ML at Facebook: An Infrastructure View

Yangqing Jia

Director, Facebook AI Infra

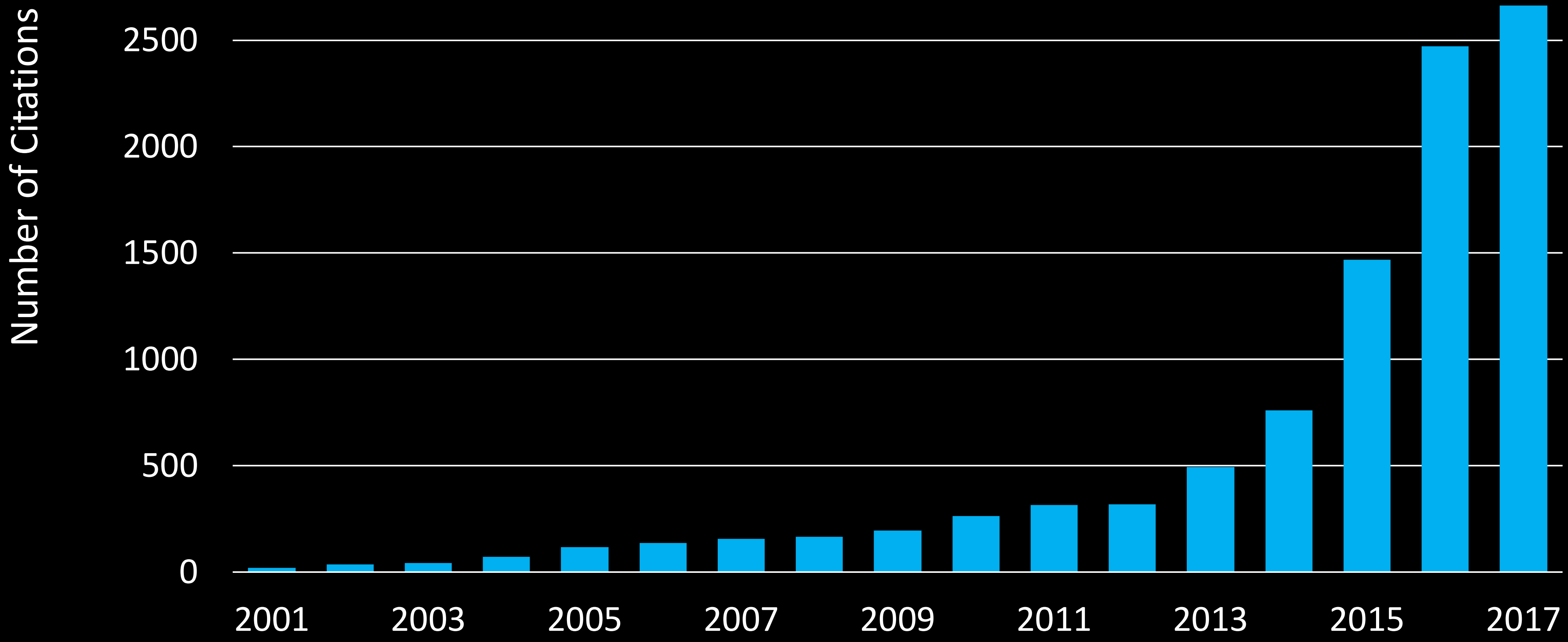
INFRAI

(* This is not
Facebook AI Infra)

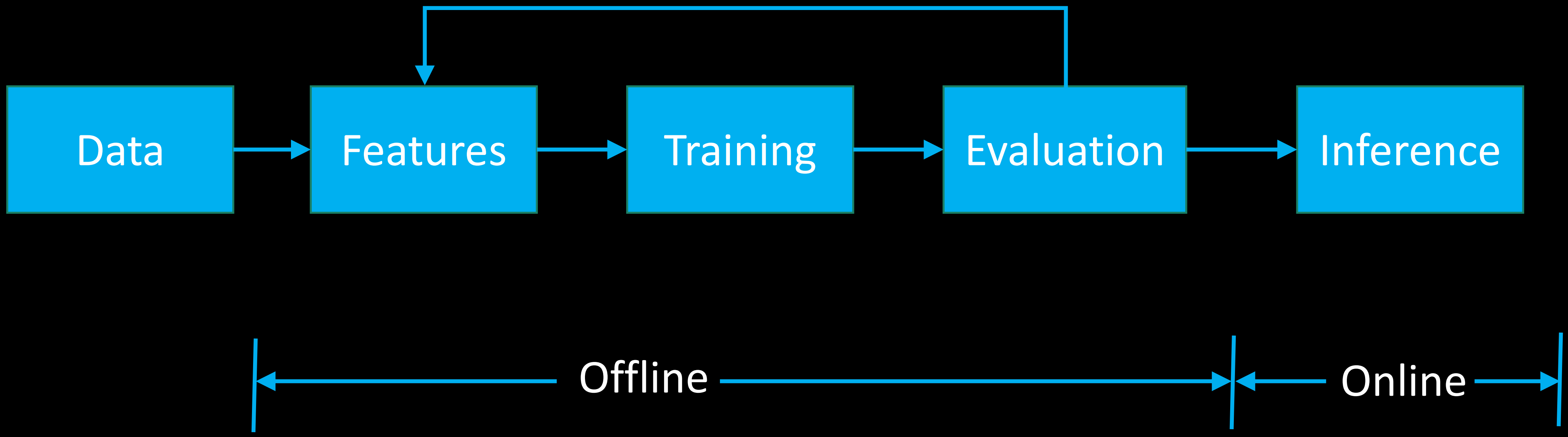
INFRAI

INFRA AI

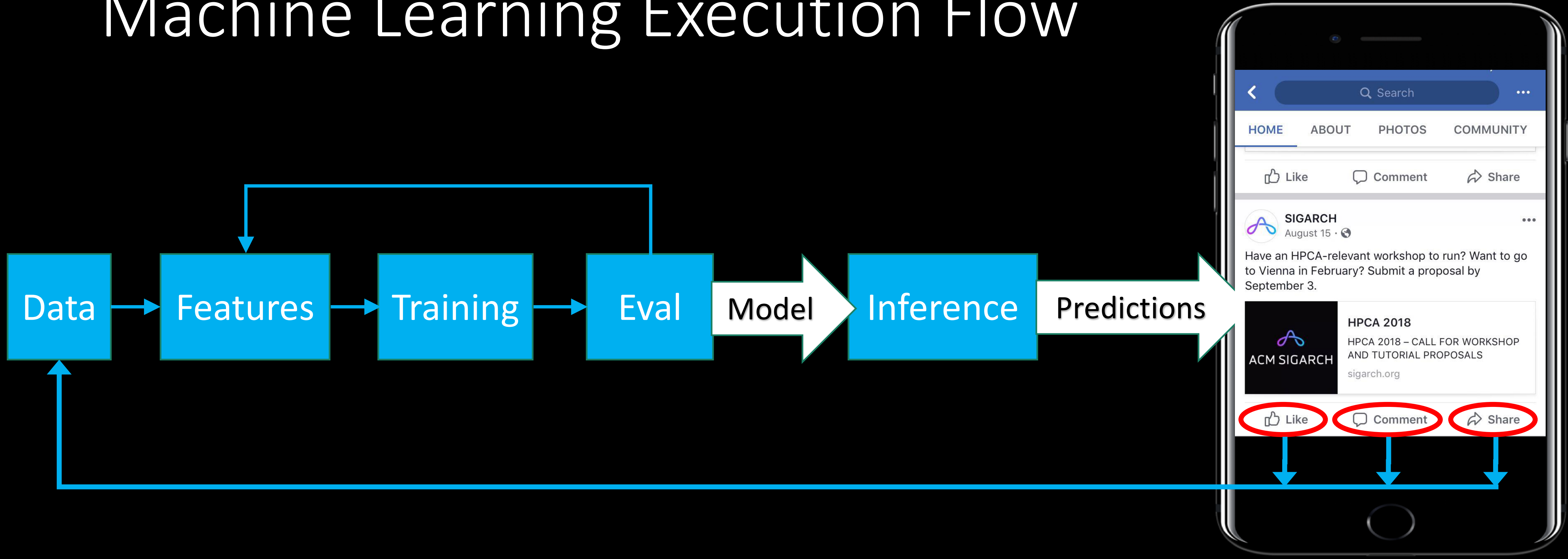
The Machine Learning Moore's Law?



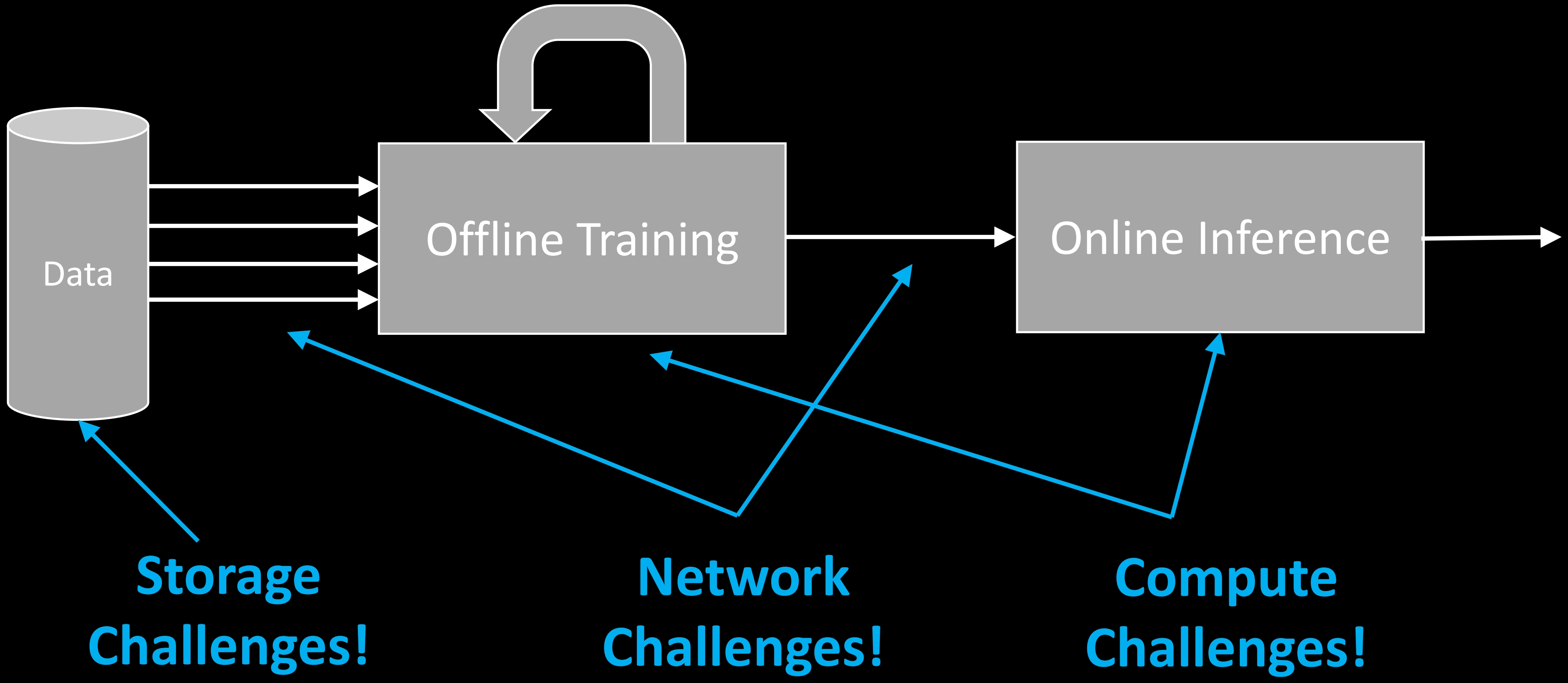
Machine Learning Execution Flow



Machine Learning Execution Flow



It’s an infrastructure challenge



Let’s Answer Some Pressing Questions

- Does Facebook leverage machine learning?
- Does Facebook design hardware?
- Does Facebook design hardware for machine learning?
- What platforms and frameworks exist; can the community use them?
- What assumptions break when supporting 2B people?

Does Facebook Use Machine Learning?

Classification & Ranking Services

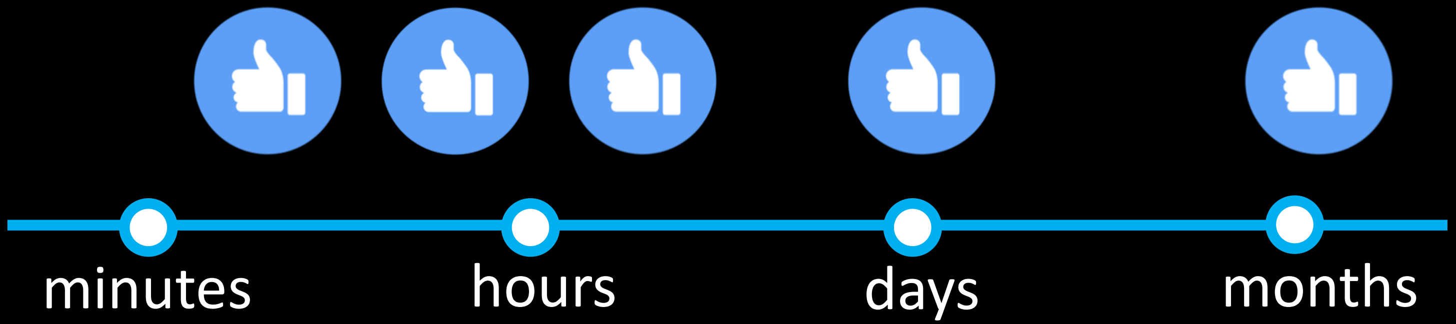
News Feed	Ads	Search
Sigma	Facer	Lumos
Language Translation	Speech Recognition	

Content Understanding

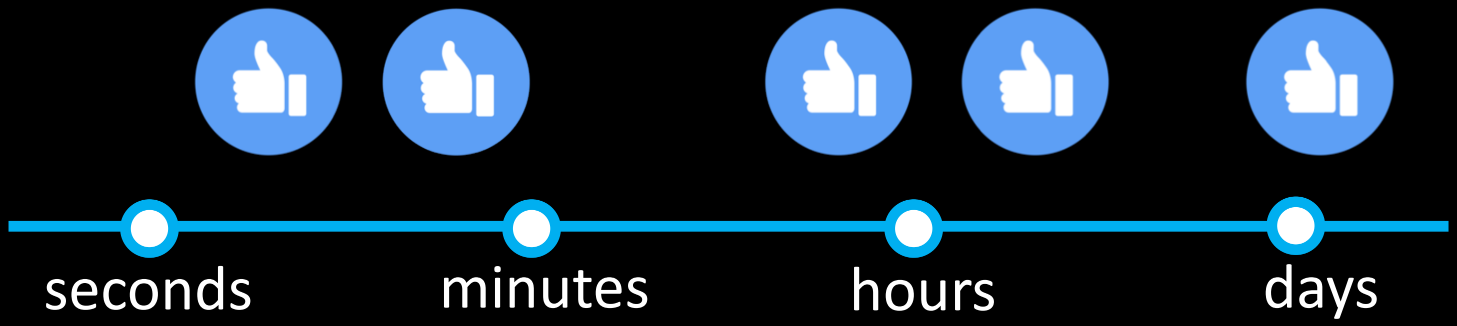
What ML Models Do We Leverage?

Support Vector Machines	Gradient-Boosted Decision Trees	Multi-Layer Perceptron	Convolutional Neural Nets	Recurrent Neural Nets
SVM	**GBDT**	**MLP**	**CNN**	**RNN**
Facer	Sigma	News Feed	Facer	Language Translation
		Ads	Lumos	Speech Rec
		Search		Content Understanding
		Sigma		

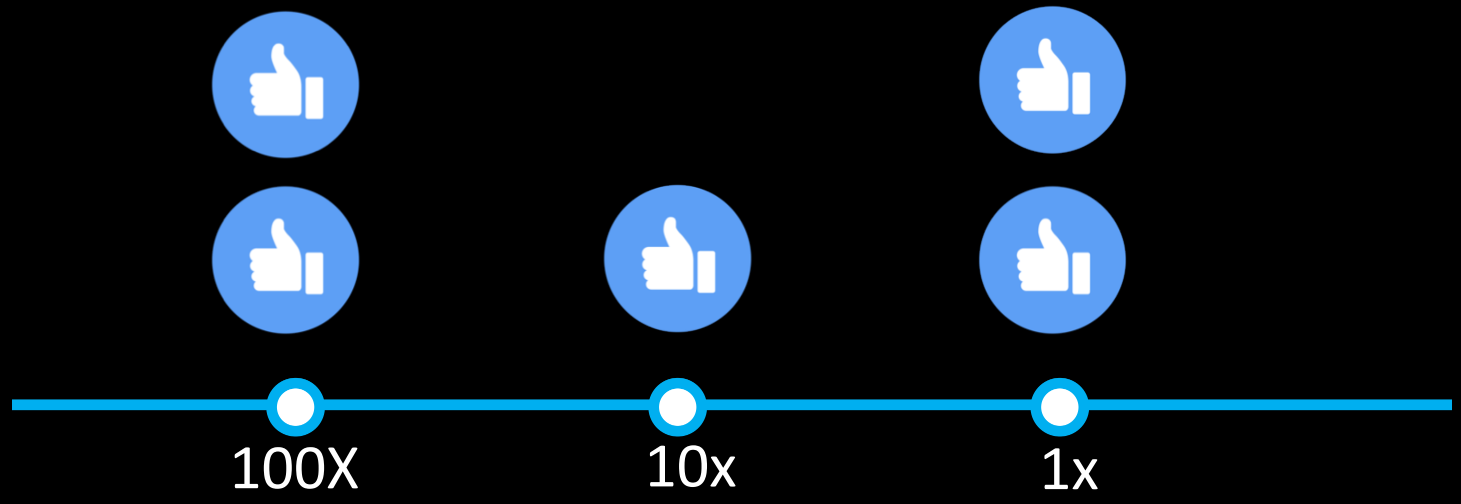
How Often Do We Train Models?



How Long Does Training Take?



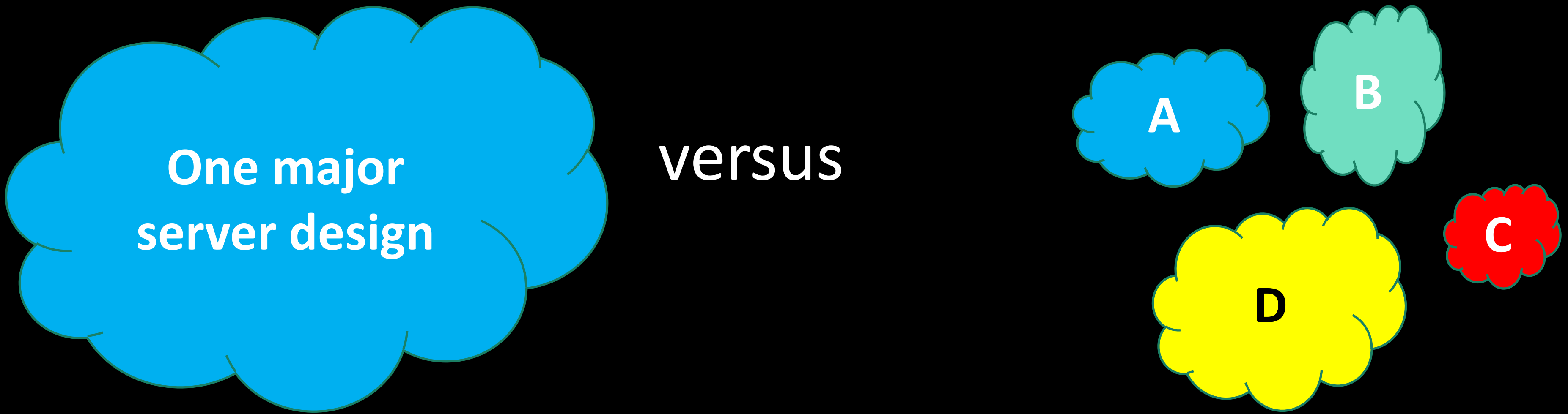
How Much Compute Does Inference Consume?



Does Facebook Design Hardware?

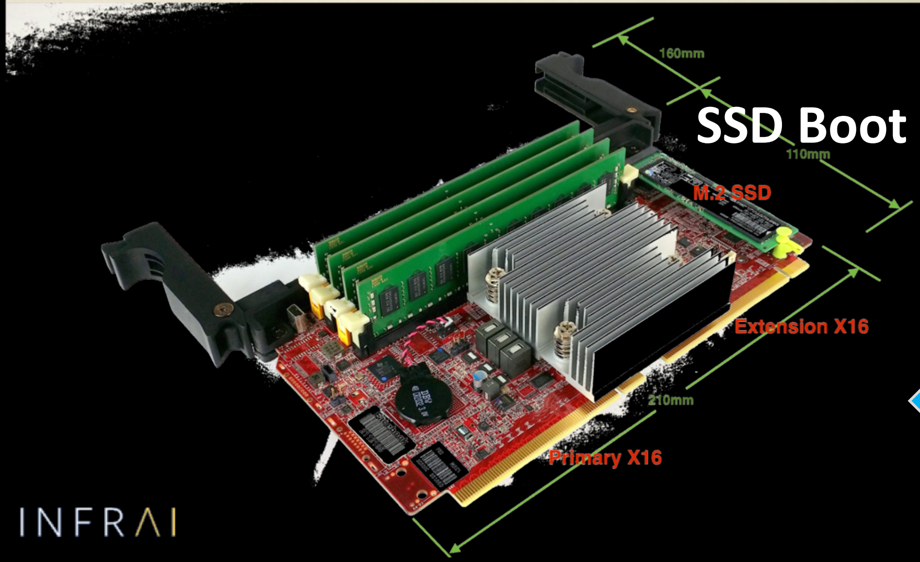
- Yes! Since 2010! All designs released through open compute!
- Facebook Server Design Philosophy
    - Identify a **small number** of **major services** with unique resource requirements
    - Design servers for those major services

Global shared pool **Customized, dedicated hardware**



Does Facebook Design Hardware?

4-way
Shared NIC



SSD Boot

Chassis

Server Card

Yosemite/Twin Lakes:

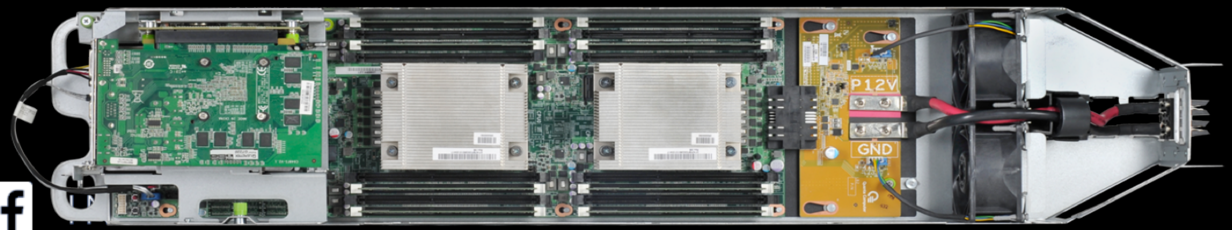
For the web tier and other "stateless services"

Open Compute "Sleds" are 2U x 3 Across in an Open Compute Rack

Rack

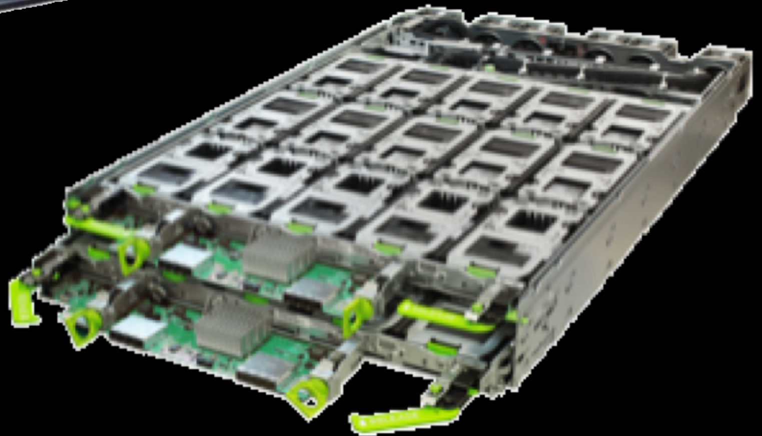
Does Facebook Design Hardware?

Tioga Pass:
For compute or memory-intensive workloads:



Bryce Canyon:

For storage-heavy workloads:



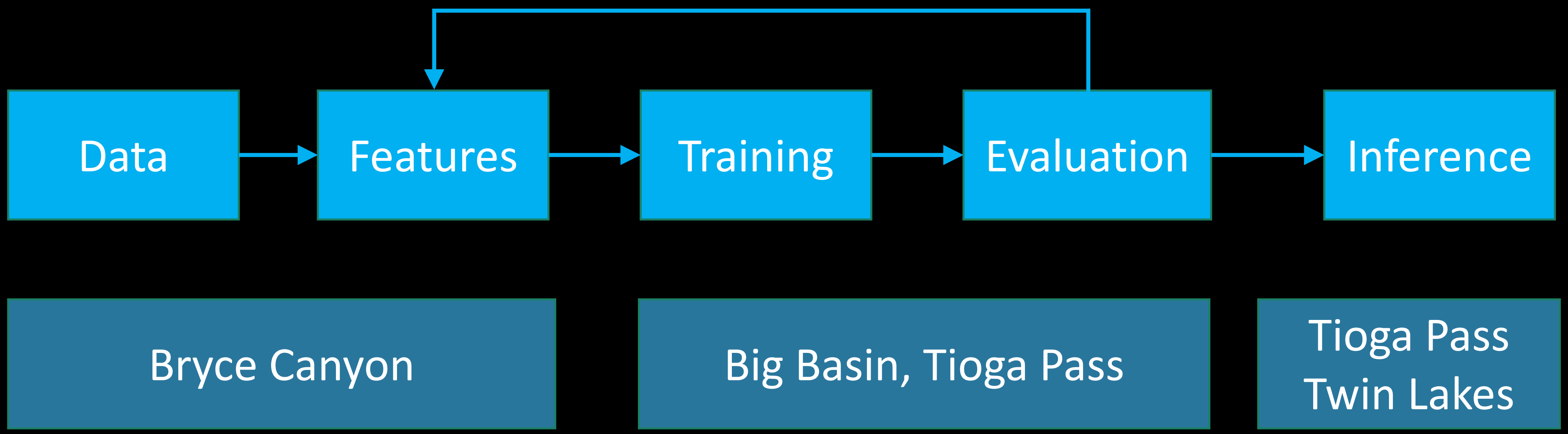
Does Facebook Design Hardware for AI/ML?

- HP SL270s (2013): learning serviceability, thermal, perf, reliability, cluster mgmt.
- Big Sur (M40) -> Big Basin (P100) -> Big Basin Volta (V100)

Big Sur
Integrated Compute
8 Nvidia M40 GPUs

Big Basin
JBOG Design (CPU headnode)
8 Nvidia P100 / V100 GPUs

Putting it Together



Let’s Answer Some Pressing Questions

- Does Facebook leverage machine learning?
- Does Facebook design hardware?
- Does Facebook design hardware for machine learning?
- What platforms and frameworks exist; can the community use them?
- What assumptions break when supporting 2B people?

Facebook AI Frameworks

Caffe2

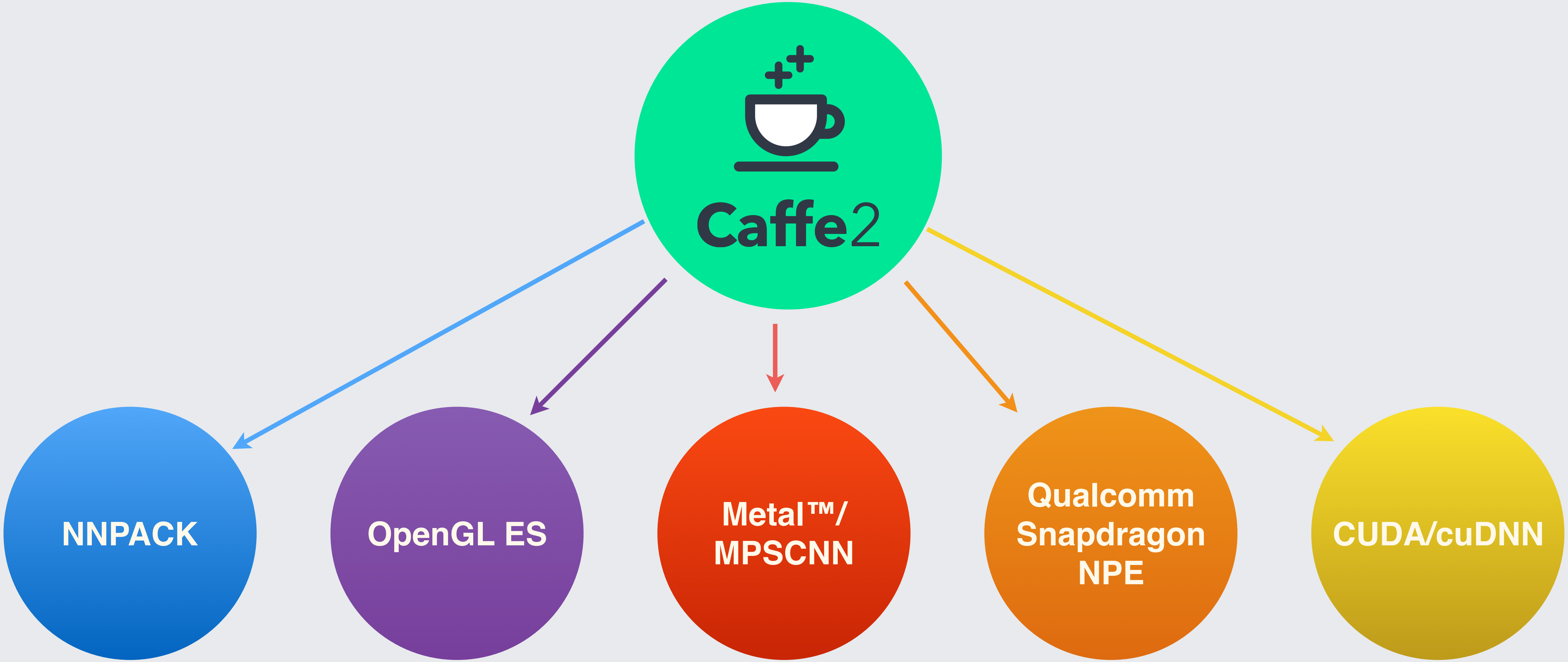
Infra Efficiency for Production

- Stability
- Scale & Speed
- Data Integration
- Relatively Fixed

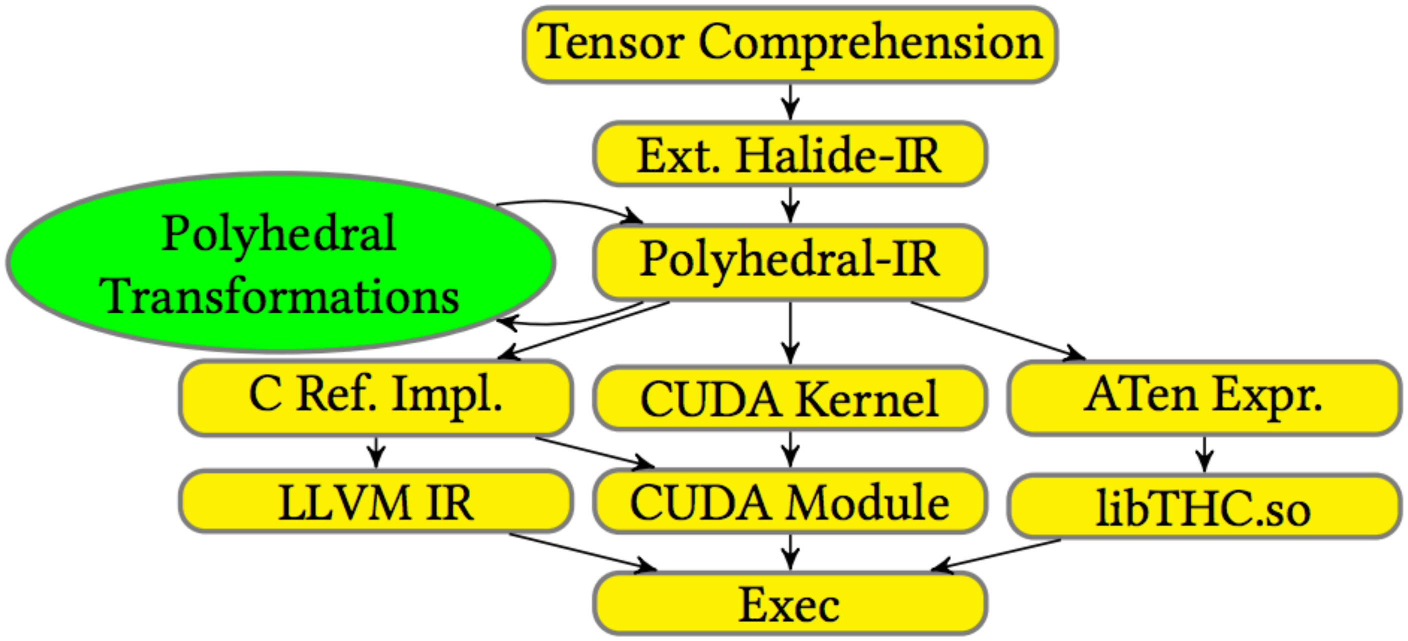
PYTORCH

Developer Efficiency for Research

- Flexible
- Fast Iteration
- Highly Debuggable
- Less Robust


Caffe2
NNPACK
OpenGL ES
Metal™/
MPSCNN
Qualcomm
Snapdragon
NPE
CUDA/cuDNN

```
def sgemm(float a, float b,
          float(N,M) A, float(M,K) B) → (C) {
  C(i,j)  = b * C(i,j)                # initialization
  C(i,j) += a * A(i,k) * B(k,j)       # accumulation
}
```



Tensor Comprehensions

```python
# -*- coding: utf-8 -*-
import numpy as np

# N is batch size; D_in is input dimension;
# H is hidden dimension; D_out is output dimension.
N, D_in, H, D_out = 64, 1000, 100, 10

# Create random input and output data
x = np.random.randn(N, D_in)
y = np.random.randn(N, D_out)

# Randomly initialize weights
w1 = np.random.randn(D_in, H)
w2 = np.random.randn(H, D_out)

learning_rate = 1e-6
for t in range(500):
    # Forward pass: compute predicted y
    h = x.dot(w1)
    h_relu = np.maximum(h, 0)
    y_pred = h_relu.dot(w2)

    # Compute and print loss
    loss = np.square(y_pred - y).sum()
    print(t, loss)

    # Backprop to compute gradients of w1 and w2 with respect to loss
    grad_y_pred = 2.0 * (y_pred - y)
    grad_w2 = h_relu.T.dot(grad_y_pred)
    grad_h_relu = grad_y_pred.dot(w2.T)
    grad_h = grad_h_relu.copy()
    grad_h[h < 0] = 0
    grad_w1 = x.T.dot(grad_h)

    # Update weights
    w1 -= learning_rate * grad_w1
    w2 -= learning_rate * grad_w2
```

Numpy

```python
import torch


dtype = torch.FloatTensor
# dtype = torch.cuda.FloatTensor # Uncomment this to run on GPU

# N is batch size; D_in is input dimension;
# H is hidden dimension; D_out is output dimension.
N, D_in, H, D_out = 64, 1000, 100, 10

# Create random input and output data
x = torch.randn(N, D_in).type(dtype)
y = torch.randn(N, D_out).type(dtype)

# Randomly initialize weights
w1 = torch.randn(D_in, H).type(dtype)
w2 = torch.randn(H, D_out).type(dtype)

learning_rate = 1e-6
for t in range(500):
    # Forward pass: compute predicted y
    h = x.mm(w1)
    h_relu = h.clamp(min=0)
    y_pred = h_relu.mm(w2)

    # Compute and print loss
    loss = (y_pred - y).pow(2).sum()
    print(t, loss)

    # Backprop to compute gradients of w1 and w2 with respect to loss
    grad_y_pred = 2.0 * (y_pred - y)
    grad_w2 = h_relu.t().mm(grad_y_pred)
    grad_h_relu = grad_y_pred.mm(w2.t())
    grad_h = grad_h_relu.clone()
    grad_h[h < 0] = 0
    grad_w1 = x.t().mm(grad_h)

    # Update weights using gradient descent
    w1 -= learning_rate * grad_w1
    w2 -= learning_rate * grad_w2
```

PyTorch

Andrej Karpathy @karpathy

Following

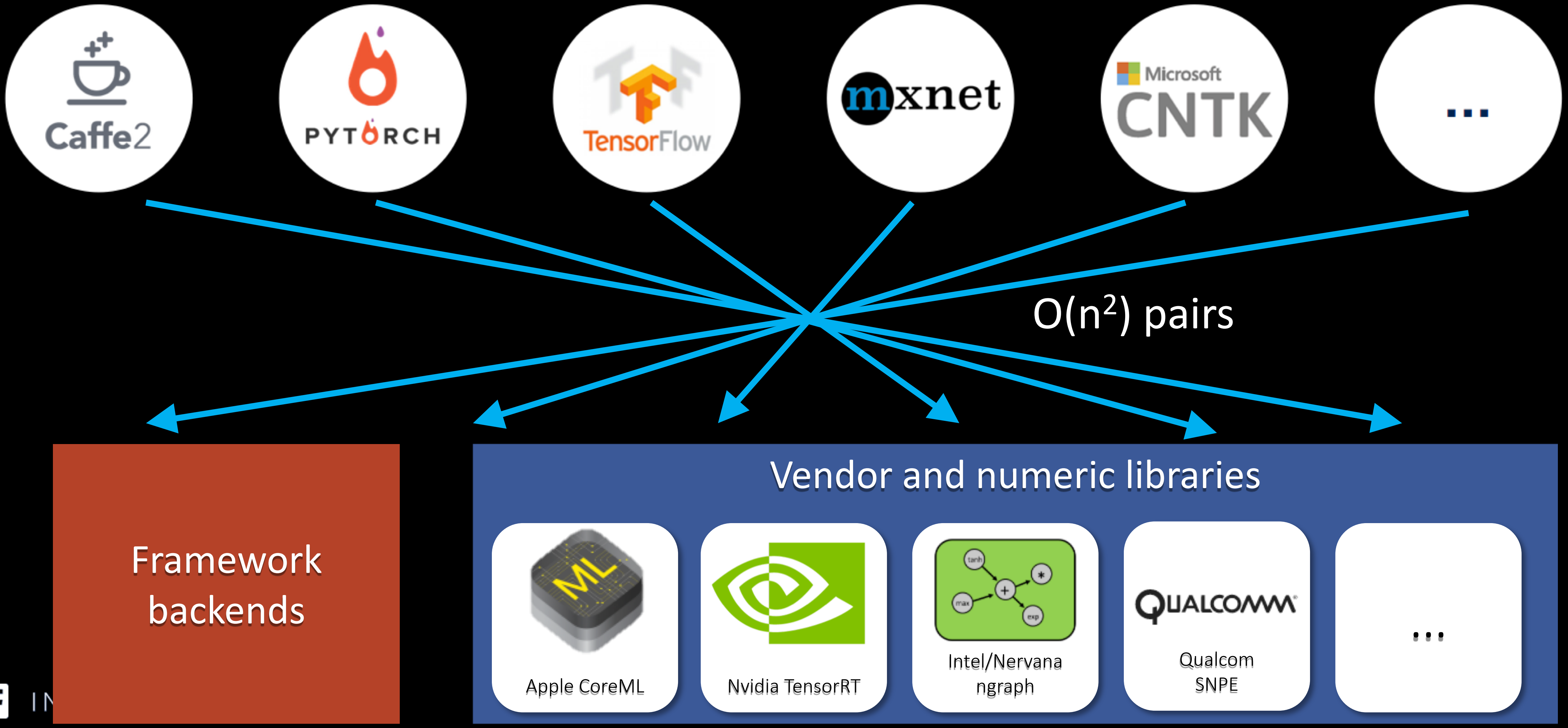
I've been using PyTorch a few months now and I've never felt better. I have more energy. My skin is clearer. My eye sight has improved.

Sean Robertson @sprobertson · 26 May 2017

Replying to @karpathy

Talk to your doctor to find out if PyTorch is right for you.

Deep Learning Frameworks



Open Neural Network Exchange

Caffe2

PYTORCH

TensorFlow

mxnet

Microsoft CNTK

...

ONNX

Shared model and operator representation

From $O(n^2)$ to $O(n)$ pairs

Framework backends

Vendor and numeric libraries

Apple CoreML

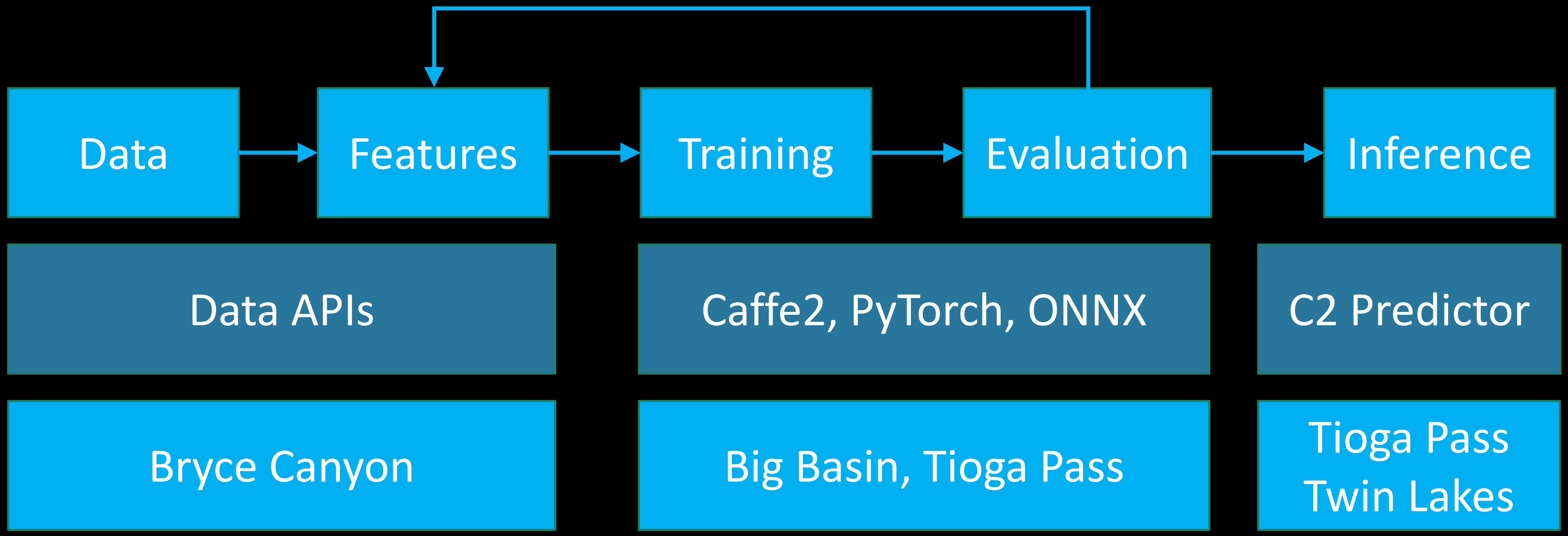
Nvidia TensorRT

Intel/Nervana ngraph

Qualcom SNPE

...

Putting it Together



Facebook AI Ecosystem

Frameworks: Core ML Software
Caffe2 / PyTorch / ONNX

Platforms: Workflow Management, Deployment
FB Learner

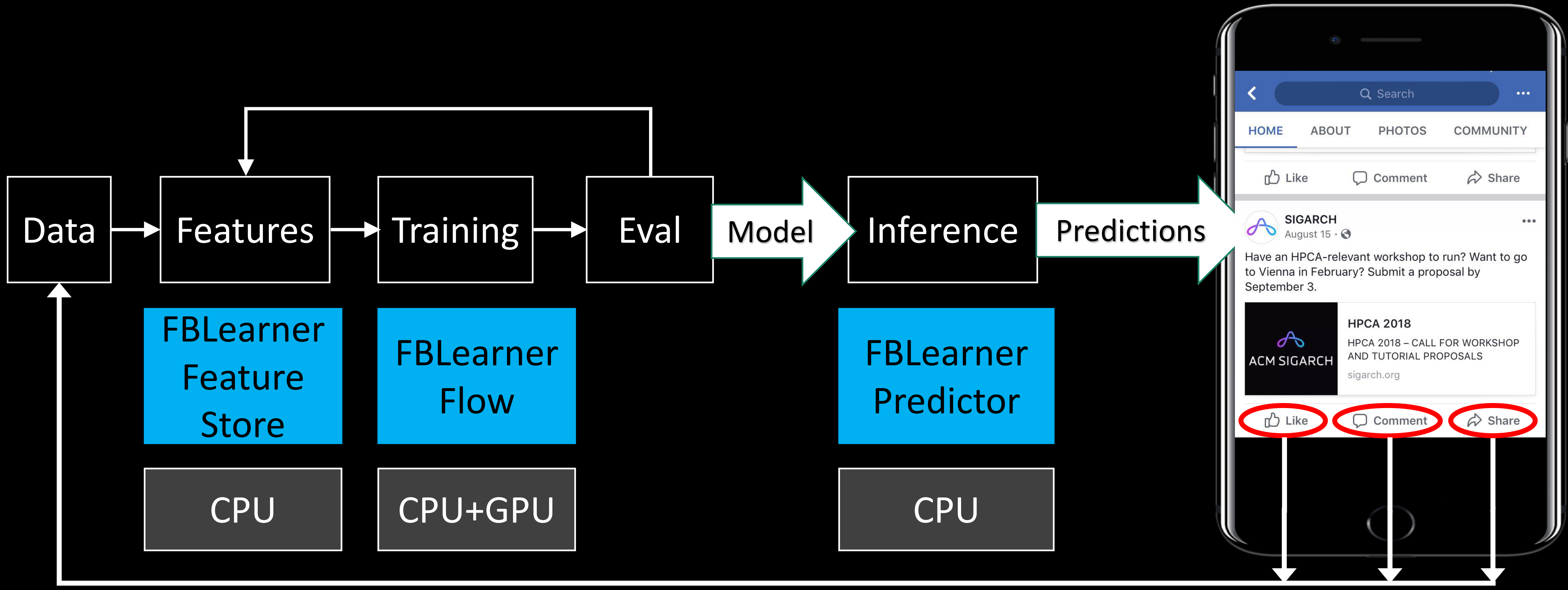
Infrastructure: Servers, Storage, Network Strategy
Open Compute Project

FB Learner Platform

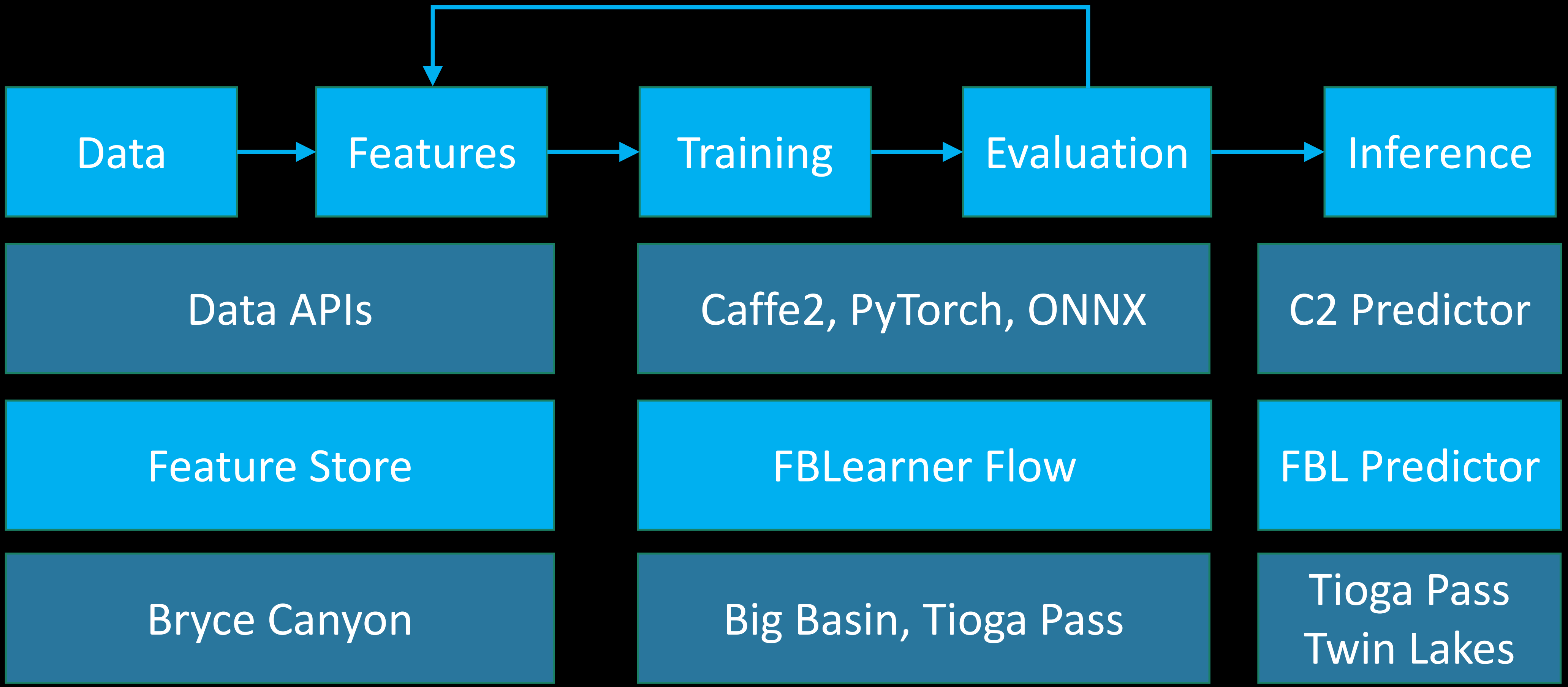
- AI Workflow
- Model Management and Deployment

FB Learner Feature Store → FB Learner Flow → FB Learner Predictor

FBLearner in ML

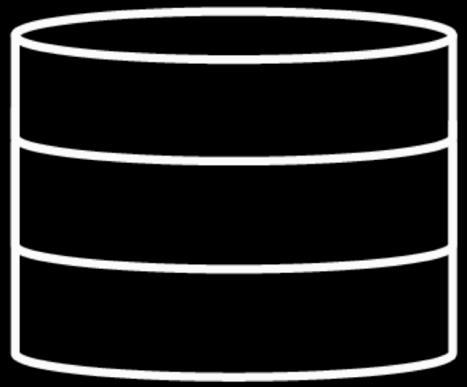


Putting it All Together

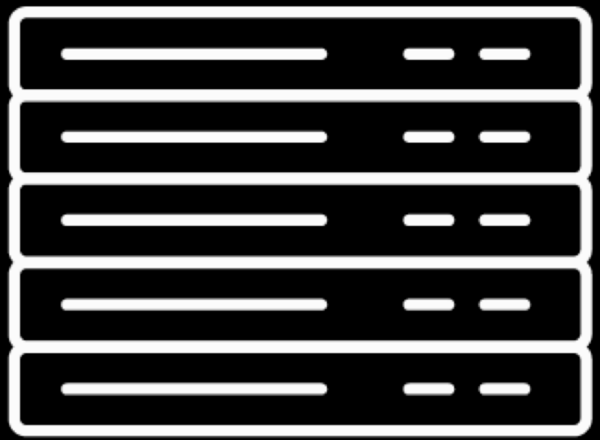



What changes when you scale to over
2 Billion People

Scaling Challenges / Opportunities

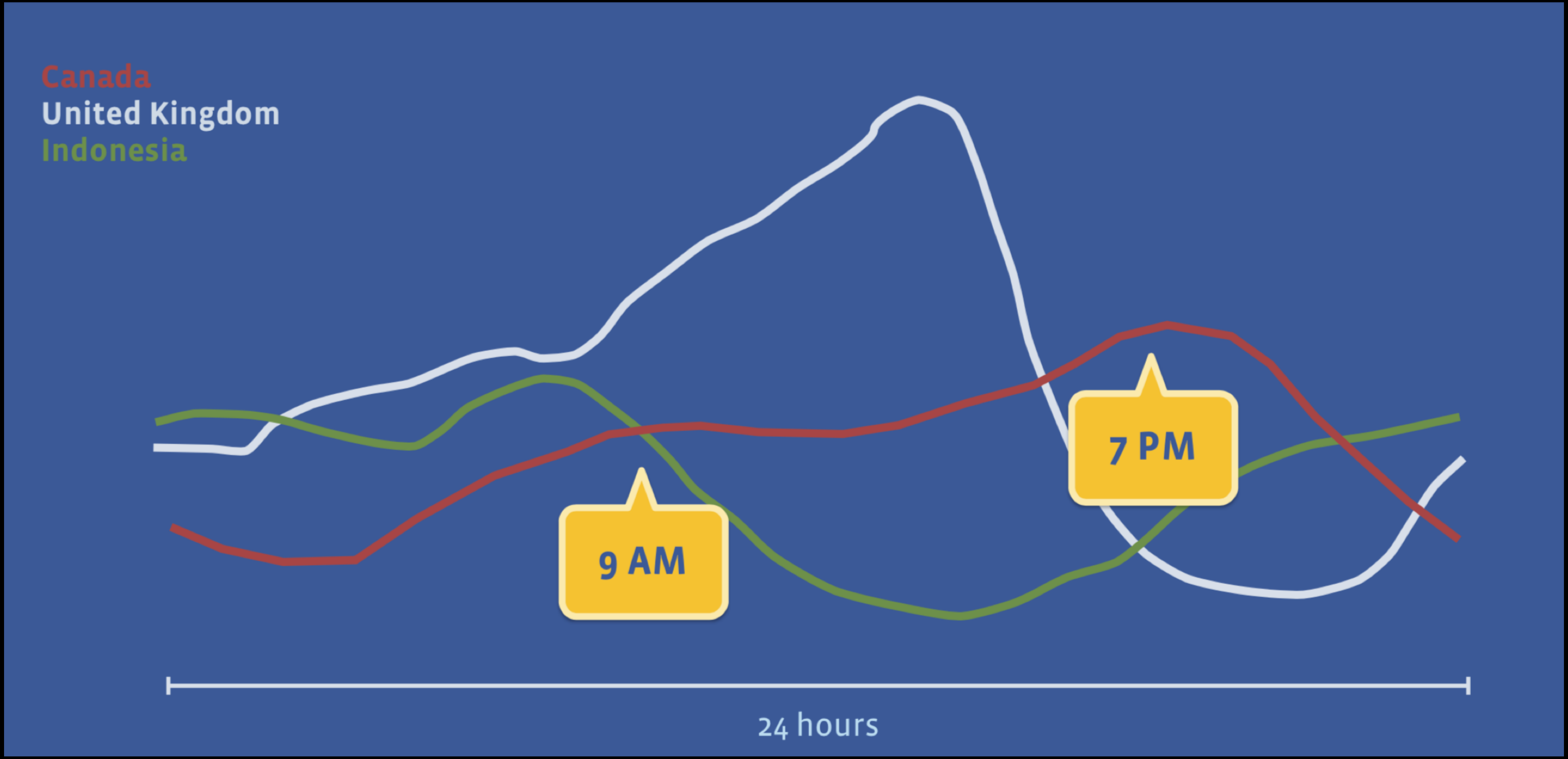


Lots of Data



Lots of Compute

Scaling Opportunity: Free Compute!



Santa Clara, California
Ashburn, Virginia
Prineville, Oregon
Forest City, North Carolina
Lulea, Sweden
Altoona, Iowa
Fort Worth, Texas
Clonee, Ireland
Los Lunas, New Mexico
Odense, Denmark
New Albany, Ohio
Papillion, Nebraska

Key Takeaways

Facebook AI

Lots of Data

Wide variety of models

Full stack challenges

Global scale


Kim Hazelwood

Sarah Bird

David Brooks

Soumith Chintala

Utku Diril

Dmytro Dzhulgakov

Mohamed Fawzy

Bill Jia

Yangqing Jia

Aditya Kalro

James Law

Kevin Lee

Jason Lu

Pieter Noordhuis

Misha Smelyanskiy

Liang Xiong

Xiaodong Wang

INFRAI