaluation metrics to prevent test set leakage.
    TaskDecontamination([Winogrande(), Squad(), TriviaQA()])
])
# Execute the pipeline on your dataset
curated_dataset = curation_pipeline(dataset)
```

[GitHub](#)[NeMo framework container](#)[PyPI](#)



# NeMo Curator - Resources

## Getting Started

- [NeMo Framework Container](#)
- [GitHub](#)
- [PyPI](#)
- [Installation Guide](#)
- [Developer Page](#)
- [User Guide/ Docs](#)
- [API Docs](#)
- [Classifier Models](#)
- [Examples](#)
- [Best Practices](#)
- [Bugs](#)
- [Discussions](#)

## Tutorials & Blogs

### Pre-training / DAPT

- [Curating data for LLM training \(Blog\)](#)
- [Curating non-English data \(Blog\)](#)
- [How-to run classifier models](#)
- [All blogs](#)

### Fine-tuning

- [Curating data for PEFT \(Blog\)](#)
- [Curating synthetic data for PEFT](#)
- [SDG using Llama 3.1 405B & Nemotron 4-340B](#)
- [SDG using Nemotron 4-340B](#)



# Customers



70% cost savings, 33% performance improvement



Process rel. between 10 million biological entities through more than a billion edges.

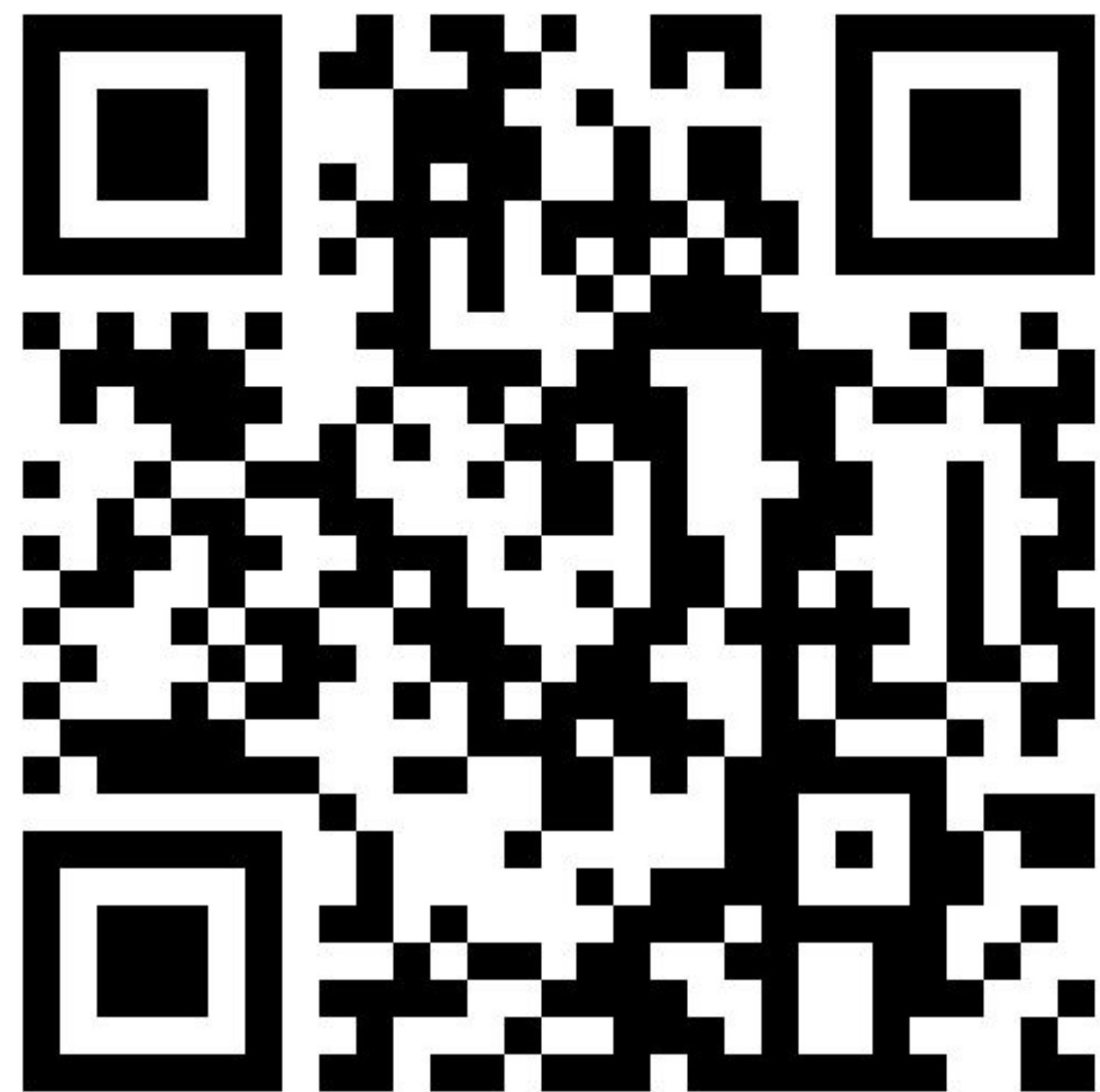


100x faster feature engineering, 20x faster model training, increased forecast accuracy



# Developer Tools and Resources

Accelerate innovation and growth



Learn more: [developer.nvidia.com](https://developer.nvidia.com)

## Individuals

### Software

100s of APIs, models, SDKs, microservices, and early access to NVIDIA tech

### Training

Hands-on self-paced courses, instructor-led workshops, and certifications

### GPU Sandbox

Approval basis, multi-GPU and multi-node

### Learning

Tutorials, self-paced courses, blogs, documentation, code samples

### Community

Dedicated developer forums, meetups, hackathons

### Ecosystem

GTC, NVIDIA Partner Network

## Organizations

### Startups

Cloud credits, engineering resources, technology discounts, exposure to VCs

### Venture Capital

Deal flow and portfolio support for Venture Capital firms

### Higher Education

Teaching kits, training, curriculum co-development, grants

### ISVs and SIs

Engineering guidance, discounts, marketing opportunities

### Research

Grant programs, collaboration opportunities

### Enterprises

Tailored developer training, skills certification, technical support





**Thank You**