RAPIDS, FOSDEM’19

Dr. Christoph Angerer, Manager AI Developer Technologies, NVIDIA

HPC & AI TRANSFORMS INDUSTRIES

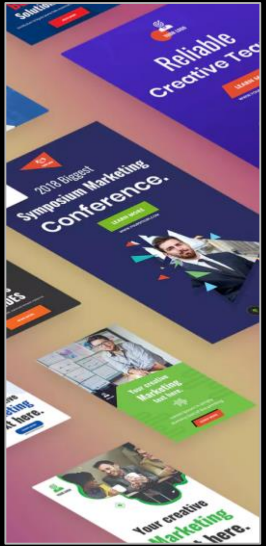
Computational & Data Scientists Are Driving Change

Healthcare

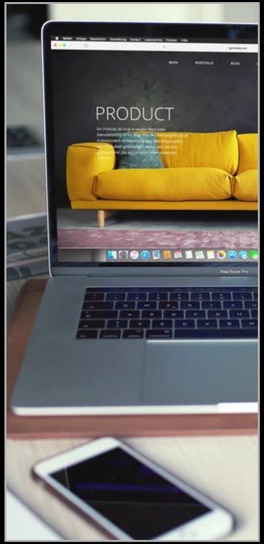
Industrial

Consumer Internet

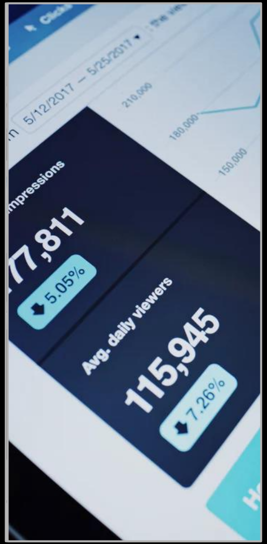
Automotive



Ad Tech / MarTech



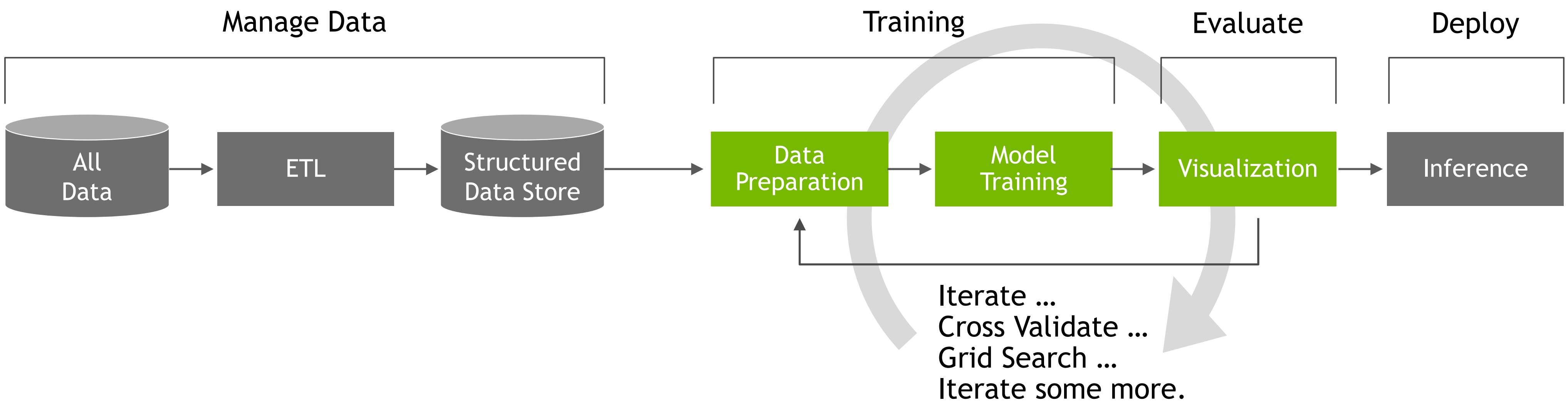
Retail



Financial / Insurance

DATA SCIENCE IS NOT A LINEAR PROCESS

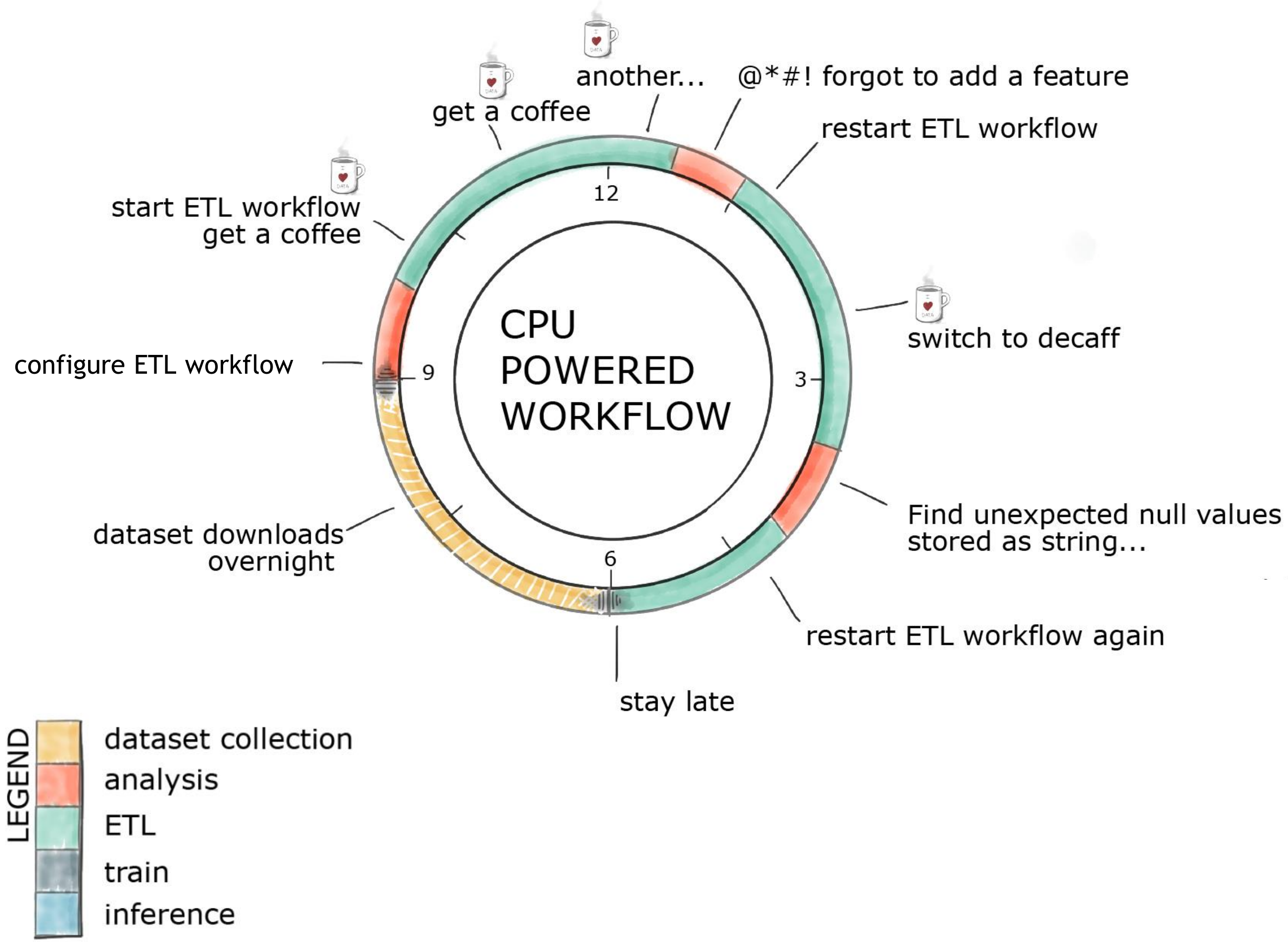
It Requires Exploration and Iterations



Accelerating`Model Training` only does have benefit but doesn't address the whole problem

DAY IN THE LIFE

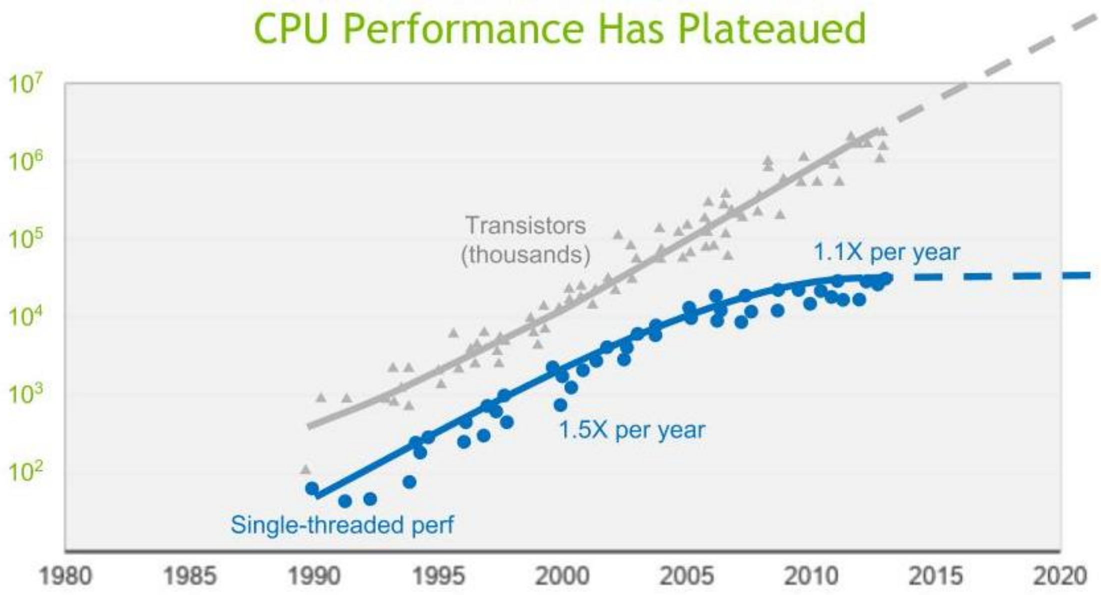
Or: Why did I want to become a Data Scientist?



Data Scientist are valued resources.
Why not give them the environment to
be more productive

PERFORMANCE AND DATA GROWTH

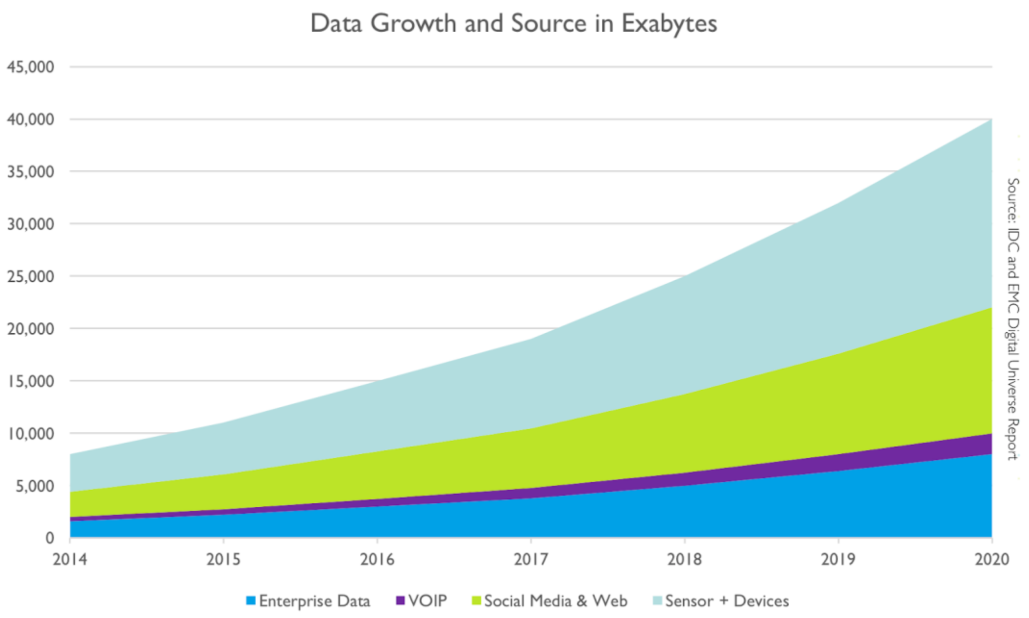
Post-Moore's law



Moore's law is no longer a predictor of capacity in CPU market growth

Distributing CPUs exacerbates the problem

Data sizes continue to grow



TRADITIONAL DATA SCIENCE CLUSTER

Workload Profile:

Fannie Mae Mortgage Data:

- 192GB data set
- 16 years, 68 quarters
- 34.7 Million single family mortgage loans
- 1.85 Billion performance records
- XGBoost training set: 50 features

300 Servers | \$3M | 180 kW

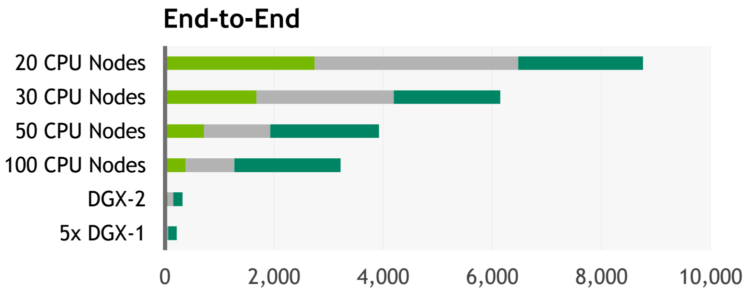
GPU-ACCELERATED MACHINE LEARNING CLUSTER

NVIDIA Data Science Platform with DGX-2

1 DGX-2 | 10 kW

1/8 the Cost | 1/15 the Space

1/18 the Power



DELIVERING DATA SCIENCE VALUE

Maximized Productivity

Top Model Accuracy

Lowest TCO

Oak Ridge National Labs

215x

Speedup Using RAPIDS with XGBoost

Global Retail Giant

$1B

Potential Saving with 4% Error Rate Reduction

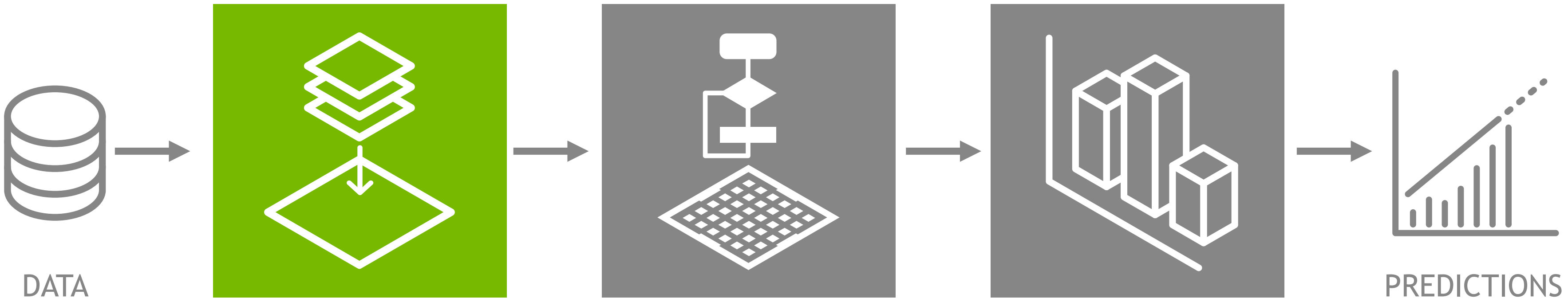
Streaming Media Company

$1.5M

Infrastructure Cost Saving

DATA SCIENCE WORKFLOW WITH RAPIDS

Open Source, End-to-end GPU-accelerated Workflow Built On CUDA



DATA PREPARATION

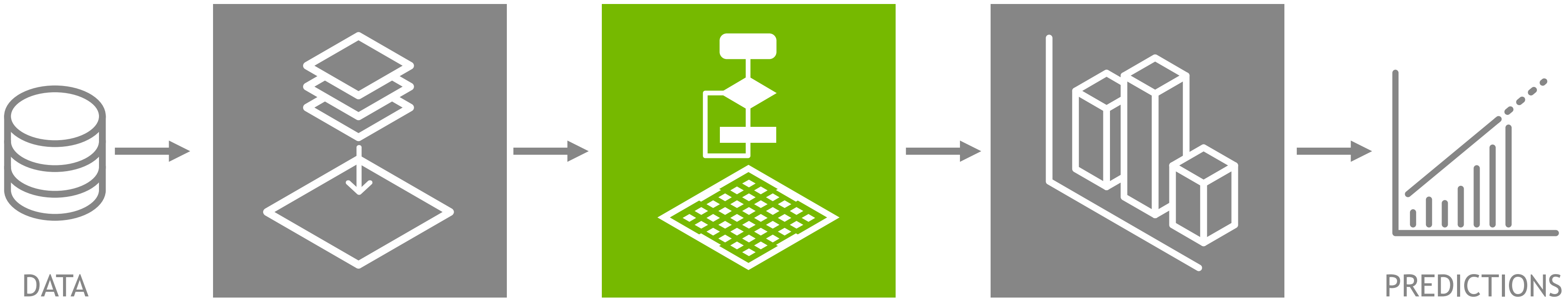
GPUs accelerated compute for in-memory data preparation

Simplified implementation using familiar data science tools

Python drop-in Pandas replacement built on CUDA C++. GPU-accelerated Spark (in development)

DATA SCIENCE WORKFLOW WITH RAPIDS

Open Source, End-to-end GPU-accelerated Workflow Built On CUDA



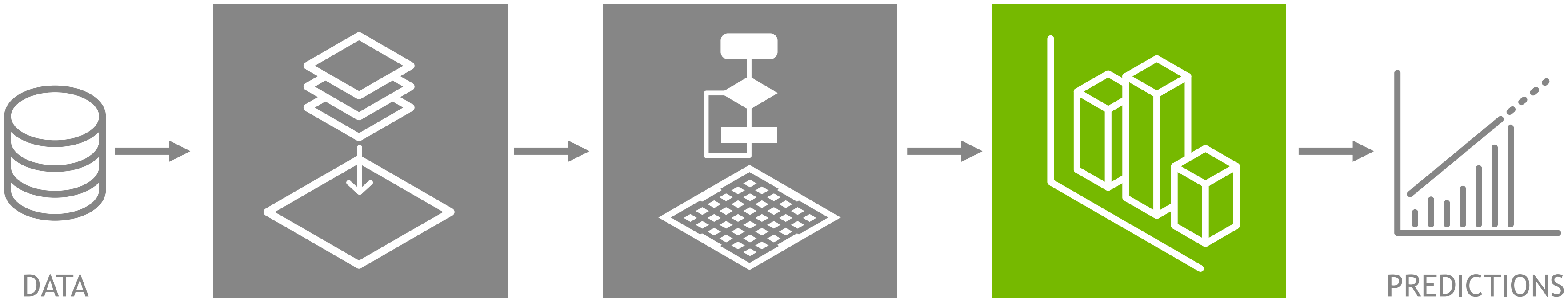
MODEL TRAINING

GPU-acceleration of today’s most popular ML algorithms

XGBoost, PCA, K-means, k-NN, DBScan, tSVD …

DATA SCIENCE WORKFLOW WITH RAPIDS

Open Source, End-to-end GPU-accelerated Workflow Built On CUDA



VISUALIZATION

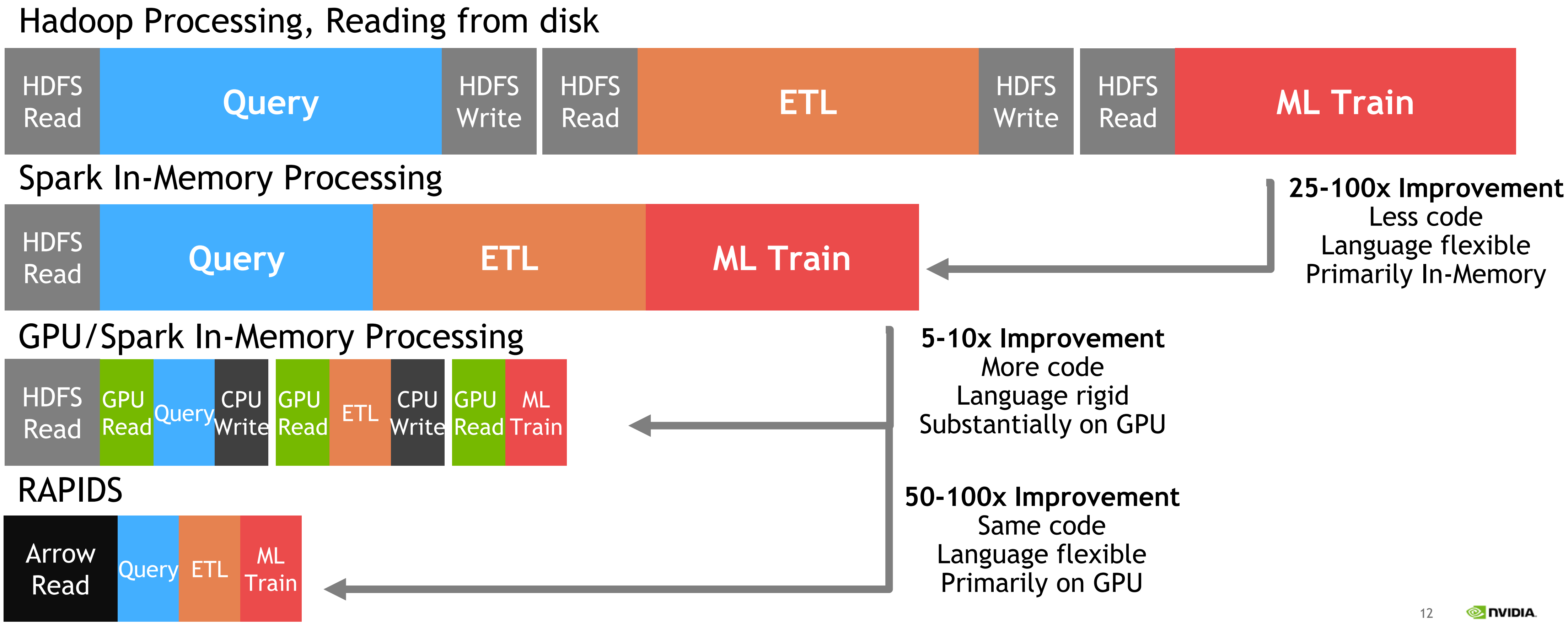
Effortless exploration of datasets, billions of records in milliseconds

Dynamic interaction with data = faster ML model development

Data visualization ecosystem (Graphistry & OmniSci), integrated with RAPIDS

THE EFFECTS OF END-TO-END ACCELERATION

Faster Data Access Less Data Movement



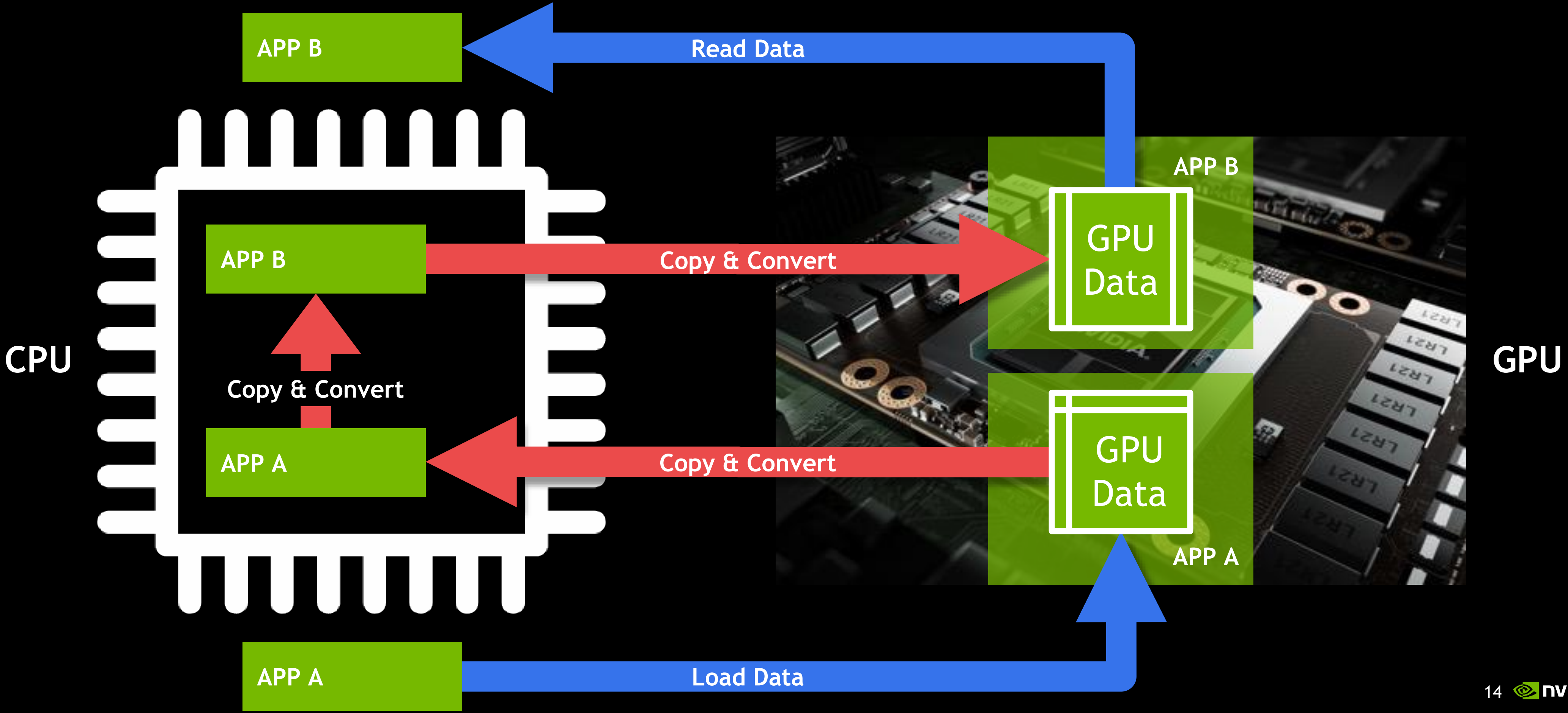
ADDRESSING CHALLENGES IN GPU ACCELERATED DATA SCIENCE

Yes GPUs are fast but …

- Too much data movement
- Too many makeshift data formats
- Writing CUDA C/C++ is involved
- No ***Python*** API for data manipulation

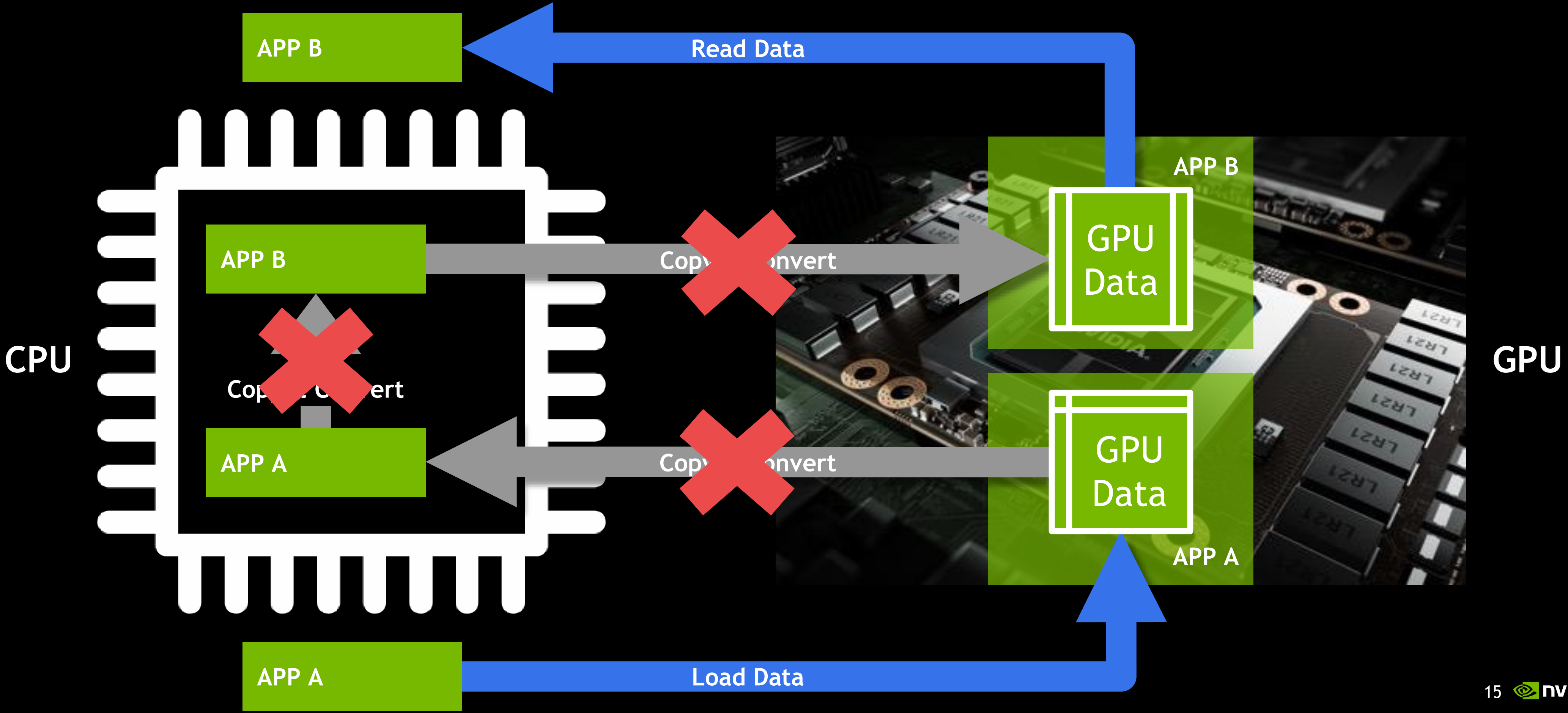
DATA MOVEMENT AND TRANSFORMATION

The bane of productivity and performance

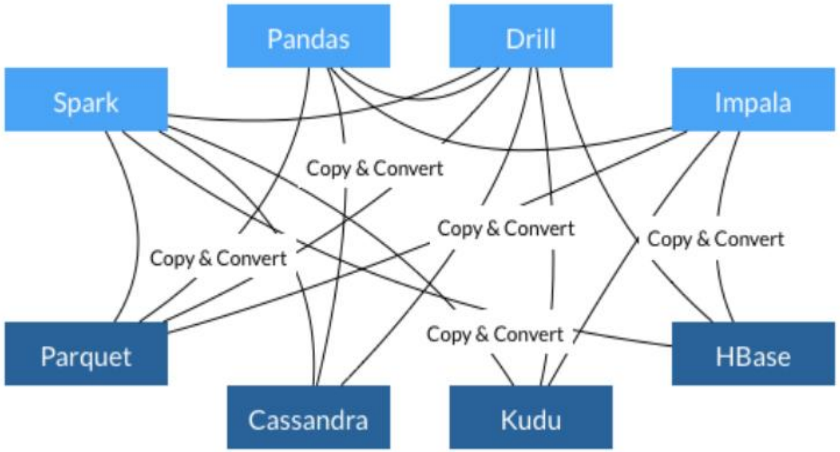


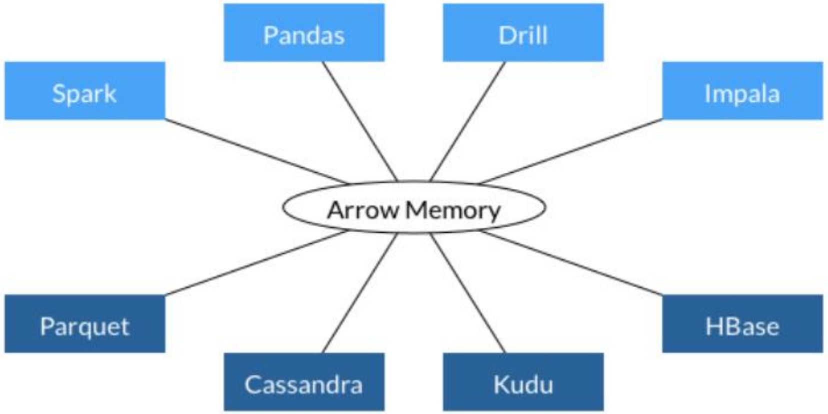
DATA MOVEMENT AND TRANSFORMATION

What if we could keep data on the GPU?



LEARNING FROM APACHE ARROW





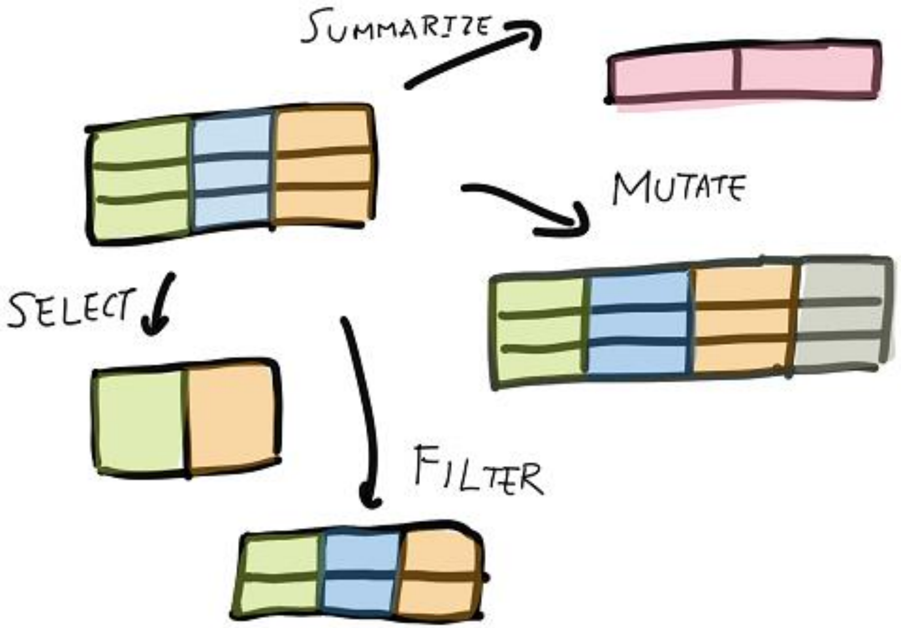
- Each system has its own internal memory format
- 70-80% computation wasted on serialization and deserialization
- Similar functionality implemented in multiple projects

- All systems utilize the same memory format
- No overhead for cross-system communication
- Projects can share functionality (eg, Parquet-to-Arrow reader)

From Apache Arrow Home Page - https://arrow.apache.org/

CUDA DATA FRAMES IN PYTHON

GPUs at your Fingertips




```
df2.withColumn('AgeTimesFare', df2.Age*df2.Fare).show()
```

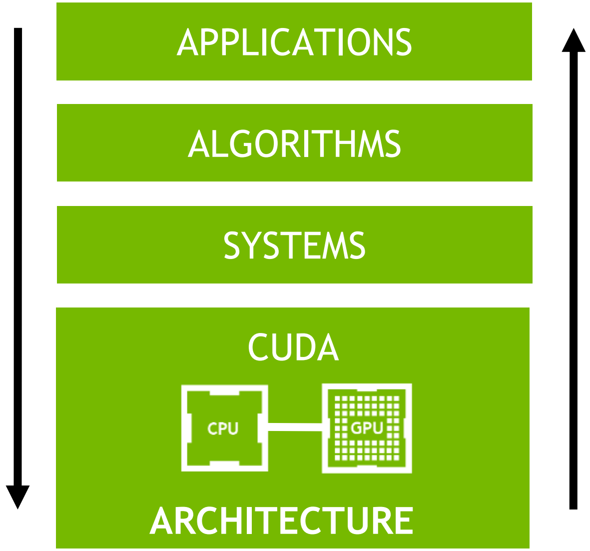
```
+-----------+---+----+------+------------+
|PassengerId|Age|Fare|Pclass|AgeTimesFare|
+-----------+---+----+------+------------+
|          1| 22| 7.3|     3|       160.6|
|          2| 38|71.3|     1|      2709.4|
|          3| 26| 7.9|     3|       205.4|
|          4| 35|53.1|     1|      1858.5|
|          5| 35| 8.0|     3|       280.0|
+-----------+---+----+------+------------+
```

Illustrations from https://changhsinlee.com/pyspark-dataframe-basics/

RAPIDS
OPEN GPU DATA SCIENCE

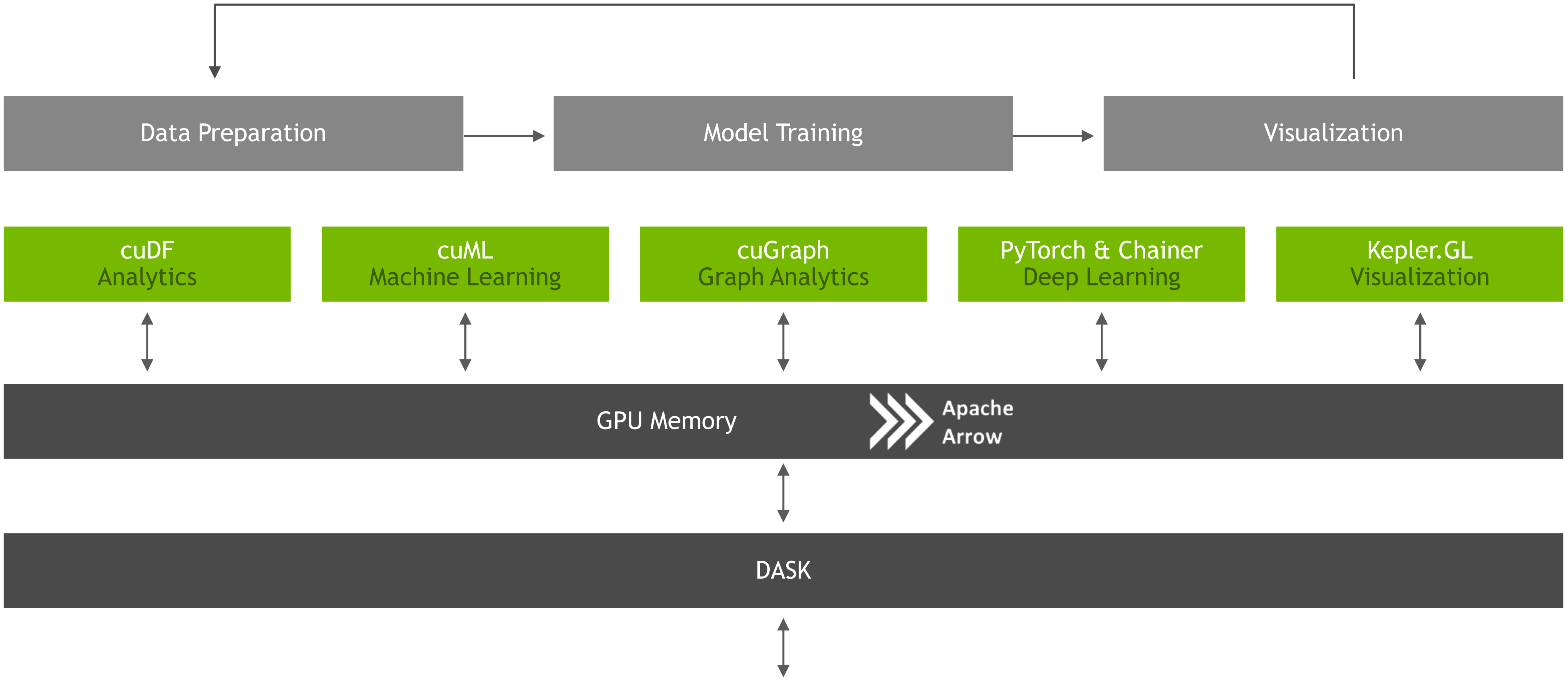
RAPIDS

Open GPU Data Science

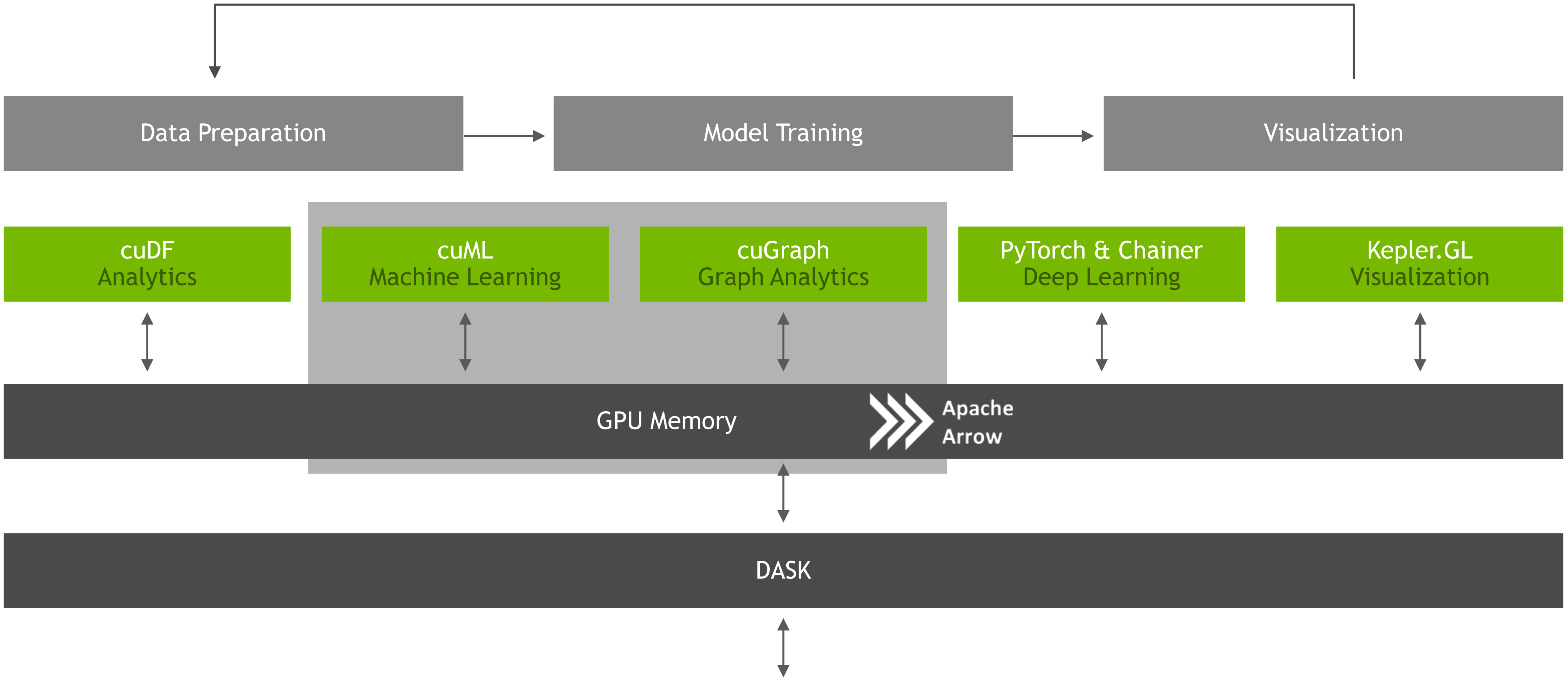


- Learn what the data science community needs
- Use best practices and standards
- Build scalable systems and algorithms
- Test Applications and workflows
- Iterate

RAPIDS COMPONENTS

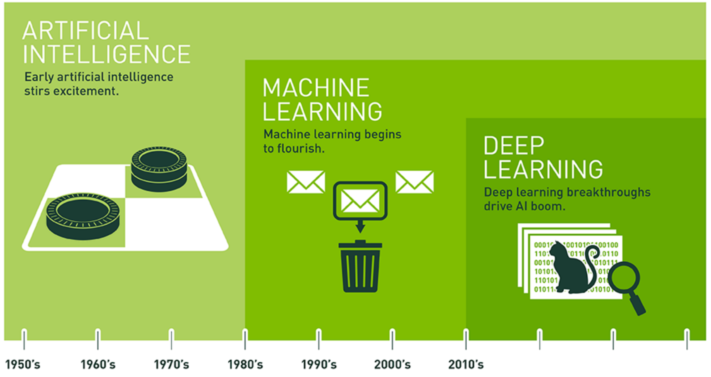


CUML & CUGRAPH



AI LIBRARIES

cuML & cuGraph



Since an early flush of optimism in the 1950s, smaller subsets of artificial intelligence – first machine learning, then deep learning, a subset of machine learning – have created ever larger disruptions.

Accelerating more of the AI ecosystem

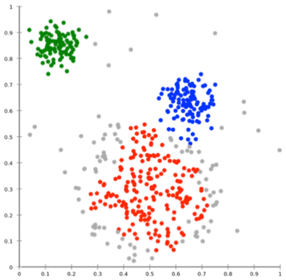
Graph Analytics is fundamental to network analysis

Machine Learning is fundamental to prediction, classification, clustering, anomaly detection and recommendations.

Both can be accelerated with NVIDIA GPU

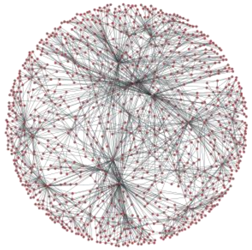
8x V100 20-90x faster than dual socket CPU

Machine Learning



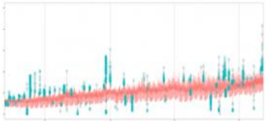
Decisions Trees
Random Forests
Linear Regressions
Logistics Regressions
K-Means
K-Nearest Neighbor
DBSCAN
Kalman Filtering
Principal Components
Single Value Decomposition
Bayesian Inferencing

Graph Analytics



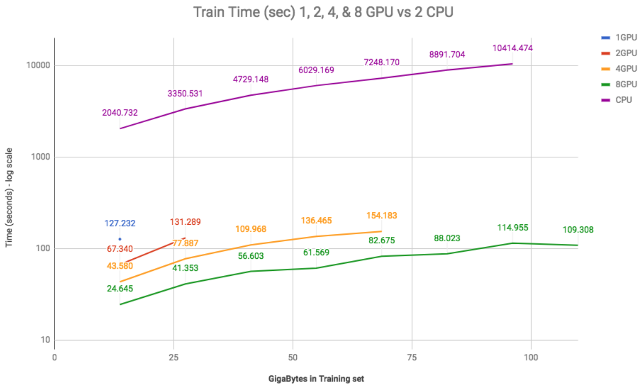
PageRank
BFS
Jaccard Similarity
Single Source Shortest Path
Triangle Counting
Louvain Modularity

Time Series



ARIMA
Holt-Winters

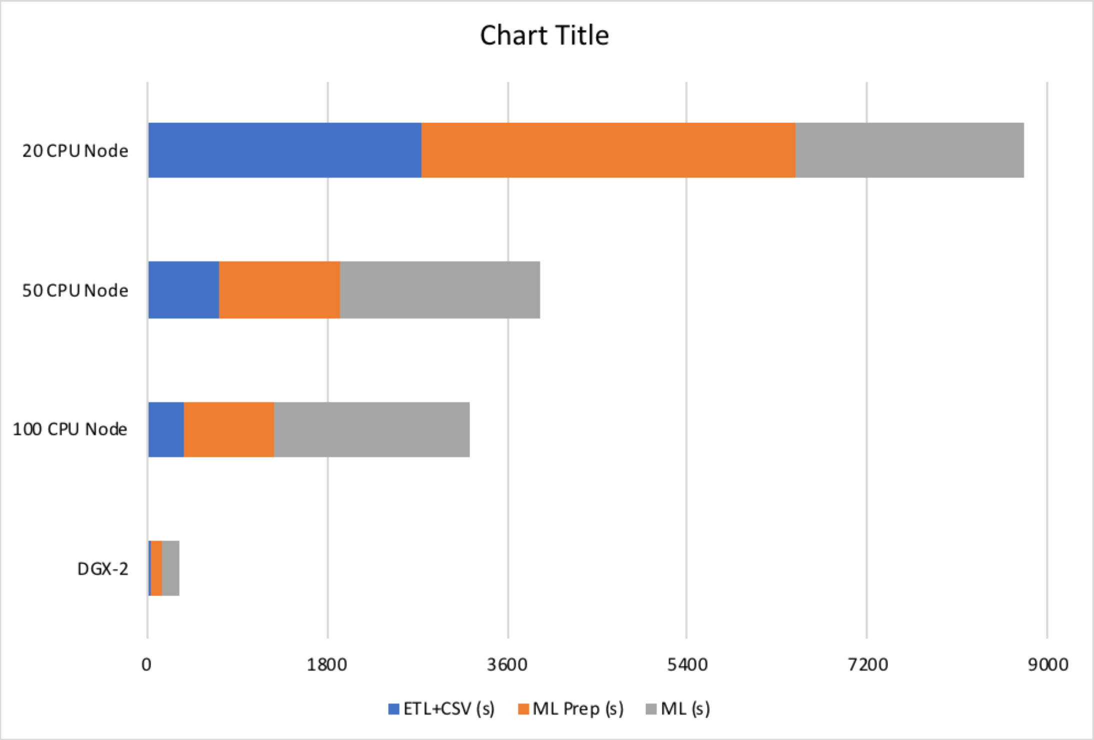
XGBoost, Mortgage Dataset, 90x



3 Hours to 2 mins on 1 DGX-1

CUDF + XGBOOST

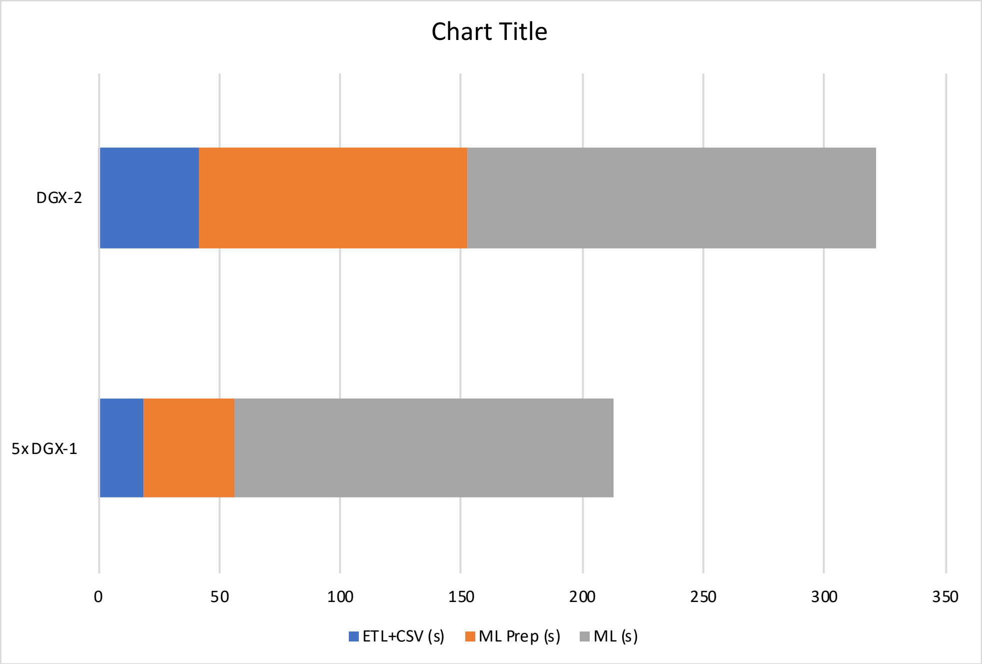
DGX-2 vs Scale Out CPU Cluster



- Full end to end pipeline
- Leveraging Dask + PyGDF
- Store each GPU results in sys mem then read back in
- Arrow to Dmatrix (CSR) for XGBoost

CUDF + XGBOOST

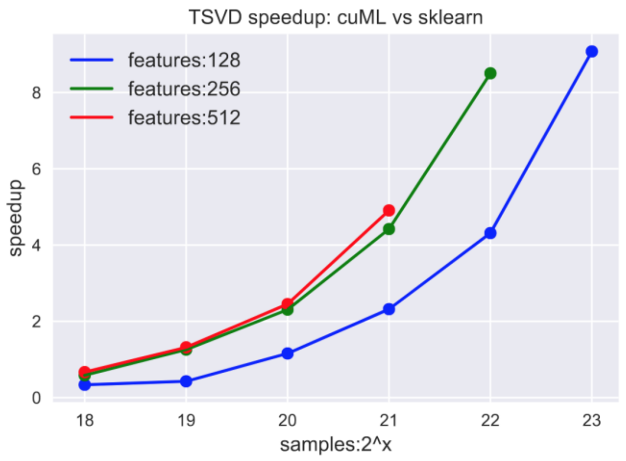
Scale Out GPU Cluster vs DGX-2

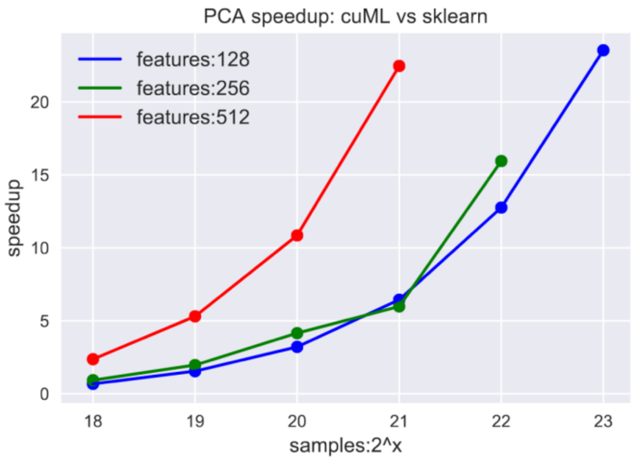


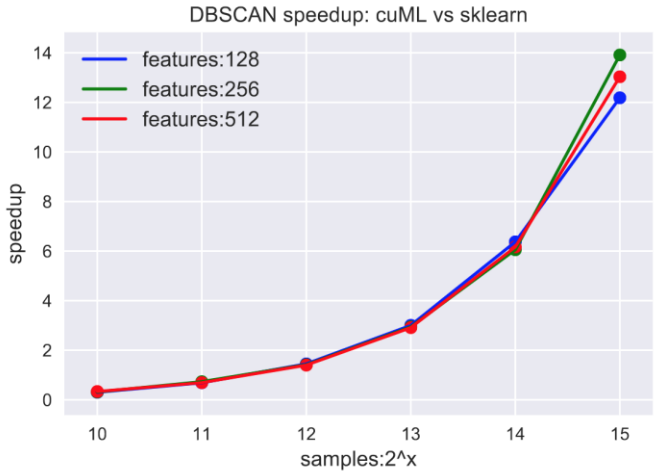
- Full end to end pipeline
- Leveraging Dask for multi-node + PyGDF
- Store each GPU results in sys mem then read back in
- Arrow to Dmatrix (CSR) for XGBoost

CUML

Benchmarks of initial algorithms







NEAR FUTURE WORK ON CUML

Additional algorithms in development right now

K-means - Released

K-NN - Released

Kalman filter – v0.5

GLM – v0.5

Random Forests - v0.6

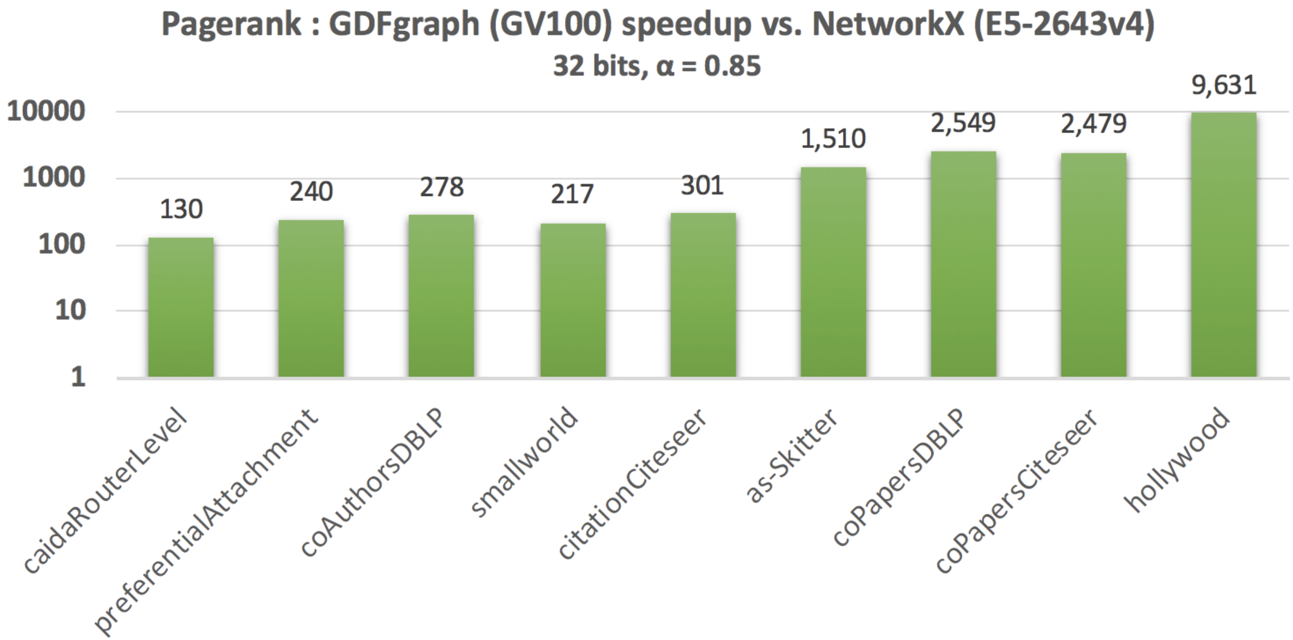
ARIMA – v0.6

UMAP – v0.6

Collaborative filtering – Q2 2019

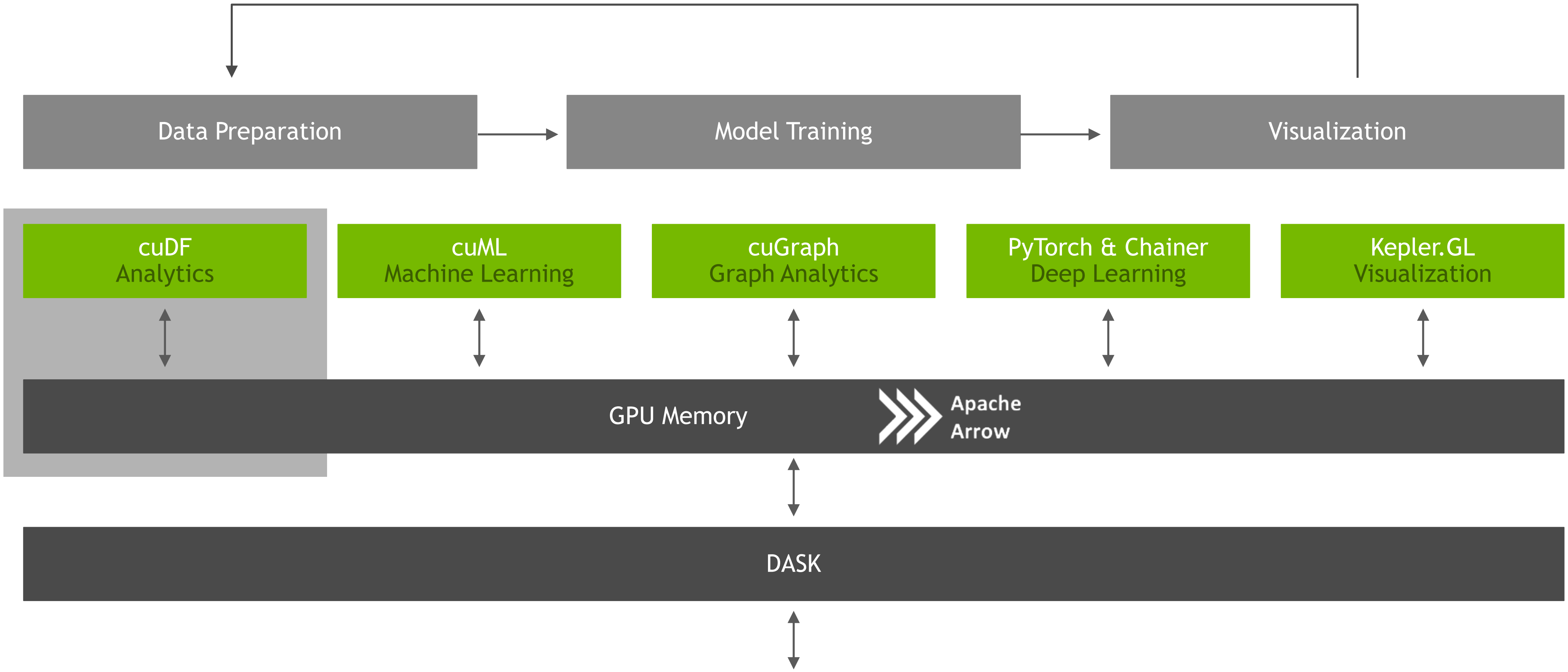
CUGRAPH

GPU-Accelerated Graph Analytics Library



Coming Soon:
Full NVGraph Integration Q1 2019

CUDF


Data Preparation
Model Training
Visualization
cuDF
Analytics
cuML
Machine Learning
cuGraph
Graph Analytics
PyTorch & Chainer
Deep Learning
Kepler.GL
Visualization
GPU Memory
Apache
Arrow
DASK

CUDF

GPU DataFrame library

	Area Abbreviation	Area Code	Area	Item Code	Item	Element Code	Element	Unit	latitude	longitude	...	Y2004	Y2005	Y2006	Y2007	Y2008
0	AF	2	Afghanistan	2511	Wheat and products	5142	Food	1000 tonnes	33.94	67.71	...	3249.0	3486.0	3704.0	4164.0	4252.0
1	AF	2	Afghanistan	2805	Rice (Milled Equivalent)	5142	Food	1000 tonnes	33.94	67.71	...	419.0	445.0	546.0	455.0	490.0
2	AF	2	Afghanistan	2513	Barley and products	5521	Feed	1000 tonnes	33.94	67.71	...	58.0	236.0	262.0	263.0	230.0
3	AF	2	Afghanistan	2513	Barley and products	5142	Food	1000 tonnes	33.94	67.71	...	185.0	43.0	44.0	48.0	62.0
4	AF	2	Afghanistan	2514	Maize and products	5521	Feed	1000 tonnes	33.94	67.71	...	120.0	208.0	233.0	249.0	247.0
5	AF	2	Afghanistan	2514	Maize and products	5142	Food	1000 tonnes	33.94	67.71	...	231.0	67.0	82.0	67.0	69.0
6	AF	2	Afghanistan	2517	Millet and products	5142	Food	1000 tonnes	33.94	67.71	...	15.0	21.0	11.0	19.0	21.0
7	AF	2	Afghanistan	2520	Cereals, Other	5142	Food	1000 tonnes	33.94	67.71	...	2.0	1.0	1.0	0.0	0.0
8	AF	2	Afghanistan	2531	Potatoes and products	5142	Food	1000 tonnes	33.94	67.71	...	276.0	294.0	294.0	260.0	242.0
9	AF	2	Afghanistan	2536	Sugar cane	5521	Feed	1000 tonnes	33.94	67.71	...	50.0	29.0	61.0	65.0	54.0
10	AF	2	Afghanistan	2537	Sugar beet	5521	Feed	1000 tonnes	33.94	67.71	...	0.0	0.0	0.0	0.0	0.0

- Apache Arrow data format
- Pandas-like API

- Unary and Binary Operations
- Joins / Merges
- GroupBys
- Filters
- User-Defined Functions (UDFs)
- Accelerated file readers
- Etc.

CUDF

Today

CUDA

- Low level library containing function implementations and C/C++ API
- Importing/exporting Apache Arrow using the CUDA IPC mechanism
- CUDA kernels to perform element-wise math operations on GPU DataFrame columns
- CUDA sort, join, groupby, and reduction operations on GPU DataFrames

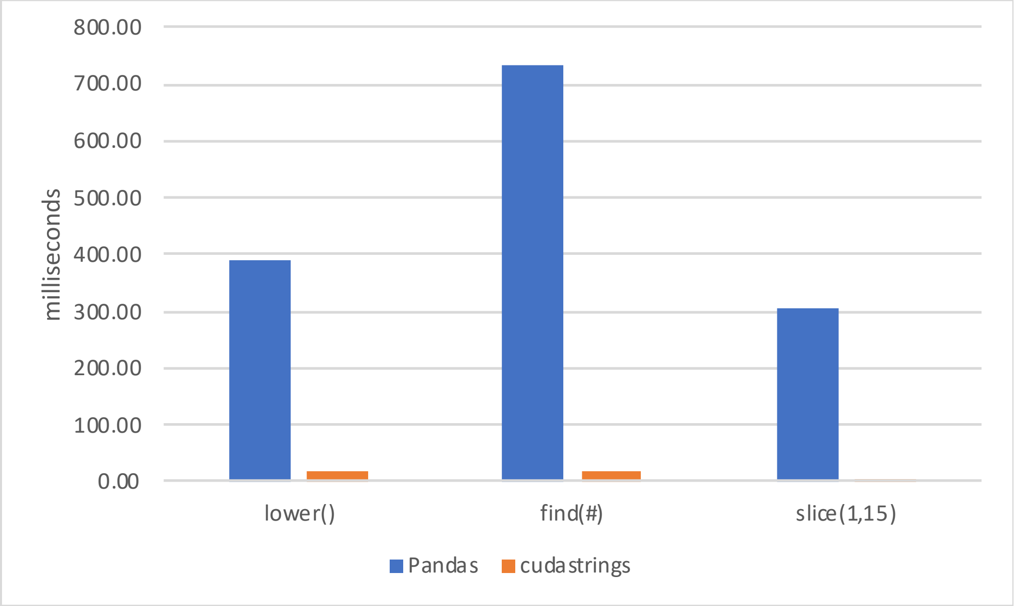
With Python Bindings

- A Python library for manipulating GPU DataFrames
- Python interface to CUDA C++ with additional functionality
- Creating Apache Arrow from Numpy arrays, Pandas DataFrames, and PyArrow Tables
- JIT compilation of User-Defined Functions (UDFs) using Numba

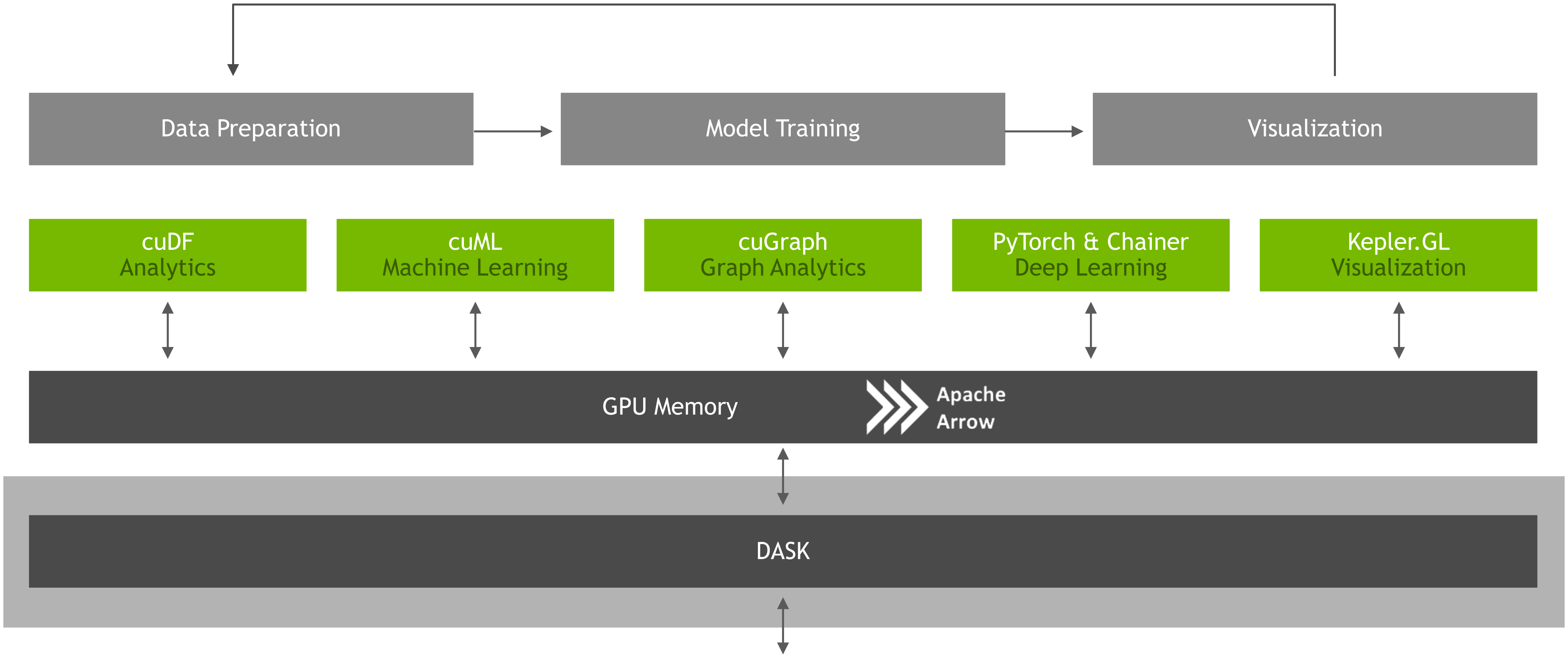
CUSTRING & NVSTRING

GPU-Accelerated string functions with a Pandas-like API

- API and functionality is following Pandas: https://pandas.pydata.org/pandas-docs/stable/api.html#string-handling
- lower()
    - ~22x speedup
- find()
    - ~40x speedup
- slice()
    - ~100x speedup



DASK



Data Preparation
Model Training
Visualization
cuDF
Analytics
cuML
Machine Learning
cuGraph
Graph Analytics
PyTorch & Chainer
Deep Learning
Kepler.GL
Visualization
GPU Memory
Apache
Arrow
DASK

DASK

What is Dask and why does RAPIDS use it for scaling out?

- Dask is a distributed computation scheduler built to scale Python workloads from laptops to supercomputer clusters.
- Extremely modular with scheduling, compute, data transfer, and out-of-core handling all being disjointed allowing us to plug in our own implementations.
- Can easily run multiple Dask workers per node to allow for an easier development model of one worker per GPU regardless of single node or multi node environment.

DASK

Scale up and out with cuDF

- Use cuDF primitives underneath in map-reduce style operations with the same high level API

- Instead of using typical Dask data movement of pickling objects and sending via TCP sockets, take advantage of hardware advancements using a communications framework called OpenUCX:

    - For intranode data movement, utilize NVLink and PCIe peer-to-peer communications

    - For internode data movement, utilize GPU RDMA over Infiniband and RoCE

DASK

https://github.com/rapidsai/dask_gdf

DASK

Scale up and out with cuML

- Native integration with Dask + cuDF
- Can easily use Dask workers to initialize NCCL for optimized gather / scatter operations
  - Example: this is how the dask-xgboost included in the container works for multi-GPU and multi-node, multi-GPU
- Provides easy to use, high level primitives for synchronization of workers which is needed for many ML algorithms

LOOKING TO THE FUTURE

GPU DATAFRAME

Next few months

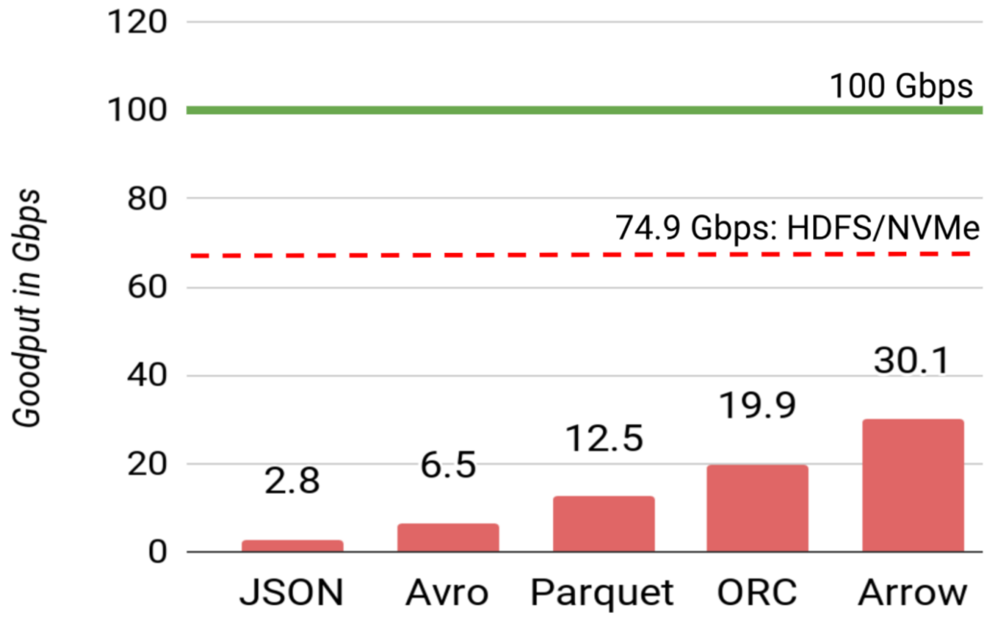
- Continue improving performance and functionality
  - Single GPU
  - Single node, multi GPU
  - Multi node, multi GPU
- String Support
  - Support for specific “string” dtype with GPU-accelerated functionality similar to Pandas
- Accelerated Data Loading
  - File formats: CSV, Parquet, ORC – to start

ACCELERATED DATA LOADING

CPUs bottleneck data loading in high throughput systems

- CSV Reader
    - Follows API of pandas.read_csv
    - Current implementation is >10x speed improvement over pandas
- Parquet Reader
    - Work in progress: https://github.com/gpuopenanalytics/libgdf/pull/85
    - Will follow API of pandas.read_parquet
- ORC Reader

- Additionally looking towards GPU-accelerating decompression for common compression schemes



Source: *Apache Crail blog: SQL Performance: Part 1 - Input File Formats*

PYTHON CUDA ARRAY INTERFACE

Interoperability for Python GPU Array Libraries

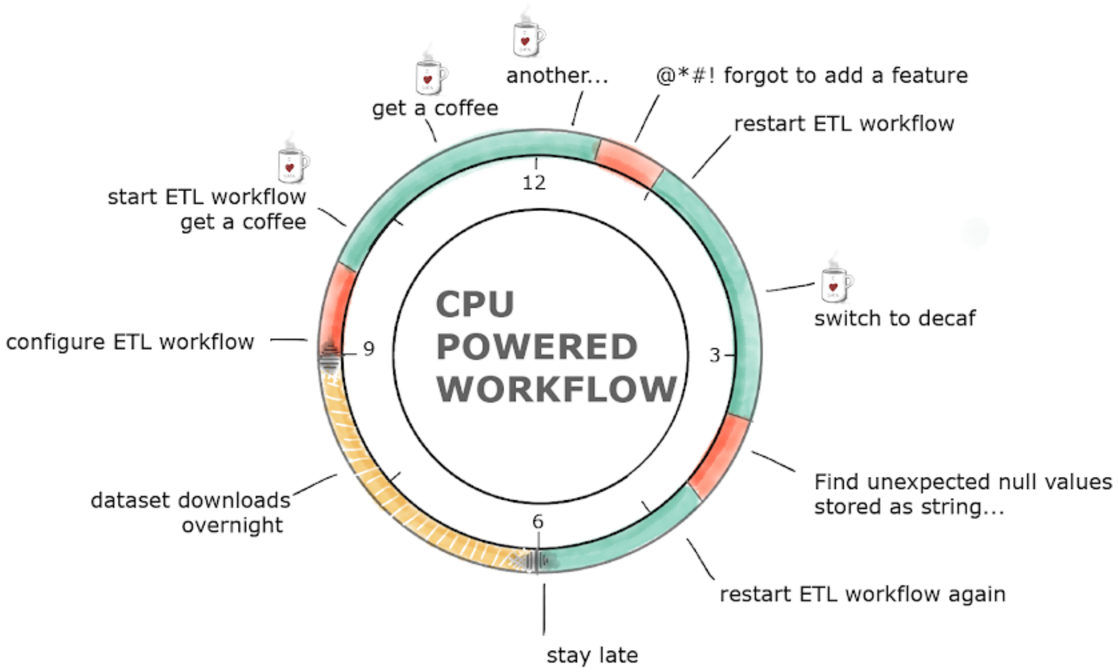
- The CUDA array interface is a standard format that describes a GPU array to allow sharing GPU arrays between different libraries without needing to copy or convert data

- Numba, CuPy, and PyTorch are the first libraries to adopt the interface:
    - https://numba.pydata.org/numba-doc/dev/cuda/cuda_array_interface.html
    - https://github.com/cupy/cupy/releases/tag/v5.0.0b4
    - https://github.com/pytorch/pytorch/pull/11984

CONCLUDING REMARKS

A DAY IN THE LIFE

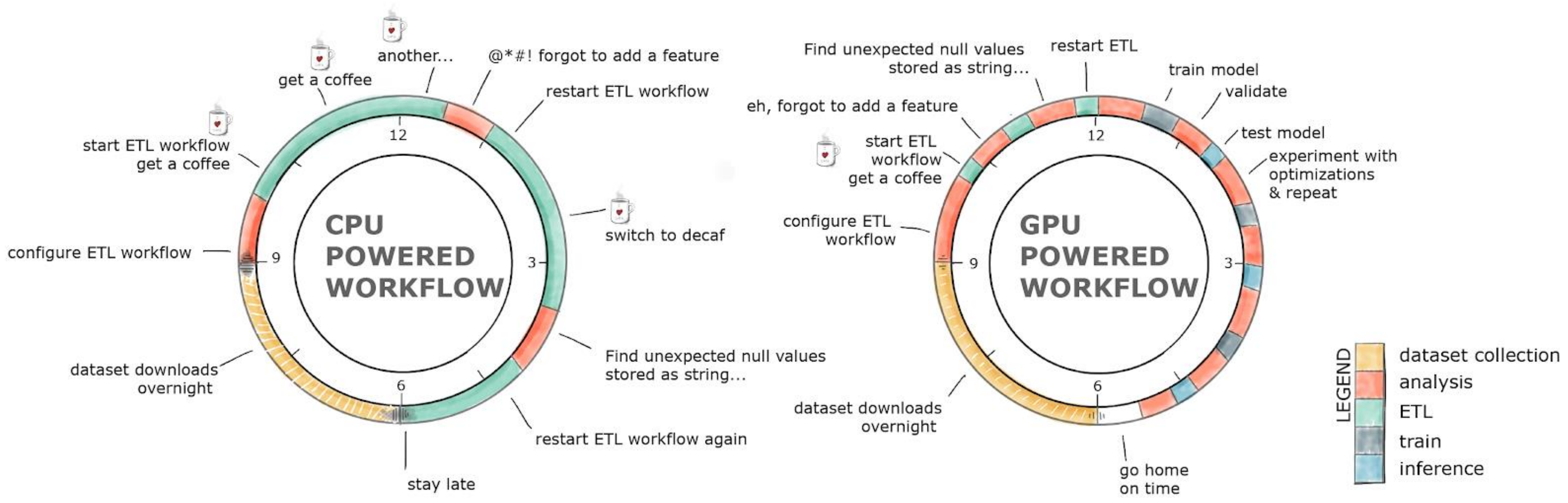
Or: Why did I want to become a Data Scientist?



A DAY IN THE LIFE

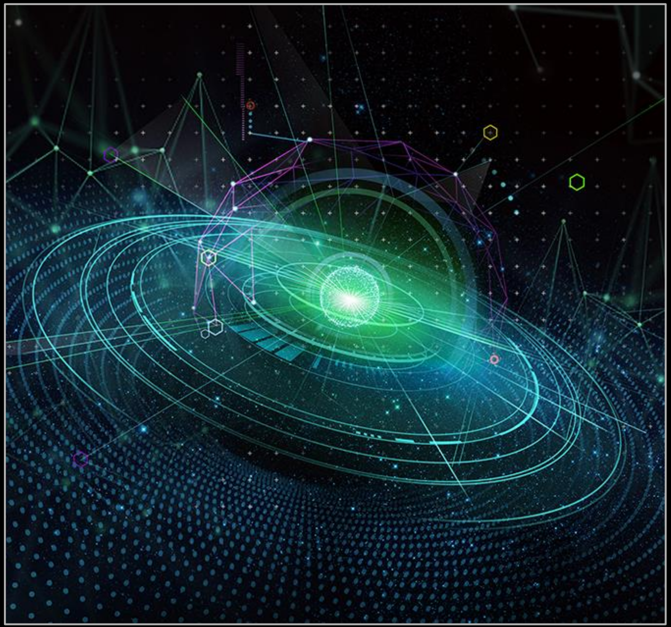
Or: Why did I want to become a Data Scientist?

A: For the Data Science. And coffee.

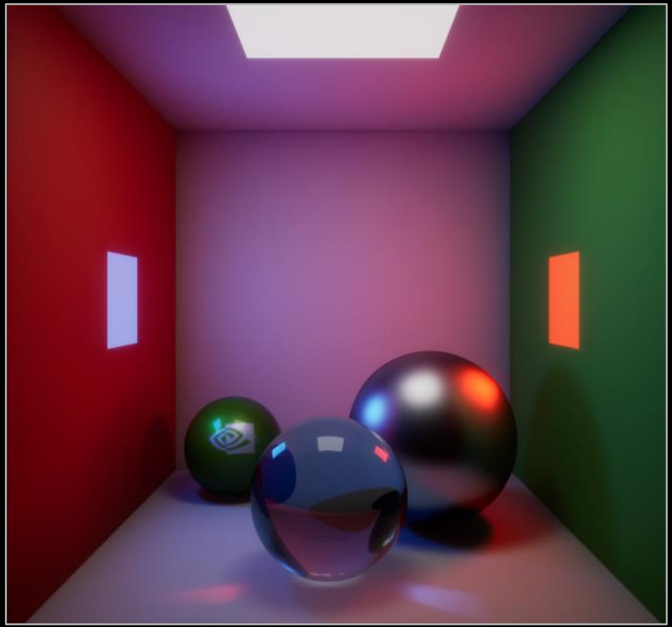


ONE ARCHITECTURE FOR HPC AND DATA SCIENCE

Simulation



Data Analytics



Visualization

RAPIDS

How do I get the software?

- https://github.com/rapidsai
- https://anaconda.org/rapidsai/
- WIP:
  - https://pypi.org/project/cudf
  - https://pypi.org/project/cuml

- https://ngc.nvidia.com/registry/nvidia-rapidsai-rapidsai
- https://hub.docker.com/r/rapidsai/rapidsai/

JOIN THE MOVEMENT

Everyone Can Help!

APACHE ARROW

https://arrow.apache.org/

@ApacheArrow

RAPIDS

https://rapids.ai

@RAPIDSAI

GPU Open Analytics Initiative

http://gpuopenanalytics.com/

@GPUOAI

Integrations, feedback, documentation support, pull requests, new issues, or code donations welcomed!

THANK YOU