Prediction Serving

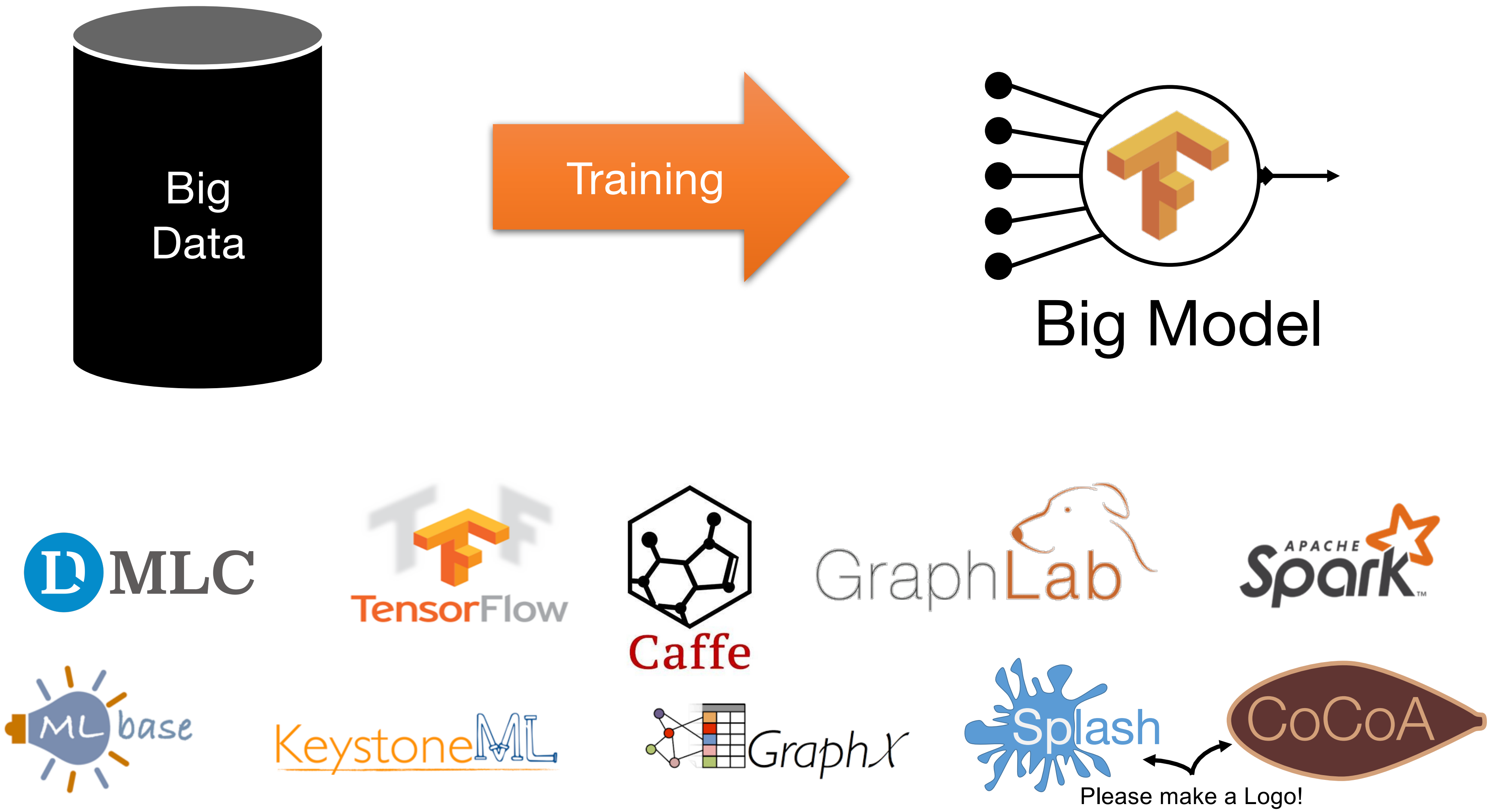
Joseph E. Gonzalez

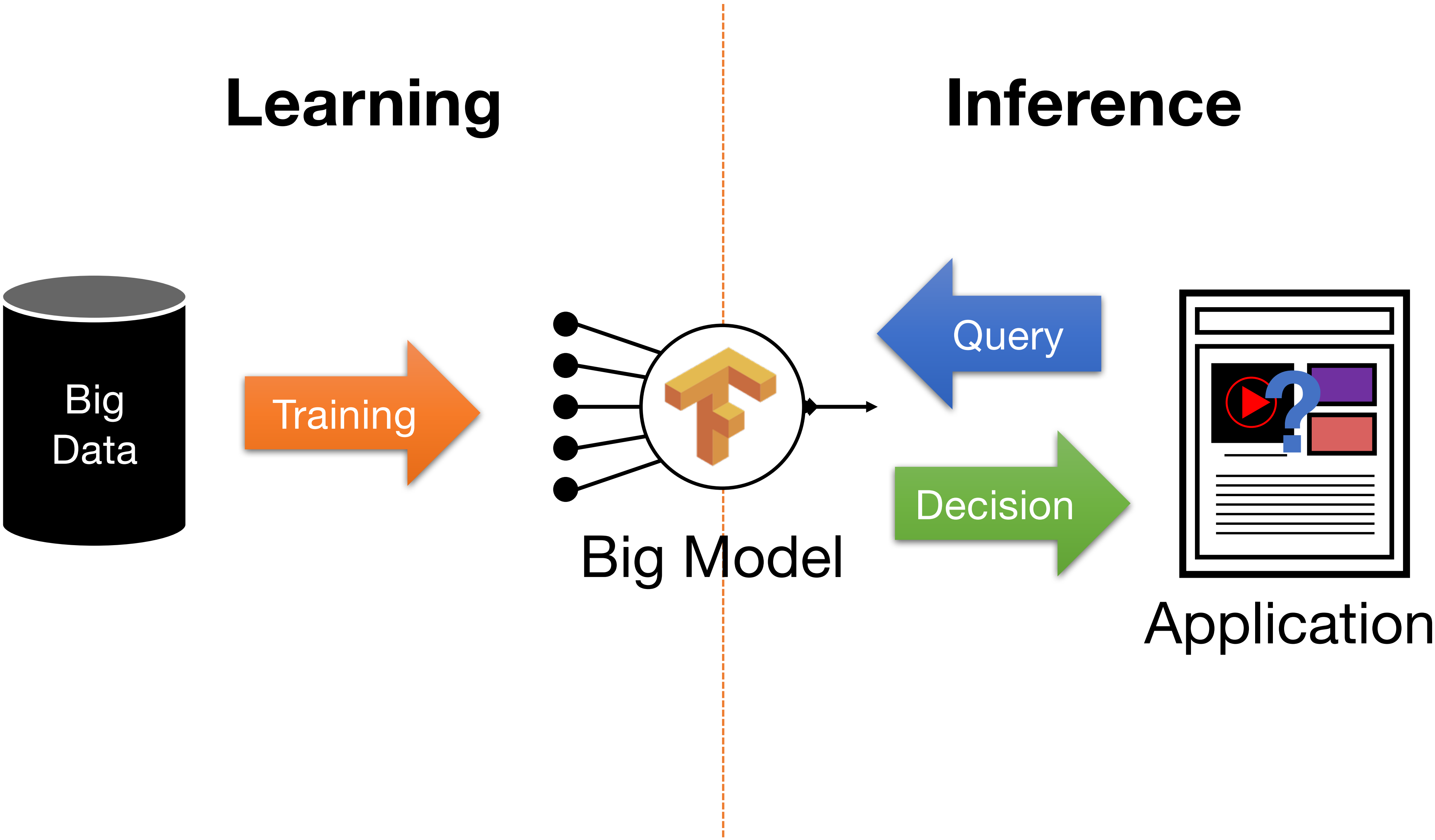
Asst. Professor, UC Berkeley

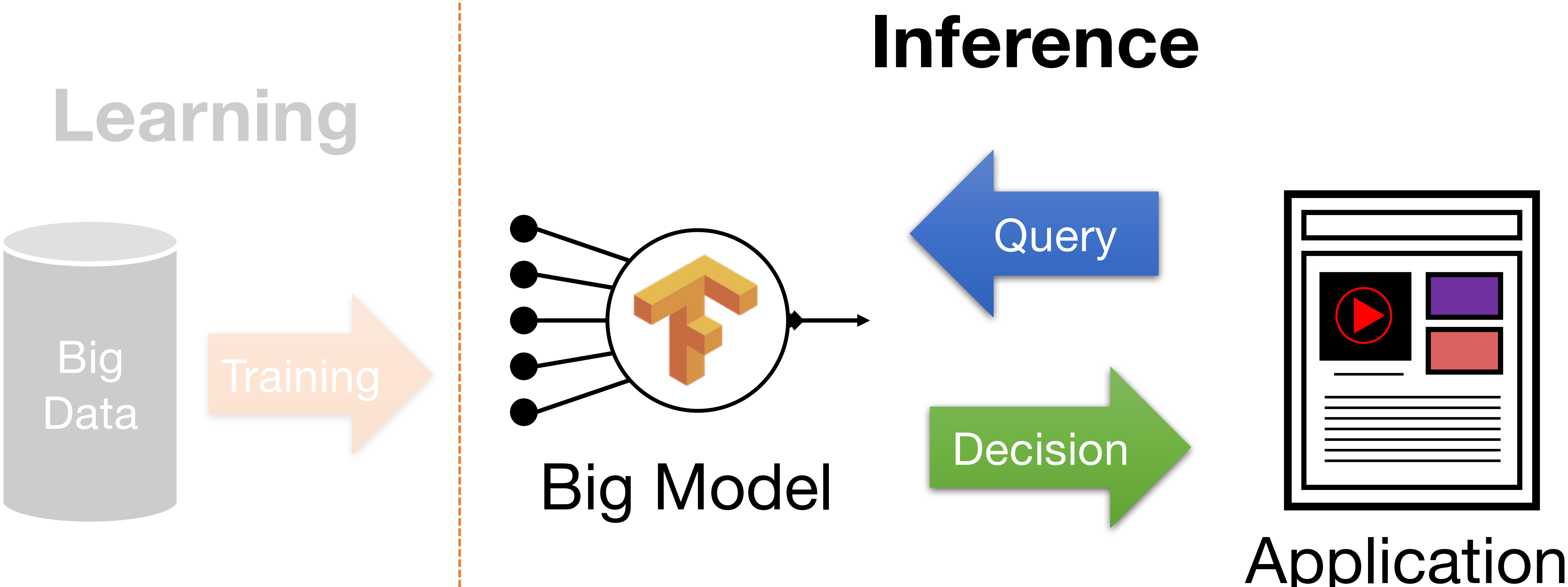
jegonzal@cs.berkeley.edu

Systems for Machine Learning

Timescale: minutes to days
Systems: offline and batch optimized
*Heavily studied ... primary focus of the **ML research***


Big Data
Training
Big Model
DMLC
TensorFlow
Caffe
GraphLab
APACHE Spark
MLbase
KeystoneML
GraphX
Splash
CoCoA
Please make a Logo!


Learning
Inference
Big
Data
Training
Big Model
Query
Decision
Application

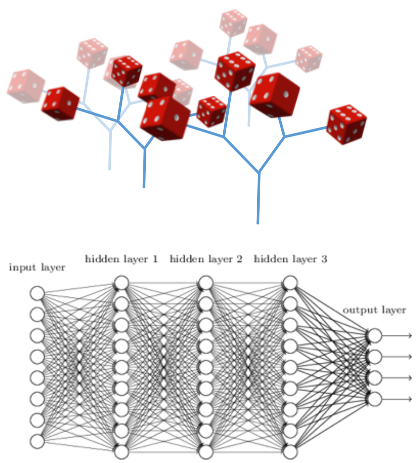


Timescale: ~10 milliseconds
Systems: *online* and *latency* optimized
Less Studied …

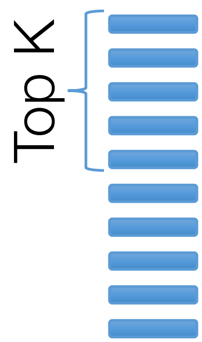
why is **Inference** challenging?

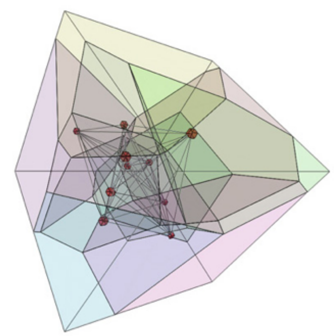
Need to render **low latency** (< 10ms) predictions for **complex**

Models



Queries





Features

SELECT * FROM
users JOIN items,
click_logs, pages
WHERE …

under **heavy load** with system **failures**.

Basic Linear Models (Often High Dimensional)

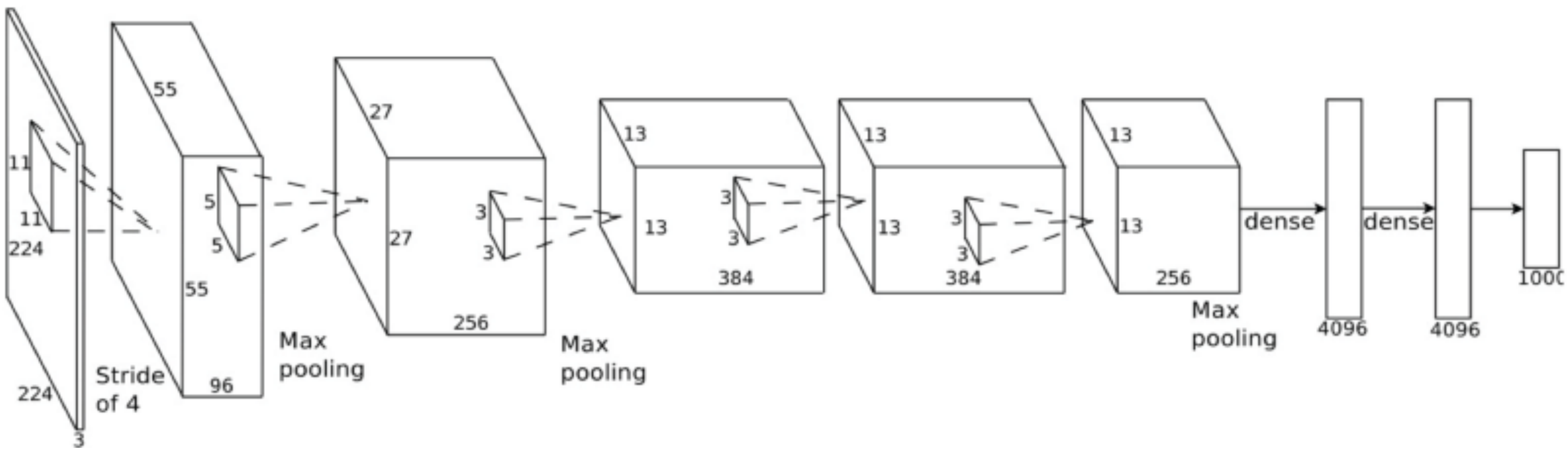
- Common for click prediction and text filter models (spam)
- Query *x* encoded in sparse Bag-of-Words:
  - x = “The quick brown” = {(”brown”, 1), (”the”, 1), (“quick”, 1)}
- Rendering a prediction:

$$\mathbf{Predict}(x) = \sigma\left(\sum_{(w,c)\in x} \theta_w c\right)$$

- θ is a large vector of weights for each possible word
  - or word combination (n-gram models) …
  - McMahan et al.: billions of coefficients

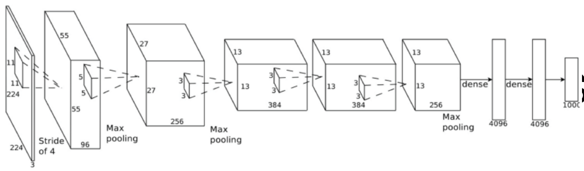
Computer Vision and Speech Recognition

- Deep Neural Networks (will cover in more detail later):



- 100's of millions of parameters + convolutions & unrolling
- Requires hardware acceleration

Computer Vision and Speech Recognition

- 

(will cover in more detail later):

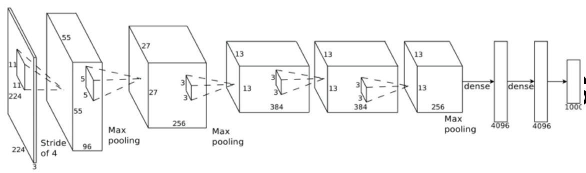
Network: GoogLeNet	Batch Size	Titan X (FP32)	Tegra X1 (FP32)	Tegra X1 (FP16)
Inference Performance	1	138 img/sec	33 img/sec	33 img/sec
Power		119.0 W	5.0 W	4.0 W
Performance/Watt		1.2 img/sec/W	6.5 img/sec/W	8.3 img/sec/W
Inference Performance	128 (Titan X) 64 (Tegra X1)	863 img/sec	52 img/sec	75 img/sec
Power		225.0 W	5.9 W	5.8 W
Performance/Watt		3.8 img/sec/W	8.8 img/sec/W	12.8 img/sec/W

Table 3 GoogLeNet inference results on Tegra X1 and Titan X. Tegra X1's total memory capacity is not sufficient to run batch size 128 inference.

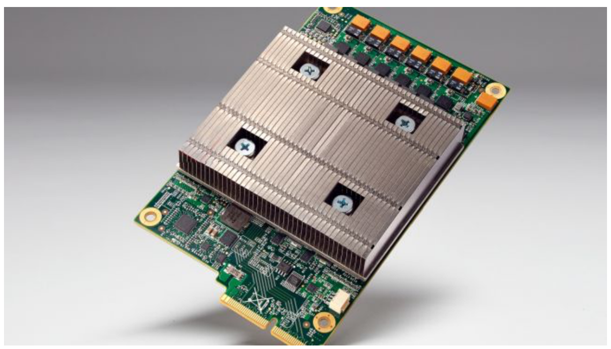
- 100’s of millions of parameters + convolutions & unrolling
- Requires hardware acceleration

http://www.nvidia.com/content/tegra/embedded-systems/pdf/jetson_tx1_whitepaper.pdf

Computer Vision and Speech Recognition

- 

(will cover in more detail later):



>1000 photos a second
on a cluster of ASICs

Using Google's fleet of TPUs, we can find all the text in the Street View database in less than five days. In Google Photos, each TPU can process ***[more than] 100 million photos a day****.*
-- Norm Jouppi (Google)

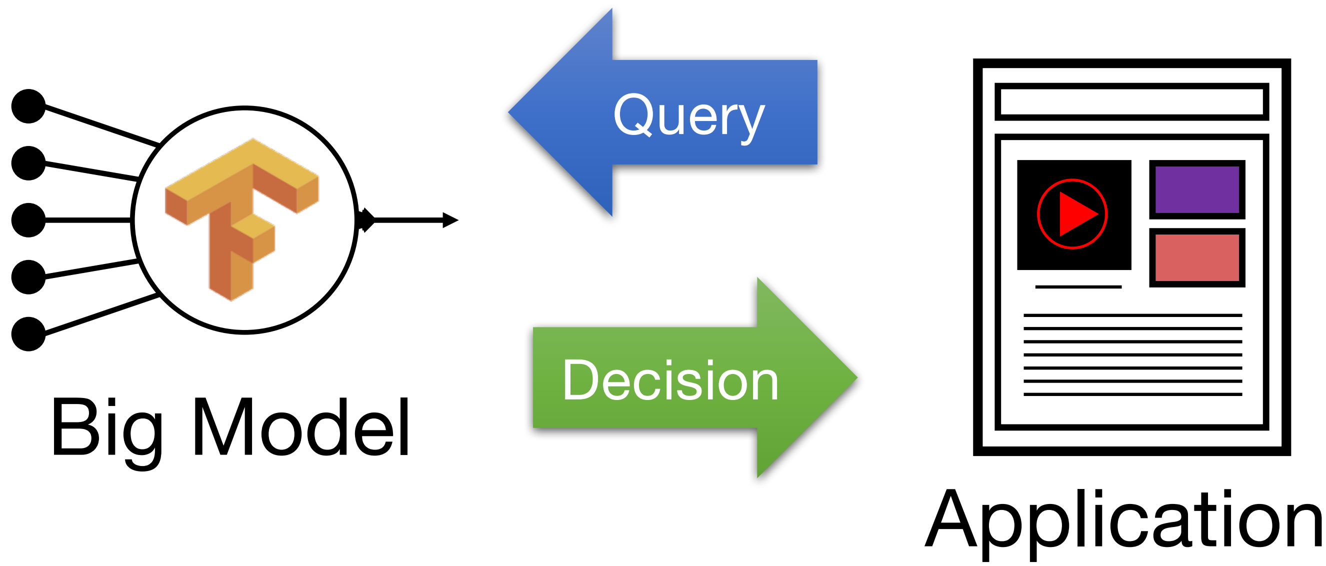
- 100’s of millions of parameters + convolutions & unrolling
- Requires hardware acceleration

http://www.techradar.com/news/computing-components/processors/google-s-tensor-processing-unit-explained-this-is-what-the-future-of-computing-looks-like-1326915

Robust Predictions

- Often want to quantify prediction accuracy (uncertainty)
- Several common techniques
  - Bayesian Inference
    - Need to maintain more statistics about each parameter
    - Often requires matrix inversion, sampling, or numeric integration
  - Bagging
    - Multiple copies of the same model trained on different subsets of data
    - Linearly increases complexity
  - Quantile Methods
    - Relatively lightweight but conservative
- In general robust predictions ➔ additional computation

Inference



Two Approaches

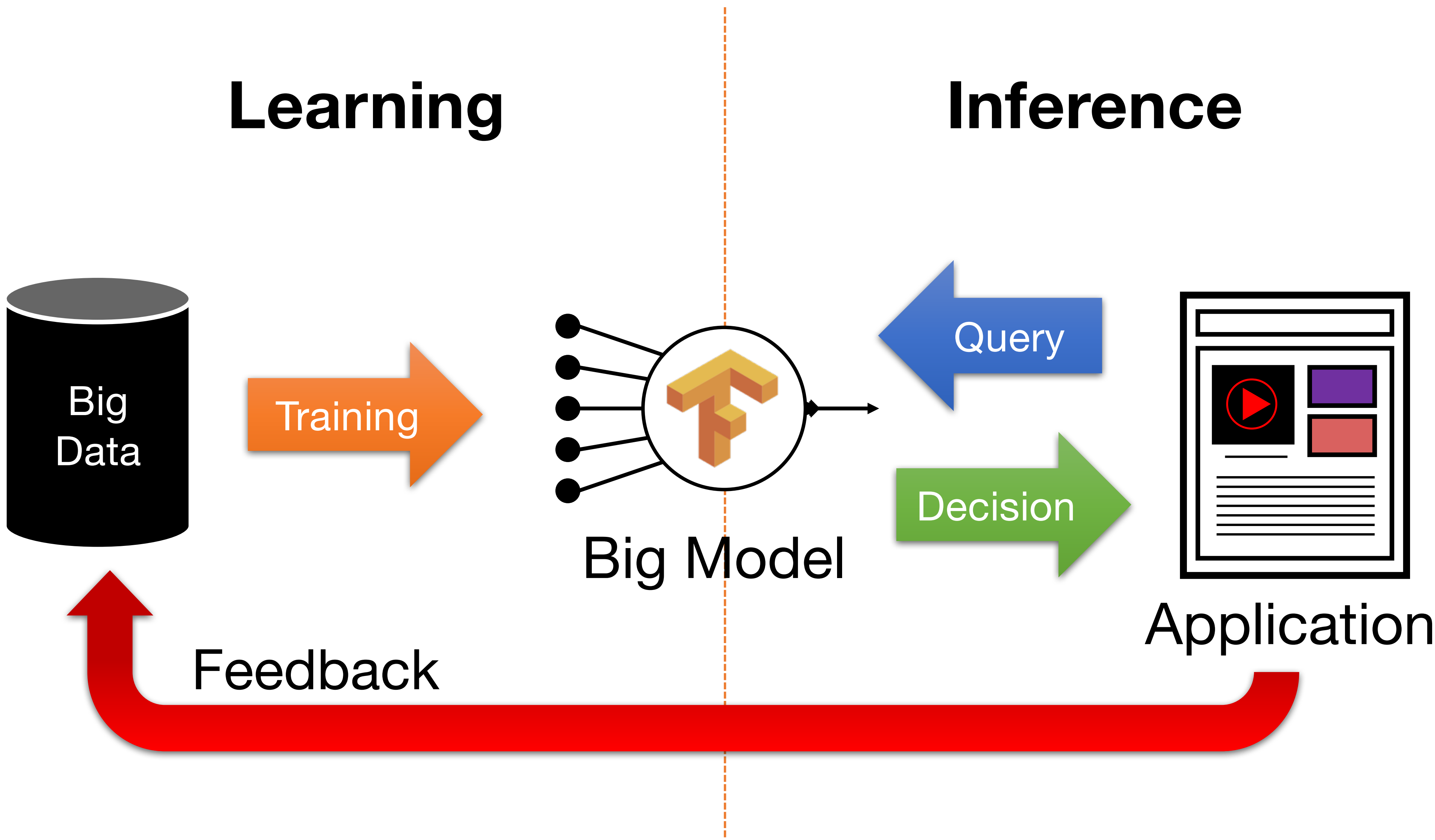
- ***Eager:*** Pre-Materialize Predictions
- ***Lazy:*** Compute Predictions on the fly

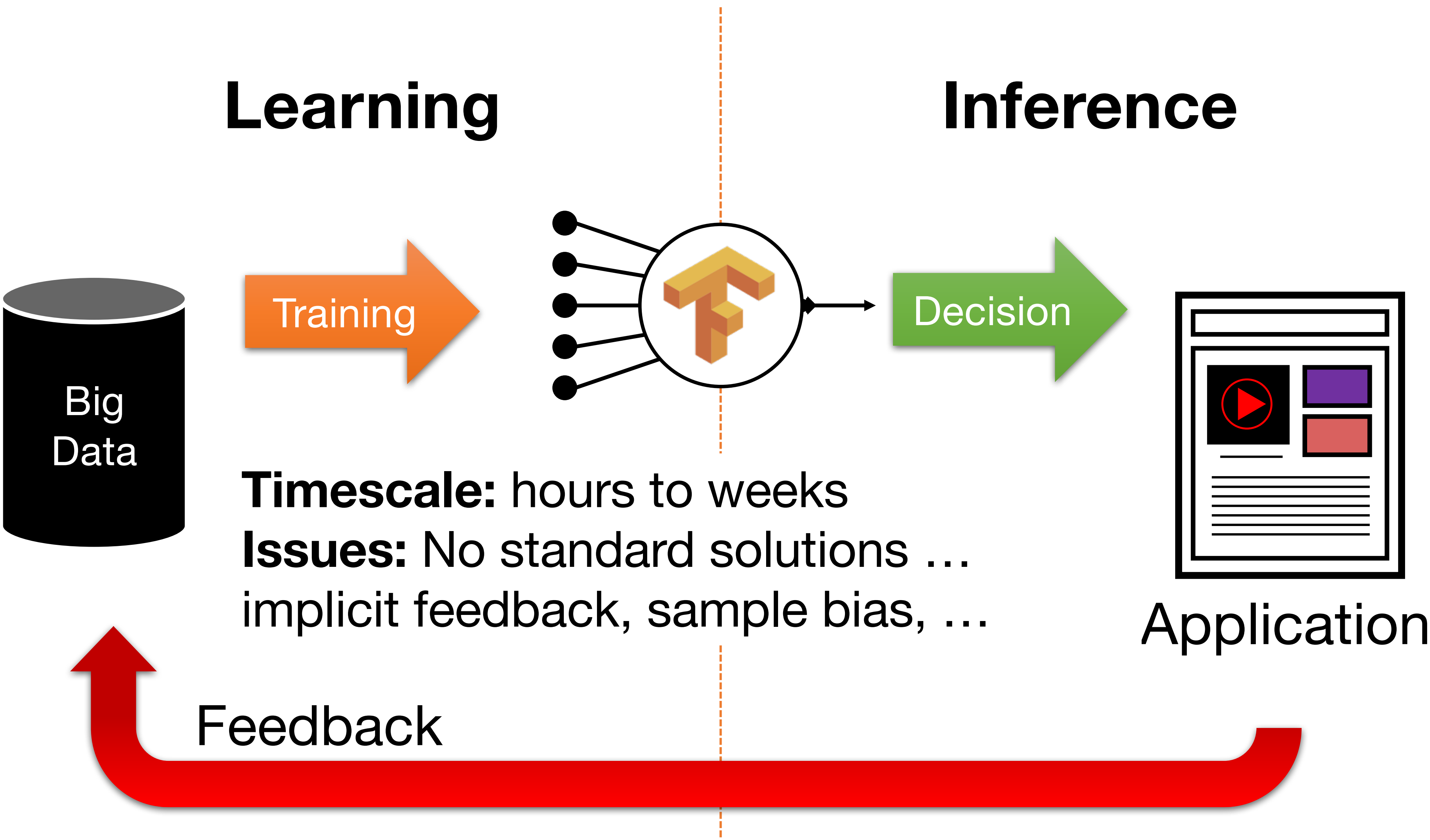
Eager: Pre-materialize Predictions

- **Examples**
  - Zillow might pre-compute popularity scores or house categories for all active listings
  - Netflix might pre-compute top k movies for each user daily
- **Advantages**
  - Use offline training frameworks for efficient batch prediction
  - Serving is done using traditional data serving systems
- **Disadvantages**
  - Frequent updates to models force substantial computation
  - Cannot be applied when set of possible queries is large (e.g., speech recognition, image tagging, …)

Lazy: Compute predictions at Query Time

- **Examples**
  - Speech recognition, image tagging
  - Ad-targeting based on search terms, available ads, user features
- **Advantages**
  - Compute only necessary queries
  - Enables models to be changed rapidly and bandit exploration
  - Queries do not need to be from small ground set
- **Disadvantages**
  - Increases complexity and computation overhead of serving system
  - Requires low and predictable latency from models


Learning
Inference
Big
Data
Training
Query
Decision
Big Model
Application
Feedback


Learning
Inference
Big Data
Training
Decision
Application
Timescale: hours to weeks
Issues: No standard solutions …
implicit feedback, sample bias, …
Feedback

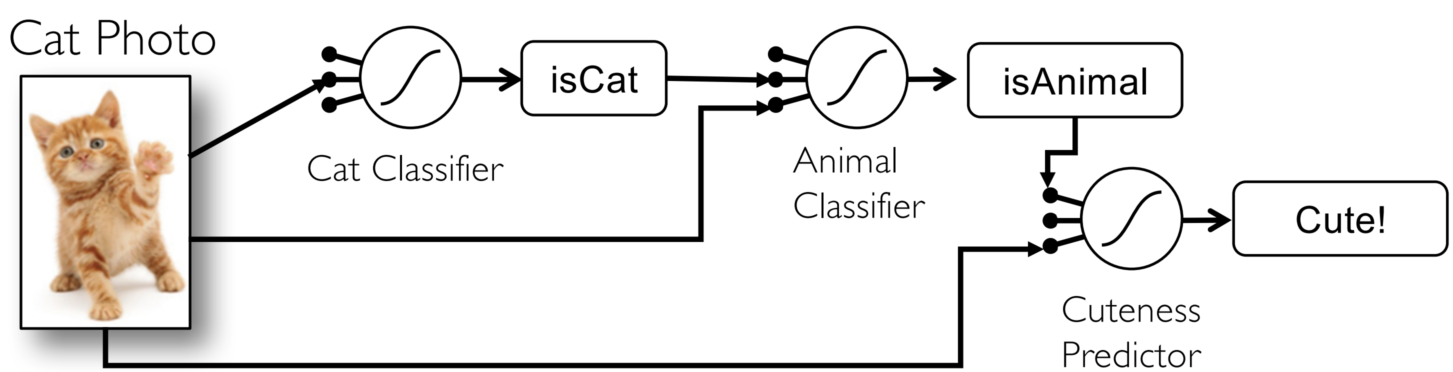
Why is **Closing the Loop** challenging?

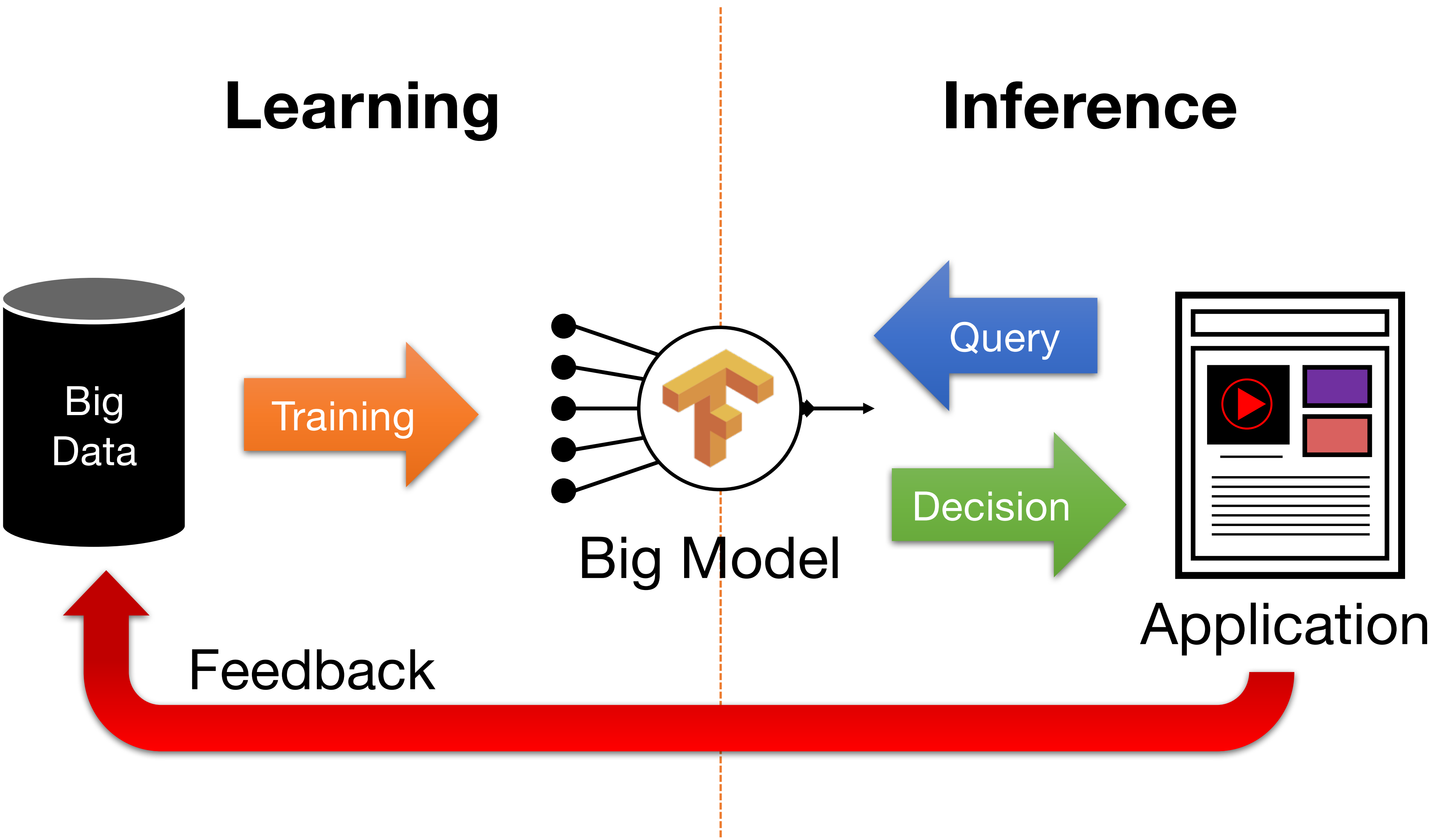
- Multiple types of feedback:
  - **implicit feedback:** absence of the correct label
  - **delayed feedback:** need to join feedback with previous prediction state
- Exposes system to **feedback loops**
  - *If we only play the top songs how will we discover new hits?*
- Need to address **concept drift** and **temporal variation**
  - How do we **forget the past** and **model time directly**

Management and Monitoring

- Desiging specifications and test for ML Systems can be difficult

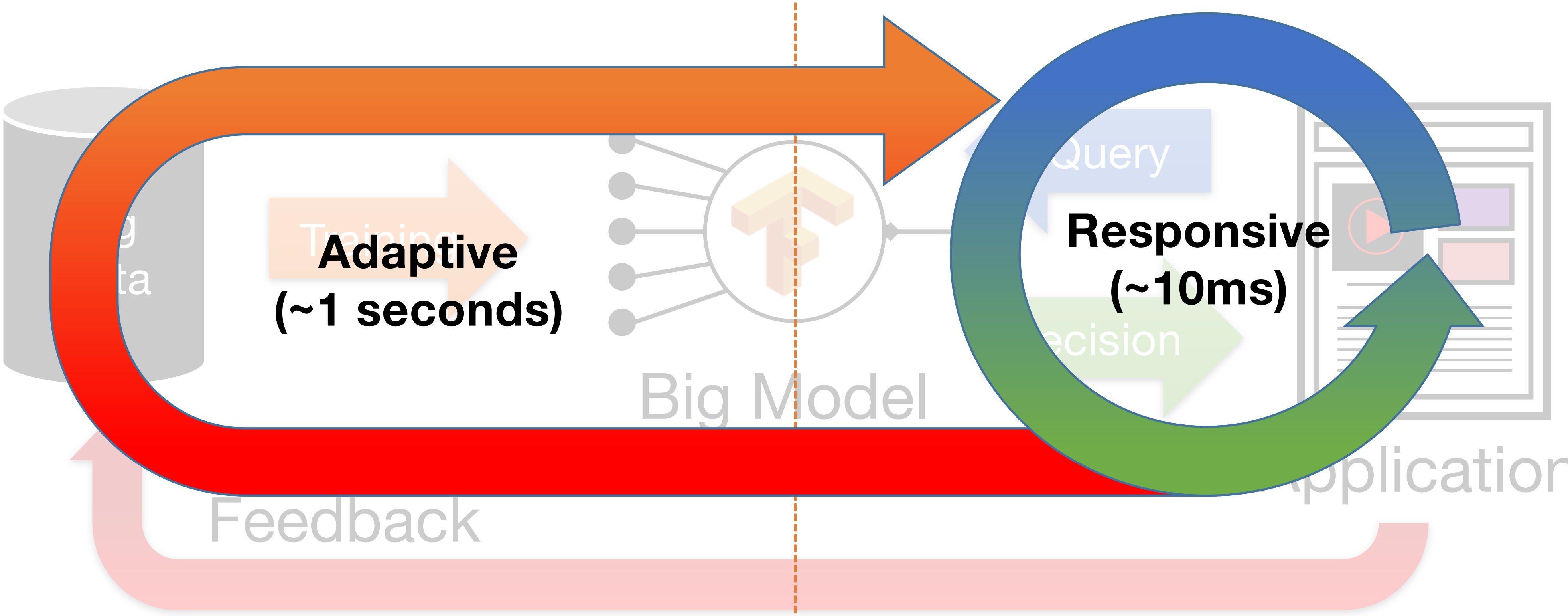
- Entagled dependencies:
  - Data and Code
  - Pipelines




Learning
Inference
Big
Data
Training
Big Model
Query
Decision
Application
Feedback

Learning

Inference

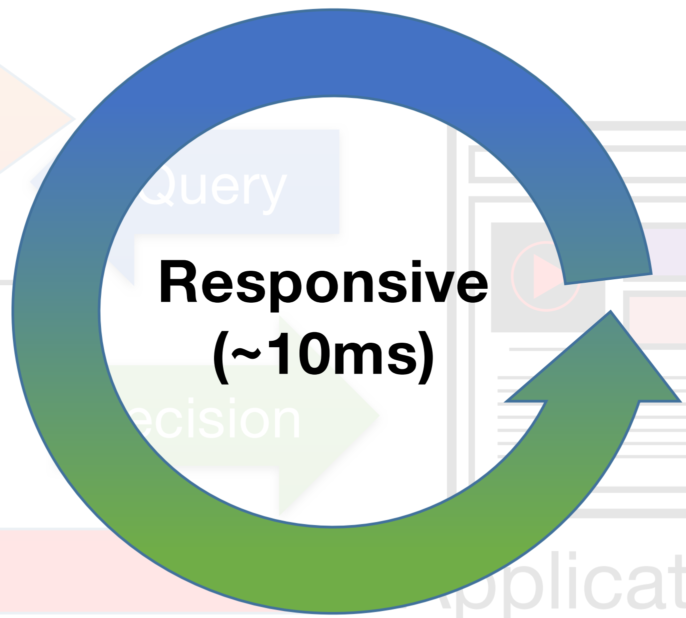


Learning

Inference

Today we will focus on
Inference and **Management**

Later in the year we will return to
Feedback.



Vertical **Solutions** to Real-time Prediction Serving

- **Ad Click Prediction and Targeting**
  - *a multi-billion dollar industry*
  - Latency sensitive, contextualized, high-dimensional models → ranking
- **Content Recommendation** (optional reading)
  - Typically simple models trained and materialized offline
  - Moving towards more online learning and adaptation
- **Face Detection** (optional reading)
  - example of early work in accelerated inference → substantial impact
  - Widely used Viola-Jones face detection algorithm (prediction cascades)
- **Automatic Speech Recognition (ASR)** (optional reading)
  - Typically cloud based with limited literature
  - Baidu Paper: deep learning + traditional beam search techniques
    - Heavy use of hardware acceleration to make ”real-time” 40ms latency

Presentations Today

- **Giulio Zhou:** challenges of deployed ML from perspective of Google & Facebook

- **Noah Golmat:** eager prediction serving from within a traditional RDBMS using hazy

- **Dan Crankshaw:** The LASER lazy prediction serving system at LinkedIn and his ongoing work on the Clipper prediction serving system.

Future Directions

Research in Faster Inference

- **Caching** (Pre-Materialization)
  - Generalize Hazy style Hölder's Inequality bounds
  - Cache warming and prefetching & approximate caching
- **Batching** → better tuning of batch sizes
- Parallel **hardware acceleration**
  - GPU → FPGA → ASIC acceleration
  - Leveraging heterogeneous hardware with low bit precision
  - Secure Hardware
- Model **compression**
  - Distillation (will cover later)
  - Context specific models
- **Cascading Models:** fast path for easy queries
- **Inference on the edge:** utilize client resources during inference

Research in Model Life-cycle Management

- **Performance monitoring**
  - Detect potential model failure with limited or no feedback
- **Incremental model updates**
  - Incorporate feedback in real-time to update entire pipelines
- **Tracking model dependencies**
  - Ensure features are not corrupted and models are updated in response to changes in upstream models
- **Automatic model selection**
  - Choosing between many candidate models for a given prediction task