

Data Management



Katherine Scott

@kscottz



One of the biggest failures I see in junior ML/CV engineers is a complete lack of interest in building data sets. While it is boring grunt work I think there is so much to be learned in putting together a dataset. It is like half the problem.

♡ 571 11:50 AM - Feb 1, 2019



mat kelcey

@mat_kelcey



for my last few ML projects the complexity hasn't been in the modelling or training; it's been in input preprocessing. find myself running out of CPU more than GPU & in one project i'm actually unsure how to optimise the python further (& am considering c++ for one piece)

♡ 130 2:01 PM - Feb 11, 2019



Vicki Boykis

@vboykis



Have been extremely curious about this for a while now, so I decided to create a poll.

"As someone titled 'data scientist' in 2019, I spend most of (60%+) my time:"

("Other") also welcome, add it in the replies.

♡ 189 8:17 AM - Jan 28, 2019



6% Picking features/models

67% Cleaning data/Moving data

4% Deploying models in prod

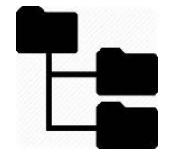
23% Analyzing/presenting data

2,116 votes • Final results

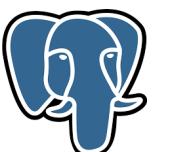
<https://veekaybee.github.io/2019/02/13/data-science-is-different/>

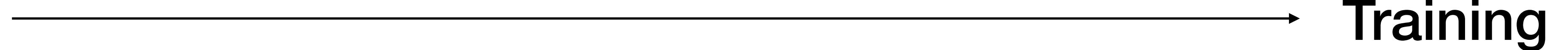
Data Sources

 Images

 Text Corpus

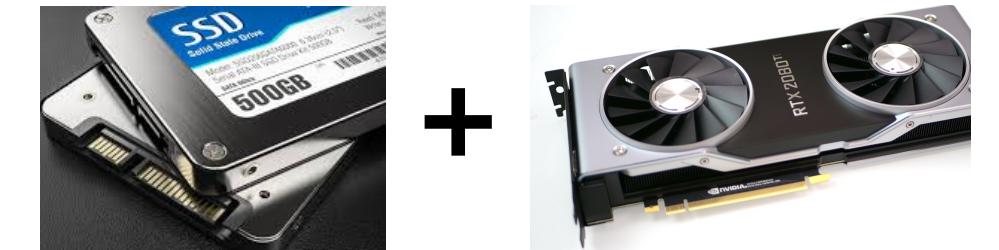
 Logs

 DB records



Different for every project / company!

Local Filesystem



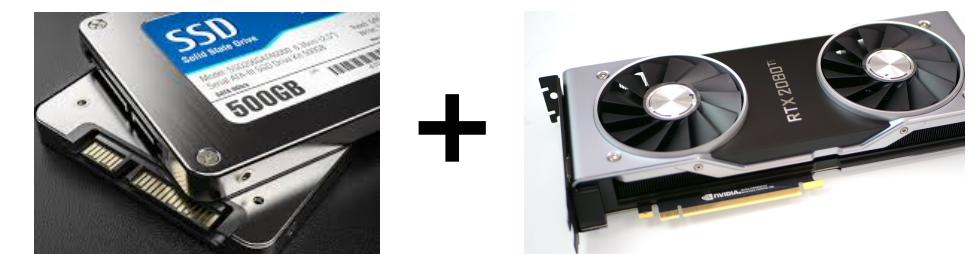
GPU

Data Sources

 Images

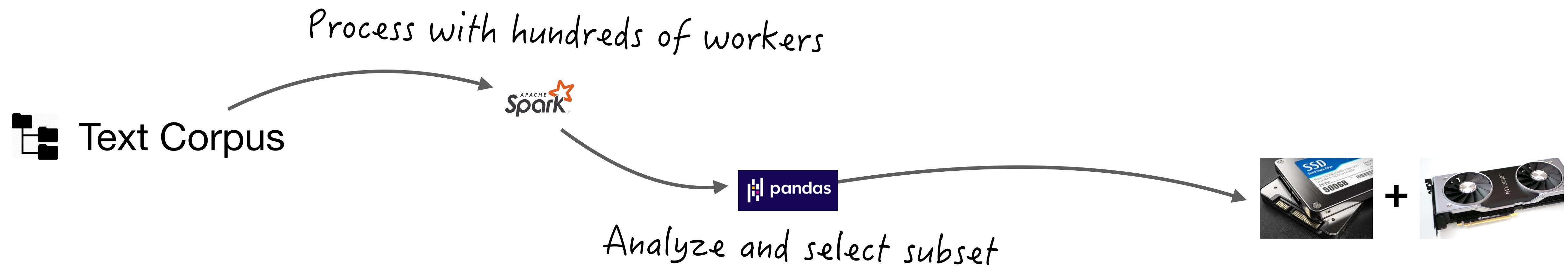
Simply Download

Training



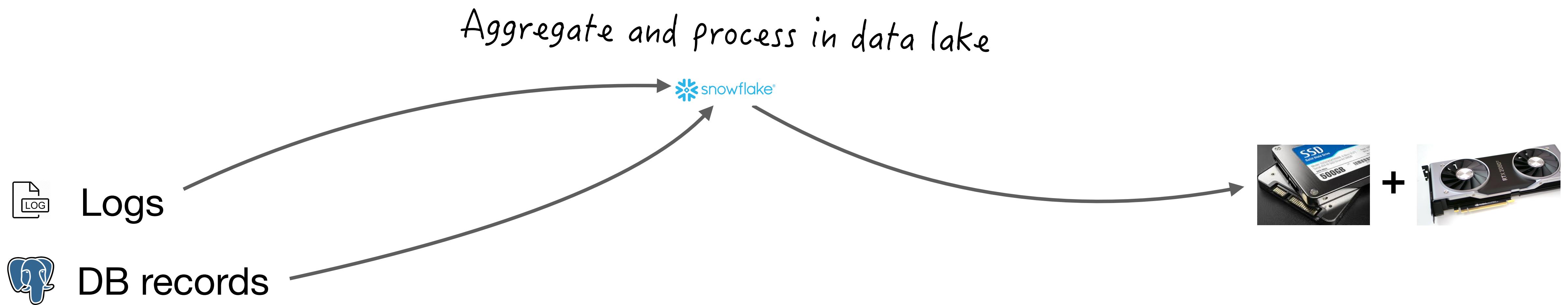
Data Sources

Training



Data Sources

Training

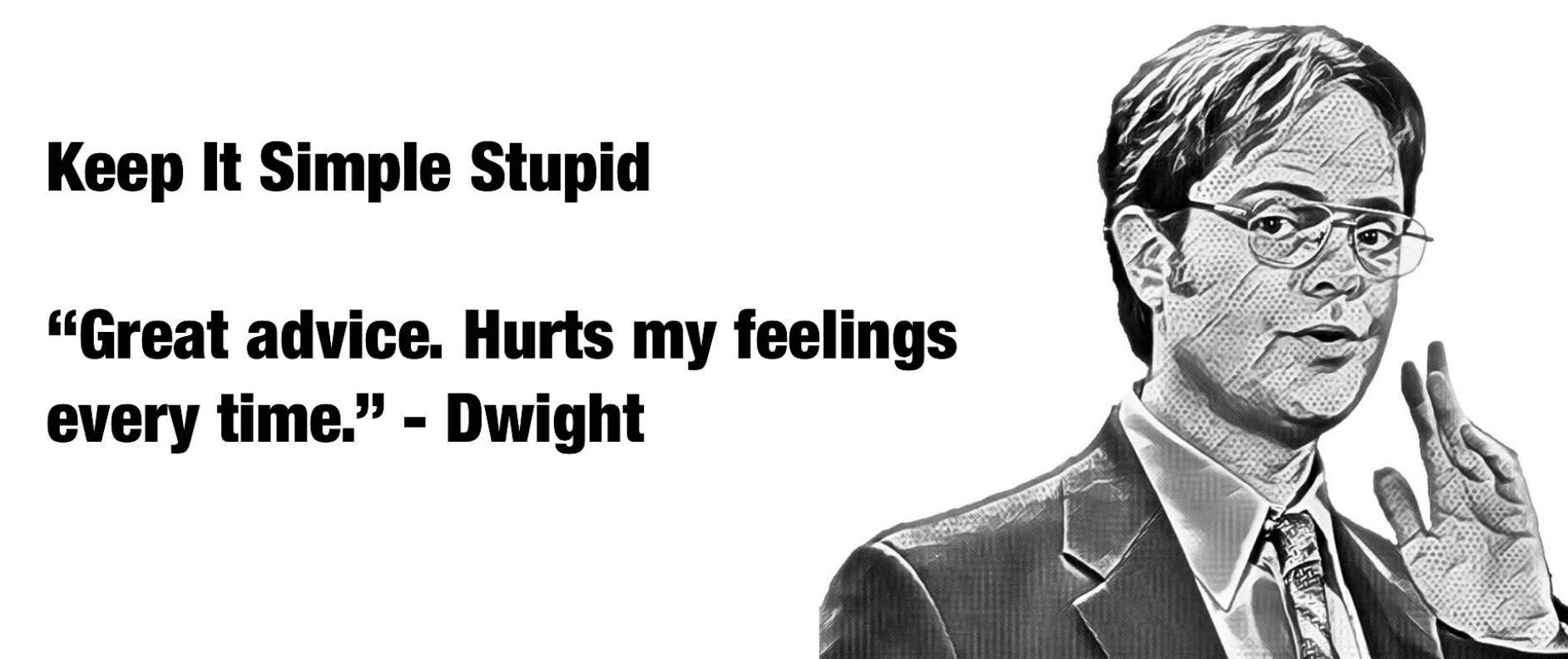


Countless possibilities

Key Points



- Spend 10x as much time exploring the data as you would like to
- Adding/augmenting data is the best way to improve performance
- KISS

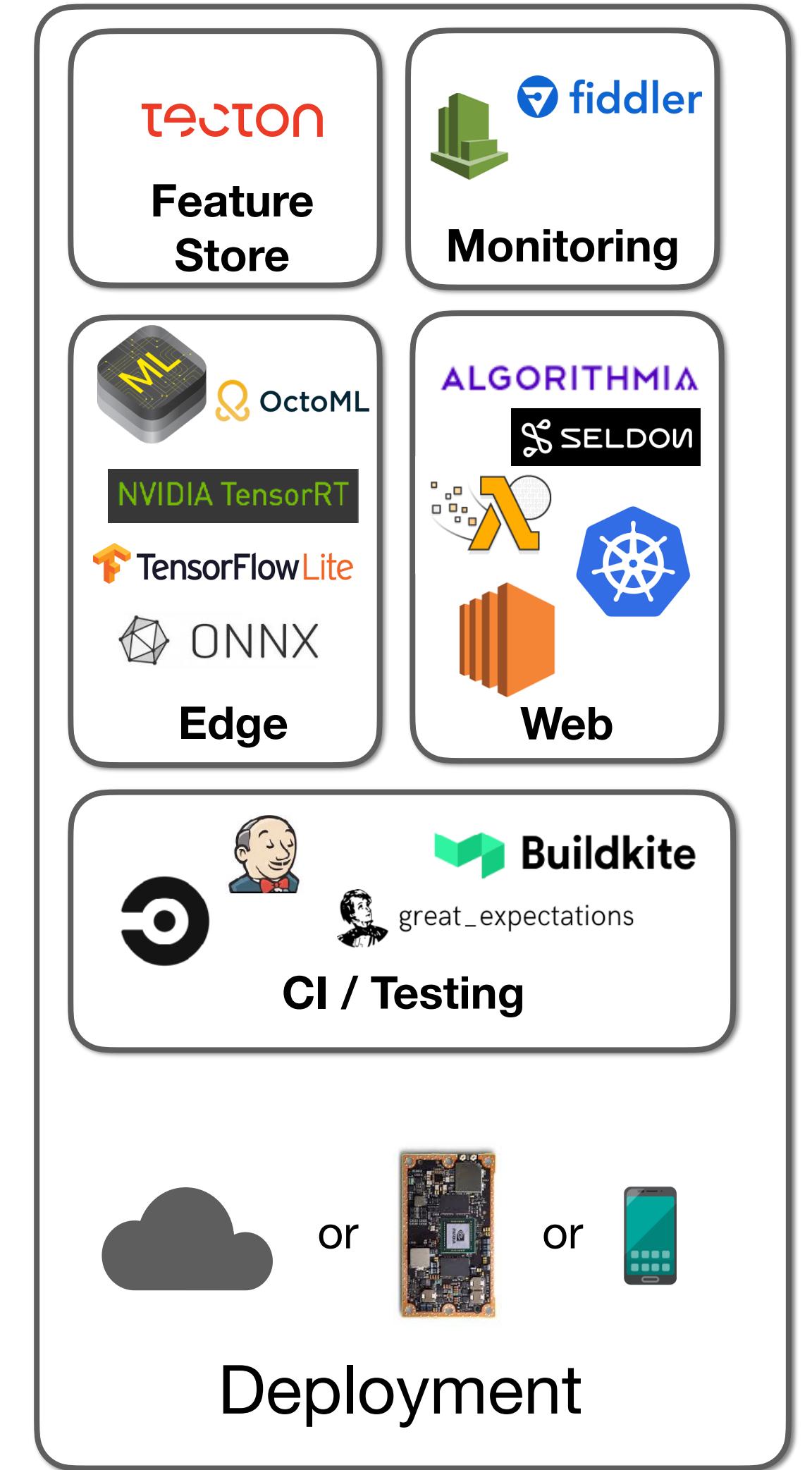
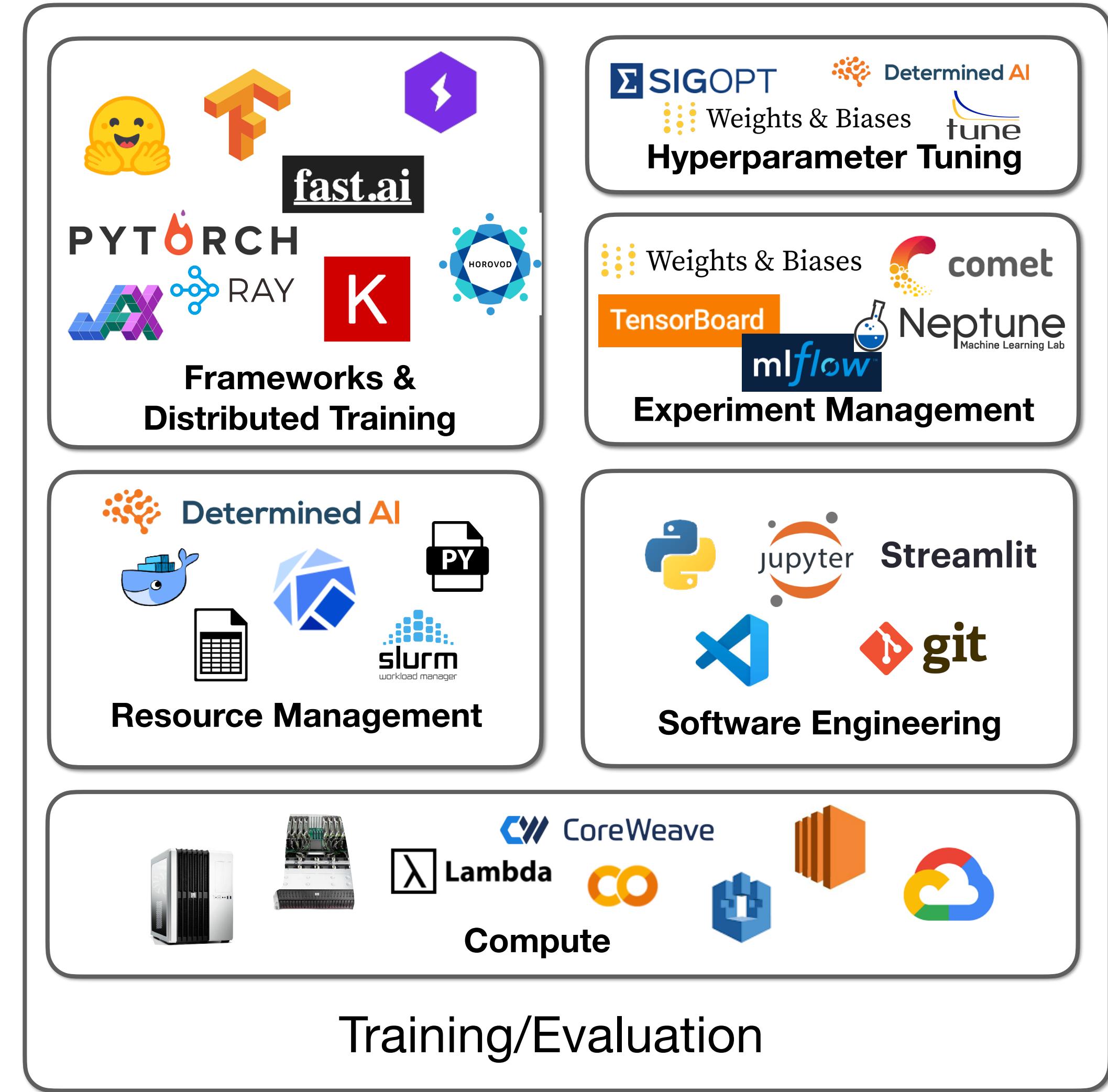


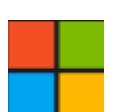


Amazon SageMaker

gradient[®]
by Paperspace

FLOYD

DOMINO
DATA LAB**“All-in-one”**



Amazon SageMaker

gradient^o
by Paperspace

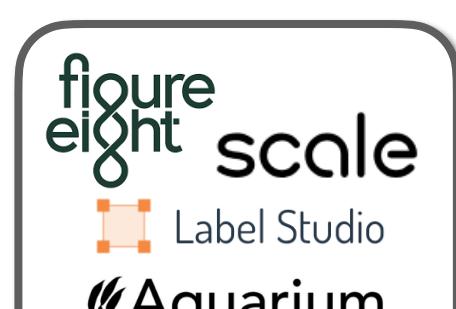
FLOYD

DOMINO
DATA LAB

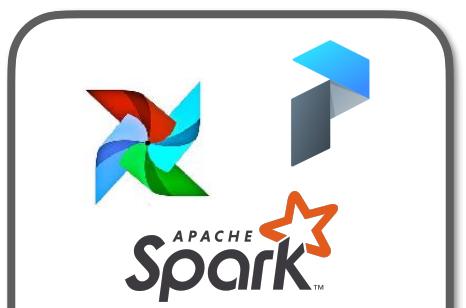
“All-in-one”



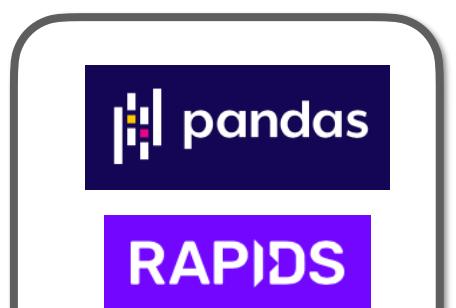
Versioning



Labeling



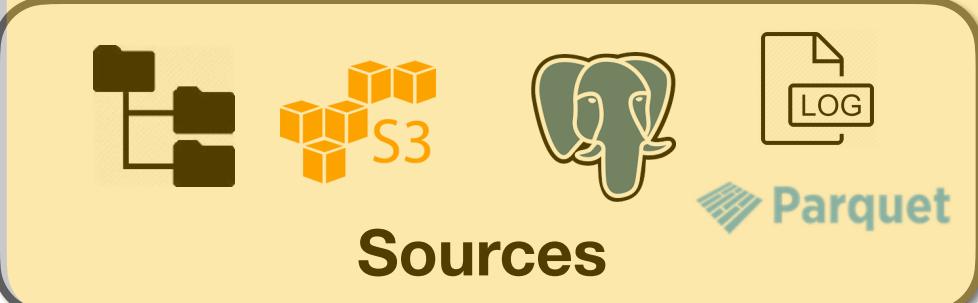
Processing



Exploration



Data Lake / Warehouse



Sources

DVC

figure-eight scale

Label Studio Aquarium

Labeling

Apache Spark

DAGSTER

dbt

pandas

RAPIDS

DASK

Exploration

snowflake

databricks

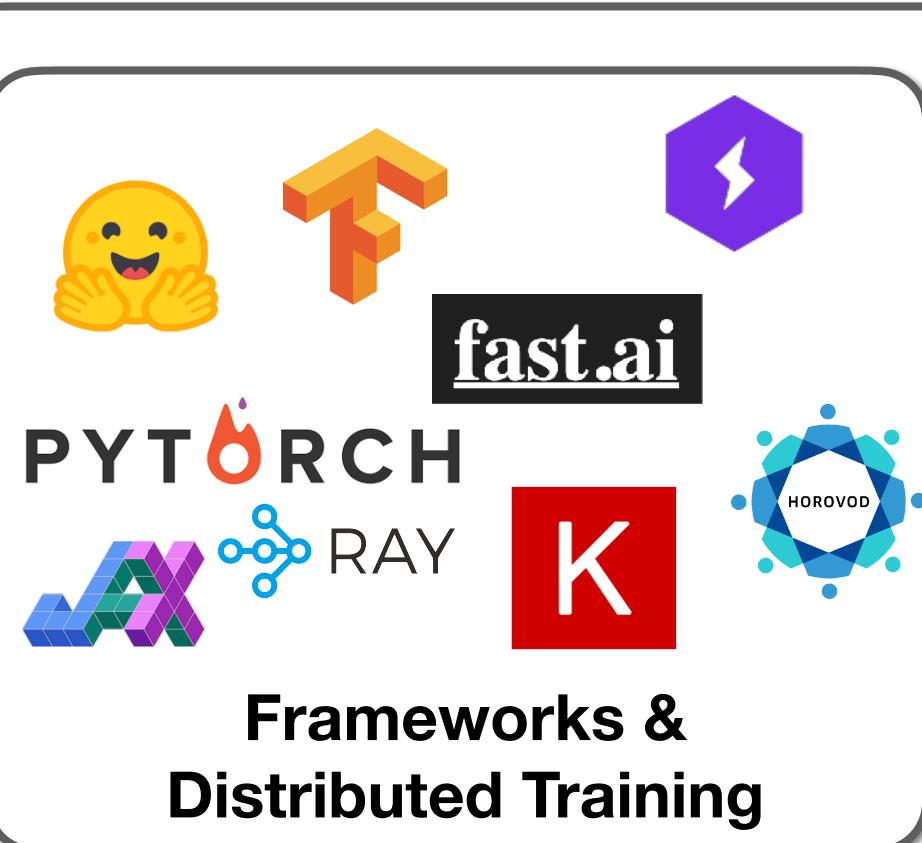
databricks

Data Lake / Warehouse

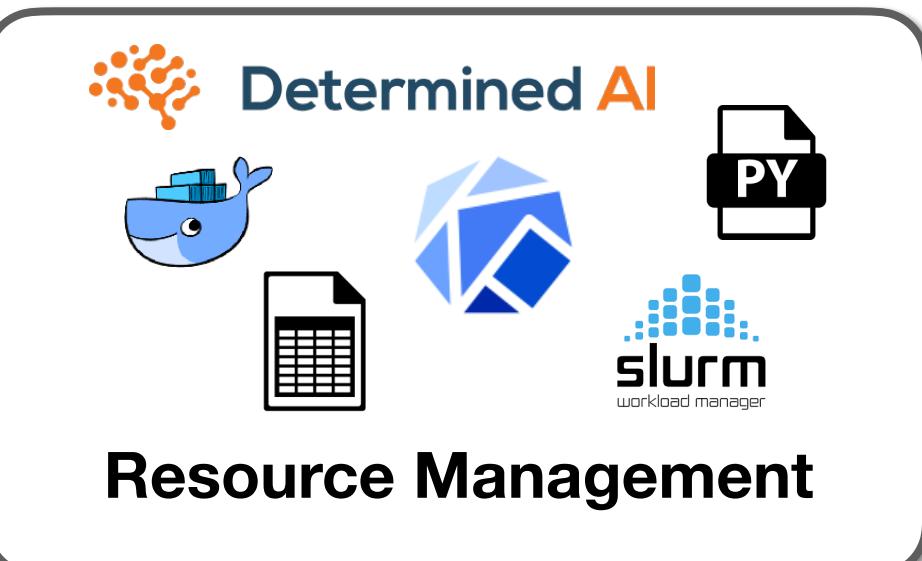
S3

Parquet

Sources



Frameworks & Distributed Training



Resource Management



Compute



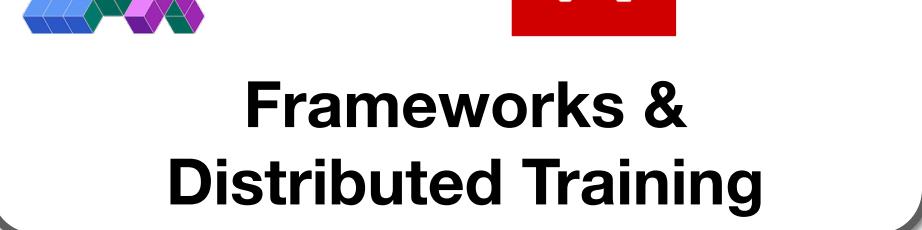
Hyperparameter Tuning



Experiment Management



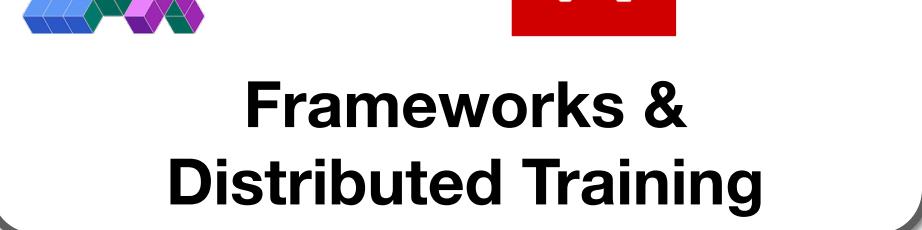
Compute



Software Engineering



Compute



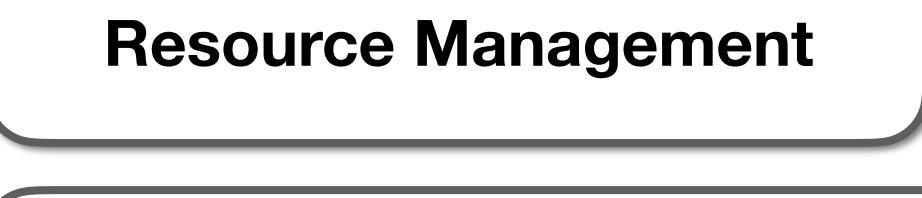
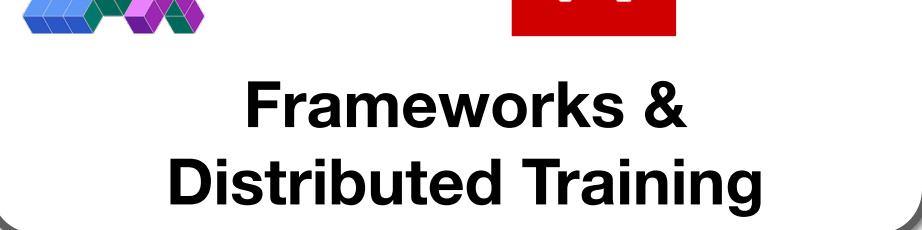
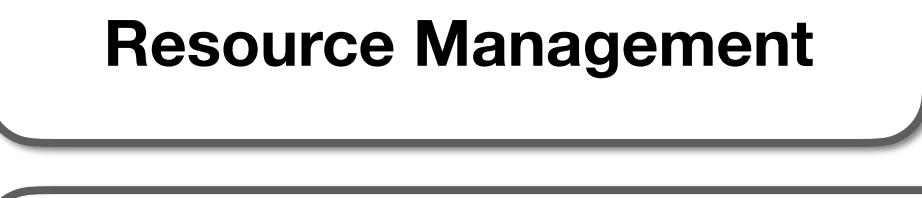
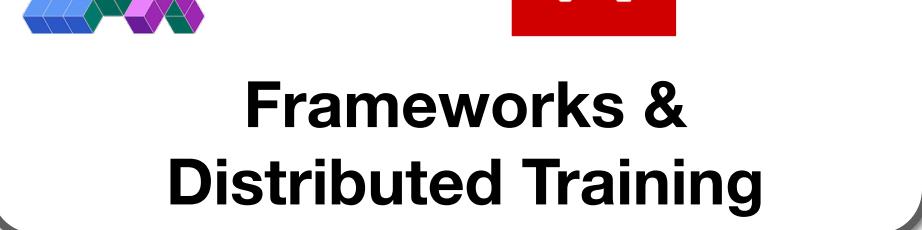
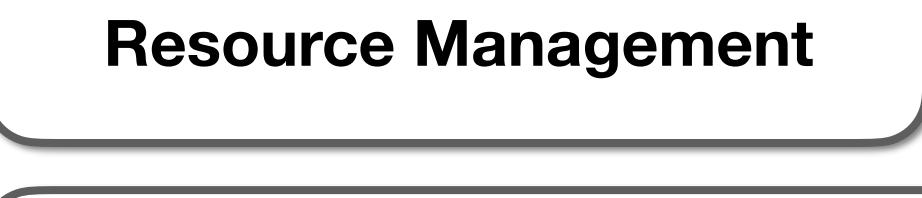
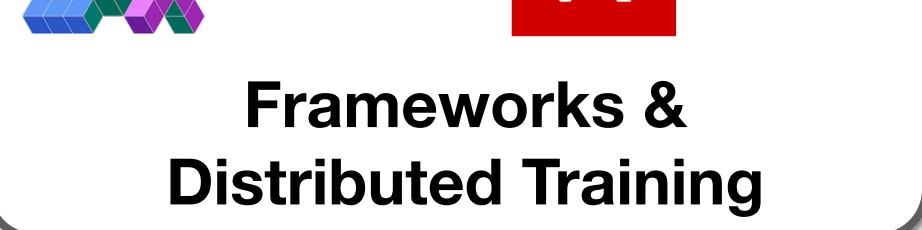
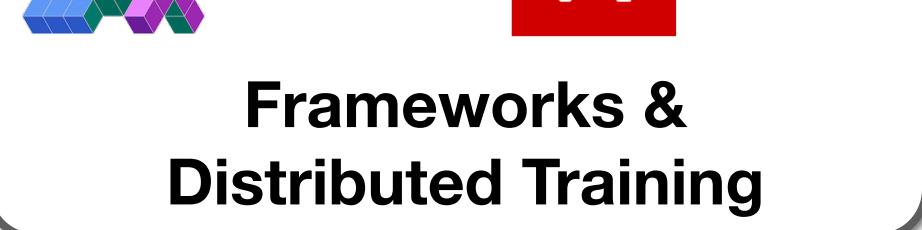
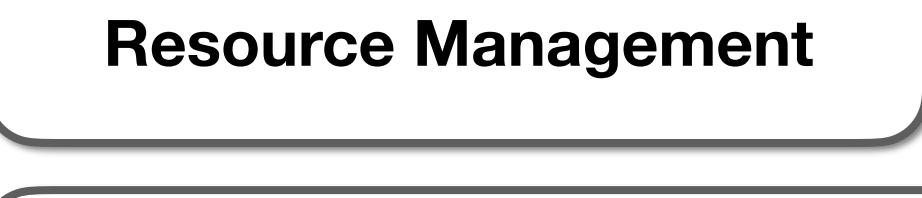
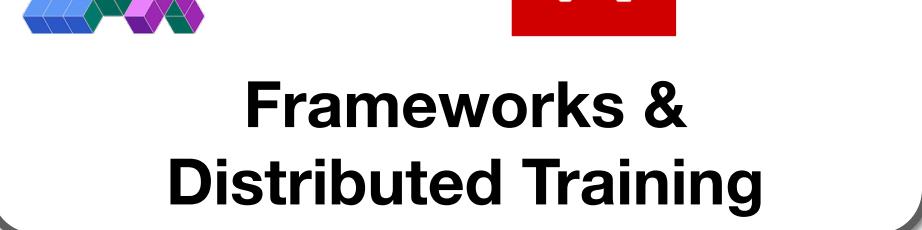
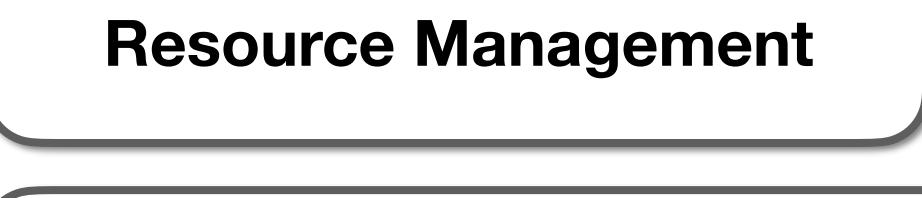
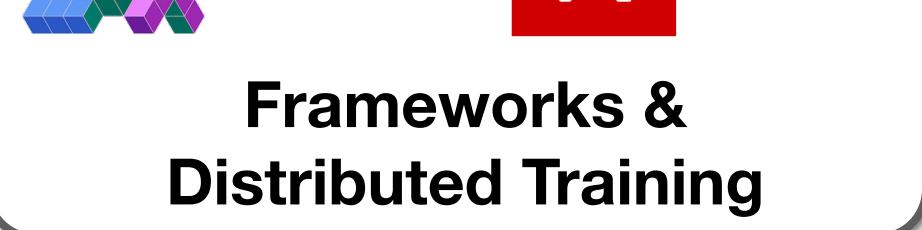
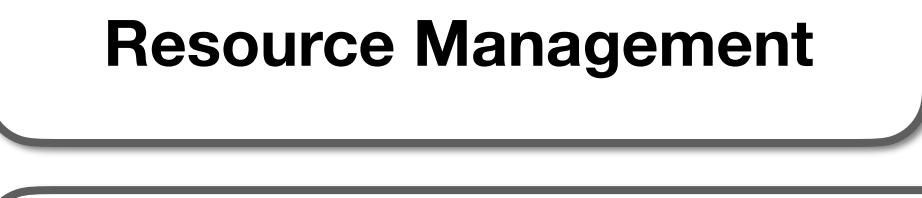
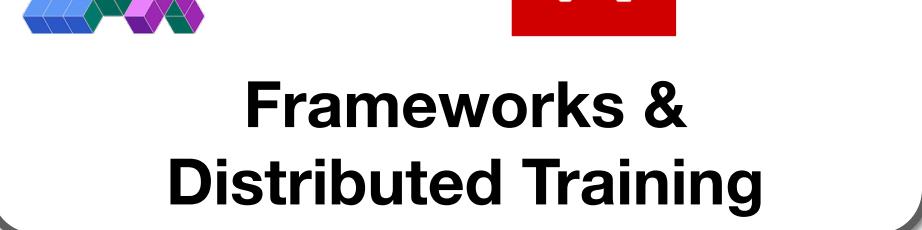
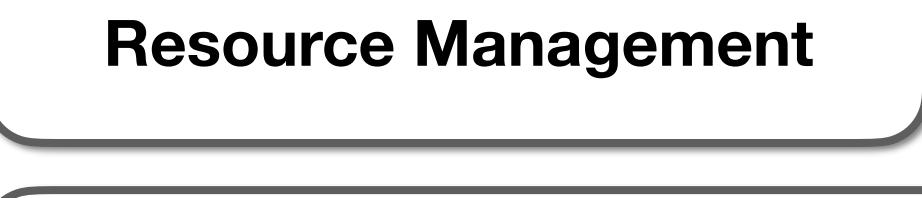
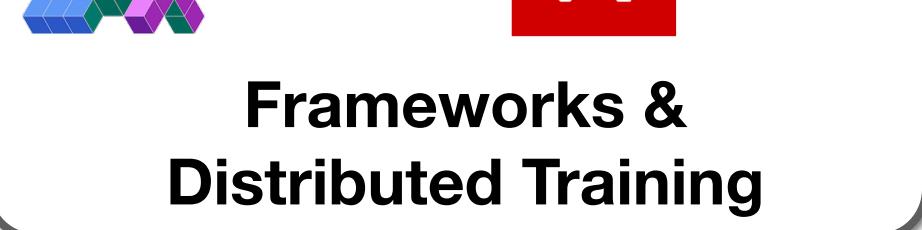
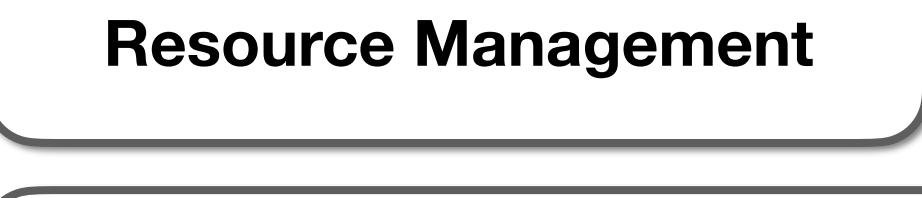
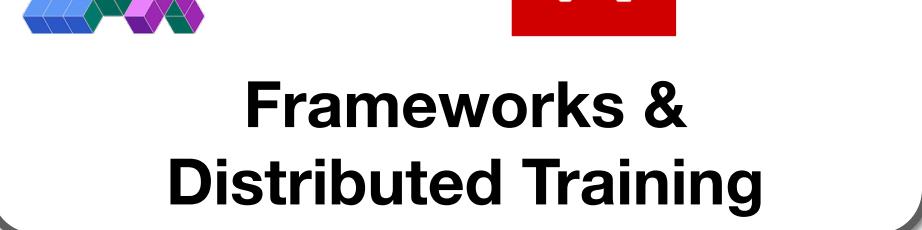
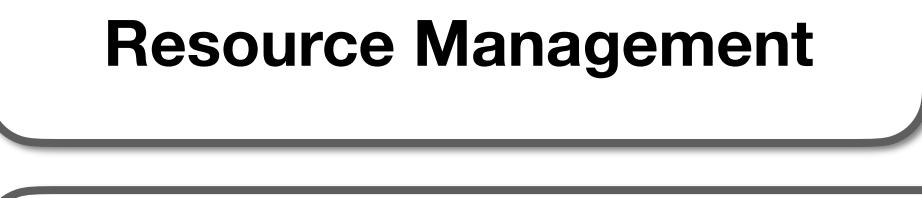
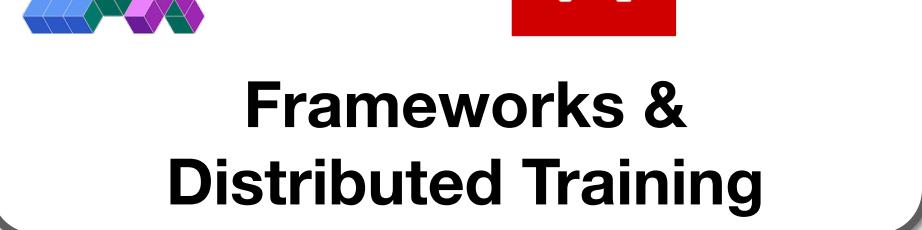
Hyperparameter Tuning

Weights & Biases

TensorBoard

CoreWeave

Compute



Data

Training/Evaluation

Deployment

Sources

- Most DL applications require lots of proprietary data
 - Exceptions: RL, GANs, GPT-3
- Publicly available datasets = No competitive advantage
 - But can serve as starting point

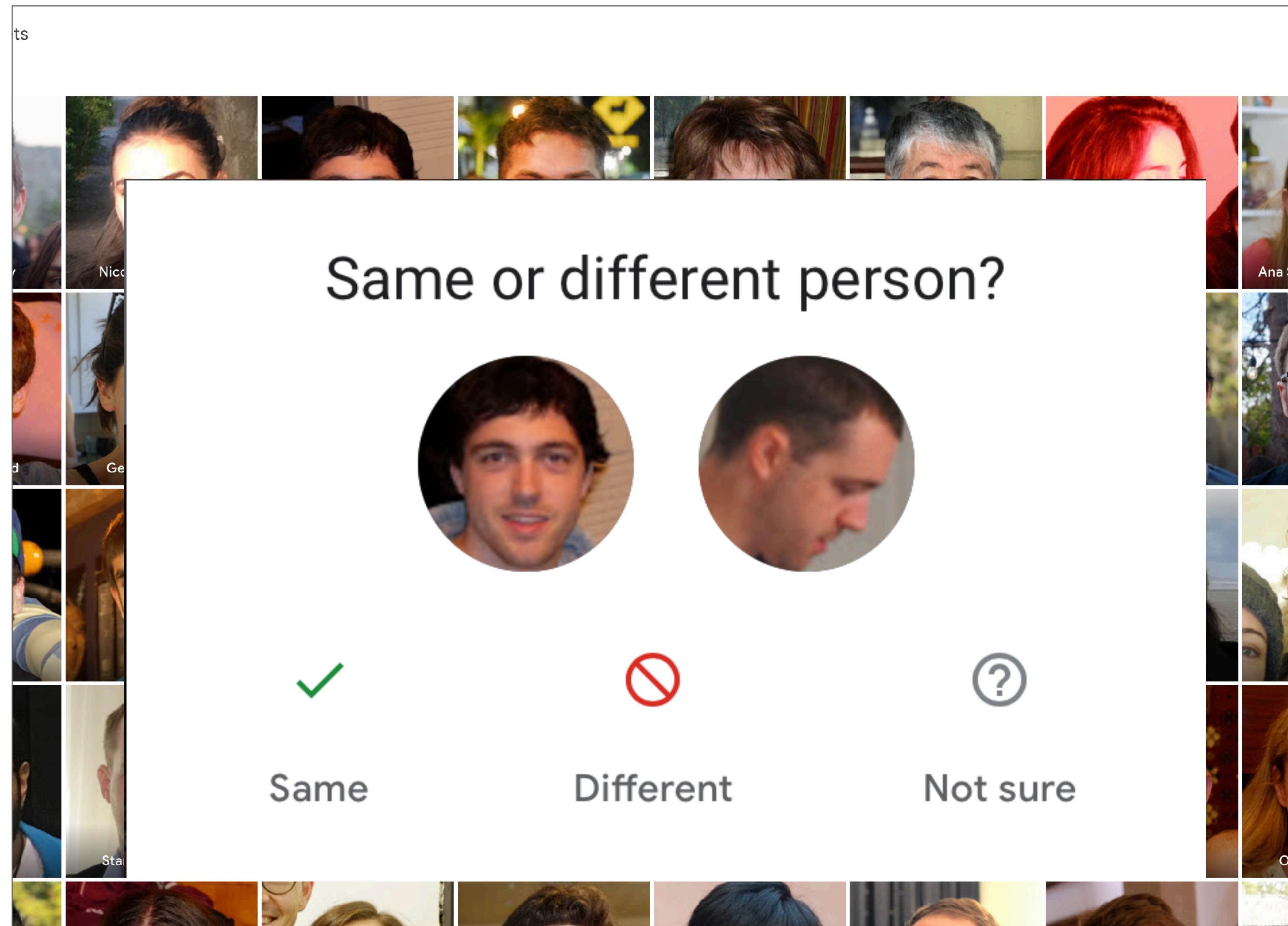
Usually: spend \$\$\$ and time to label own data



https://cdn-sv1.deepsense.ai/wp-content/uploads/2017/04/sample_image_from_the_training_set.jpg

Data flywheel

Enables rapid improvement with user labels



Semi-supervised learning

- ▶ Predict any part of the input from any other part.
- ▶ Predict the **future** from the **past**.
- ▶ Predict the **future** from the **recent past**.
- ▶ Predict the **past** from the **present**.
- ▶ Predict the **top** from the **bottom**.
- ▶ Predict the **occluded** from the **visible**
- ▶ **Pretend there is a part of the input you don't know and predict that.**

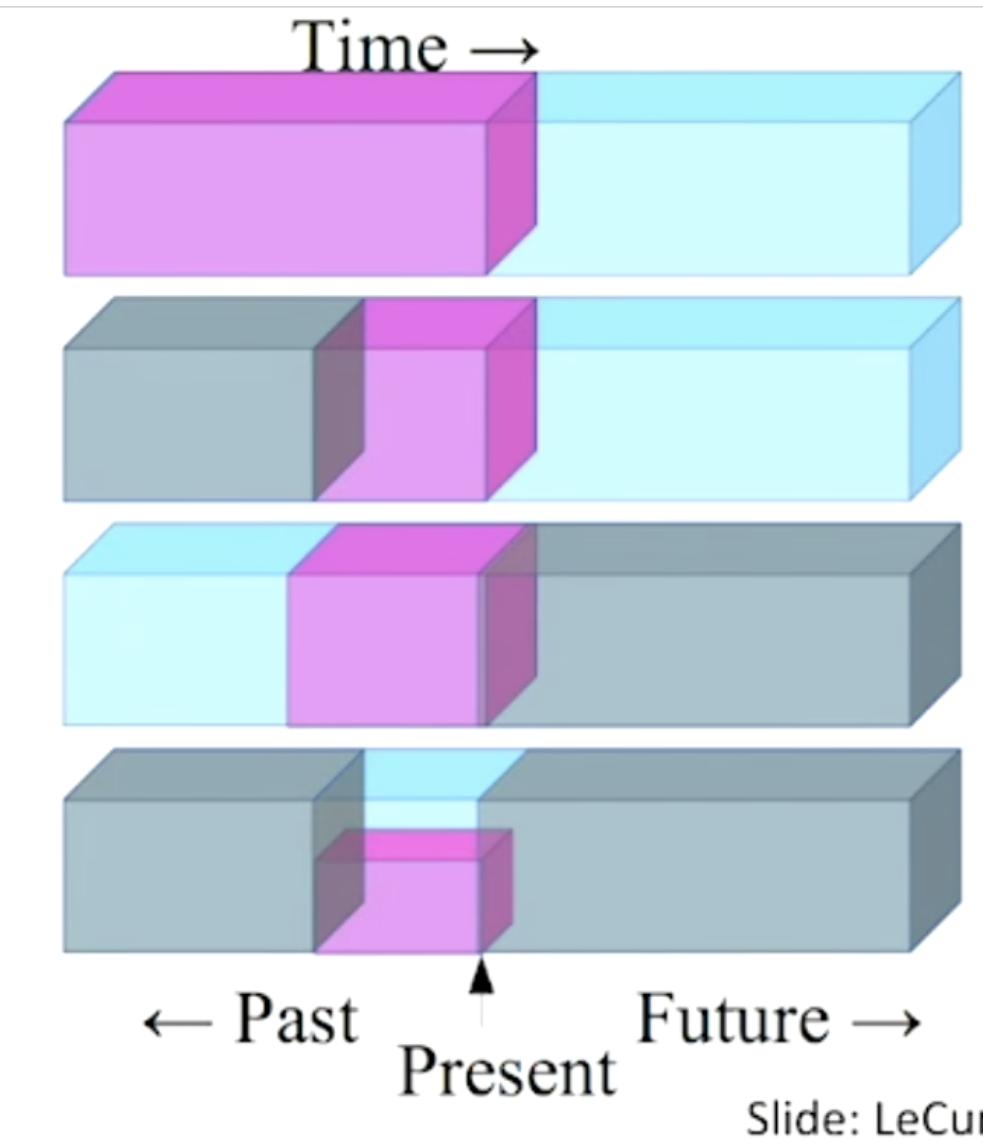
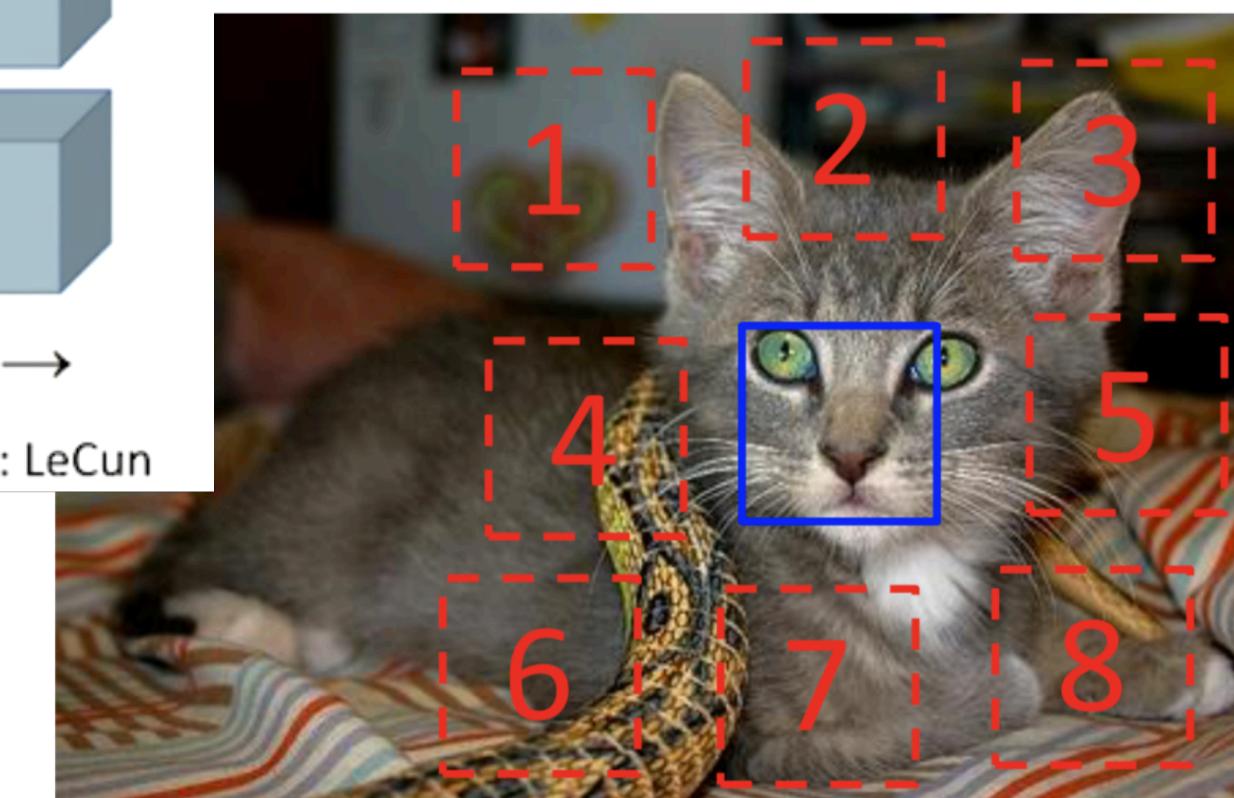


Fig. 1. A great summary of how self-supervised learning tasks can be constructed (Image source: LeCun's talk)

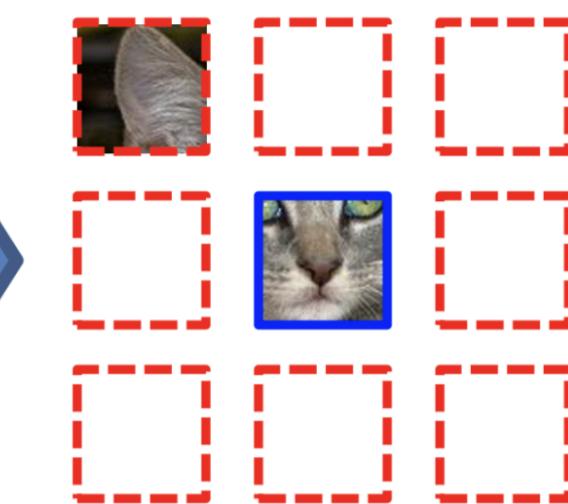
Use parts of data to label other parts

Very important idea!



$$X = (\text{cat eye}, \text{ear}); Y = 3$$

Example:



Question 1:



Question 2:



Fig. 4. Illustration of self-supervised learning by predicting the relative position of two random patches. (Image source: Doersch et al., 2015)

<https://ai.facebook.com/blog/self-supervised-learning-the-dark-matter-of-intelligence>

<https://lilianweng.github.io/lil-log/2019/11/10/self-supervised-learning.html>

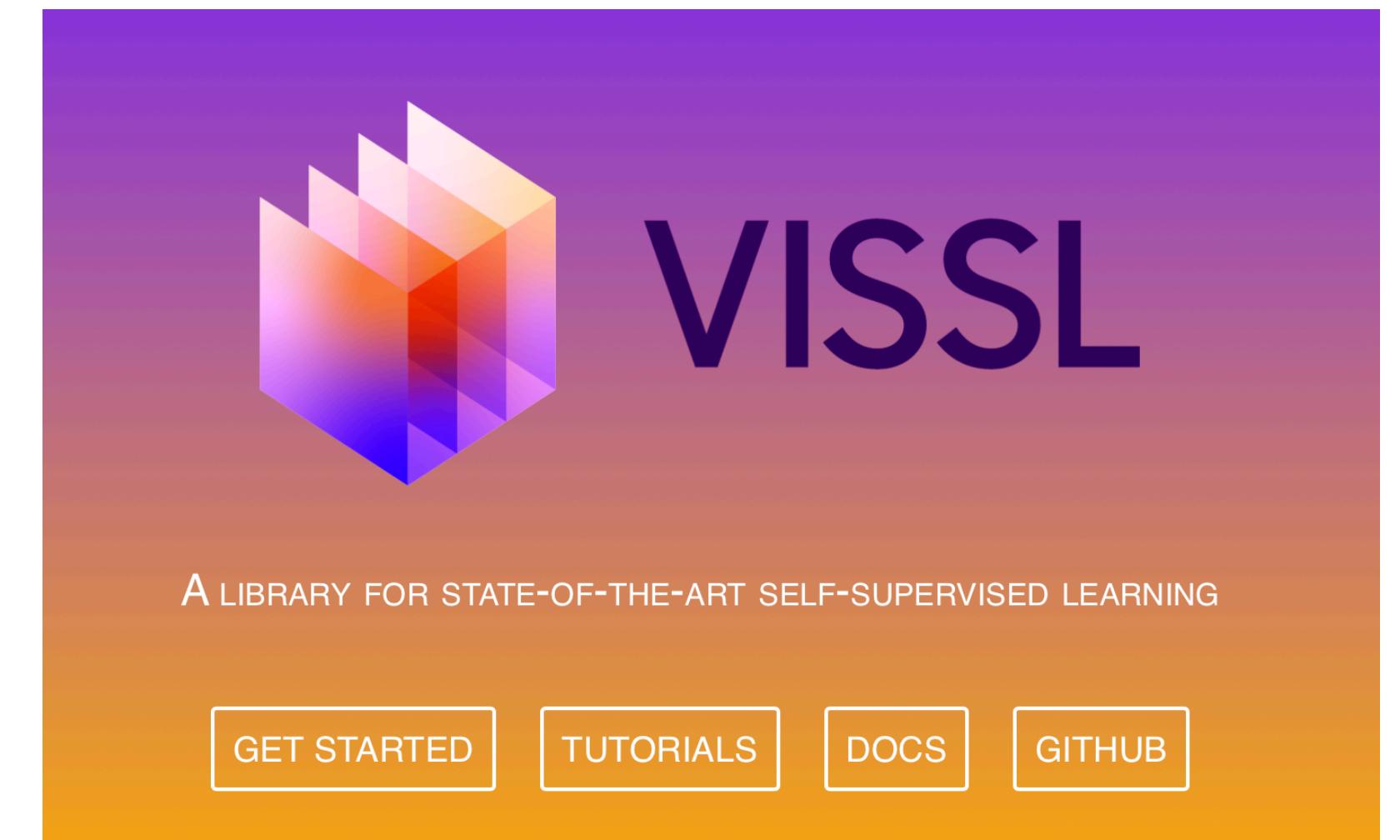
Semi-supervised learning

RESEARCH | COMPUTER VISION

SEER: The start of a more powerful, flexible,
and accessible era for computer vision

March 4, 2021

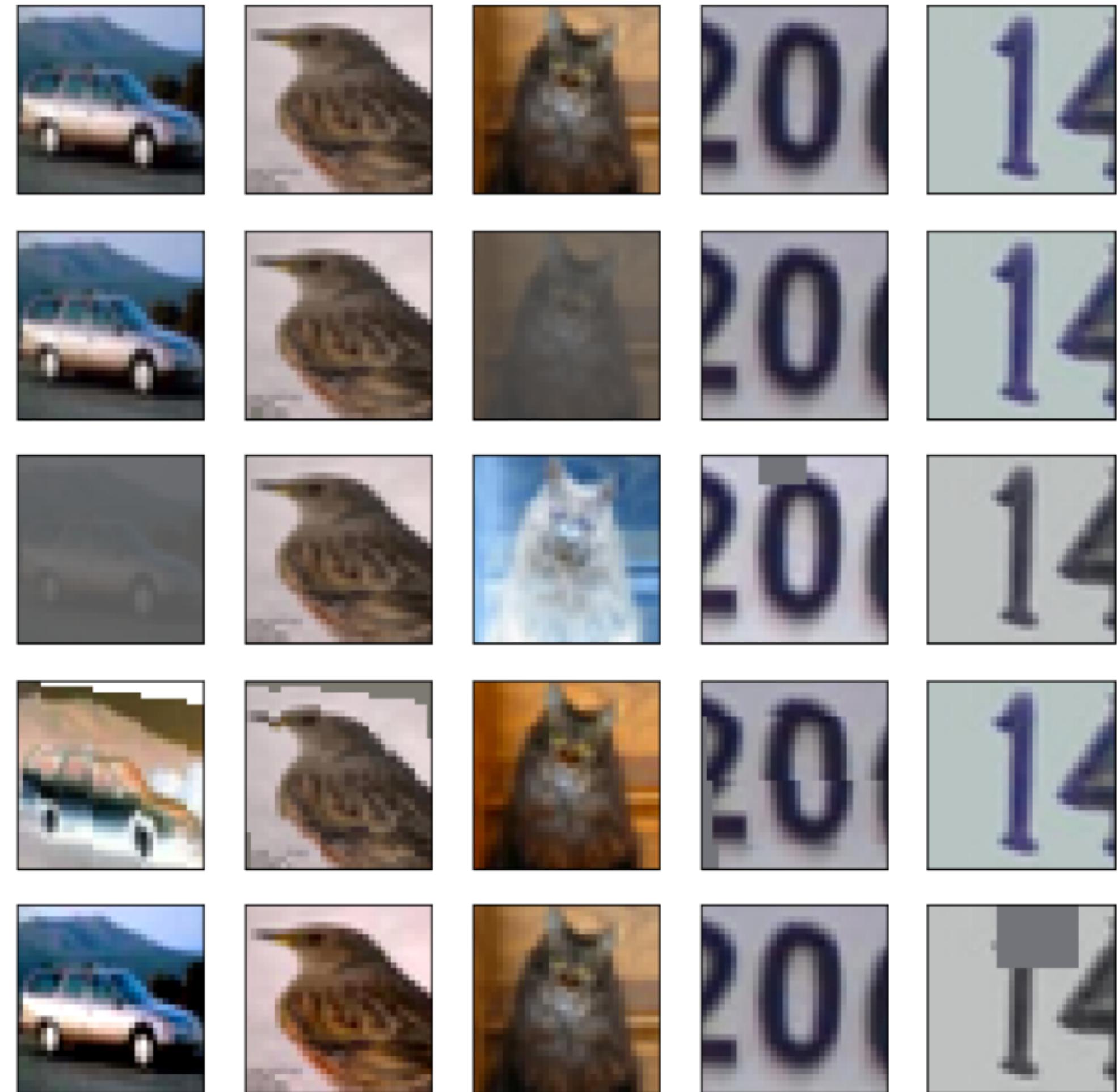
- Trained on 1B random images
- Achieved SOTA accuracy on ImageNet top-1 prediction
- Open-source library



<https://ai.facebook.com/blog/seer-the-start-of-a-more-powerful-flexible-and-accessible-era-for-computer-vision>

Image data augmentation

- **Must do** for training vision models
- Frameworks (e.g. torchvision) provide functions that do this
- Done in parallel to GPU training on the CPU

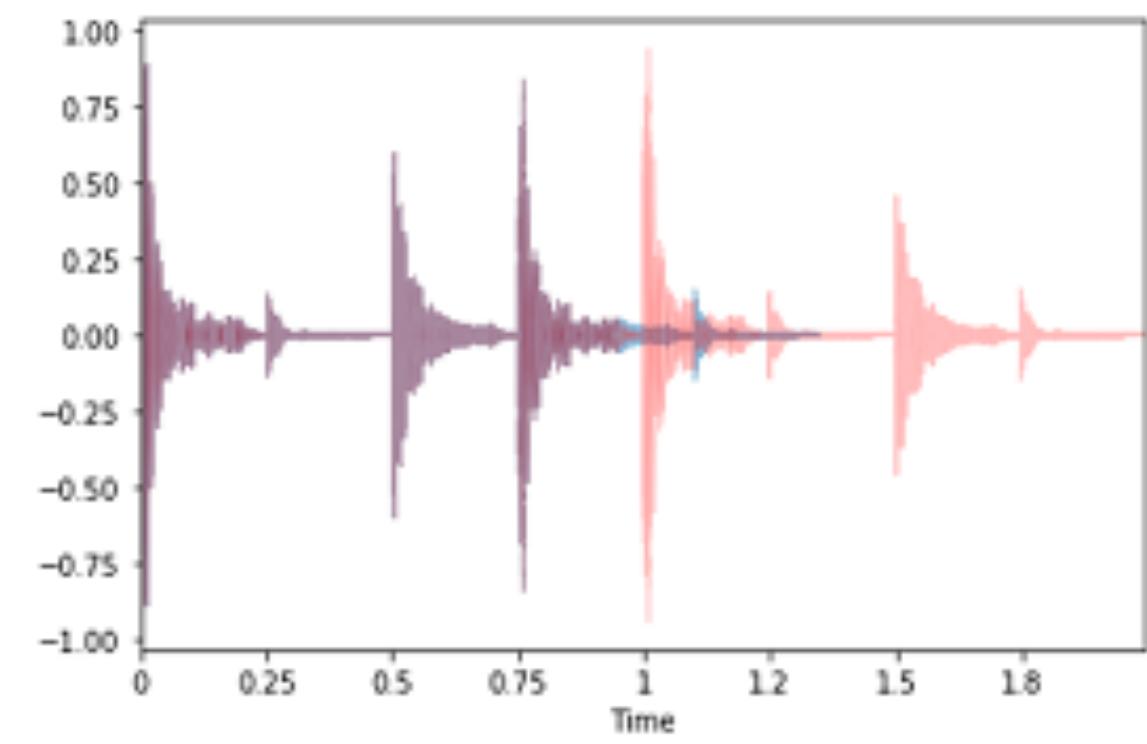


<https://towardsdatascience.com/1000x-faster-data-augmentation-b91baf8ee896c>

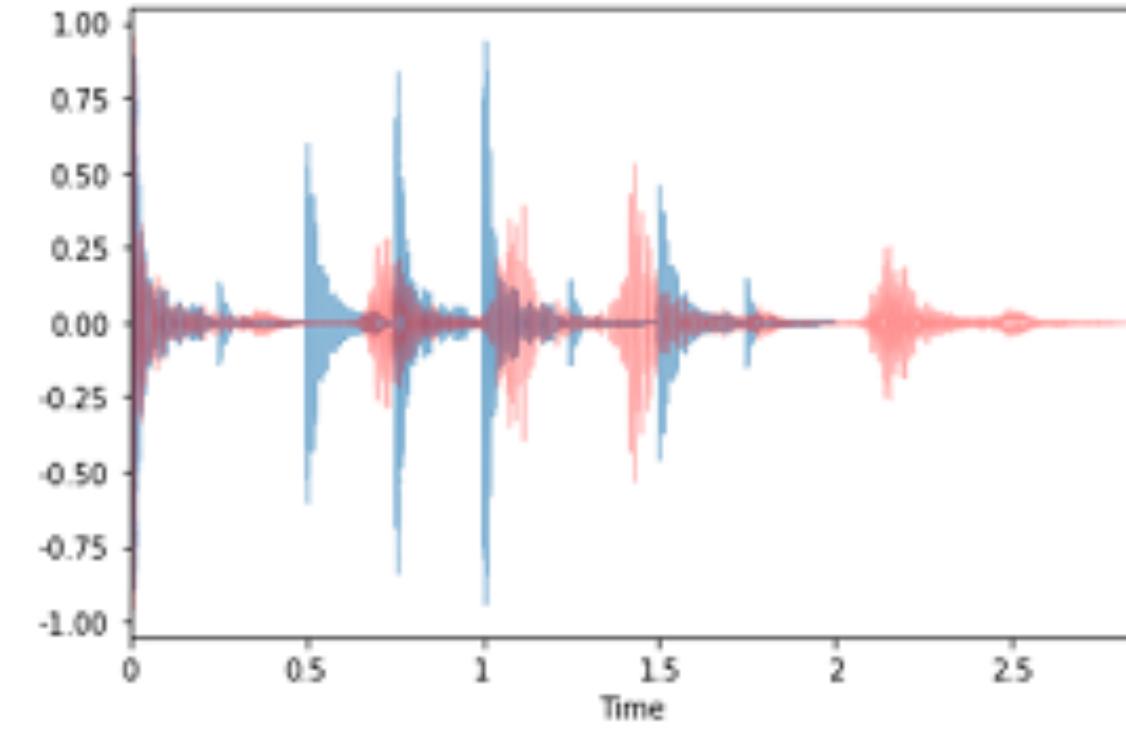
Other data augmentation

- Tabular
 - Delete some cells to simulate missing data
- Text
 - No well established techniques, but replace words with synonyms, change order of things.
- Speech/video
 - Change speed, inserts pauses, etc

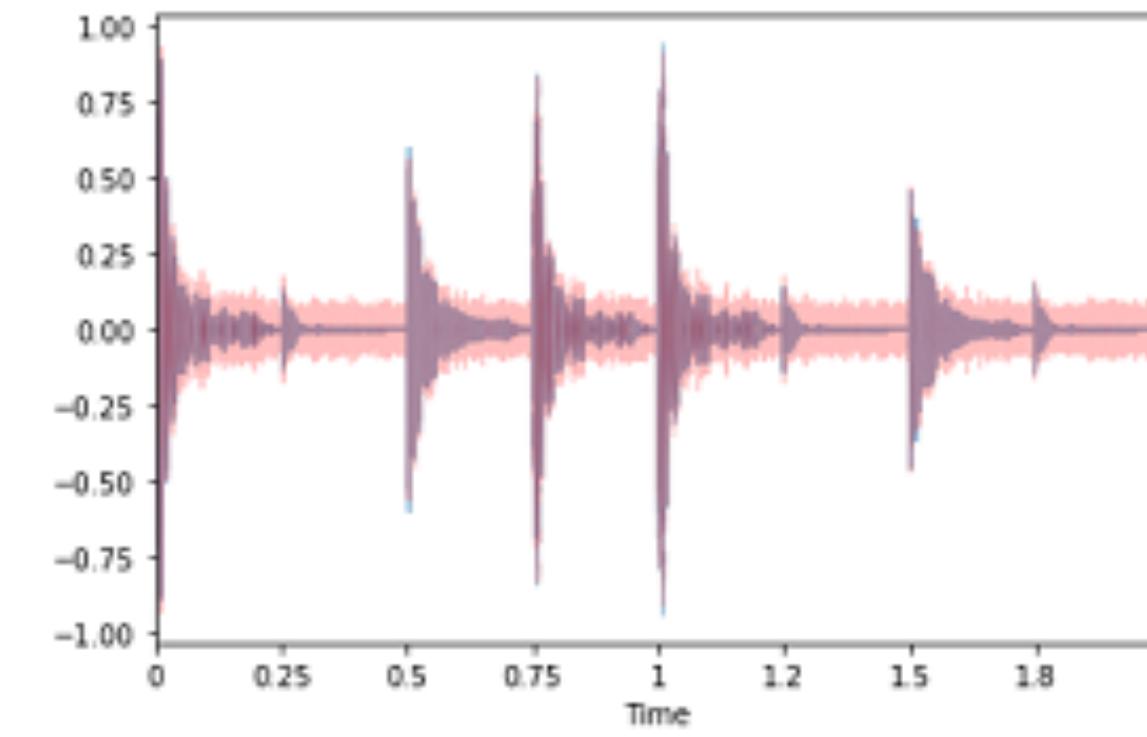
Cropping out a portion



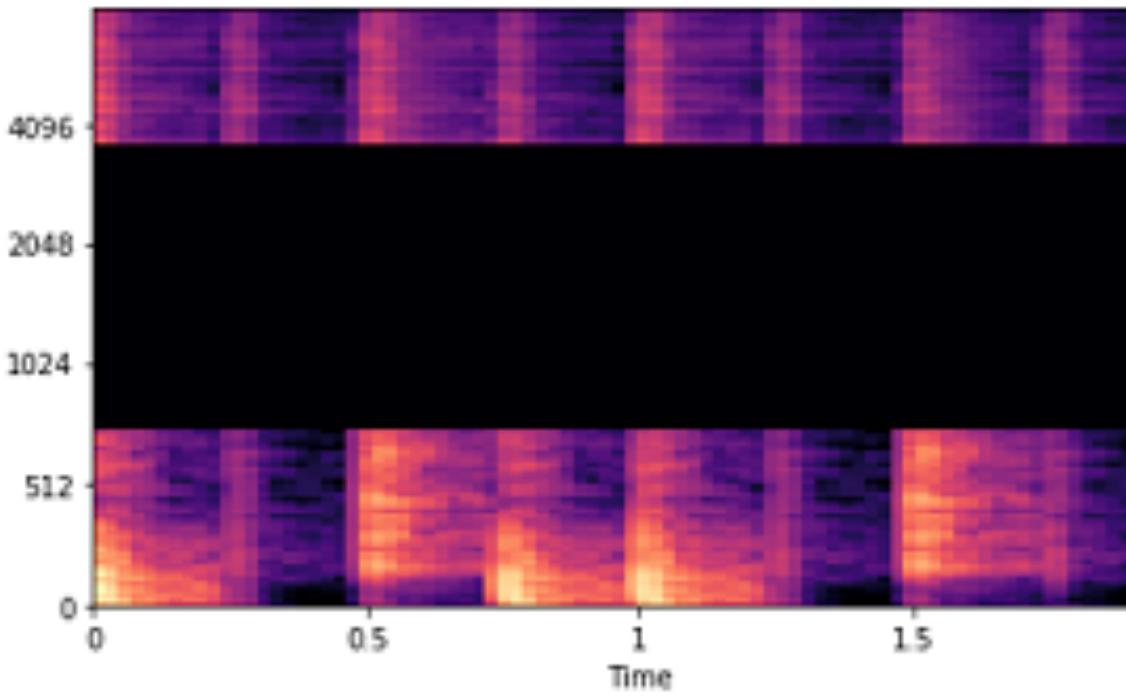
Changing Speed



Injecting Noise



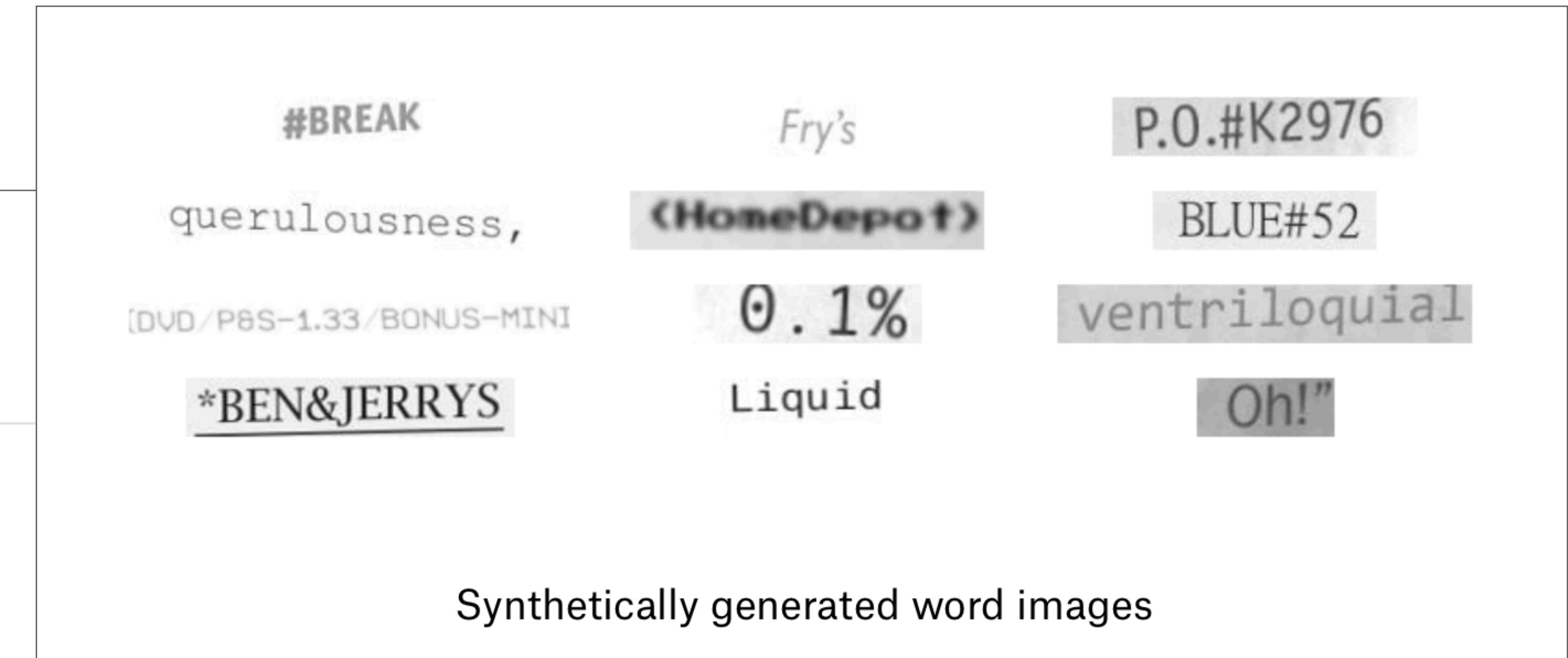
Masking Frequency



<https://github.com/makcedward/nlpaug>

Synthetic data

Underrated idea that is often worth starting with



Synthetically generated word images

Creating a Modern OCR Pipeline Using Computer Vision and Deep Learning

Brad Neuberg | April 12, 2017

728 0

<https://blogs.dropbox.com/tech/2017/04/creating-a-modern-ocr-pipeline-using-computer-vision-and-deep-learning/>

This can get pretty deep!



- Receipt crinkliness
- Receipt curvature
- Receipt alignment with camera
- Receipt ink fadedness
- Receipt paper glossiness
- Table material
- Camera flash
- Camera location
- Camera direction
- Camera exposure
- Camera focal distance
- Camera aperature size
- Environment ambient brightness
- Primary light location



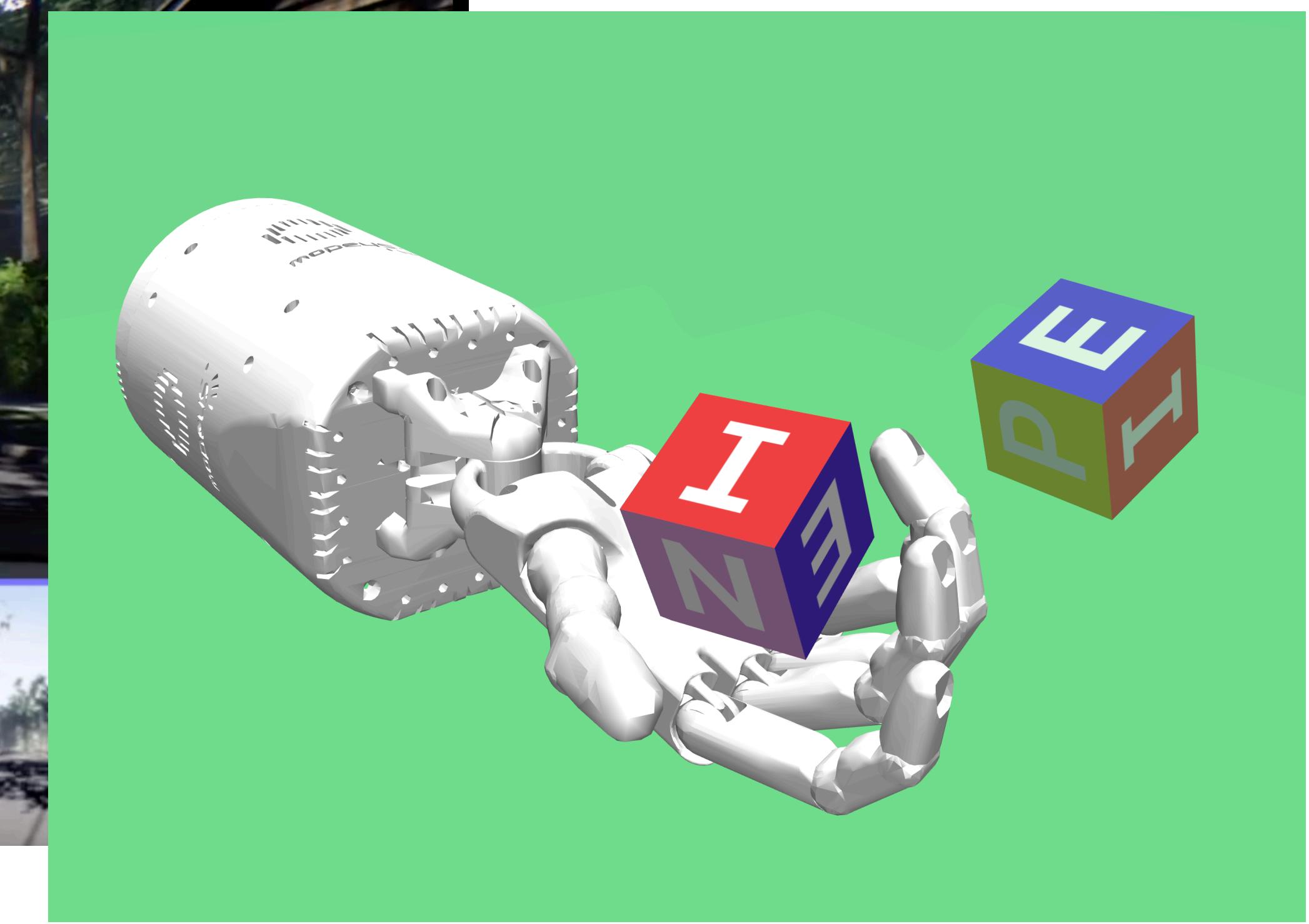
Andrew Moffat - <https://github.com/amoffat/metabrite-receipt-tests>

Especially for driving and robotics



<https://microsoft.github.io/AirSim/>

<https://openai.com/blog/ingredients-for-robotics-research/>



Questions?

Data Storage

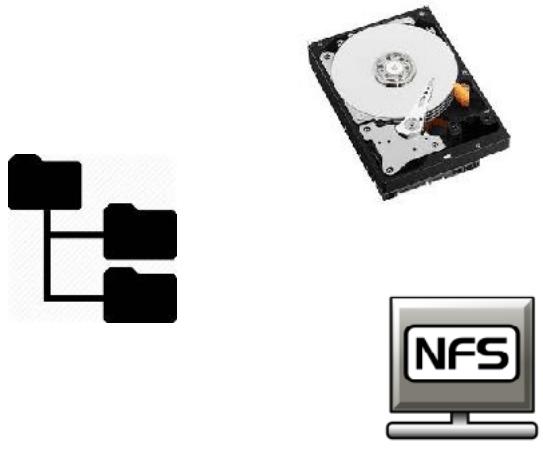
1. Building blocks

- Filesystem
- Object Storage
- Database
- Data Lake / Data Warehouse

2. What goes where

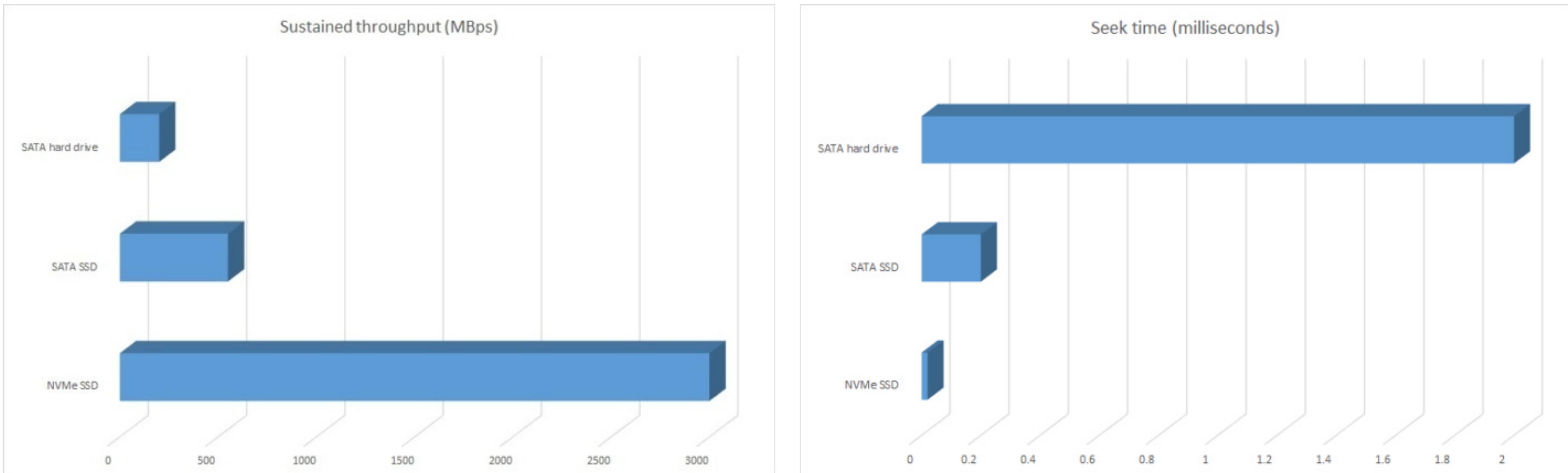
3. Where to learn more

Filesystem



- Foundational layer of storage.
- Fundamental unit is a "file", which can be text or binary, is not versioned, and is easily overwritten.
- Can be as simple as a locally mounted disk containing all the files you need.
- Can be networked (e.g. NFS): accessible over network by multiple machines.
- Can be distributed (e.g. HDFS): stored and accessed over multiple machines
- Fastest option

Hard Drive Speeds



<https://www.pcworld.com/article/2899351/everything-you-need-to-know-about-nvme.html>

Local Data Format

- Binary data: just files
 - TFRecord batches files -- doesn't seem necessary with NVMe drives
- For large tabular / text data, have choices:
 - HDF5 is powerful, but bloated and declining
 - Parquet is widespread and recommended
 - Feather is powered by Apache Arrow, up-and-coming
- Try to use native Tensorflow and PyTorch dataset classes

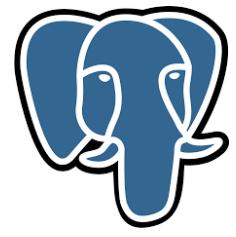
Object Storage



- An **API** over the filesystem. GET, PUT, DELETE files to a service, without worrying where they are actually stored.
- Fundamental unit is an "object". Usually binary: image, sound file, etc.
- Versioning, redundancy can be built into the service.
- Not as fast as local, but fast enough within the cloud

	S3	HDFS	S3 vs HDFS
Elasticity	Yes	No	S3 is more elastic
Cost/TB/month	\$23	\$206	10X
Availability	99.99%	99.9% (estimated)	10X
Durability	99.999999999%	99.9999% (estimated)	10X+
Transactional writes	Yes with DBIO	Yes	Comparable

Database

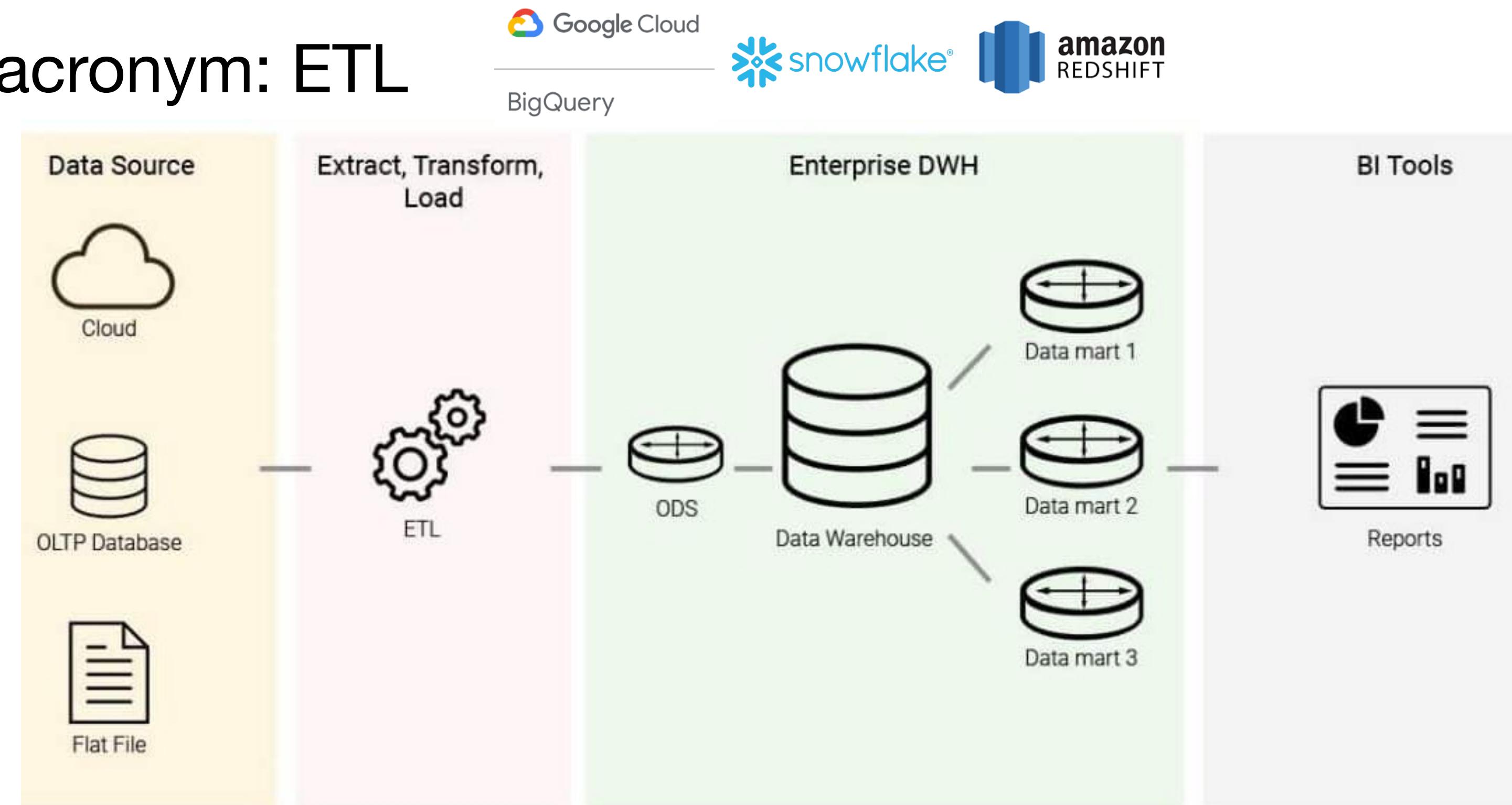


- Persistent, fast, scalable storage and retrieval of structured data that will be accessed repeatedly.
- AKA Online **Transaction** Processing (OLTP)
- Mental model: everything is actually in RAM, but software ensures that everything is logged to disk and never lost.
- Not for binary data! Store **references** instead.
- Postgres is the right choice most of the time. Supports unstructured JSON.
 - SQLite is perfectly good for small projects.
- "NoSQL" was a big craze in 2010's. Mostly avoid.
 - **Redis** is very useful when you need a simple **key-value store**.

Data Warehouse

- Structured aggregation of data for analysis
- AKA Online Analytical Processing (OLAP)

- Another acronym: ETL



<https://addepto.com/implement-data-warehouse-business-intelligence/>

SQL and DataFrames

- Most data solutions use SQL. Some, like Databricks, use **DataFrames**.
- SQL is the standard interface for structured data.
- Pandas is the main DataFrame in the Python ecosystem.
- Our advice: become fluent in both

In SQL, selection is done using a comma-separated list of columns you'd like to select (or a `*` to select all columns):

```
SELECT total_bill, tip, smoker, time  
FROM tips  
LIMIT 5;
```

With pandas, column selection is done by passing a list of column names to your DataFrame:

```
In [6]: tips[["total_bill", "tip", "smoker", "time"]].head(5)  
Out[6]:  
   total_bill  tip  smoker    time  
0      16.99  1.01     No  Dinner  
1      10.34  1.66     No  Dinner  
2      21.01  3.50     No  Dinner  
3      23.68  3.31     No  Dinner  
4      24.59  3.61     No  Dinner
```

Grouping by more than one column is done by passing a list of columns to the `groupby()` method.

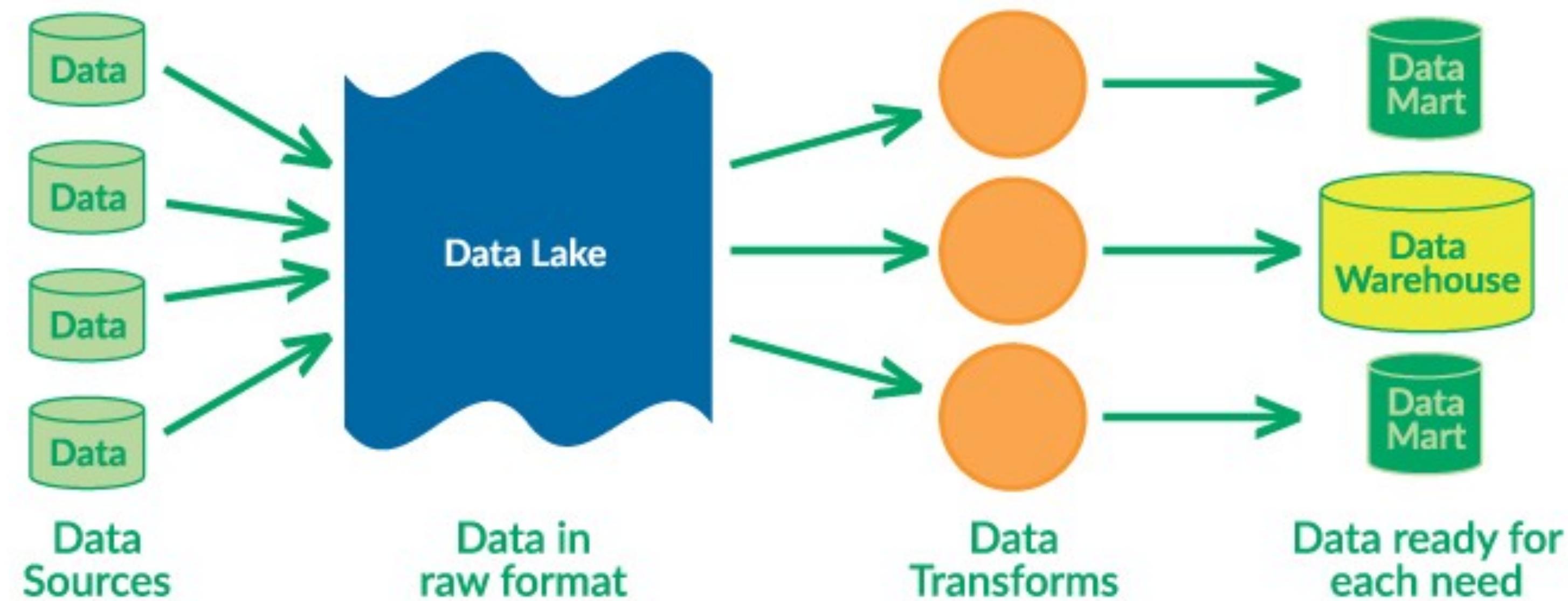
```
SELECT smoker, day, COUNT(*), AVG(tip)  
FROM tips  
GROUP BY smoker, day;  
/*  
smoker day  
No     Fri      4  2.812500  
      Sat      45 3.102889  
      Sun      57 3.167895  
      Thur     45 2.673778  
Yes    Fri      15 2.714000  
      Sat      42 2.875476  
      Sun      19 3.516842  
      Thur     17 3.030000  
*/
```

```
In [22]: tips.groupby(["smoker", "day"]).agg({"tip": [np.size, np.mean]})  
Out[22]:  
           tip  
           size      mean  
smoker day  
No     Fri    4.0  2.812500  
      Sat   45.0 3.102889  
      Sun   57.0 3.167895  
      Thur  45.0 2.673778  
Yes    Fri   15.0 2.714000  
      Sat   42.0 2.875476  
      Sun   19.0 3.516842  
      Thur  17.0 3.030000
```

https://pandas.pydata.org/docs/getting_started/comparison/comparison_with_sql.html

Data Lake

- Unstructured aggregation of data from multiple sources, e.g. databases, logs, expensive data transformations.
- ELT: dump everything in, then transform for specific needs later.

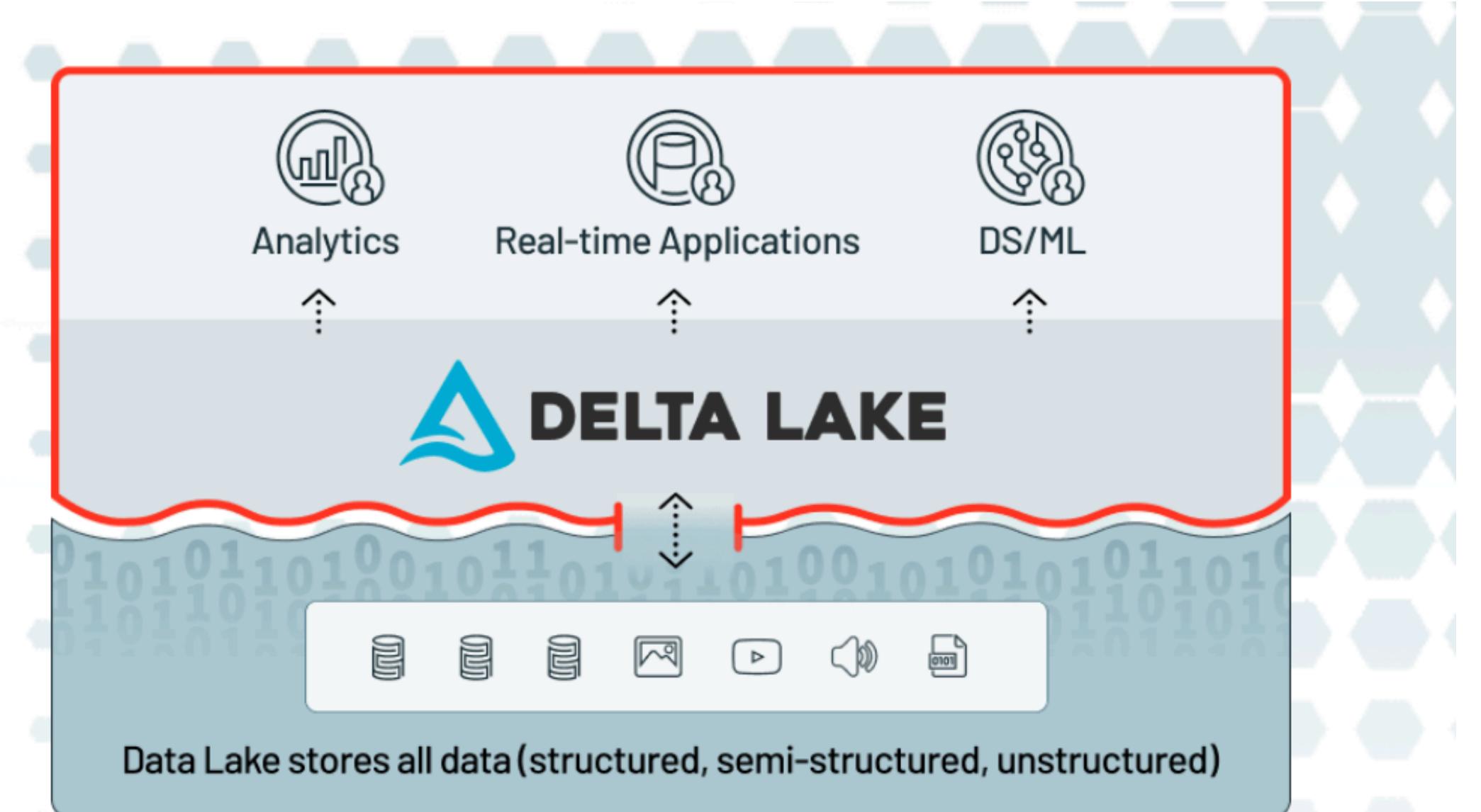


<https://medium.com/data-ops/throw-your-data-in-a-lake-32cd21b6de02>

Trend: Lake House

The Databricks Lakehouse Platform

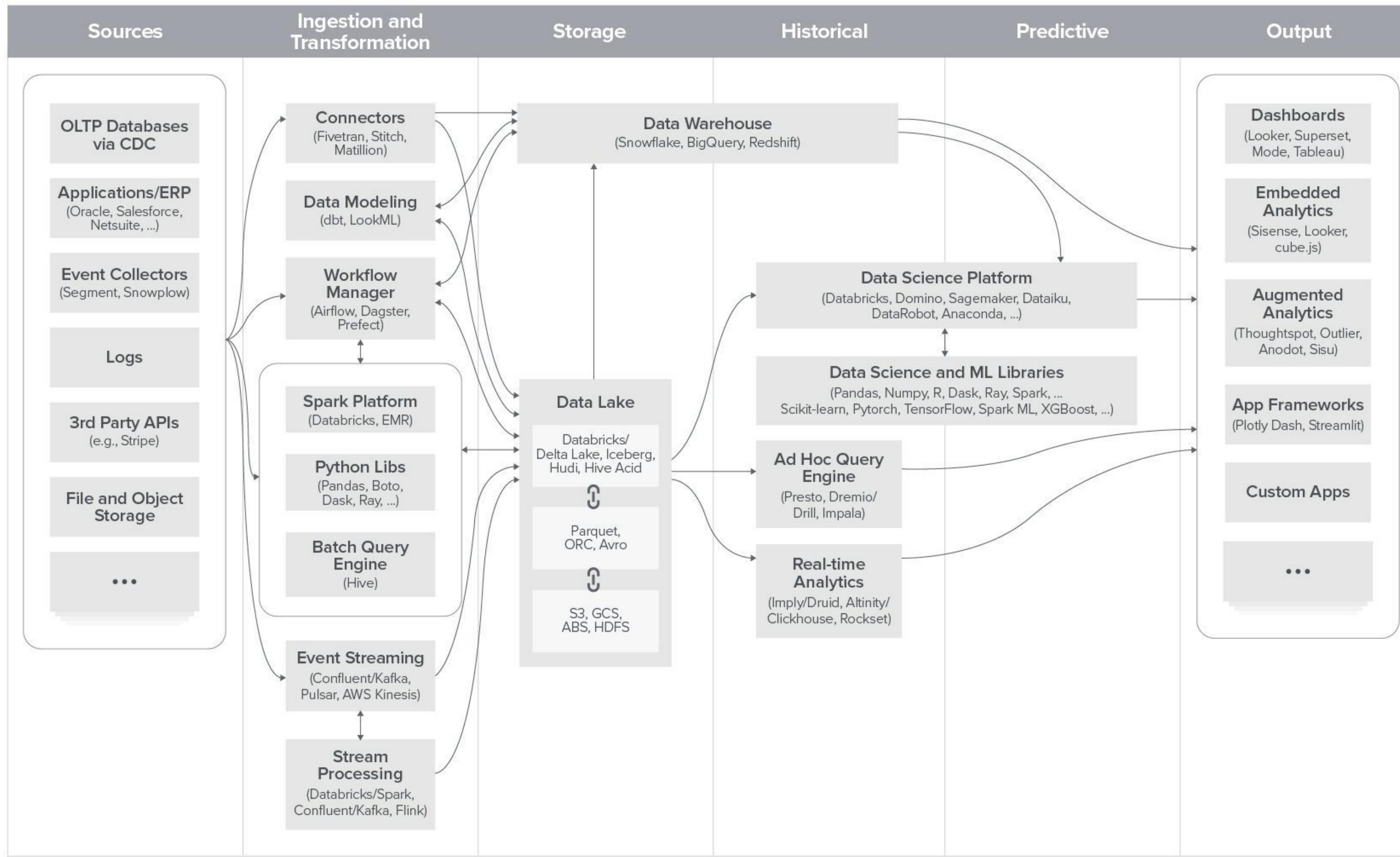
One open, simple platform to store and manage all of your data for all of your analytics workloads



For now

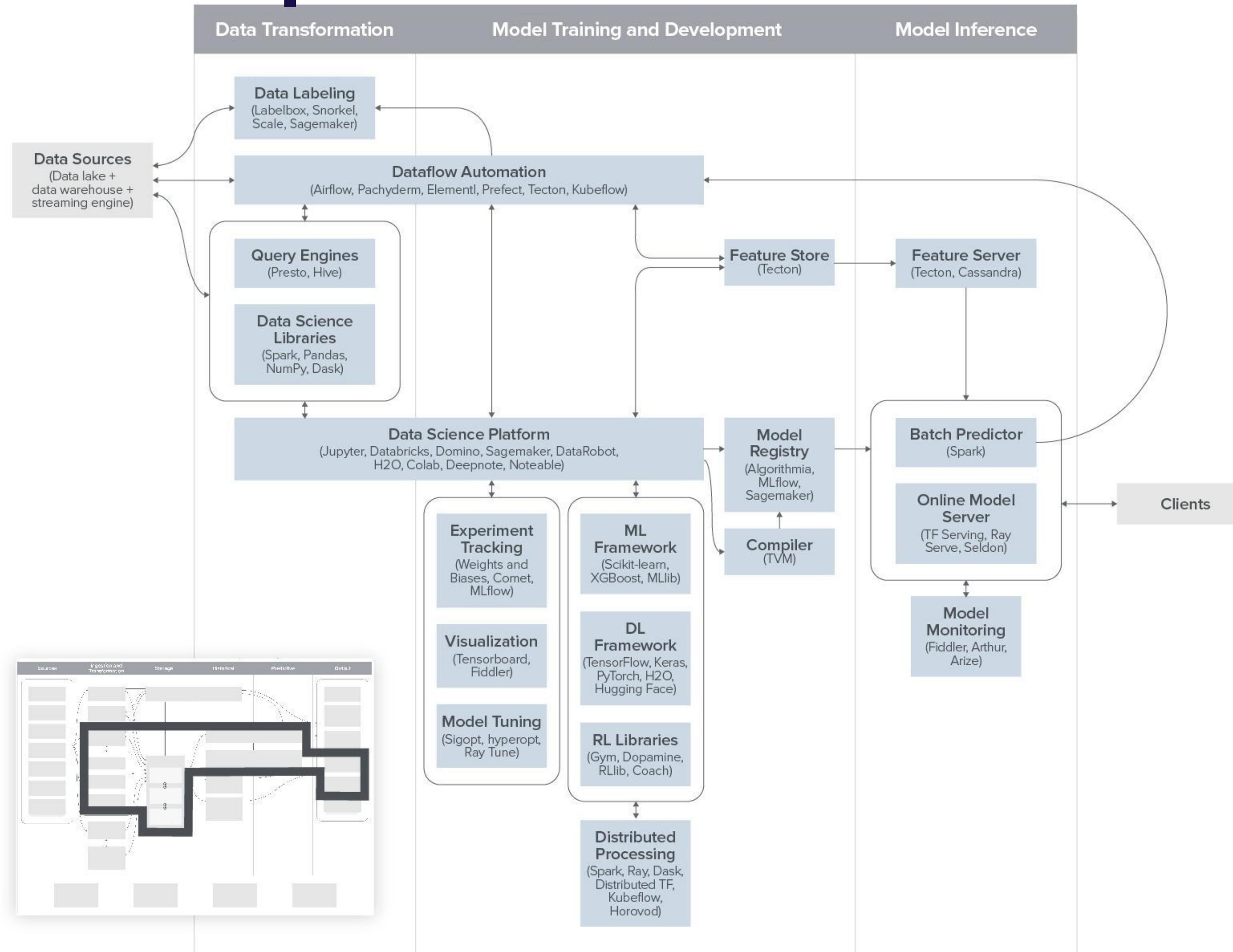
- **Binary data** (images, sound files, compressed texts) is stored as **objects**.
- **Metadata** (labels, user activity) is stored in **database**.
- If need features which are not obtainable from database (e.g logs), set up **data lake** and a process to aggregate needed data.
- At **training time**, copy the data that is needed to a **filesystem** on a fast drive.

There's a lot more to the story



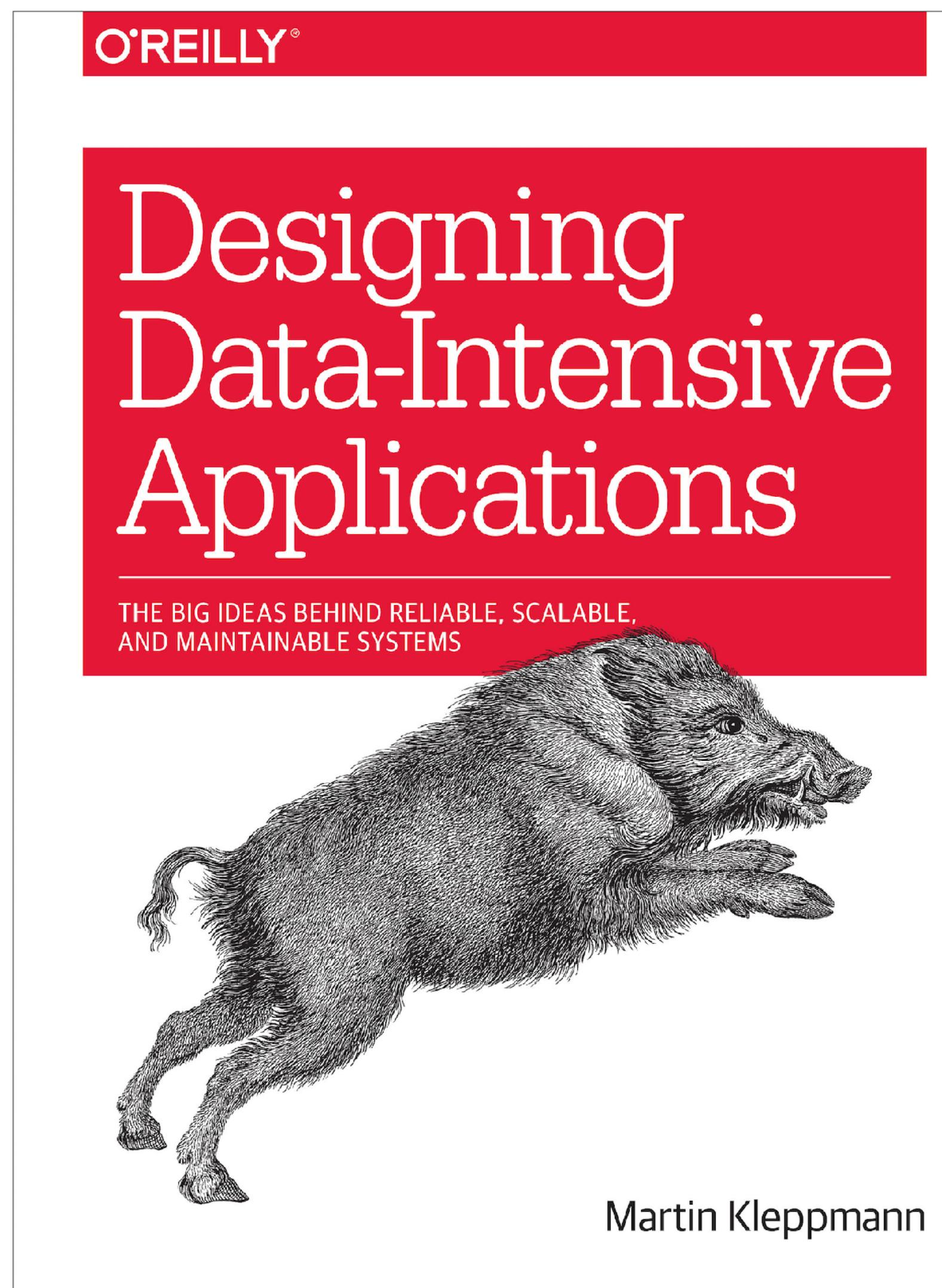
<https://a16z.com/2020/10/15/the-emerging-architectures-for-modern-data-infrastructure/>

Blueprint for AI and ML



<https://a16z.com/2020/10/15/the-emerging-architectures-for-modern-data-infrastructure/>

If you're truly interested



Don't just hack it together

NoSQL... Big Data... Scalability... CAP Theorem... Eventual Consistency... Sharding...

Nice buzzwords, but how does the stuff actually work?

<https://dataintensive.net>

Questions?



Amazon SageMaker

gradient^o
by Paperspace

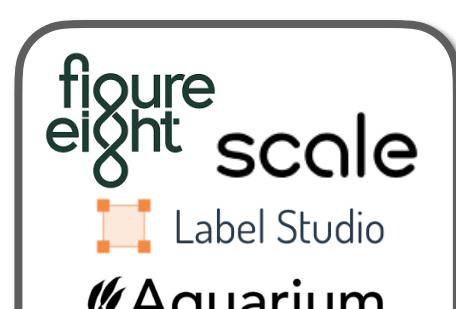
FLOYD

DOMINO
DATA LAB

“All-in-one”



Versioning



Labeling



DAGSTER dbt

Processing



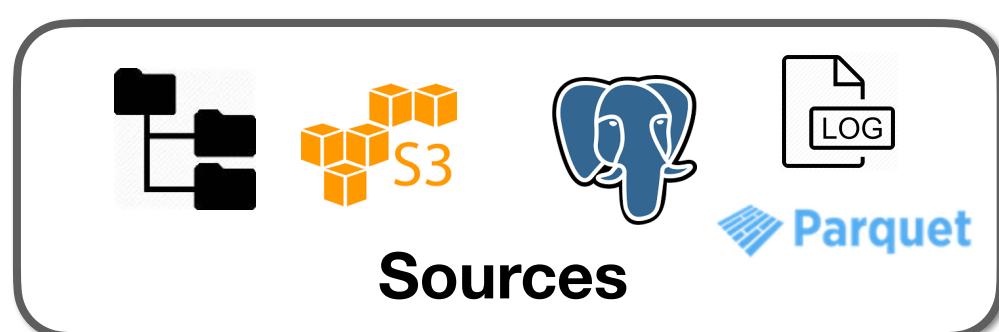
RAPIDS



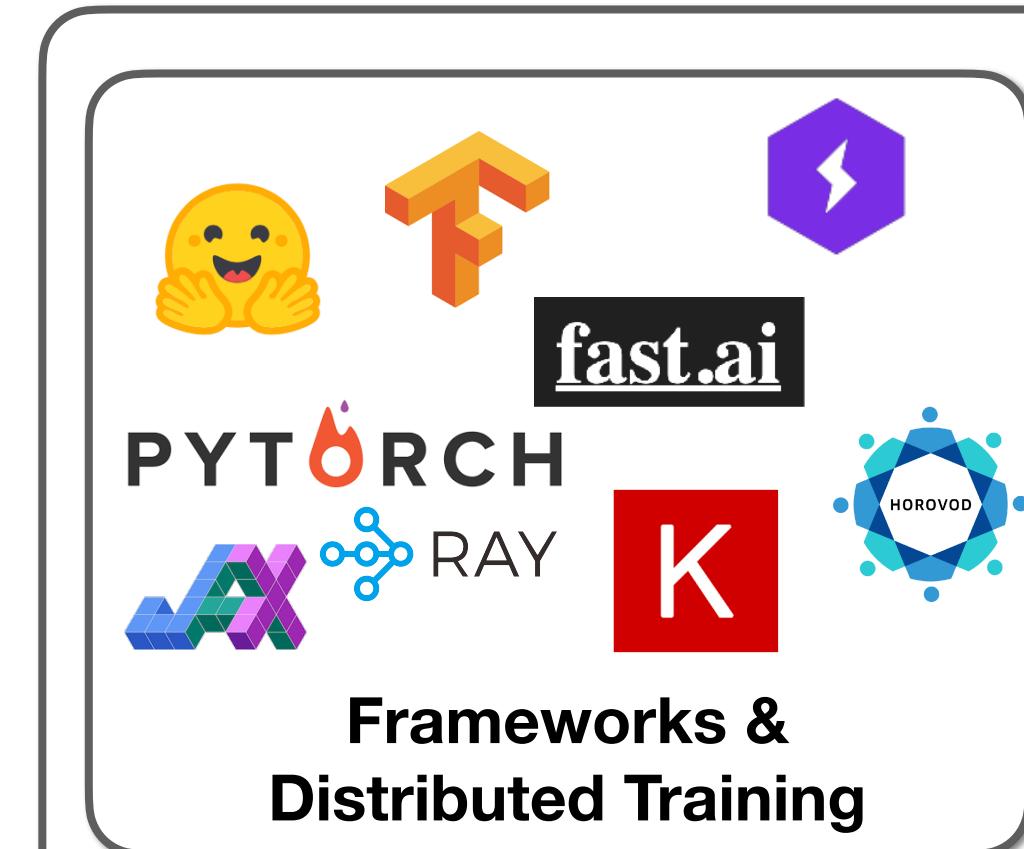
Exploration



Data Lake / Warehouse



Data



Frameworks &
Distributed Training



Resource Management



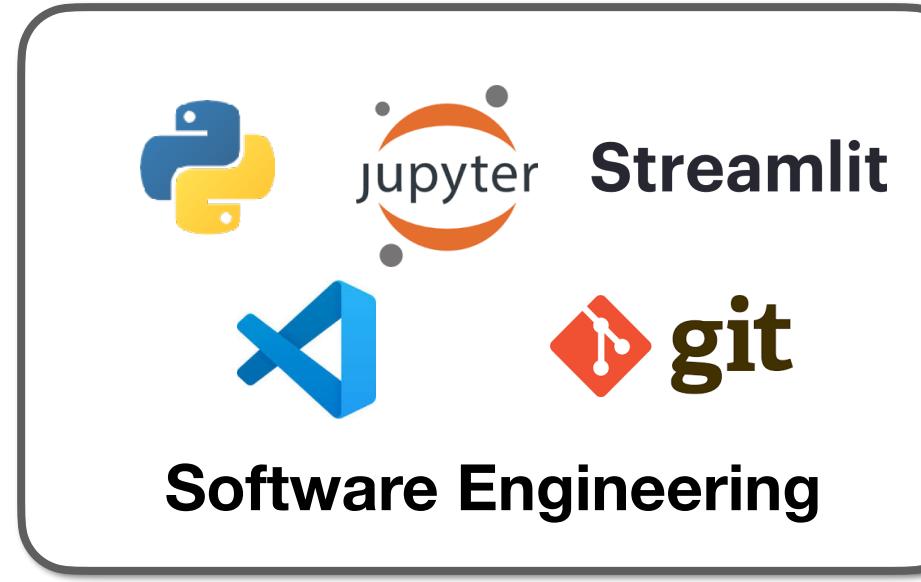
Training/Evaluation



Determined AI
Weights & Biases
Hyperparameter Tuning



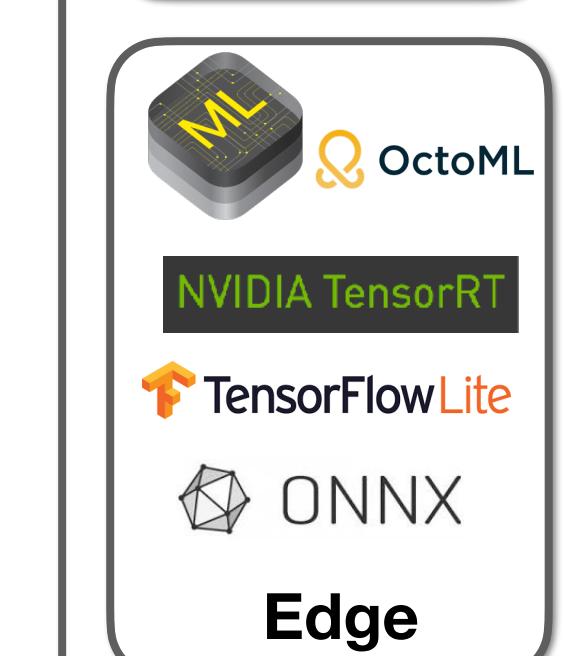
TensorBoard
mlflow
comet
Neptune
Machine Learning Lab



Software Engineering



Feature
Store



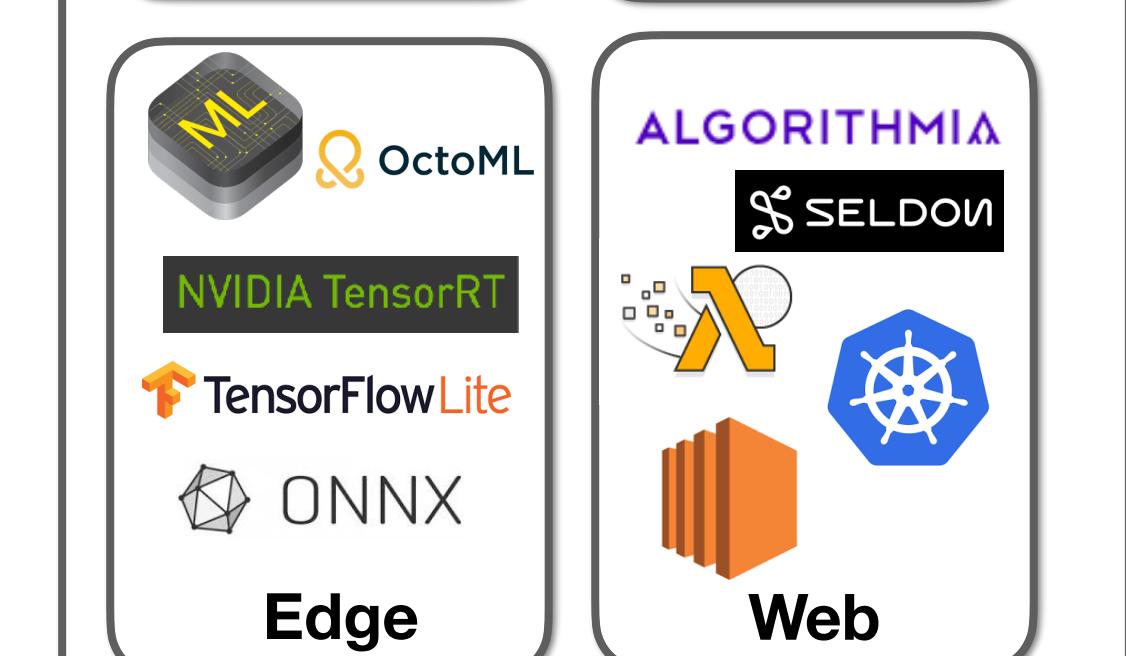
Edge



CI / Testing



Monitoring



Web



Deployment

Motivational Example

- We have to train a photo popularity predictor every night.
- For each photo, training data must include:
 - Metadata such as posting time, title, location
 - Some features of the user, such as how many times they logged in today.
 - Outputs of photo classifiers (content, style)

In database

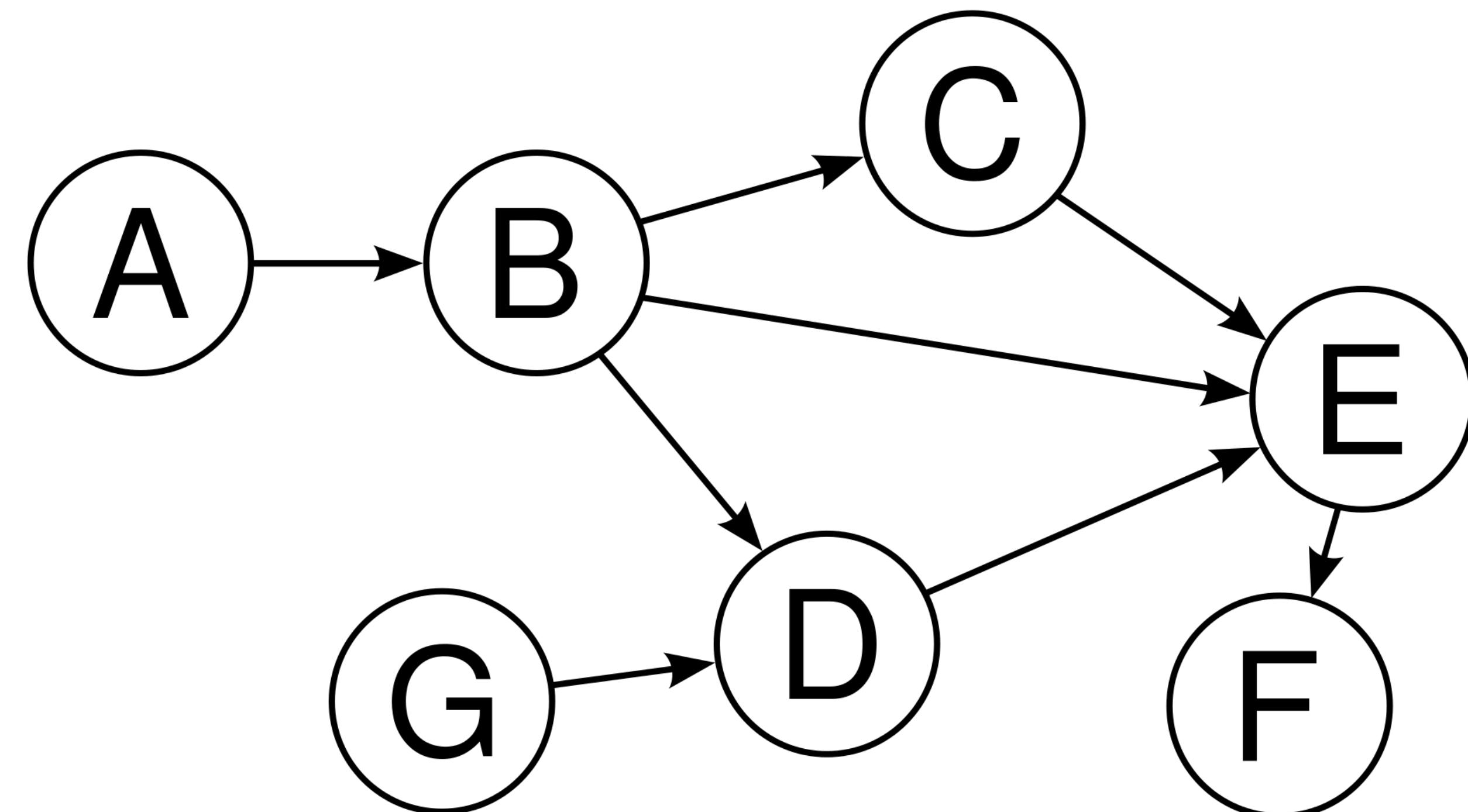
Need to compute

from logs

Need to run classifiers

Task Dependencies

- Some tasks can't be started until other tasks are finished.
- Finishing a task should "kick off" its dependencies

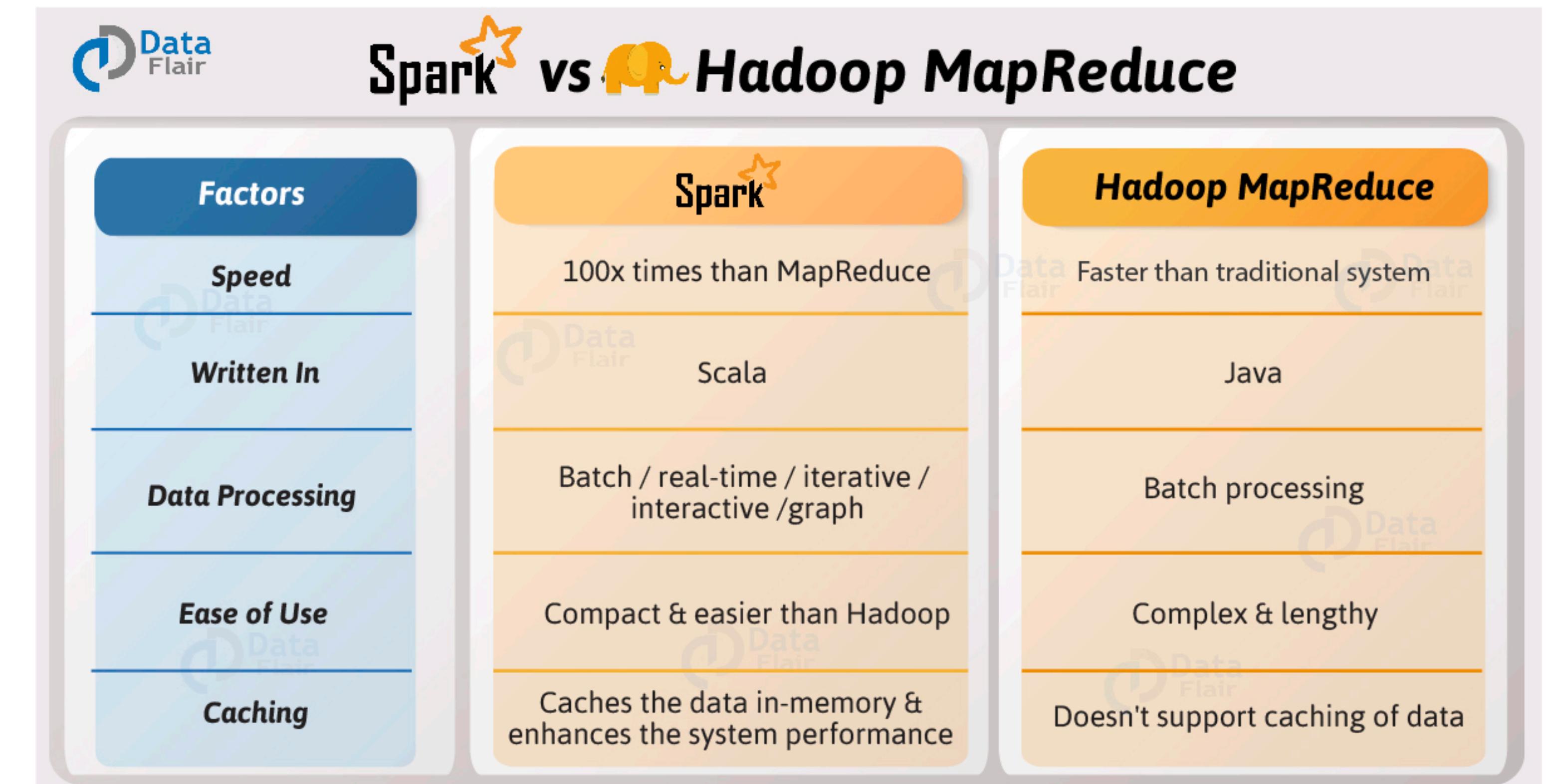


Desiderata

- Re-computation should depend on content
- Dependencies are not files, but programs and databases
- Work needs to be spread over many machines
- Many dependency graphs are executing all at once

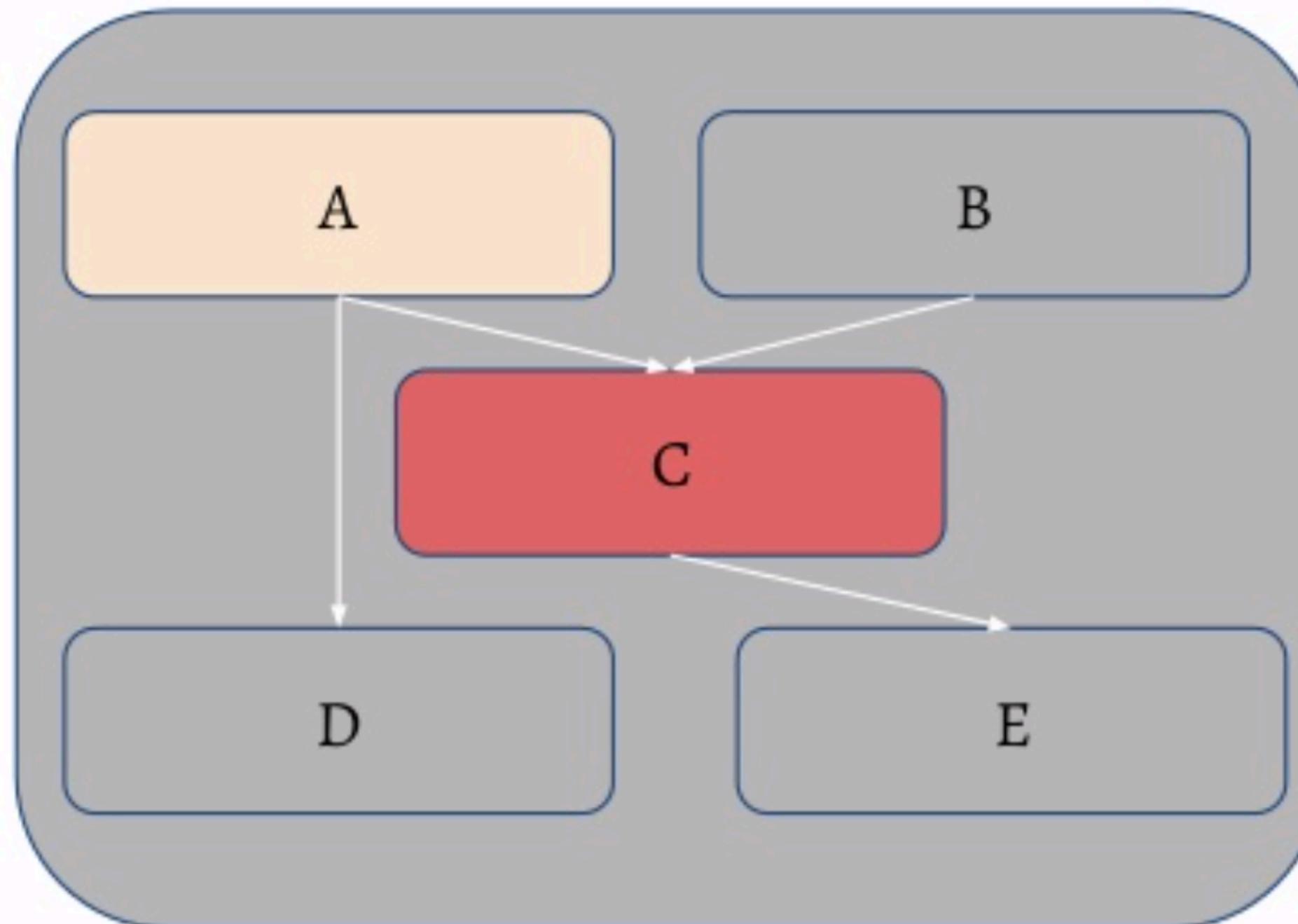
Hadoop/Spark

- Map/Reduce implementations
- Running data processing operations and simple ML on commodity hardware, with tricks to speed things up



<https://data-flair.training/blogs/spark-vs-hadoop-mapreduce/>

Airflow



```
# Airflow
dag = DAG(schedule_interval=
            timedelta(days=1),
           start_date=
               datetime(2015, 10, 6))

a = PythonOperator(
    task_id="A",
    python_callable=ClassA,
    dag=dag)

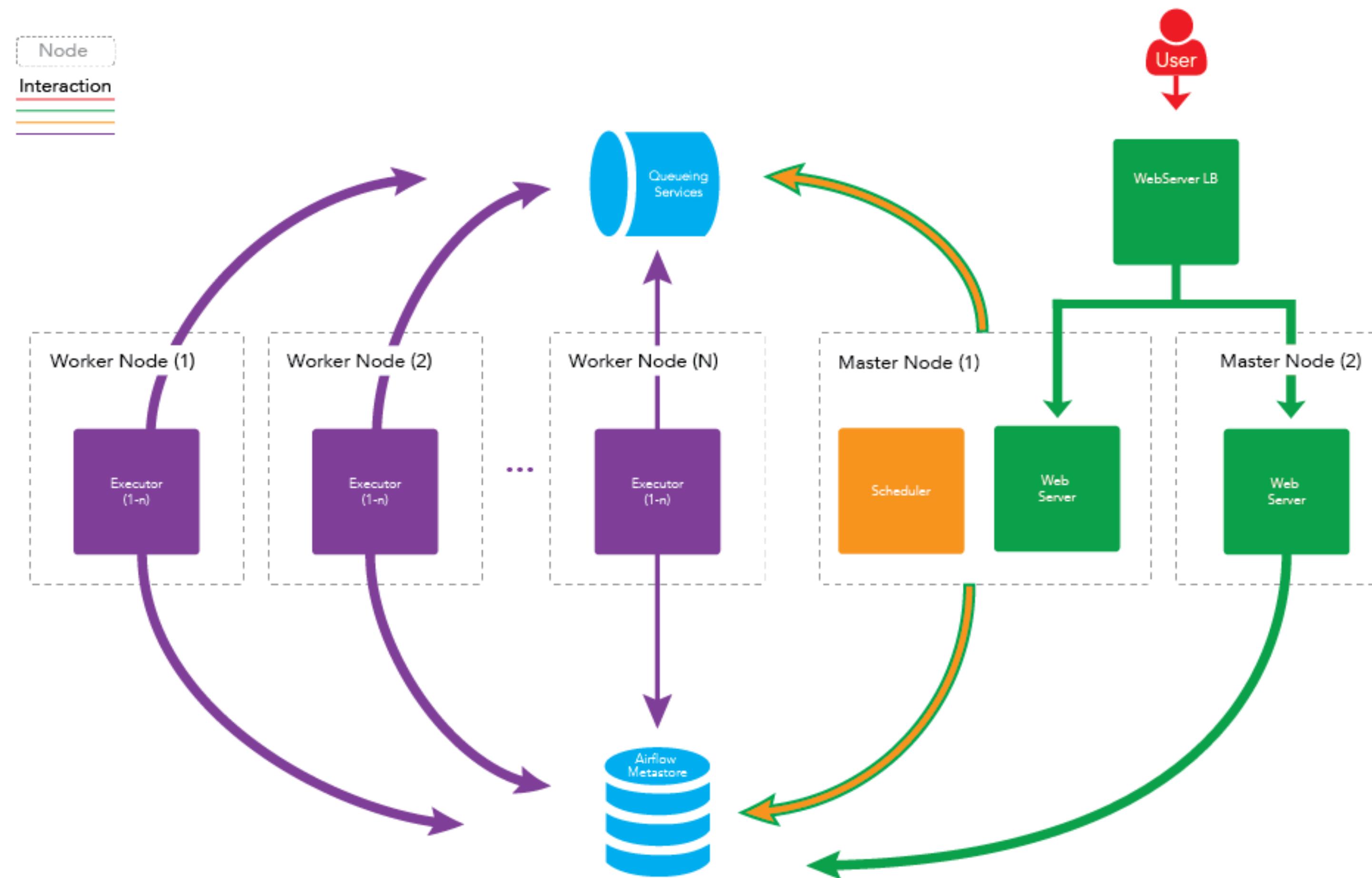
c = MySQLOperator(
    task_id="B",
    sql="DROP TABLE hello",
    dag=dag)

c.set_upstream(a)
```

<https://www.slideshare.net/PyData/how-i-learned-to-time-travel-or-data-pipelining-and-scheduling-with-airflow-67650418>

Distributing work

- The workflow manager has a **queue** for the tasks, and manages workers that pull from it, restarting jobs if they fail.



<http://site.clairvoyantsoft.com/making-apache-airflow-highly-available/>

Tensorflow Datasets + Apache Beam

TensorFlow > Resources > Datasets > Guide



Generating big datasets with Apache Beam

For example,
the 7TB
Colossal
Clean Corpus

Some datasets are too big to be processed on a single machine. [tfds](#) supports generating data across many machines by using [Apache Beam](#).

This doc has two sections:

- For user who want to generate an existing Beam dataset
- For developers who want to create a new Beam dataset

Generating a Beam dataset

Below are different examples of generating a Beam dataset, both on the cloud or locally.

⚠ Warning: When generating the dataset with the [tfds build CLI](#), make sure to specify the dataset config you want to generate or it will default to generate all existing configs. For example, for [wikipedia](#), use `tfds build wikipedia/20200301.en` instead of `tfds build wikipedia`.

On Google Cloud Dataflow

To run the pipeline using [Google Cloud Dataflow](#) and take advantage of distributed computation, first follow the [Quickstart instructions](#).

Once your environment is set up, you can run the [tfds build CLI](#) using a data directory on [GCS](#) and specifying the [required options](#) for the `--beam_pipeline_options` flag.

https://www.tensorflow.org/datasets/beam_datasets

Prefect

```
1 from prefect import task, Flow  
2  
3 @task  
4 def say_hello():  
5     print("Hello, world!")  
6  
7  
8 with Flow("My First Flow") as flow:  
9     say_hello()  
10  
11  
12 flow.run() # "Hello, world!"
```

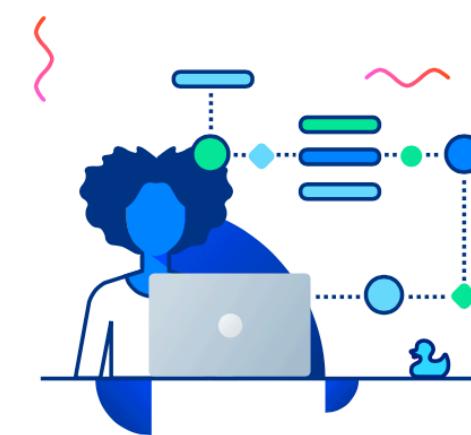


Focus on Your Code

Use a flexible Python framework to easily combine tasks into workflows, then deploy, schedule, and monitor their execution through the Prefect UI or API.

[LEARN MORE](#)[CORE DOCS](#)

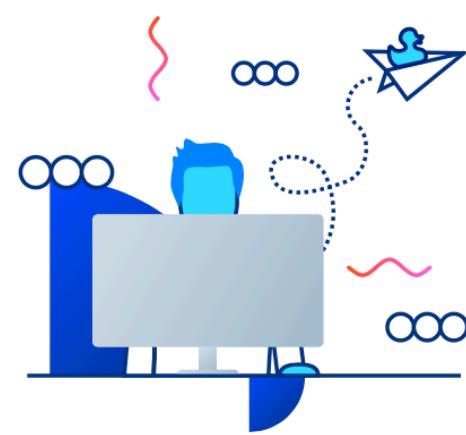
Execution in Your Cloud; Orchestration in Ours



STEP 01 - CORE

Build Your Flow

Design and test your workflow with our open-source Prefect Core framework.



STEP 02 - CLOUD

Register Your Flow

Send metadata (but never code!) to Prefect Cloud in order to register your flow for scheduling and execution.



STEP 03 - CORE

Run the Flow

Flow runs are executed on your private infrastructure, keeping data secure. State updates are sent to Cloud as metadata.



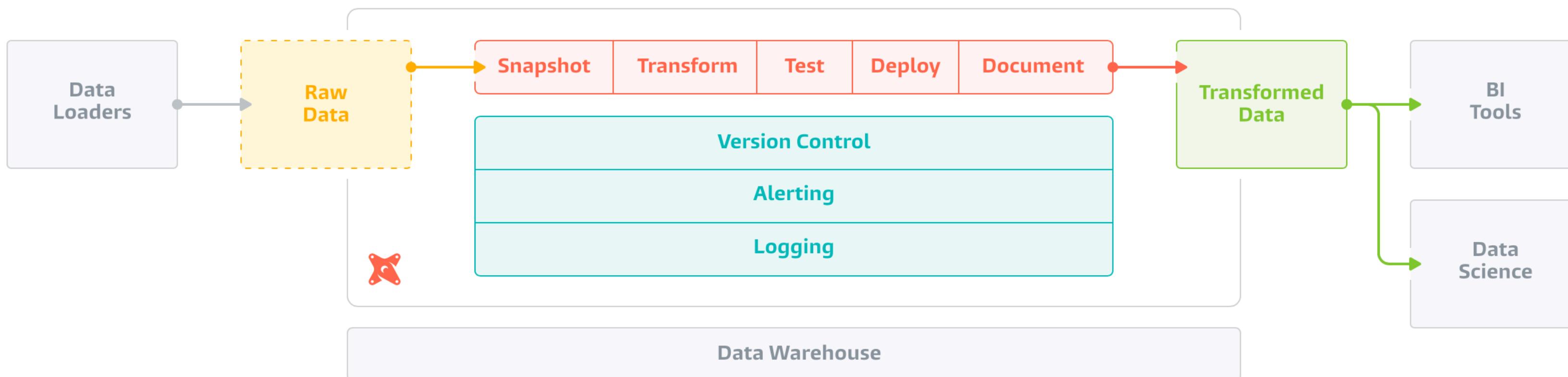
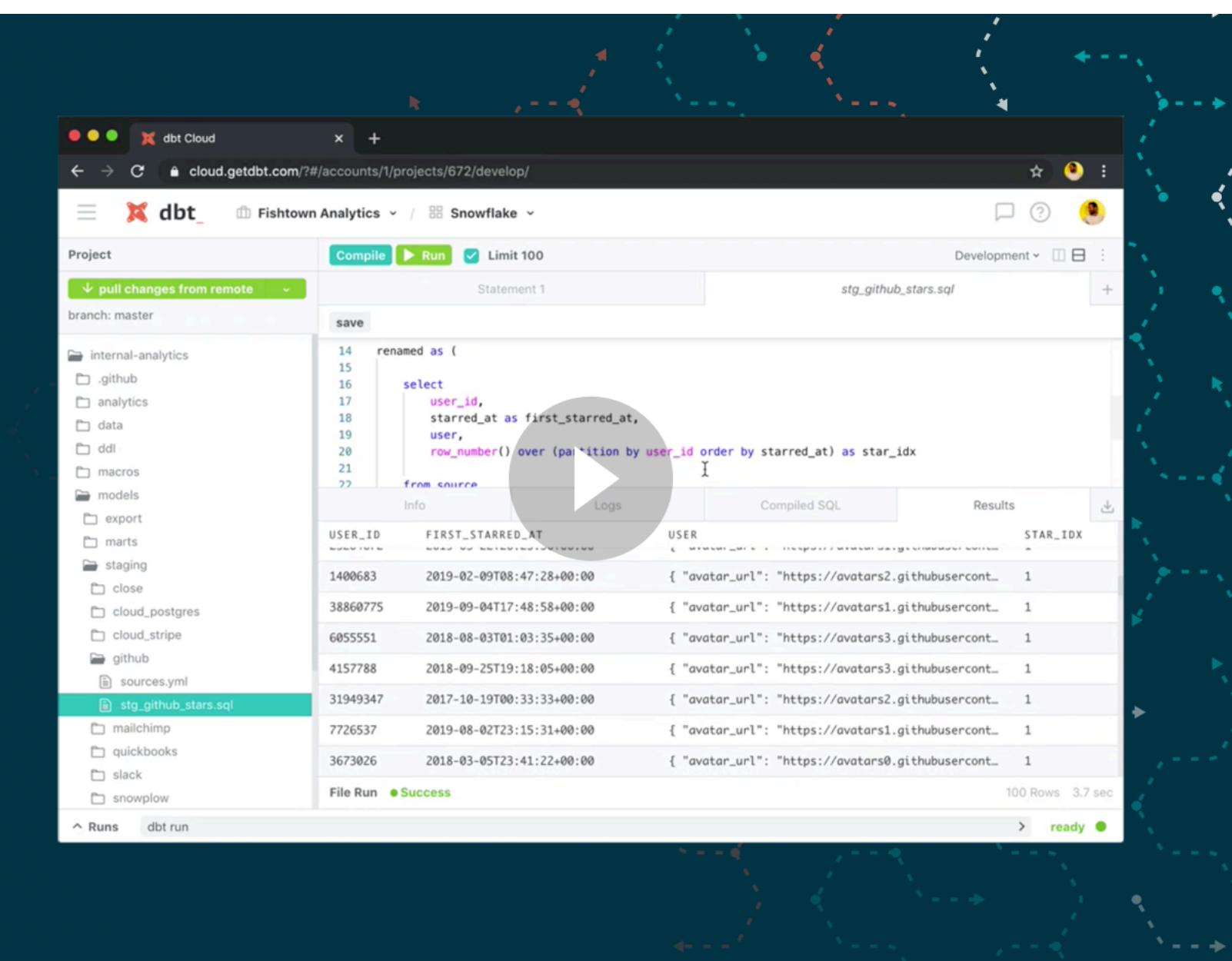
STEP 04 - CLOUD

Monitor and Manage

Use the Cloud UI to monitor all of your flows, no matter where or how often they run.

Your entire analytics engineering workflow

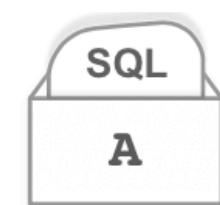
Analytics engineering is the data transformation work that happens between loading data into your warehouse and analyzing it. dbt allows anyone comfortable with SQL to own that workflow.

[Sign Up](#)
[Learn More >](#)


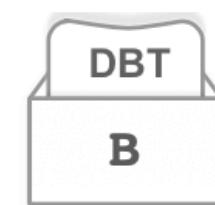
Dagster



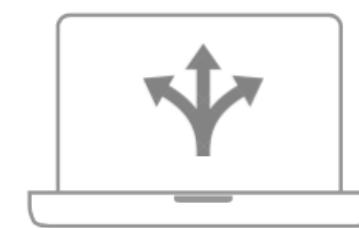
Dagster is a data orchestrator for machine learning, analytics, and ETL



Implement components in any tool, such as Pandas, Spark, SQL, or DBT.



Define your pipelines in terms of the data flow between reusable, logical components.



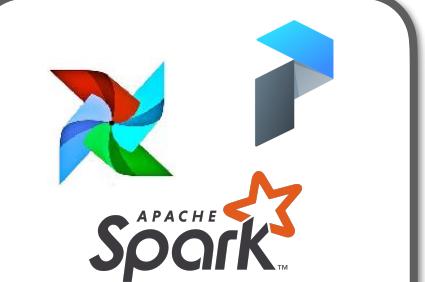
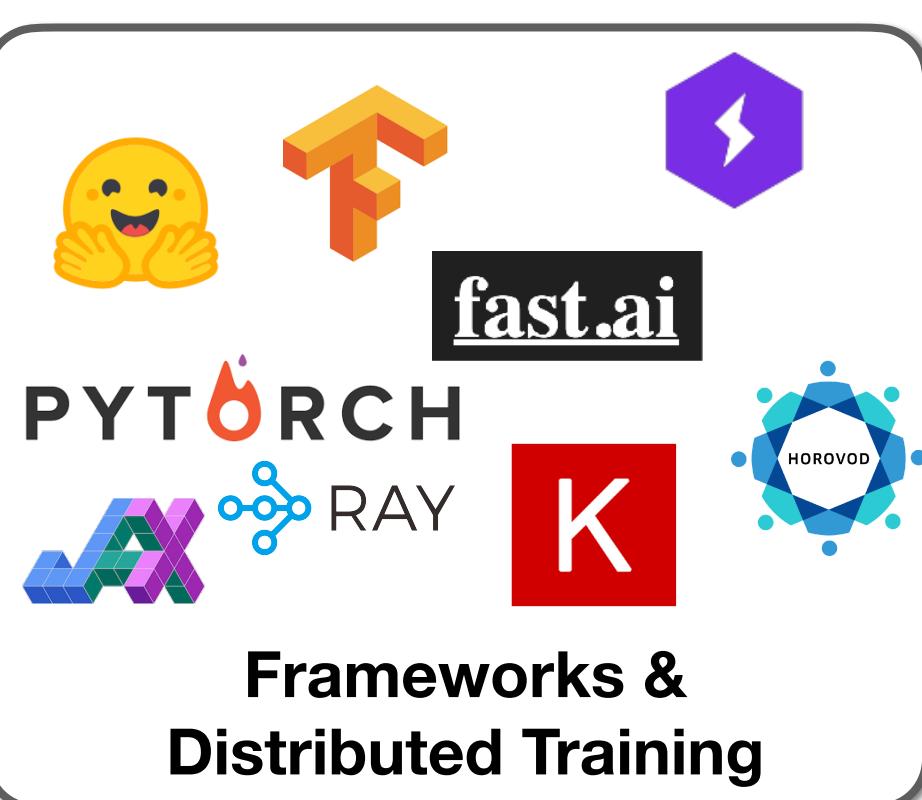
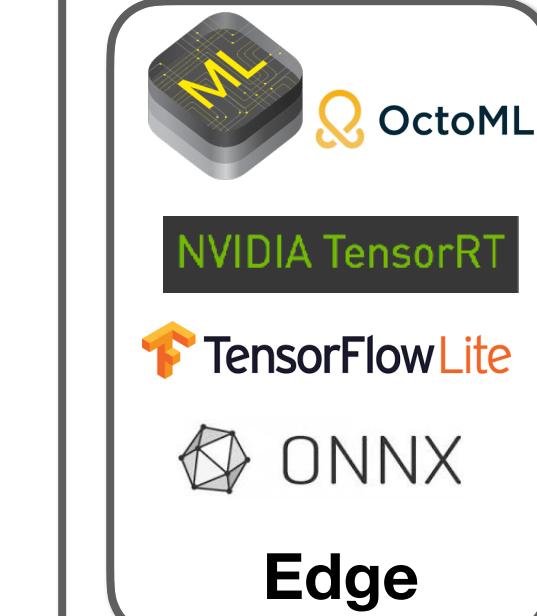
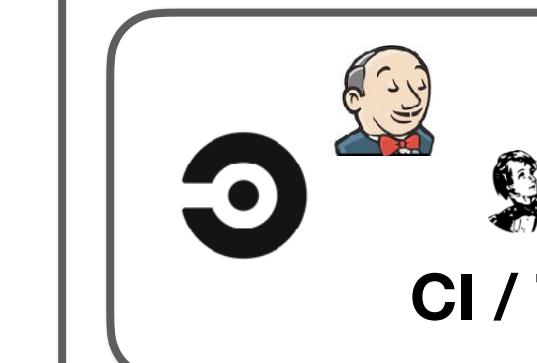
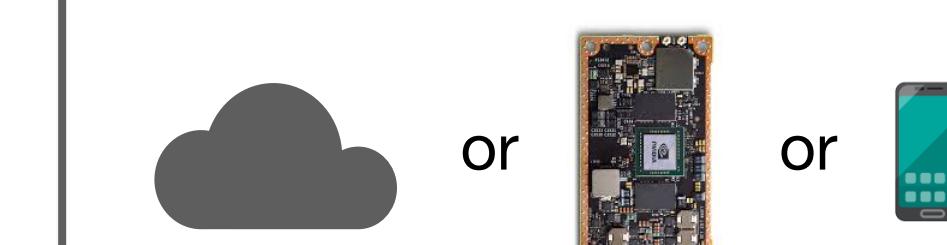
Test locally and run anywhere with a unified view of data pipelines and assets.



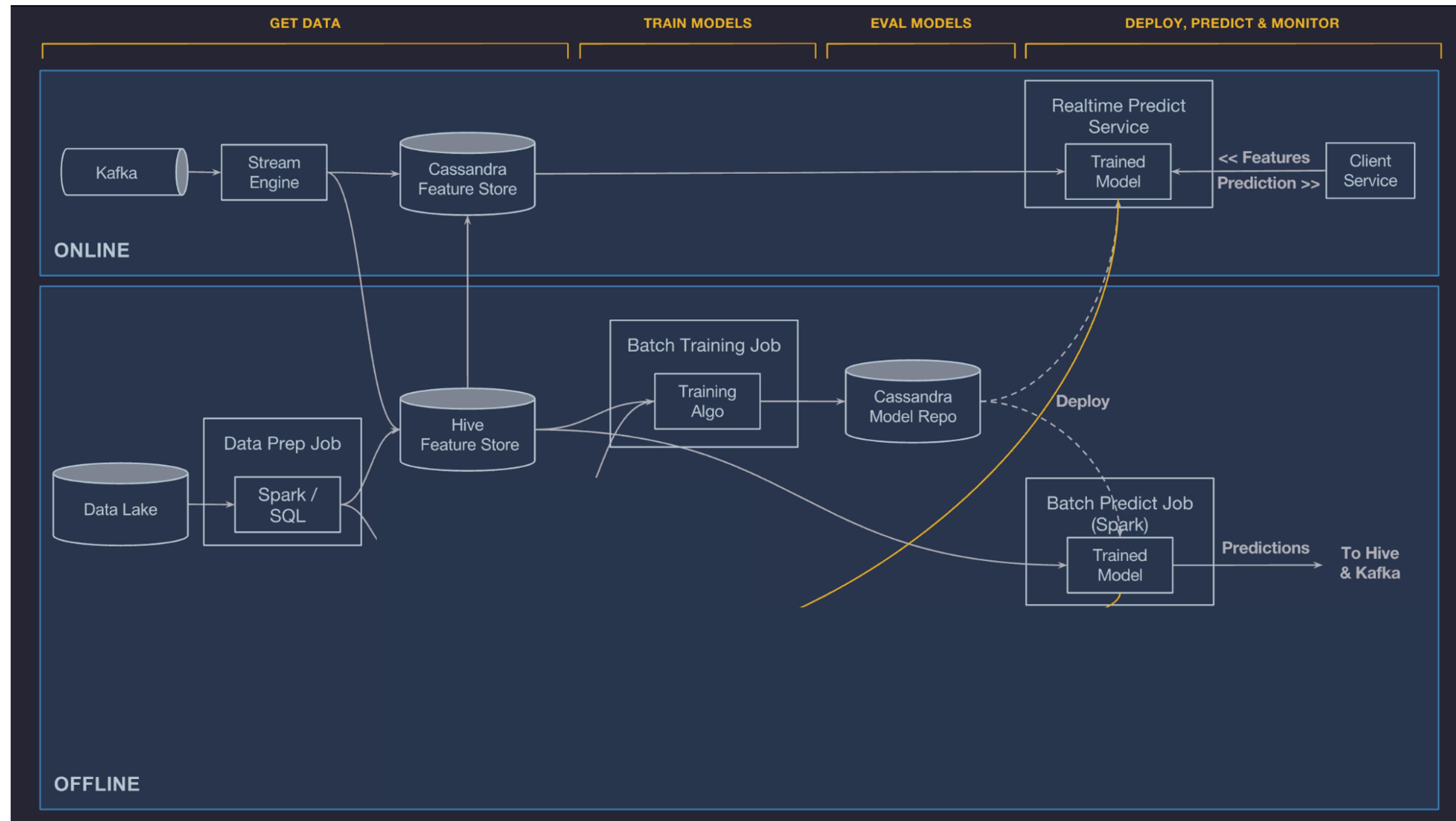
Amazon SageMaker

gradient[®]
by Paperspace

FLOYD

DOMINO
DATA LAB**“All-in-one”****Versioning****Labeling****Processing****Data Lake / Warehouse****Data****Frameworks & Distributed Training****Resource Management****Training/Evaluation****Hyperparameter Tuning****Experiment Management****Software Engineering****Feature Store****Edge****CI / Testing****Deployment**

Feature Store



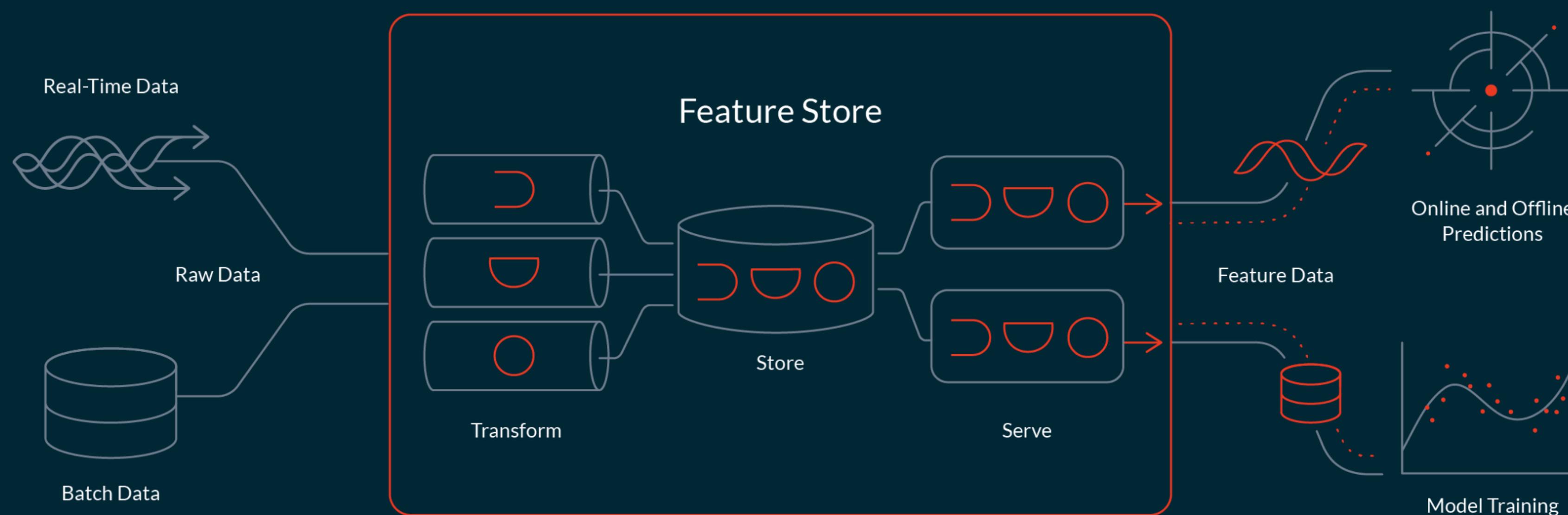
<https://eng.uber.com/michelangelo-machine-learning-platform/>

The Enterprise Feature Store for Machine Learning

Build a library of great features. Serve them in production. Do it at scale.

Corporate Email*

Request a free trial



<https://www.tecton.ai>



An open source feature store for machine learning

Feast is the bridge between your data and your machine learning models. It allows teams to register, ingest, serve, and monitor features in production.

[Quickstart](#)[Learn More](#)

```
customer_features = [
    'credit_score', 'balance', 'total_purchases', 'last_active'
]

# Fetch historical features
historical_features_df = fs.get_historical_features(customer_ids, customer_features)

# Train model
my_model = ml.fit(historical_features_df)

# Fetch online features
online_features = fs.get_online_features(customer_ids, customer_features)

# Predict using online features
prediction = my_model.predict(online_features)
```

Register

Feast provides a registry through which to explore, develop, collaborate on, and publish new feature definitions. The registry is the central interface for all interactions with the feature store.

Ingest

Easily ingest data from both batch and streaming sources into both online and offline feature stores, automating data management and making features available for serving.

Serve

A feature retrieval interface that provides a consistent view of features in stores. Feast provides a point-in-time correct interface for training data, and a low-latency API for online serving.

Monitor

Feast allows teams to confidently operate machine learning systems by publishing operational metrics, statistics, and logs to their existing production monitoring infrastructure.

Try to keep things simple

- Don't overengineer
- For example, UNIX has powerful parallelism, streaming, highly optimized tools

```
find . -type f -name '*.pgn' -print0 \  
| xargs -0 -n4 -P4 mawk '/Result/ { split($0, a, "-"); res = substr(a[1], length(a[1]), 1); if (res == 1) white++; if (res == 0) black++; if (res == 2) draw++ } END { print white+black+draw, white, black, draw }' \  
| mawk '{games += $1; white += $2; black += $3; draw += $4; } END { print games, white, black, draw }'
```

Command-line Tools can be 235x Faster than your Hadoop Cluster

January 18, 2014



26 minutes

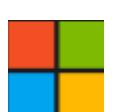
in parallel

```
cat *.pgn | grep "Result" | sort | uniq -c
```

70 seconds

<https://adamdralke.com/command-line-tools-can-be-235x-faster-than-your-hadoop-cluster.html>

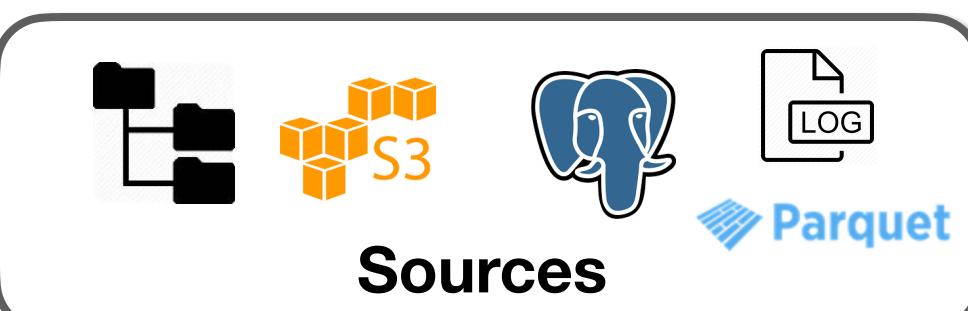
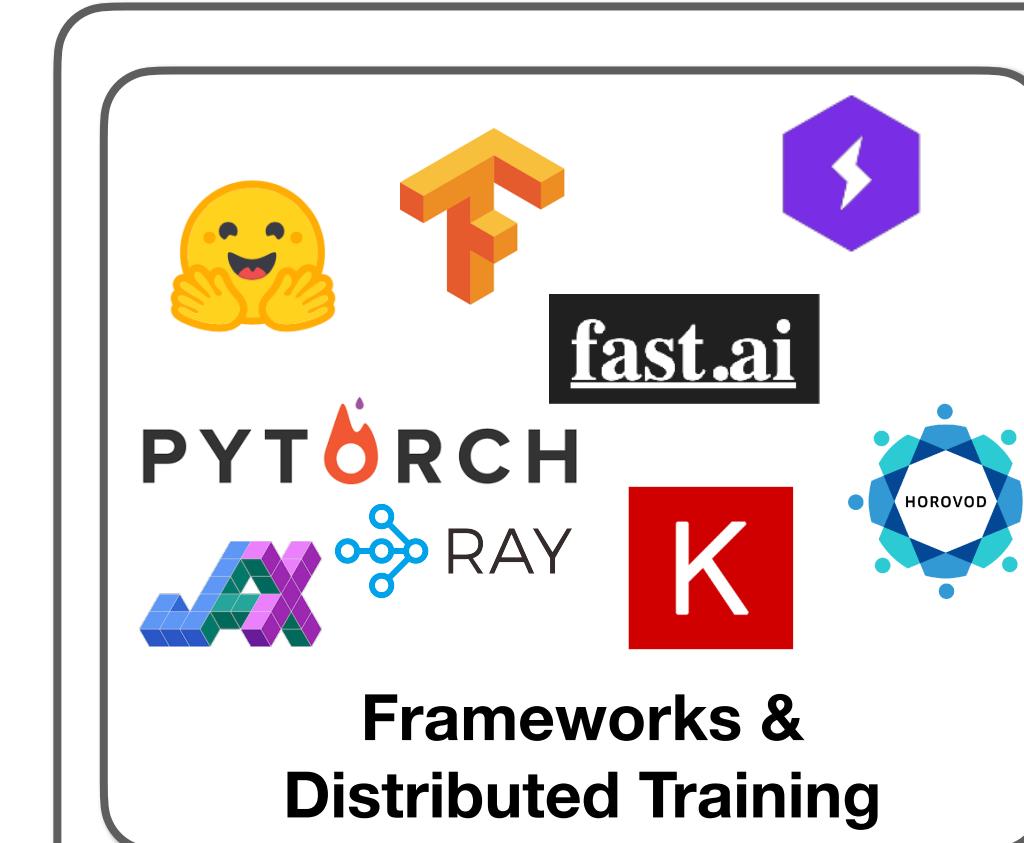
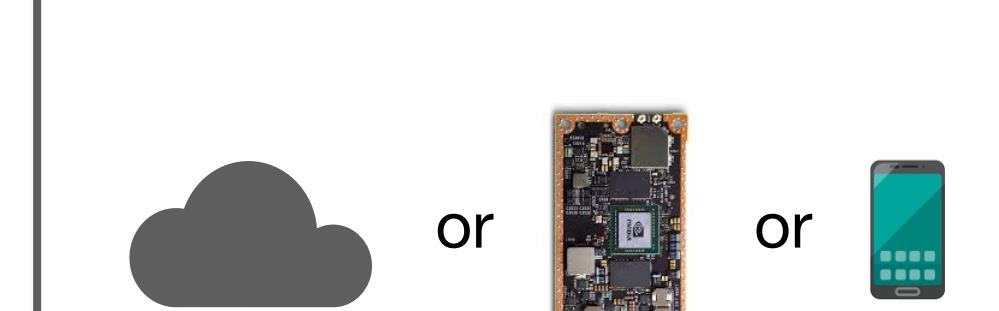
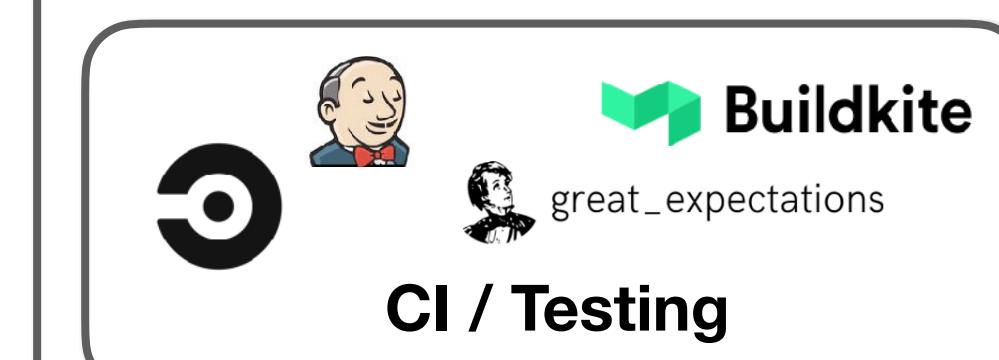
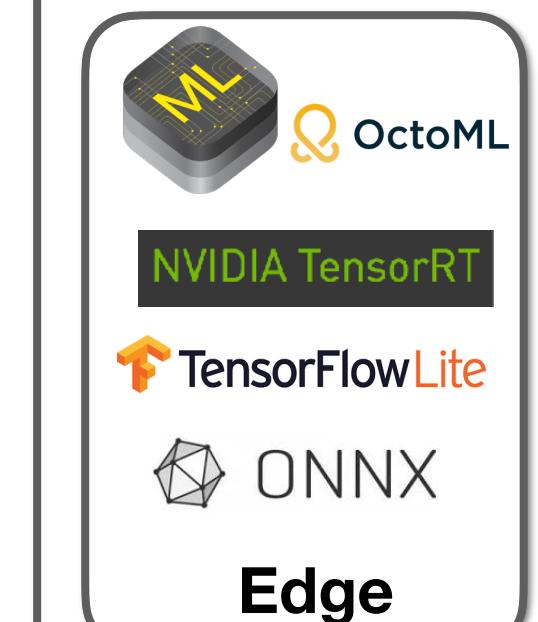
Questions?



Amazon SageMaker

gradient[®]
by Paperspace

FLOYD

DOMINO
DATA LAB**“All-in-one”****Data****Training/Evaluation**

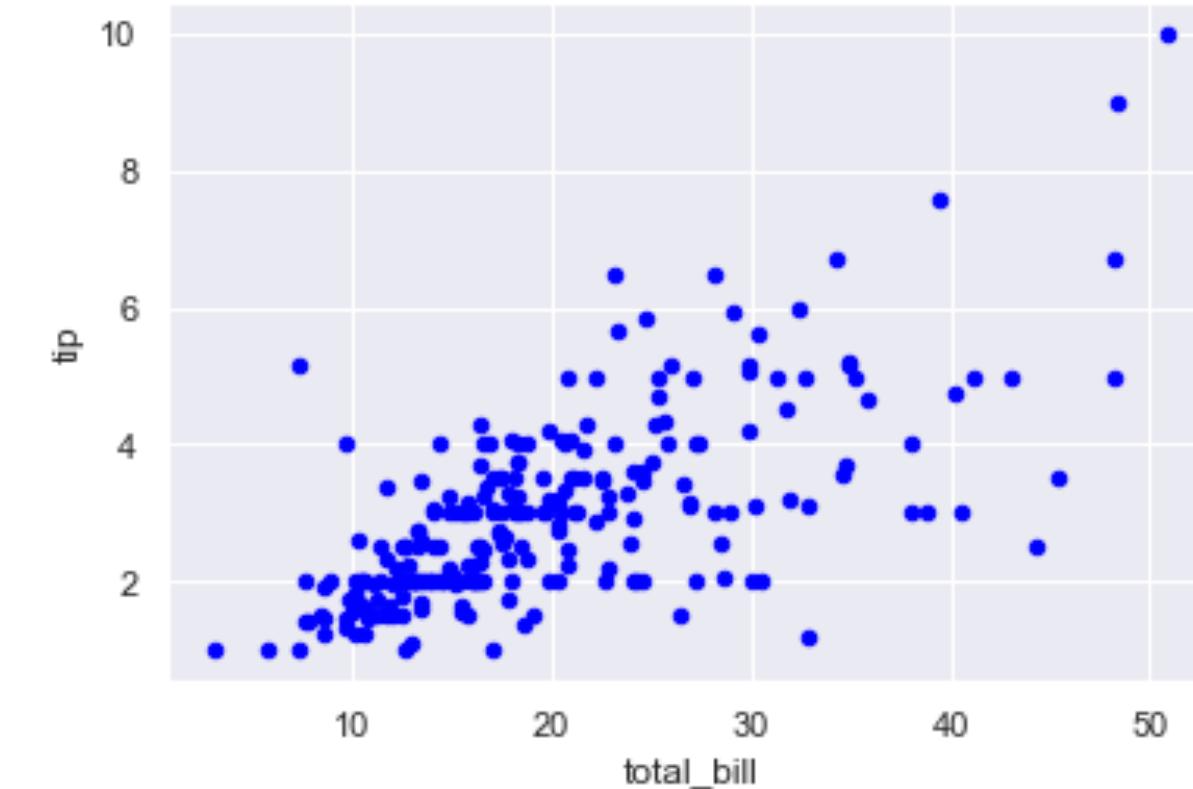
Pandas

In [12]:

```
import numpy as np
import pandas as pd
import matplotlib.pyplot as plt

tips = pd.read_csv("tips.csv")
tips.plot.scatter(x="total_bill", y="tip", c="Blue")
```

Out[12]: <matplotlib.axes._subplots.AxesSubplot at 0xc5120b0>



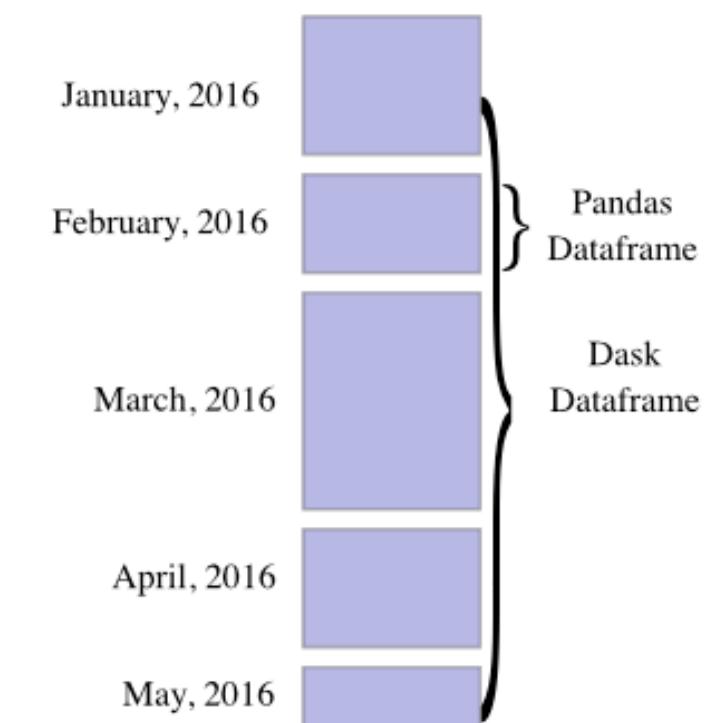
- The workhorse of Python data science
- Definitely do a few projects using it if you haven't used it before

<https://projectcodeed.blogspot.com/2019/08/setting-up-jupyter-notebooks-for-data.html>

Dask



pandas



Dask dataframes scale pandas workflows,
enabling applications in time series,
business intelligence, and general data
munging on big data.

RAPIDS

Open GPU Data Science

ⓘ ACCELERATED DATA SCIENCE

The RAPIDS suite of open source software libraries gives you the freedom to execute end-to-end data science and analytics pipelines entirely on GPUs.

[Learn about RAPIDS »](#)

⚡ SCALE OUT ON GPUS

Seamlessly scale from GPU workstations to multi-GPU servers and multi-node clusters with Dask.

[Learn about Dask »](#)

⚙ PYTHON INTEGRATION

Accelerate your Python data science toolchain with minimal code changes and no new tools to learn.

[Learn about our libraries »](#)

GETTING STARTED

The RAPIDS data science framework is designed to have a familiar look and feel to data scientist working in Python. Here's a code snippet where we read in a CSV file and output some descriptive statistics:

```
import cudf

gdf = cudf.read_csv('path/to/file.csv')
for column in gdf.columns:
    print(gdf[column].mean())
```

Find more details on our [get started section »](#)

⚡ TRY NOW ONLINE

Jump right into a GPU powered RAPIDS notebook online.

Try with [BlazingSQL \(RAPIDS 0.15+\) »](#)

Or with [Colabatory \(RAPIDS 0.14 only\) »](#)

📖 10 MINUTES TO CUDF

Modeled after 10 Minutes to Pandas, this is a short introduction to cuDF that is geared mainly for new users.

[Go to guide »](#)

📖 MULTI-GPU WITH DASK-CUDF

A short introduction to multi-GPU solutions with a distributed DataFrame via Dask-cuDF.

[Go to guide »](#)

📖 EXAMPLE NOTEBOOKS

A Github repository with our introductory examples of XGBoost, cuML demos, cuGraph demos, and more.

[Go to repo »](#)

📖 EXAMPLE COMMUNITY NOTEBOOKS

A second Github repository with our extended collection of community contributed notebook examples.

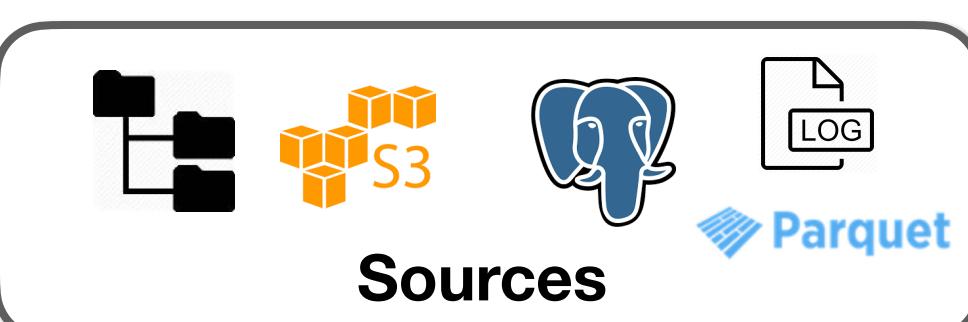
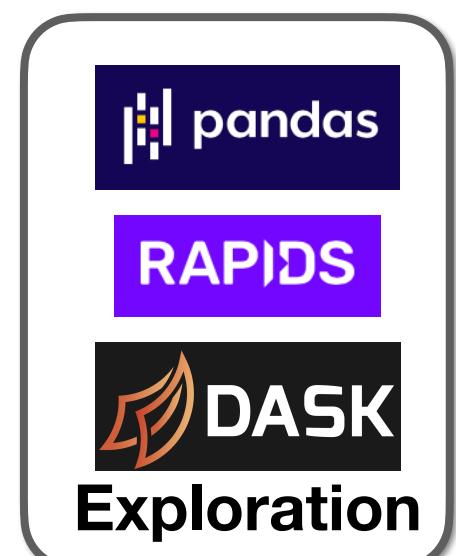
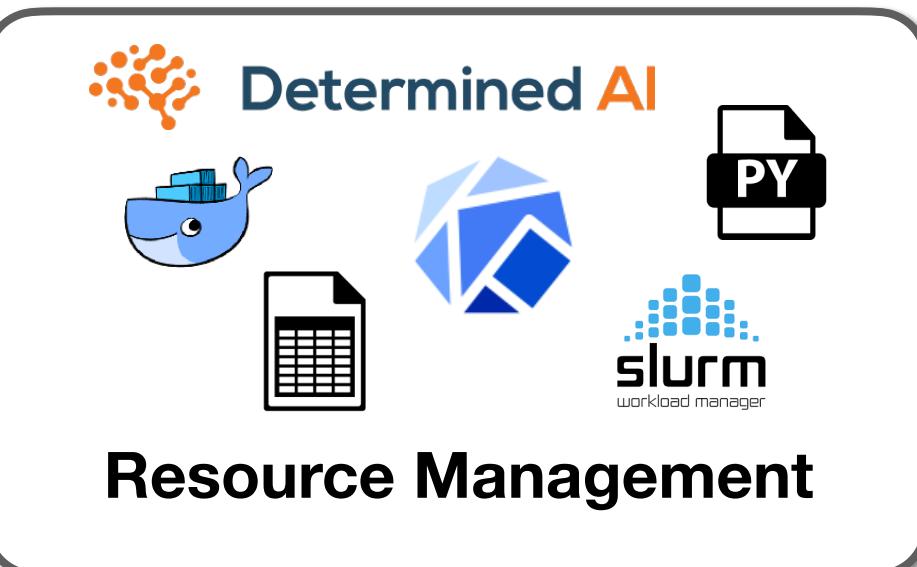
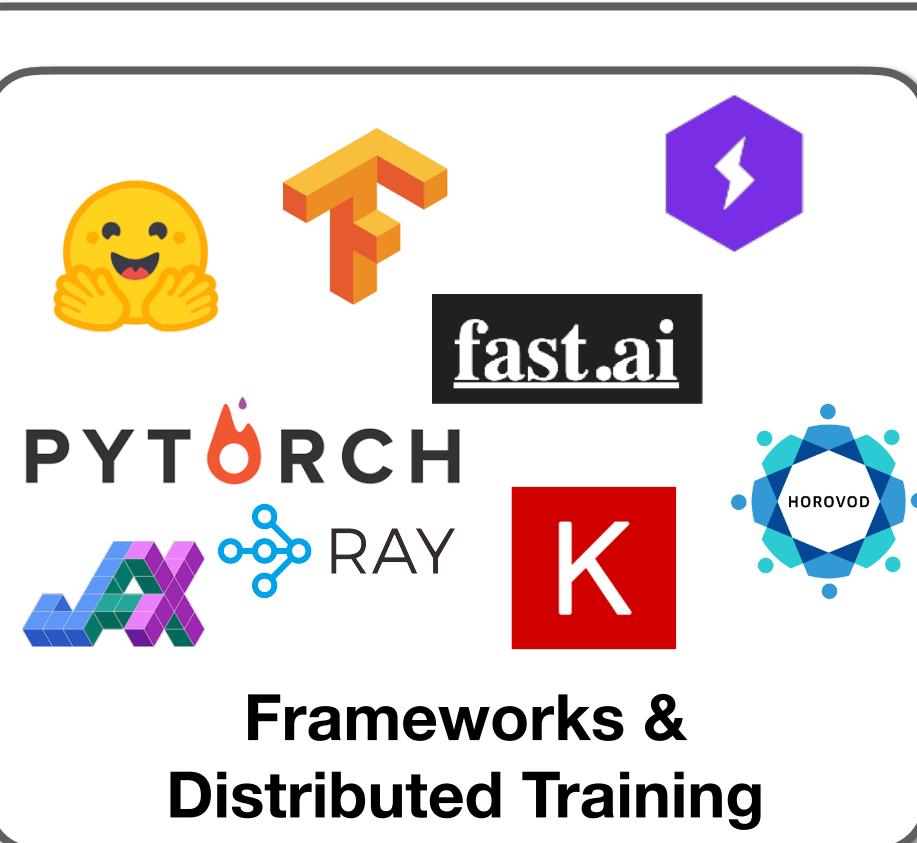
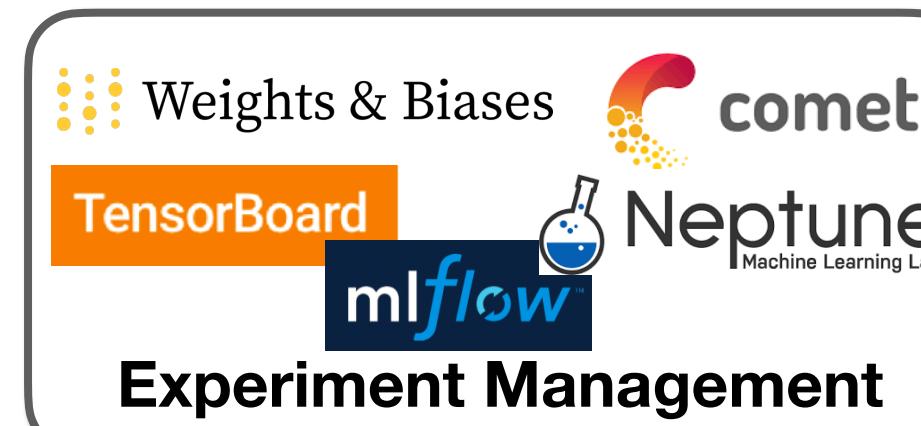
[Go to repo »](#)



Amazon SageMaker

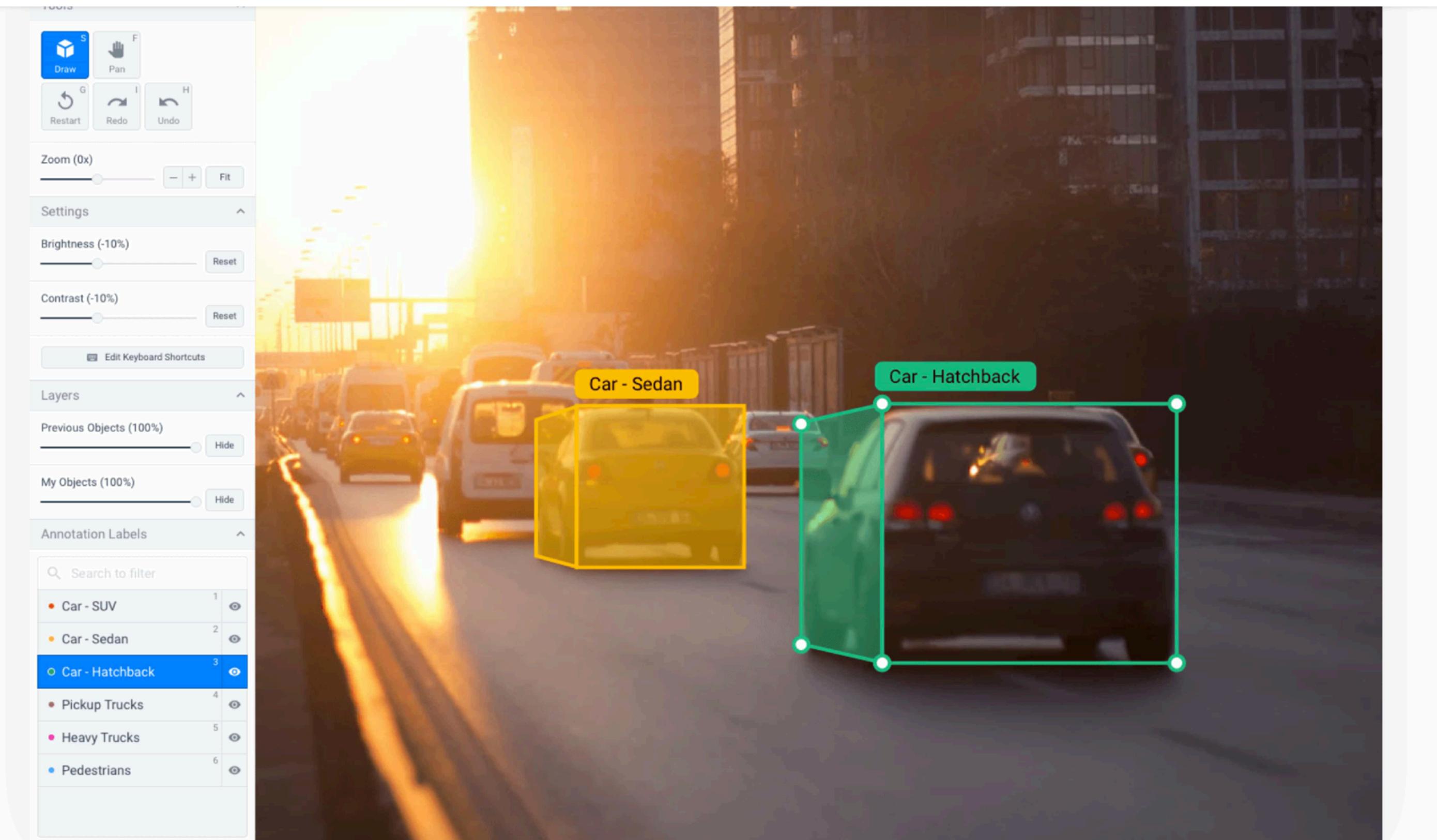
gradient[®]
by Paperspace

FLOYD

DOMINO
DATA LAB**“All-in-one”****Data****Training/Evaluation****Deployment**

Data Labeling

1. User Interfaces
2. Sources of labor
3. Service companies

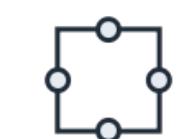


Standard set of features:

- bounding boxes, segmentations, keypoints, cuboids
- set of applicable classes



Categorization



Bounding Box



Semantic Segmentation



Line Annotation



Keypoint Annotation



Cuboid Annotation



Contour Annotation



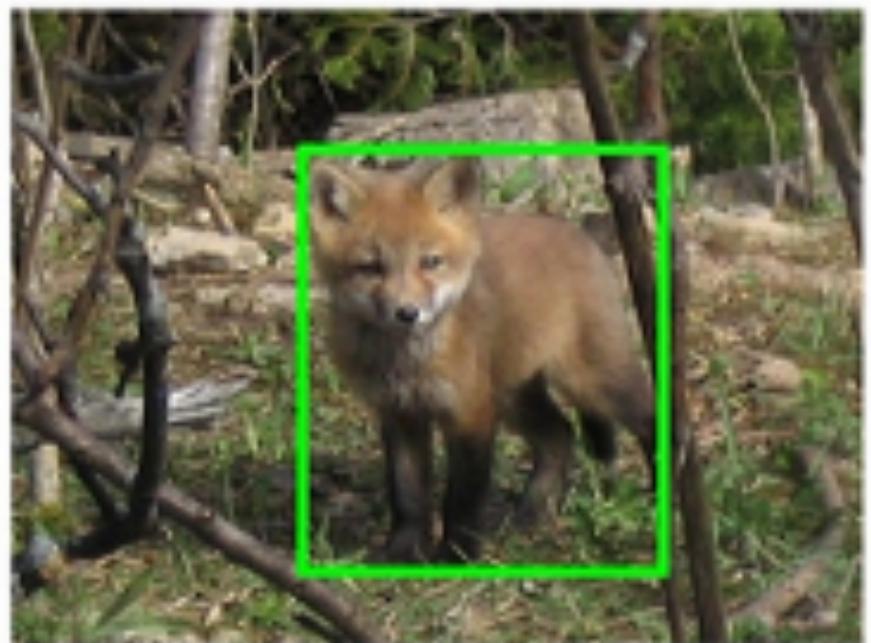
Poly Mask Annotation



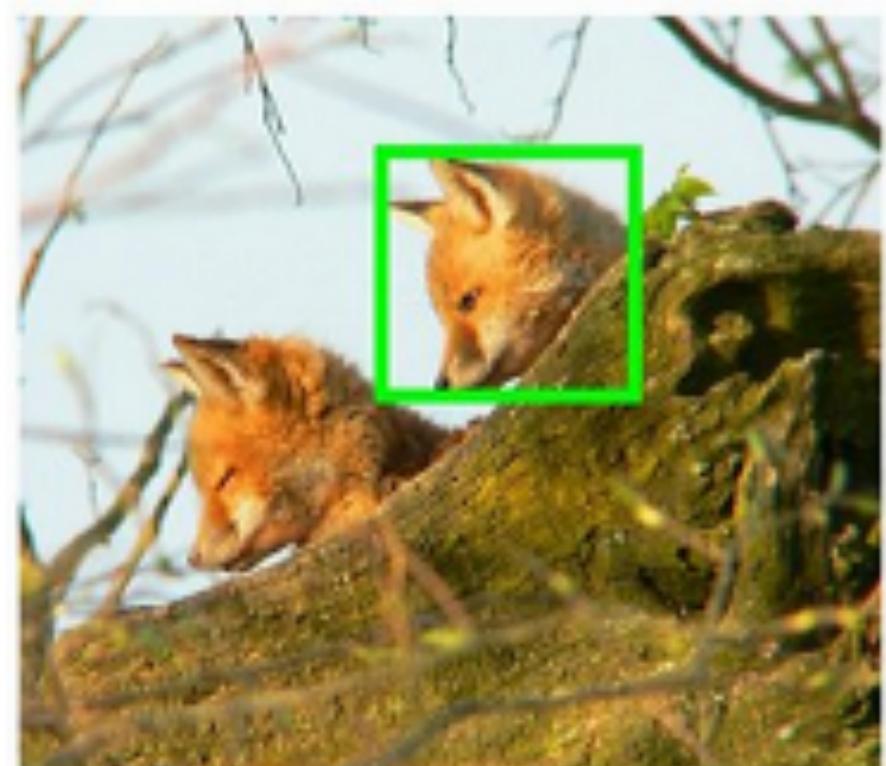
Polygonal Annotation

Training the annotators is crucial

Rule 1: Include all visible part and draw as tightly as possible.



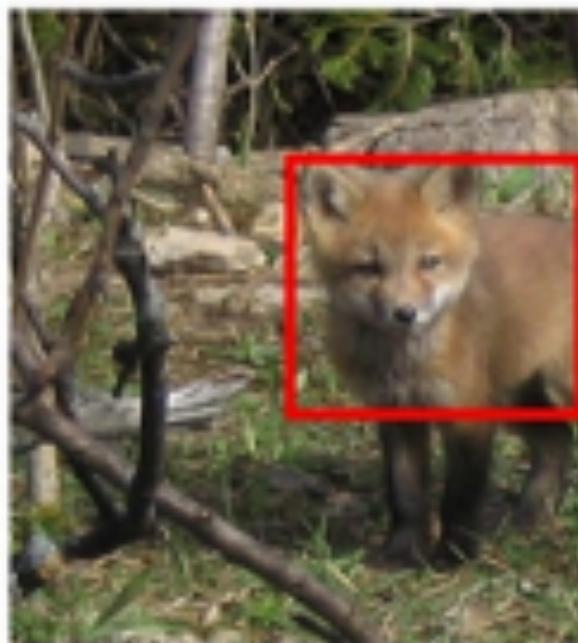
CORRECT



CORRECT



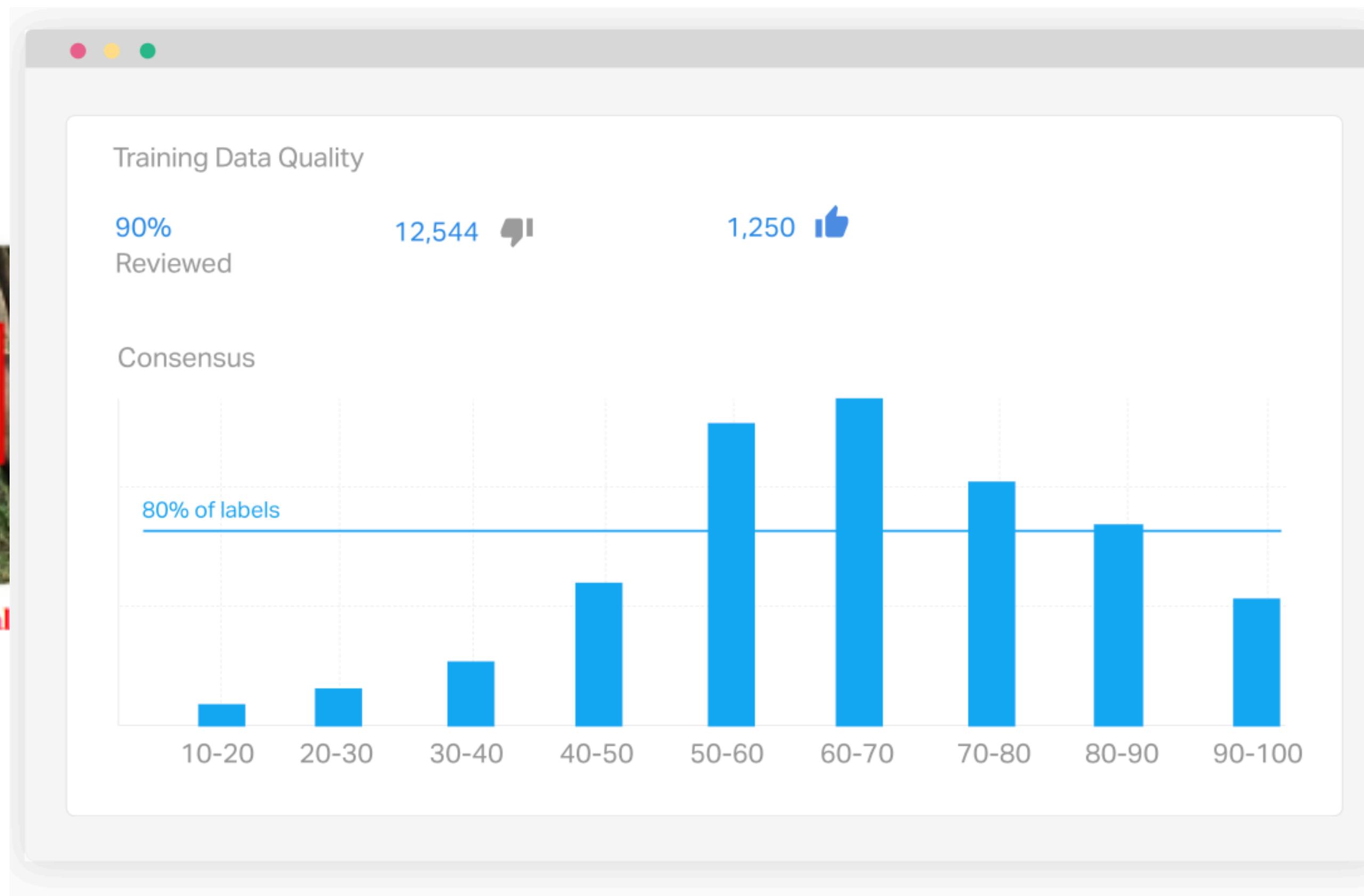
WRONG: must be as tight as possible!



WRONG: must include all parts!



WRONG: occluded parts do not matter as long as all visible parts are included.



Quality assurance is key

Source: cs.stanford.edu

Sources of Labor

- **Hire own annotators**, promote best ones to quality control
 - Pros: secure, fast (once hired), less QC needed
 - Cons: expensive, slow to scale, admin overhead
- ...or, crowdsource (**Mechanical Turk**)
 - Pros: cheaper, more scalable
 - Cons: not secure, significant QC effort required
- ...or, full-service data labeling companies

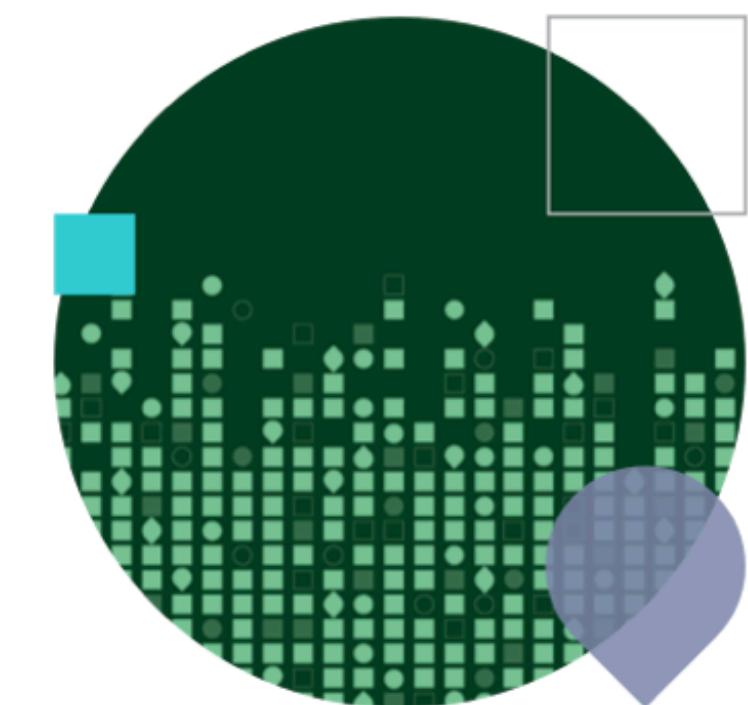
Service Companies

- Data labeling requires separate software stack, temporary labor, and quality assurance. **Makes sense to outsource.**
- Dedicate several days to selecting the best one for you:
 - Label **gold standard** data **yourself**
 - Sales calls with several contenders, ask for work sample on same data
 - Ensure agreement with your gold standard, and evaluate on value

FigureEight is the original AI data labeling company



Over 100 million images
labeled



Over 1 billion human
judgments in 2018



Over 10 years enabling
AI projects

Scale.ai is a dominant up-and-comer

The screenshot displays the Scale.ai web interface. At the top, a navigation bar includes tabs for SELF DRIVING CARS (highlighted in blue), DRONES, ROBOTICS, AR & VR, and RETAIL. Below the navigation bar are seven circular icons representing different labeling types: Sensor Fusion (purple), Video (yellow), Semantic Segmentation (pink), Cuboids (teal), Polygons (orange), 2D Boxes (green), and Lines & Splines (blue). A callout bubble contains the instruction: "Please label all cars, pedestrians, and cyclists in each frame." To the left, a code snippet shows how to create a lidar annotation task using the client library:

```
client.createLidarAnnotationTask({
  'instruction': 'Please label all cars, pedestrians, and
  cyclists in each frame.',
  'labels': ['car', 'pedestrian', 'cyclist'],
  'meters_per_unit': 2.3,
  'max_distance_meters': 30
}, (err, task) => {
  // do something with task
});
```

Below the code are two buttons: RUN CODE and READ MORE.

To the right, there is a large black rectangular area representing a camera feed or simulation screen. It features a green circular trackball control in the bottom right corner and a set of white directional keys (W, A, S, D) at the bottom center. A small loading icon is visible in the lower-left corner of the black area.

And there are a ton of others

Labelbox

Product Solutions Company Pricing Support Sign in ▶

NEW Introducing One-click Outsourcing

The best way to create and manage training data

Labelbox is a collaborative training data platform for artificial intelligence applications.

[CREATE ACCOUNT](#) [CONTACT SALES](#)



A stylized illustration showing a team of people working on AI training data. On the left, two individuals in maroon suits are writing on large yellow and orange charts. In the center, a person in a maroon suit is pointing at a computer screen displaying a neural network diagram. On the right, two individuals in yellow shirts are looking at a large orange chart. A dashed red arrow points from the charts to a neural network diagram in the upper right corner, symbolizing the connection between raw data and AI models.

CONDÉ NAST

AIRBUS

lytx

genius sports

SomaDetect

ARTURO

KEEP TRUCKIN

STOCKWELL

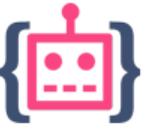
Aquabyte

nielsen

abundant ROBOTICS

neonode®

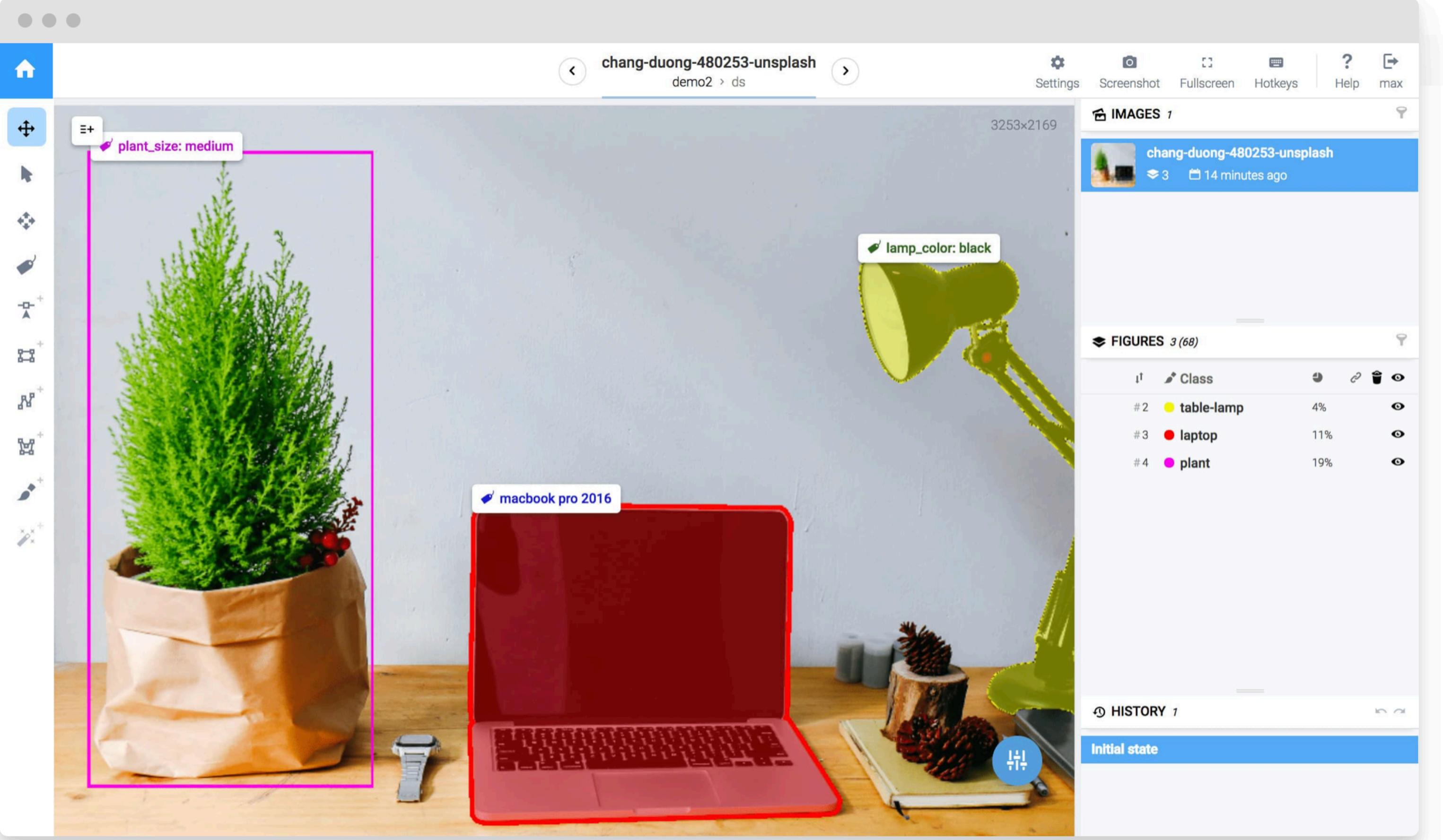
And there are a ton of others

 **SUPERVISELY**

Product Enterprise Pricing About Learn More Docs [Log in](#)

Leverage best in class annotation platform

BOUNDING BOX POLYGON PIXELWISE TAG HOTKEYS LABELING HISTORY



IMAGES 1

- chang-duong-480253-unsplash (3) 14 minutes ago

FIGURES 3 (68)

#	Class	Percentage
1	table-lamp	4%
2	laptop	11%
3	plant	19%

HISTORY 1

- Initial state

Software

- Full-service data labeling is always **pricy**
- But some companies offer their software without labor

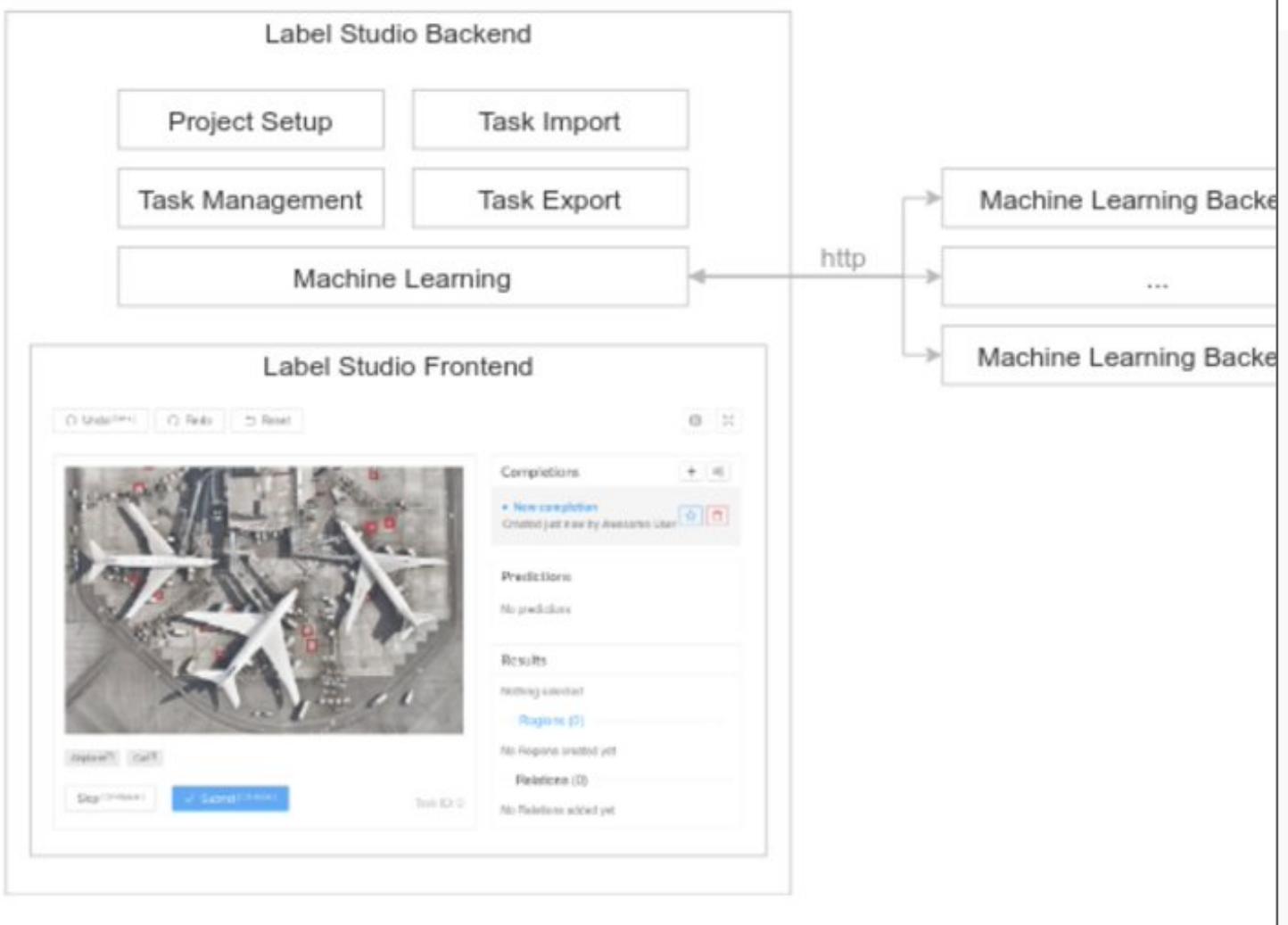
Label Studio

- Open-source edition to run yourself
- Enterprise edition for managed hosting
- Using in lab!

Main modules

The main modules of LS are

- Label Studio Backend (LSB, main repository)
- Label Studio Frontend (LSF, editor)
- Machine Learning Backends (MLB)



2. Edit Labeling config

```
1 <View>
2
3   <!-- Image with Polygons -->
4   <View style="padding: 25px;
5     box-shadow: 2px 2px 8px #AAA">
6     <Header value="Label the image with polygons"/>
7     <Image name="img" value="$image"/>
8     <Text name="text1"
9       value="Select label, start to click on image"/>
10
11   <PolygonLabels name="tag" toName="img">
12     <Label value="Airbus" background="blue"/>
13     <Label value="Boeing" background="red"/>
14   </PolygonLabels>
15
16 </View>
```

```
Multi-choices -->
  margin-top: 20px; padding: 25px;
  box-shadow: 2px 2px 8px #AAA;">
  'Classify the text'>
```

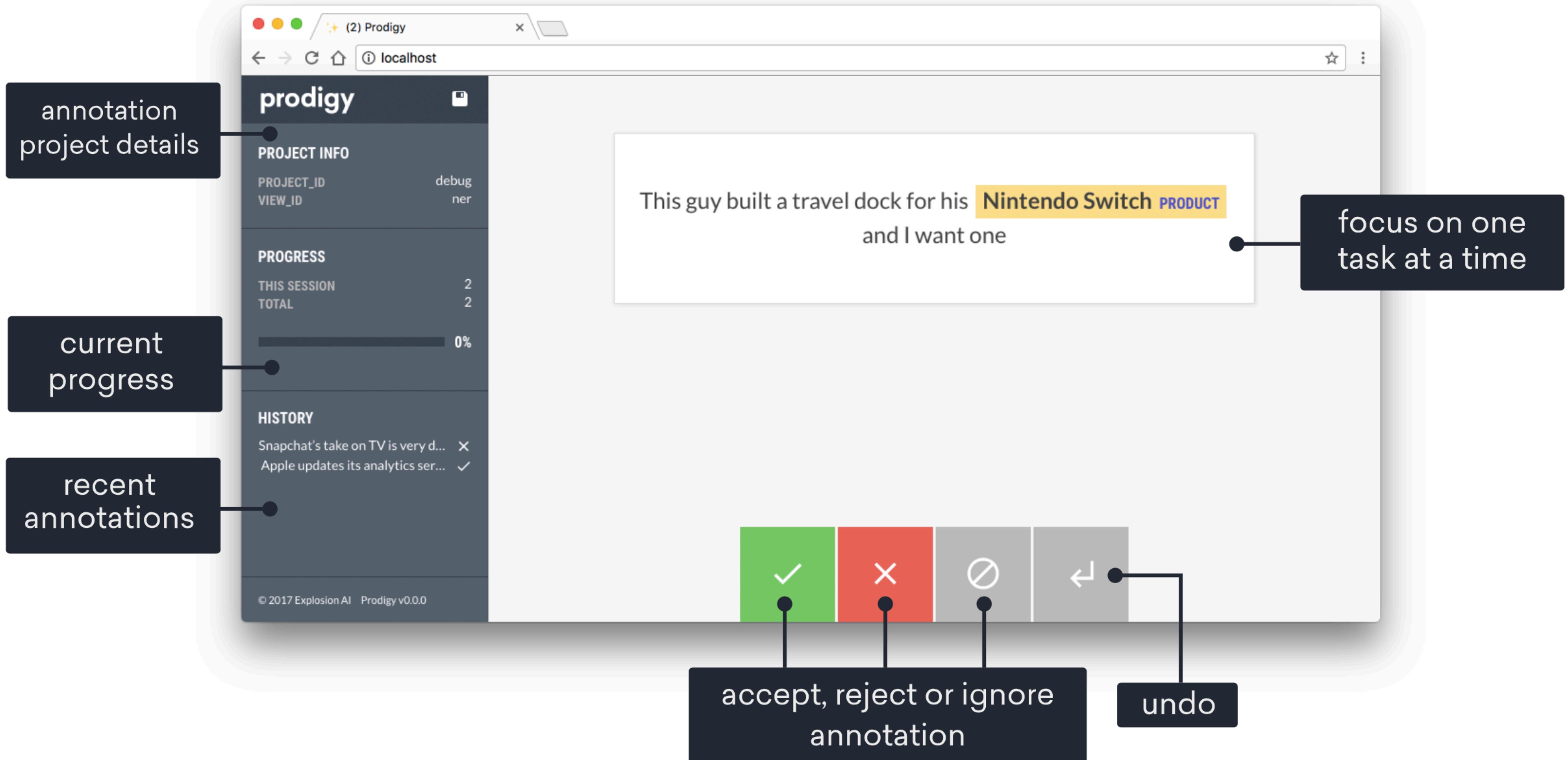
3. Inspect Interface preview

Label the image with polygons



The screenshot shows the Label Studio Frontend interface. At the top, there's a header with the Label Studio logo and tabs for Tasks, Model, Settings, Docs, and user icons. Below the header, a sidebar displays project details: Drone_dataset, Unlabeled, and a progress bar showing 14 / 14 tasks, 0 completions, and 0 predictions. The main area is a grid of 15 images, each with a checkbox and a number (e.g., 6, 11, 7, 8, 10, 13, 5, 1, 4, 12, 9, 0) in the top-left corner. The images are aerial views of outdoor scenes, such as buildings, roads, and people. The interface also includes buttons for Import, Export, and Label.

Prodigy



Aquarium

Better models through better data

Aquarium is an ML data management platform that helps you improve your models by improving your datasets.

Upload data

Aquarium's Python client API makes it easy to upload data, labels, and model inferences.

Explore your dataset

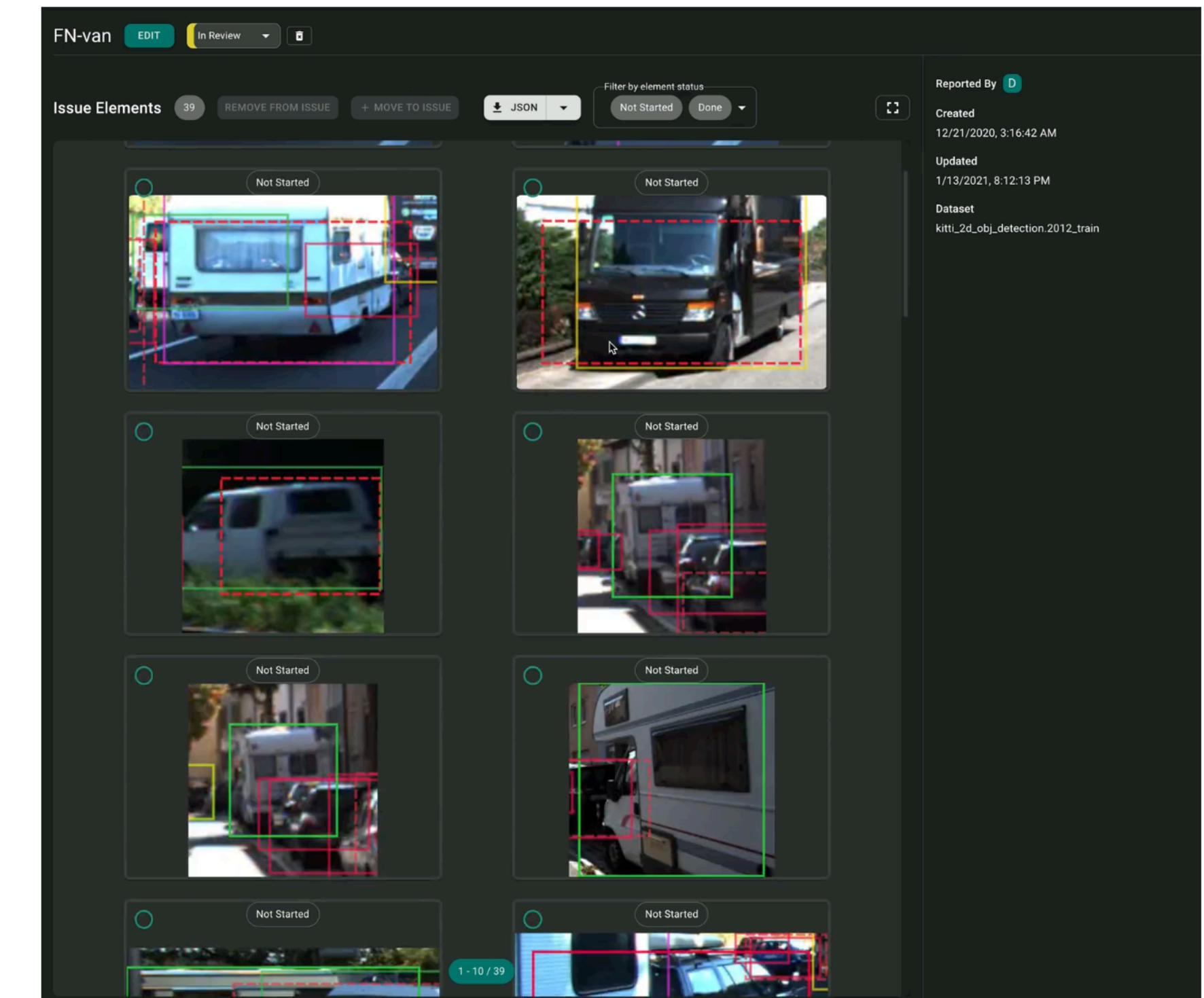
Visualize your dataset and model performance. Uncover corrupted data and difficult edge cases.

Curate data

Fix mislabeled examples and collect more examples of difficult edge cases. Aquarium's labeling service integrations lets you easily send data out for correction.

Model gets better!

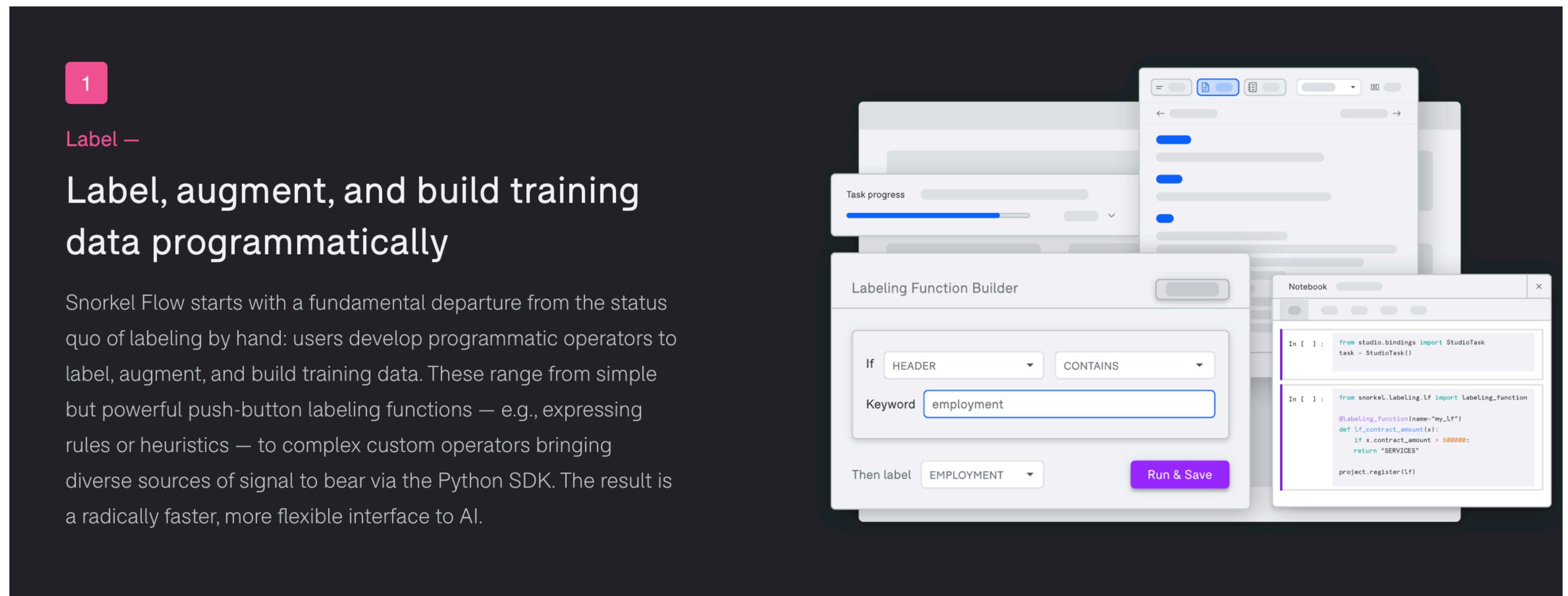
Retrain your model on the improved dataset. Use Aquarium to confirm that your new model is better. Success!



<https://www.aquariumlearning.com>

Weak supervision

- Snorkel
 - Open-source project snorkel.org
 - Commercial platform snorkel.ai



```
from snorkel.labeling import labeling_function

@labeling_function()
def lf_contains_link(x):
    # Return a label of SPAM if "http" in comment text, otherwise ABSTAIN
    return SPAM if "http" in x.text.lower() else ABSTAIN
```

- Conclusions
 - outsource to full-service company if you can afford it
 - if not, then at least use existing software
 - hiring part-time makes more sense than trying to make crowdsourcing work

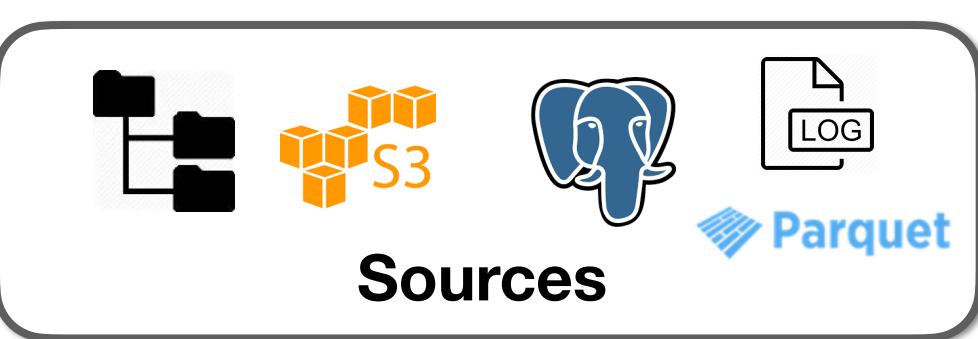
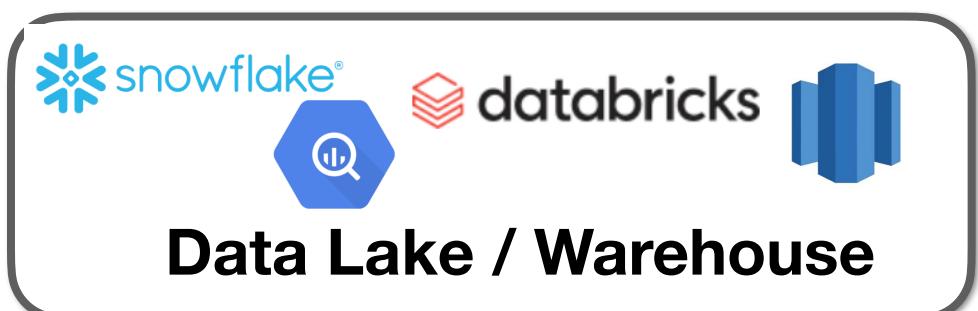
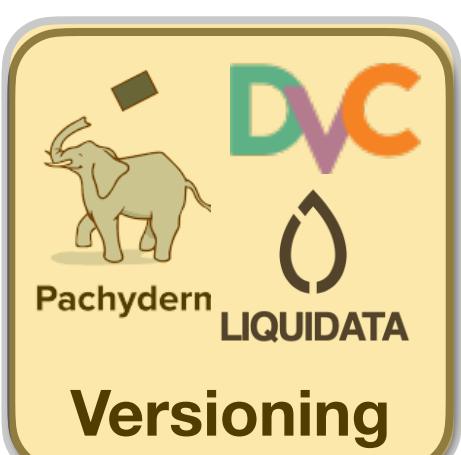
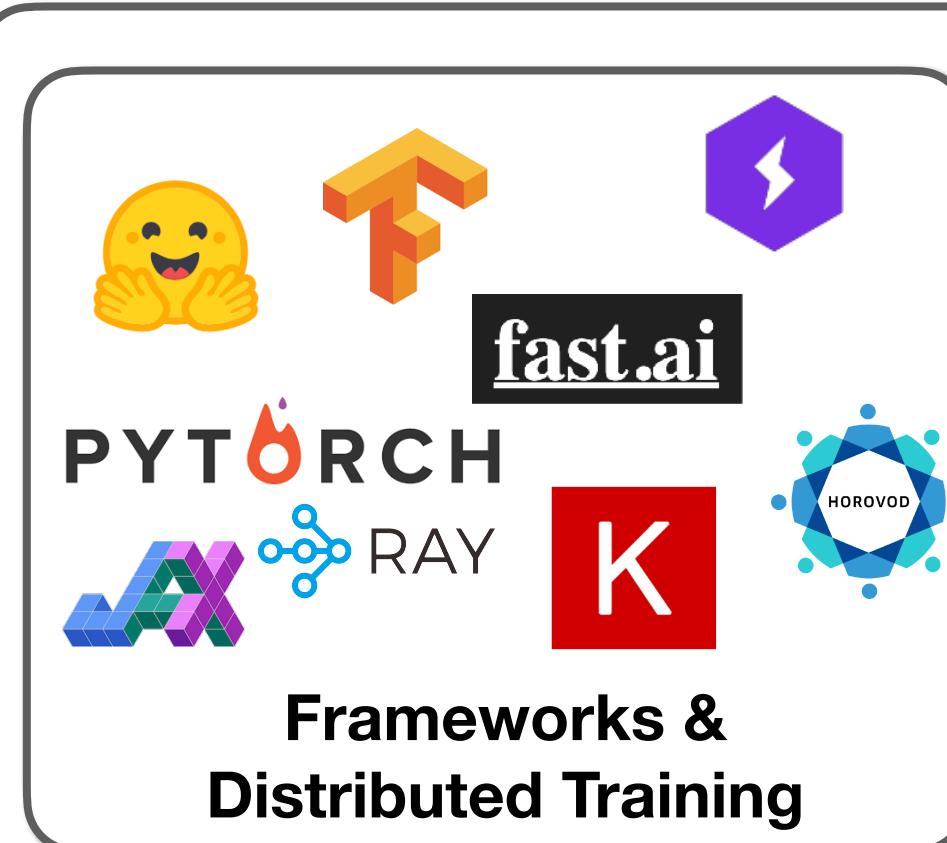
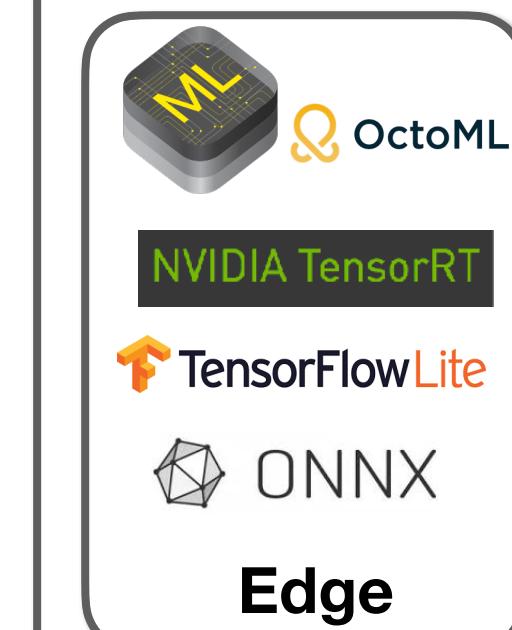
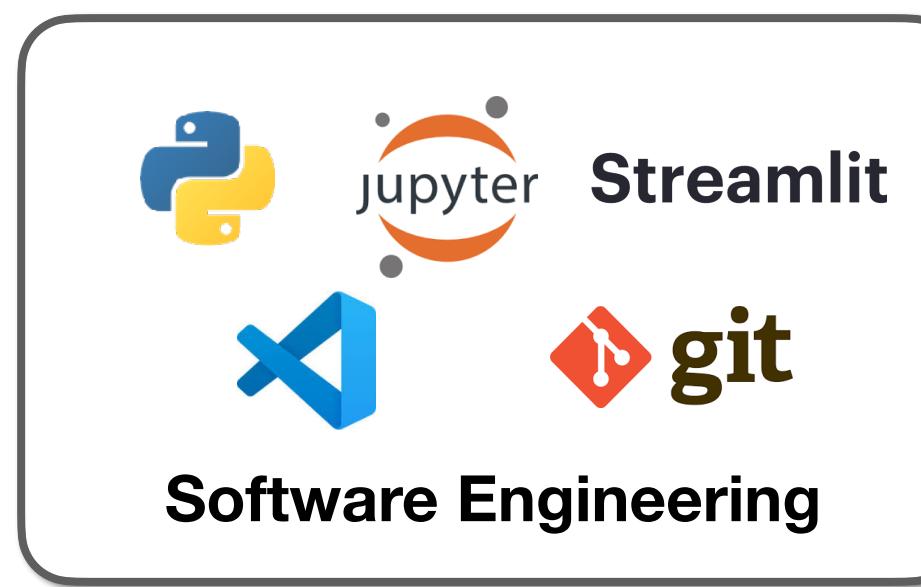
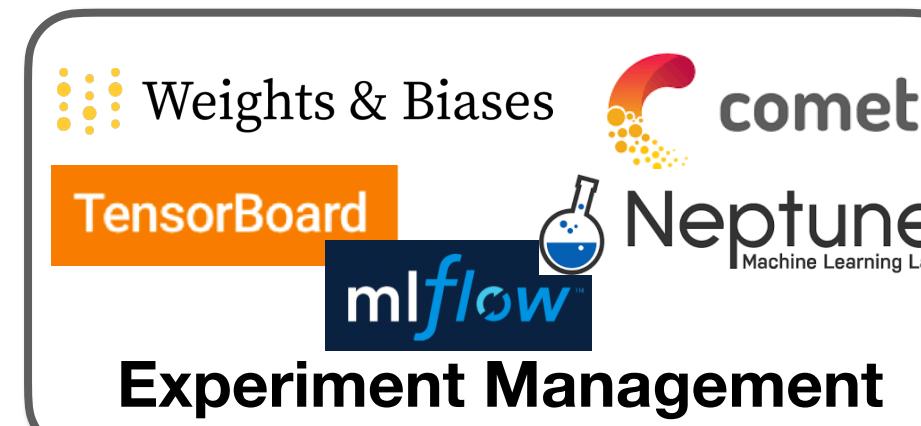
Questions?



Amazon SageMaker

gradient[®]
by Paperspace

FLOYD

DOMINO
DATA LAB**“All-in-one”****Data****Training/Evaluation****Deployment**

Data Versioning

Level 0: unversioned

Level 1: versioned via snapshot at training time

Level 2: versioned as a mix of assets and code

Level 3: specialized data versioning solution

Level 0



- Data lives are on filesystem/S3 and database
- **Problem:** Deployments must be versioned. Deployed machine learning models are part code, part data. If data is not versioned, deployed models are not versioned.
- Problem you **will** face: inability to get back to a previous level of performance

Level 1



- Data is versioned by storing a snapshot of everything at training time
- This allows you to version deployed models, and to get back to past performance, but is super hacky.
- Would be far better to be able to version data just as easily as code.



Level 2

- Data is versioned as a mix of assets and code.
- Heavy files stored in S3, with unique ids. Training data is stored as JSON or similar, referring to these ids and include relevant metadata (labels, user activity, etc).
- JSON files can get big, but using git-lfs lets us store them just as easily as code
 - Can improve further with "lazydata": only syncing files that are needed.
- The git signature + of the raw data file defines the version of the dataset
 - Often helpful to add timestamp



Level 3



- Specialized solutions for versioning data.
- Avoid these until you can fully explain how they will improve your project.
- Leading solutions are DVC, Pachyderm, Quill.

Data Versioning Solutions

	Open Source	Data Format Agnostic	Cloud/Storage Agnostic* (Supports most common cloud and storage types)	Simple to Use	Easy Support for BIG Data
 DVC	✓ (Apache 2.0)	✓	✓	✓	✗
 DELTA LAKE	✓ (Apache 2.0)	✗	✓	✗	✓
 Git Large File Storage	✓ (MIT)	✓	✗	✓	✗
 Pachyderm	✓⚠ (Non-standard license)	✓	✓	✗	✓
 DOLT	✓ (Apache 2.0)	✗	✗	✗	✗
 lakeFS	✓ (Apache 2.0)	✓	✗	✗	✓

 **DAGsHub**

<https://dagshub.com/blog/data-version-control-tools/>



DVC

Open-source Version Control System for Machine Learning Projects

2

```
$ dvc run \
  -d prepare.py -d data.xml \
  -o data.tsv -o data-test.tsv \
  python prepare.py data.xml
```

3 The first stage, feature extraction:

```
$ dvc run -d featurization.py -d data.tsv \
  -o matrix.pkl \
  python featurization.py data.tsv matrix.pkl
```

The second stage, training:

```
$ dvc run -d train.py -d matrix.pkl \
  -o model.pkl \
  python train.py matrix.pkl model.pkl
```

Let's commit meta-files that describe our pipeline:

```
$ git add .gitignore matrix.pkl.dvc model.pkl.dvc
$ git commit -m "add featurization and train steps to the pipeline"
```

1 \$ dvc add data.xml

DVC stores information about your data file in a special `.dvc` file, that has a human-readable [description](#) and can be committed to Git to track versions of your file:

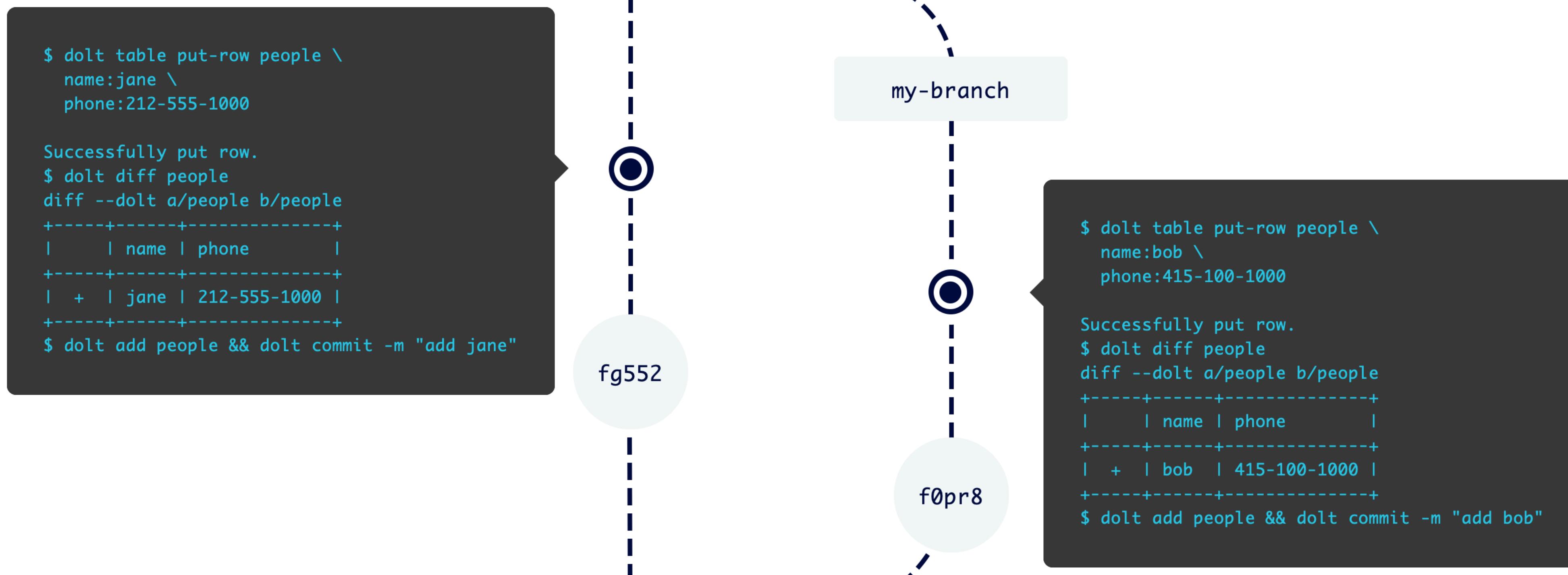
```
$ git add .gitignore data.xml.dvc
$ git commit -m "add source data to DVC"
```

4

```
$ dvc pipeline show --ascii model.pkl.dvc --outs
  -----
  | data.xml |
  '-----'
  **
  **          **
  **          **
  |
  | data.tsv |           | data-test.tsv |
  '-----'           '-----'
  **
  **          **
  **          **
  |
  | matrix.pkl |
  '-----'
  *
  *
  *
  |
  | model.pkl |
  '-----'
```

Dolt

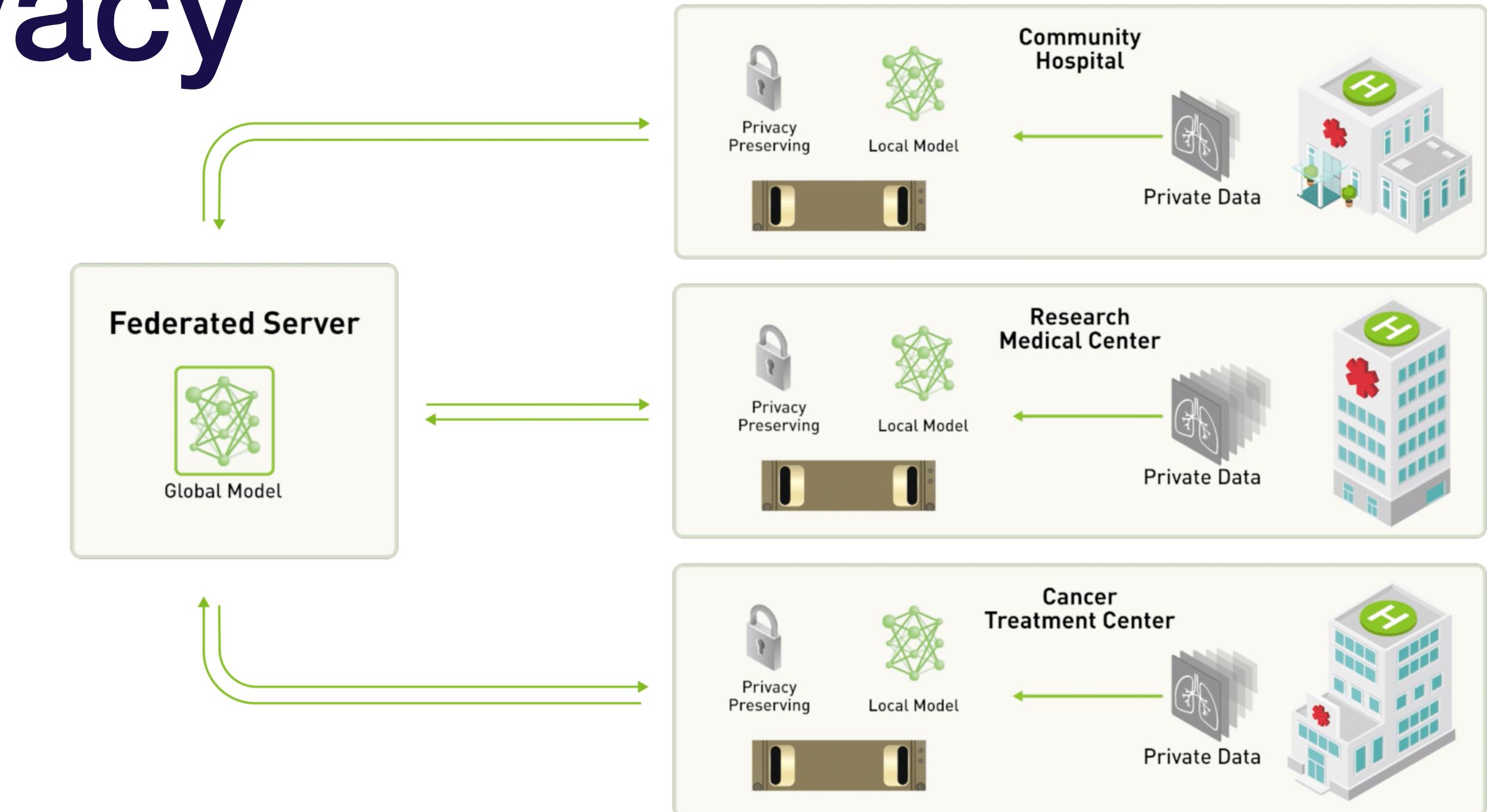
A nice simple solution for versioning databases, that speaks SQL.



Questions?

Privacy

- Federated Learning: training a global model from data on local devices, without ever having access to the data



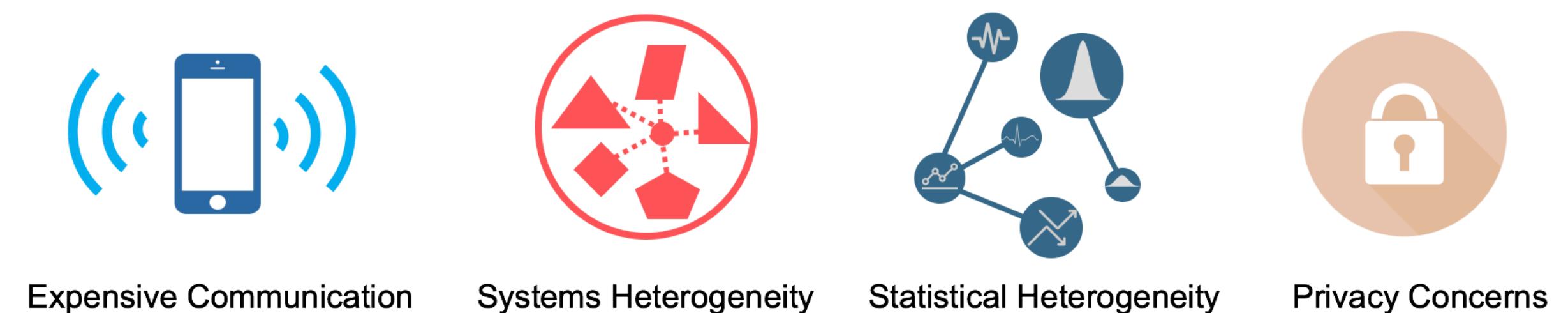
- Differential privacy: aggregating data such that individual points cannot be identified

- Another topic: Learning on encrypted data

- Let us know about the best resources!

<https://blog.ml.cmu.edu/2019/11/12/federated-learning-challenges-methods-and-future-directions/>

<https://blogs.nvidia.com/blog/2019/10/13/what-is-federated-learning/>



Thank you!