BDMI-课程编号-01510243

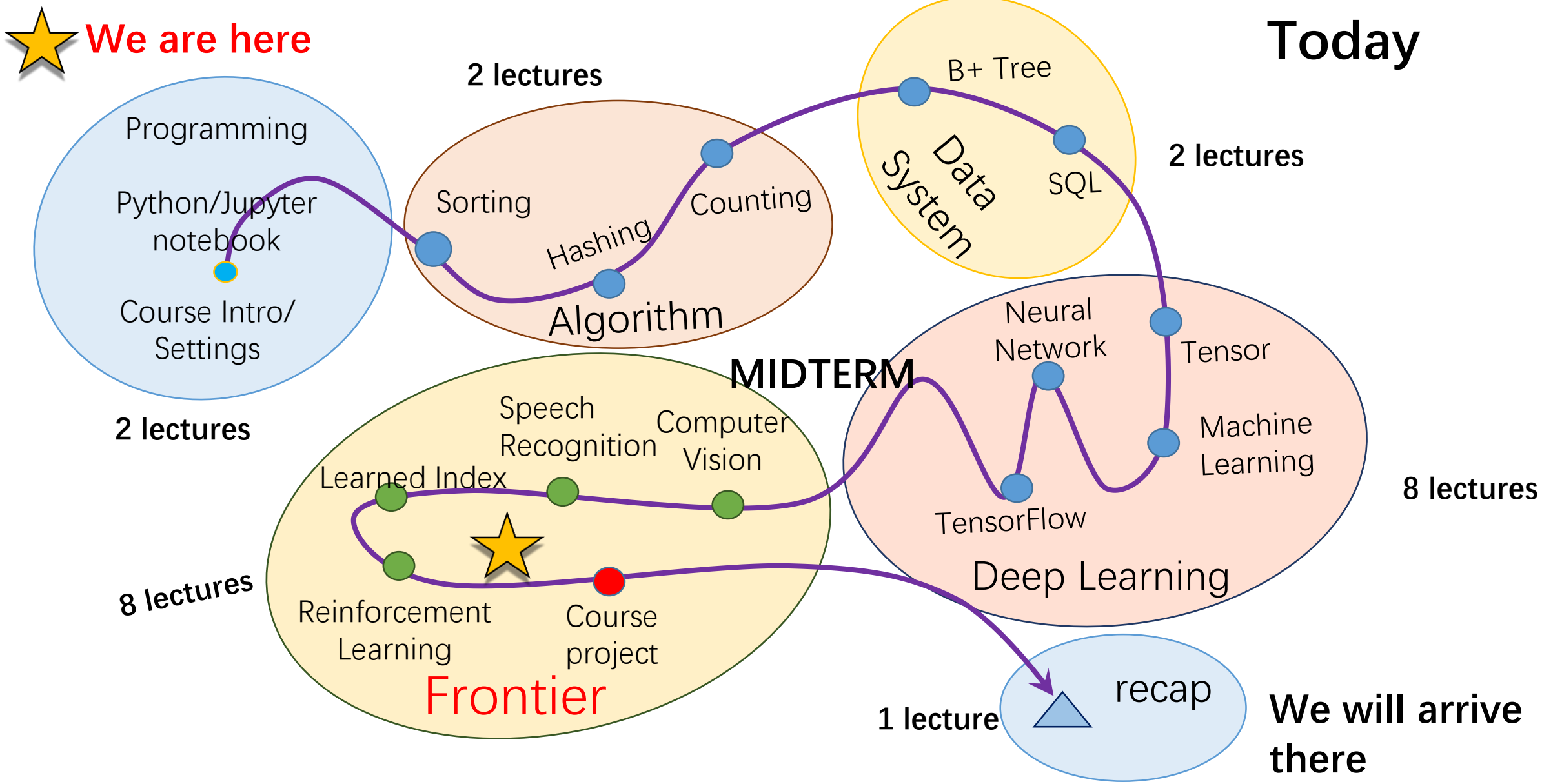
大数据与机器智能

Learned ~~Data~~ System

智能系统实验室

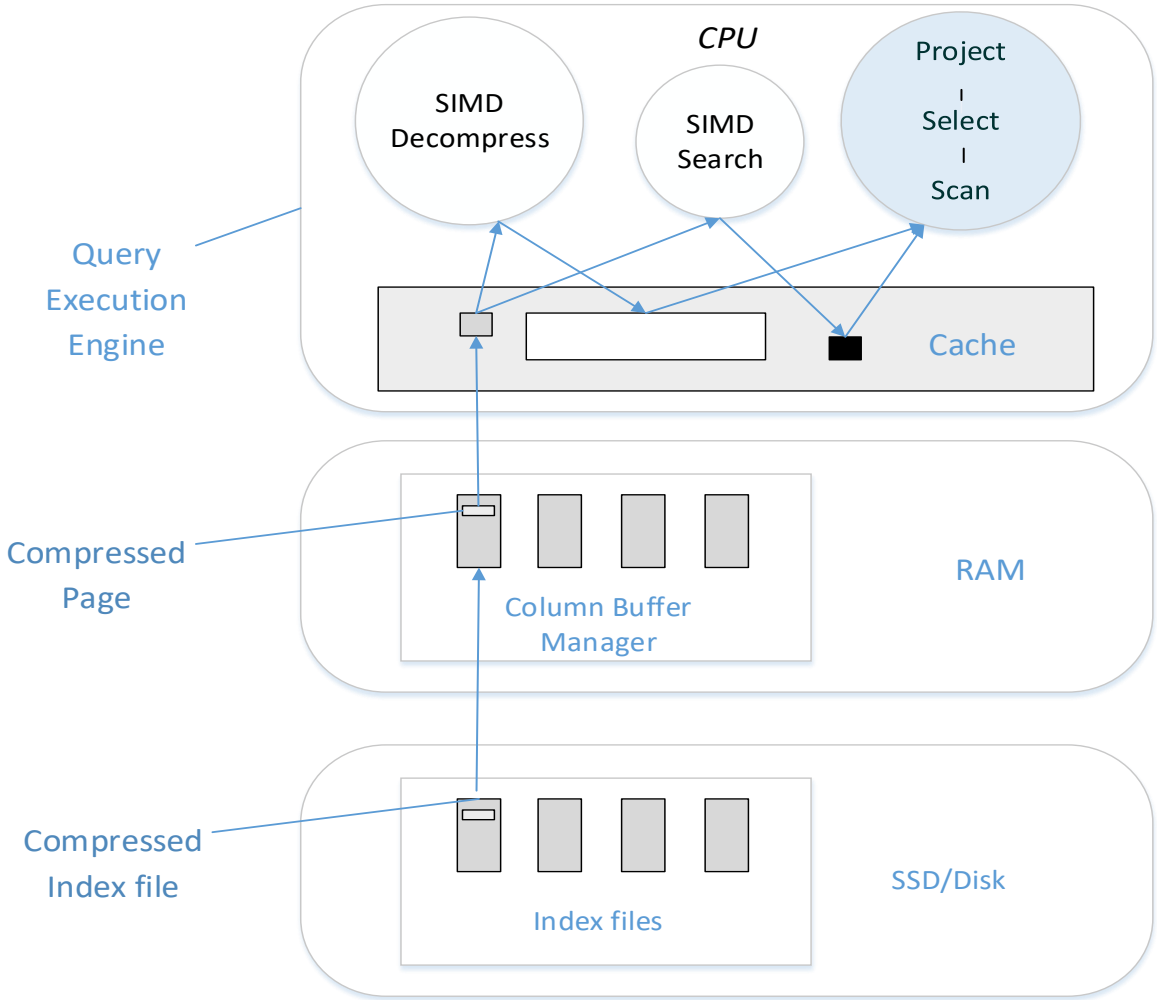
清华大学基础工业训练中心

课程路线图（Roadmap）



Computer organizations for Computer System

计算机架构的存储器等级（Memory Hierarchy）



- CPU/RAM/SSD/Disk
- Memory Hierarchy
  - Register/Cache
  - RAM(Main memory)
  - SSD(Solid-state disk)
  - Disk (magnetic)

High-level: Disk vs. Main Memory

- Random Access Memory (RAM) or **Main Memory**:
  - Fast
    - Random access, byte addressable
      - ~10x faster for sequential access
      - ~100,000x faster for random access!
  - Volatile
    - Data can be lost if e.g. crash occurs, power goes out, etc!
  - Expensive
    - For $100, get 16GB of RAM vs. 2TB of disk! (~125x)

High-level: Disk vs. Main Memory

Disk:

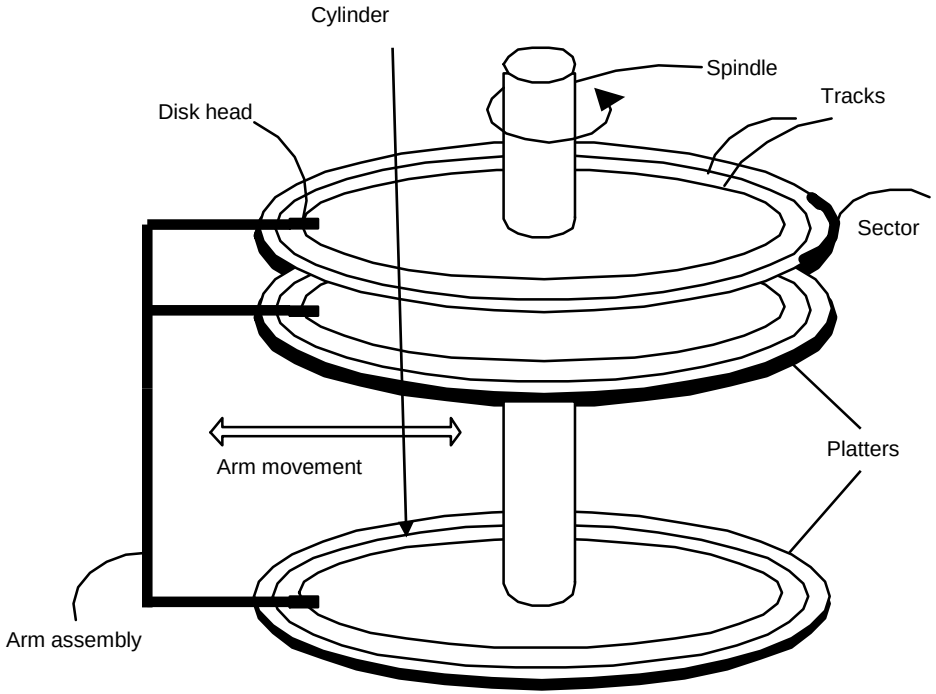
Slow seeks (8-15 msec)

Rotations: 5000-15000 rpm

Durable

- We will assume that once on disk, data is safe!

Cheap



程序员需知的时延开销

timings

execute typical instruction	1/1,000,000,000 sec = 1 nanosec
fetch from L1 cache memory	0.5 nanosec
fetch from L2 cache memory	7 nanosec
Mutex lock/unlock	25 nanosec
fetch from main memory	100 nanosec
send 2K bytes over 1Gbps network	20,000 nanosec
read 1MB sequentially from memory	250,000 nanosec
fetch from new disk location (seek)	8,000,000 nanosec
read 1MB sequentially from disk	20,000,000 nanosec
send packet US to Europe and back	150 milliseconds = 150,000,000 nanosec

(~0.25 msecs)

(~10 msecs)

(~20 msecs) (or use 100 MB/sec)

RDBMS

关系型数据库

关系数据库原理

一个**关系**或者一个**数据表**是有多个**属性**（由模式定义）的**多元组**。

A **relation** or **table** is a multiset of **tuples** having the **attributes** specified by the **schema**

Product

PName	Price	Manuf
小米Max3	1,600	小米
小米6	2,000	小米
Iphone 8	5,000	苹果
Mate 20	5,000	华为

- 表模式(Table Schemas)
- Product(Pname: *string*, Price: *float*, Category: *string*, Manufacturer: *string*)

SQL Basics

SQL语言

SQL stands for
Structured
Query
Language

- SQL代表**结构化查询语言，用于查询和操纵数据**。
- 标准：ANSI SQL，SQL92 (a.k.a. SQL2)，SQL99 (a.k.a. SQL3), ….
- SQL分为DML和DDL两部分
  - **DML（Data Manipulation Language）：**查询数据表（tables），插入删除修改数据表的元组（tuples）
  - **DDL（Data Definition Language (DDL)：**定义关系模式（relational schemata），创建、修改删除数据表和表的属性（attributes）

Example: Search for books

Billion_Books

BID	Title	Author	Published	Full_text
	….	….	….	
7003	Harry Potter	Rowling	1999	…
1001	War and Peace	Tolstoy	1869	…
1002	Crime and Punishment	Dostoyevsky	1866	…
1003	Anna Karenina	Tolstoy	1877	…
	….			

All **books** written by Rowling?'

```
SELECT *
FROM   Billion_Books
WHERE Author like 'Rowling'
```

SQL Database

- 关系数据模型和关系代数的坚实数学基础
- 成熟的关系型数据库RDBMS
- 成熟的查询语言SQL
- 严格的事务（TXNs）处理特性（ACID）

埃德加·弗兰克·科德（Edgar Frank Codd），
为关系型数据库理论做出了奠基性的贡献。
他在IBM工作期间，首创了关系模型理论。
他于1981年获得图灵奖。
数据库领域出现了四位图灵奖。

ACID stands for
Atomicity
Consistency
Isolation
Durability

Big Data: NoSQL—NewSQL

- NoSQL, Popular set of databases, largely built by web companies in the 2000s
  - Focus on scale-out and flexible schemas
  - Lots of hype, somewhat dying down

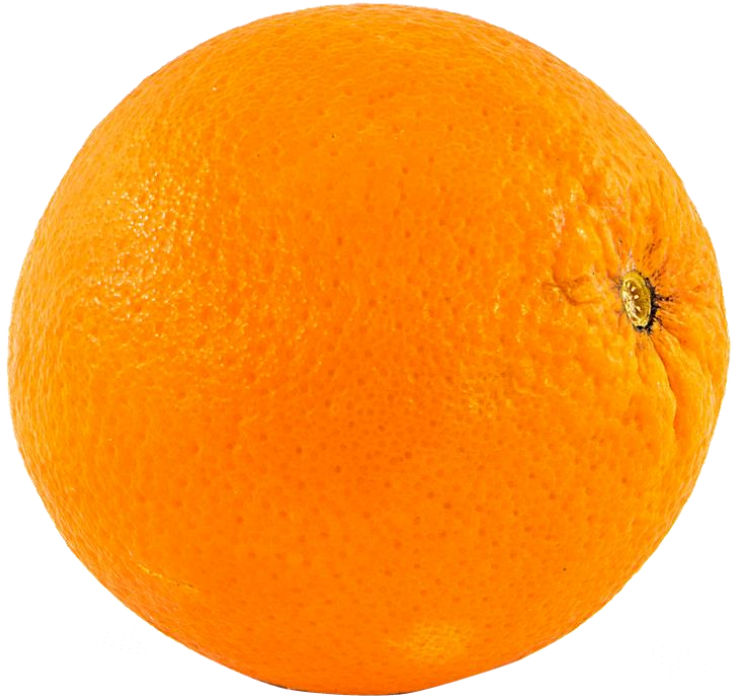
- Amazon’s Dynamo was among the first
  - Dynamo: amazon's highly available key-value store

- Open source examples:
  - MongoDB, Cassandra, Redis

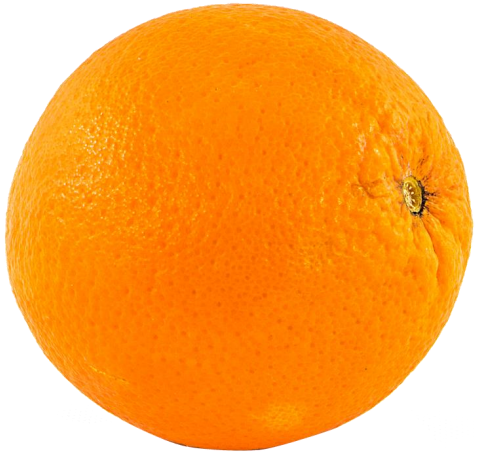
Deep Learning

深度学习

How would we do this without machine learning?



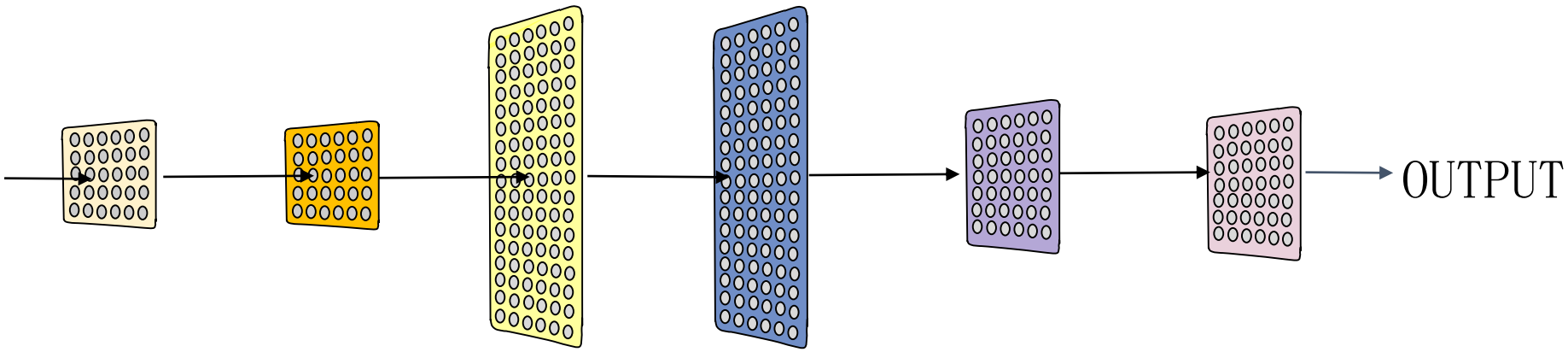
How would we do this without ML?



Stage 1: Train an ML model with examples

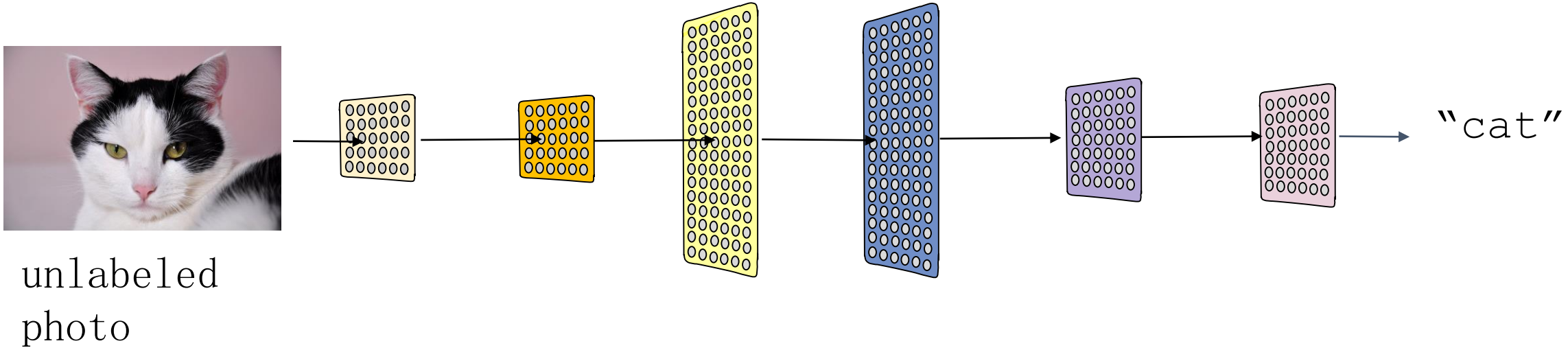
label, input

Make tiny adjustments to model function so output is closer to label for a given input



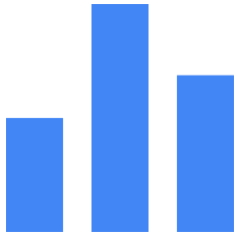
A ML model is a mathematical function

Stage 2: Predict with a trained model

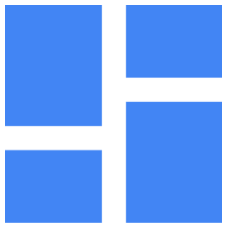


unlabeled photo

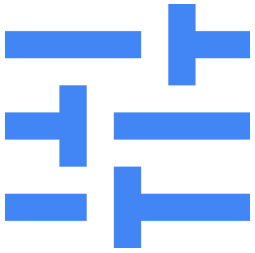
ML teaches a computer to make repeated decisions using standard algorithms and lots of data



Data



Algorithm



Predictive insight

Decision

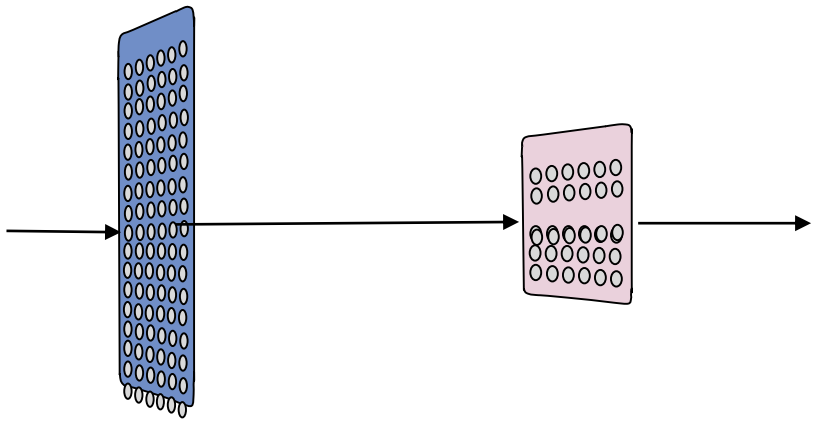
Data instead of rules

A data warehouse can be a source of training examples

```
SELECT
  gestation_weeks,
  mother_age,
  cigarette_use,
  alcohol_use,
  weight_gain_pounds
FROM
 `bigquery-public-data.samples.natality`
WHERE cigarette_use is not null AND alcohol_use
```

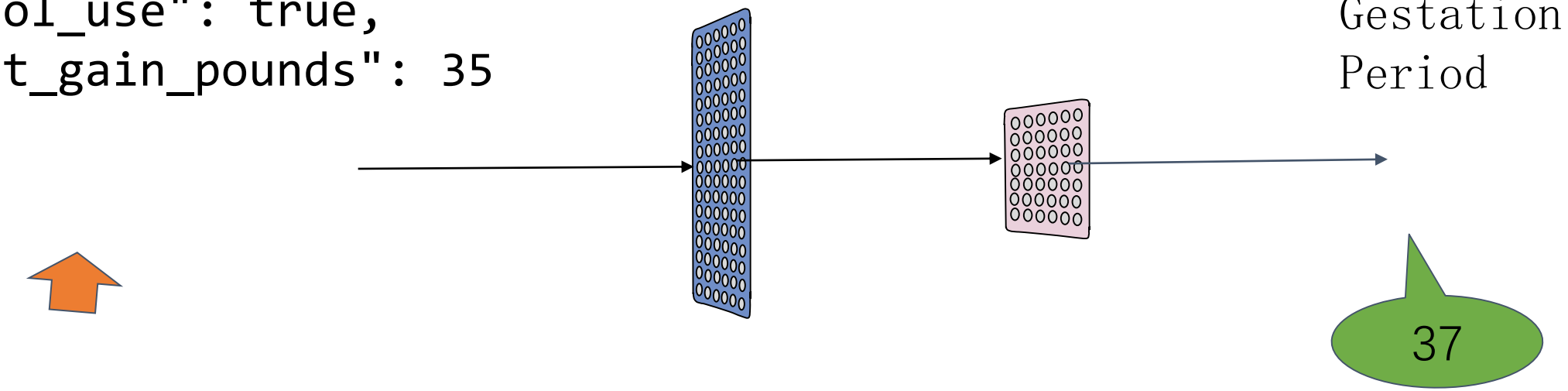
Could learn to predict gestation_WEEKS based on other factors

weight	year	mother_age	gestation_weeks	cigarette_use	alcohol_use
7.86	2003	25	39	false	false
7.5	2003	21	39	false	false
8.06	2004	29	40	false	false
7.56	2004	38	37	false	false
7.06	2003	22	38	false	false



The model is fed information collected in real-time, and used for prediction

```
{
  "mother_age": 23,
  "cigarette_use": false,
  "alcohol_use": true,
  "weight_gain_pounds": 35
}
```



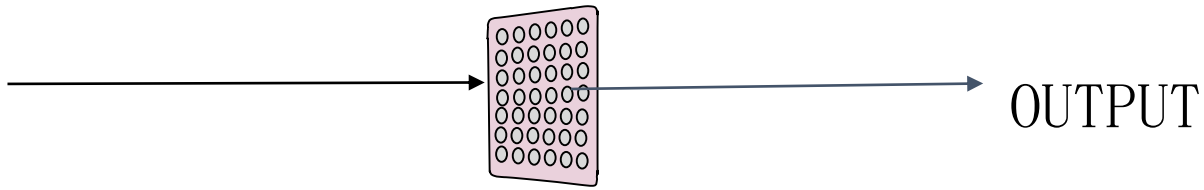
If model has just one layer, this is called linear regression

gestation_weeks	mother_age	cigarette_use	alcohol_use	weight_gain_pounds
36	23	false	false	70
37	23	false	false	25
38	21	false	false	33
39	16	false	false	26
39	23	true	true	null
41	30	false	false	8

36

41

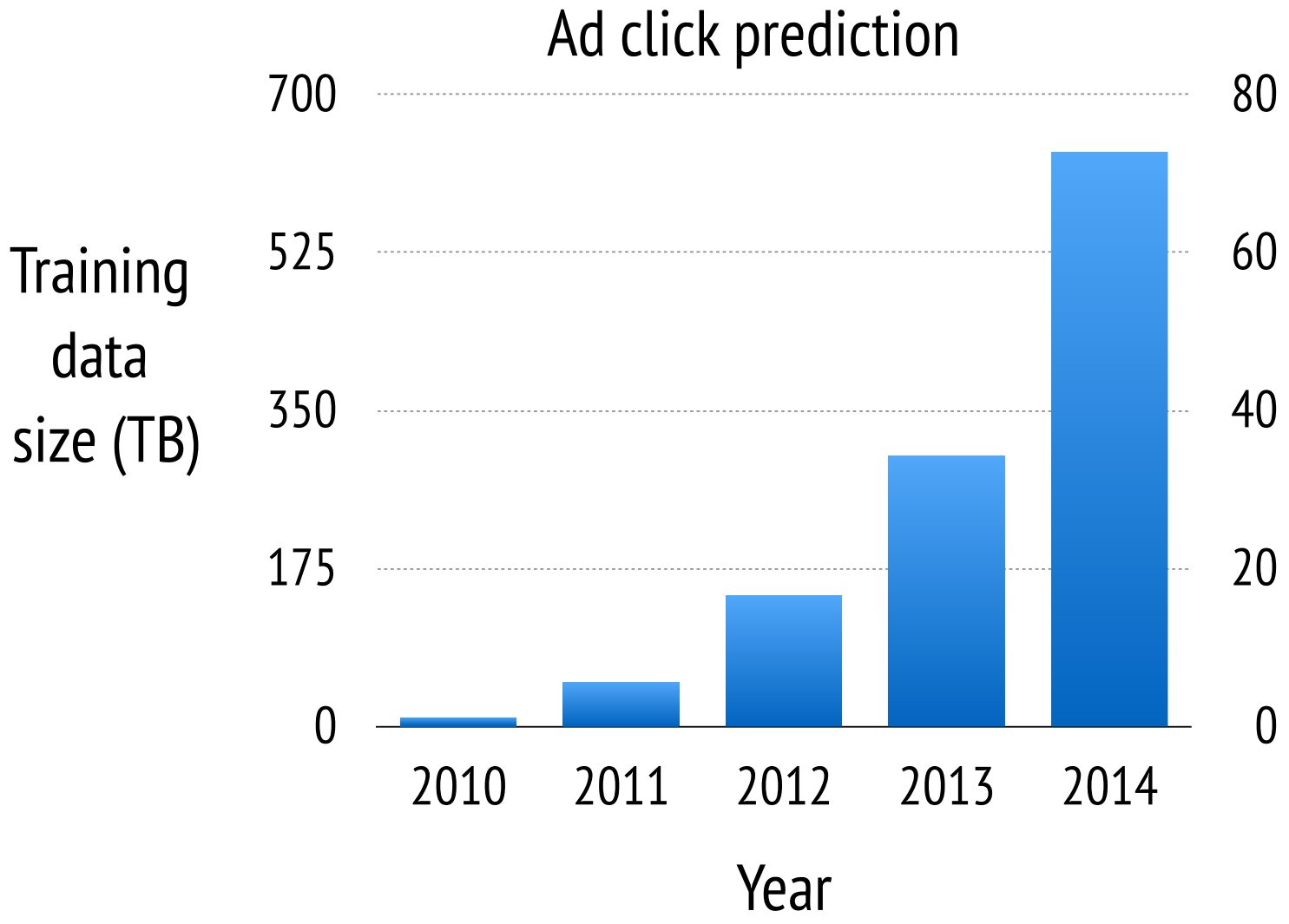
```
gestation_weeks =  w0 *
mother_age
                        + w1 *
cigarette_use
                        + w2 *
alcohol_use
                        + w3 *
weight_gain_pounds
                        + b
```



If the variable to be predicted is categorical, you would use **logistic regression** *(essentially linear regression + a logistic transformation)*

Large Scale Deep Learning including data system

Machine learning is concerned with systems that can learn from data



Hardware Trends

	2010	2018	
Storage	50+MB/s (HDD)	500+MB/s (SSD)	10X
Network	1Gbps	10Gbps	10X
CPU	~3GHz	~3GHz	☹

Idea:

- For “big models,” partition data *and* parameters
  - Called a “**parameter server**”
  - **Asynchronous training** can help!

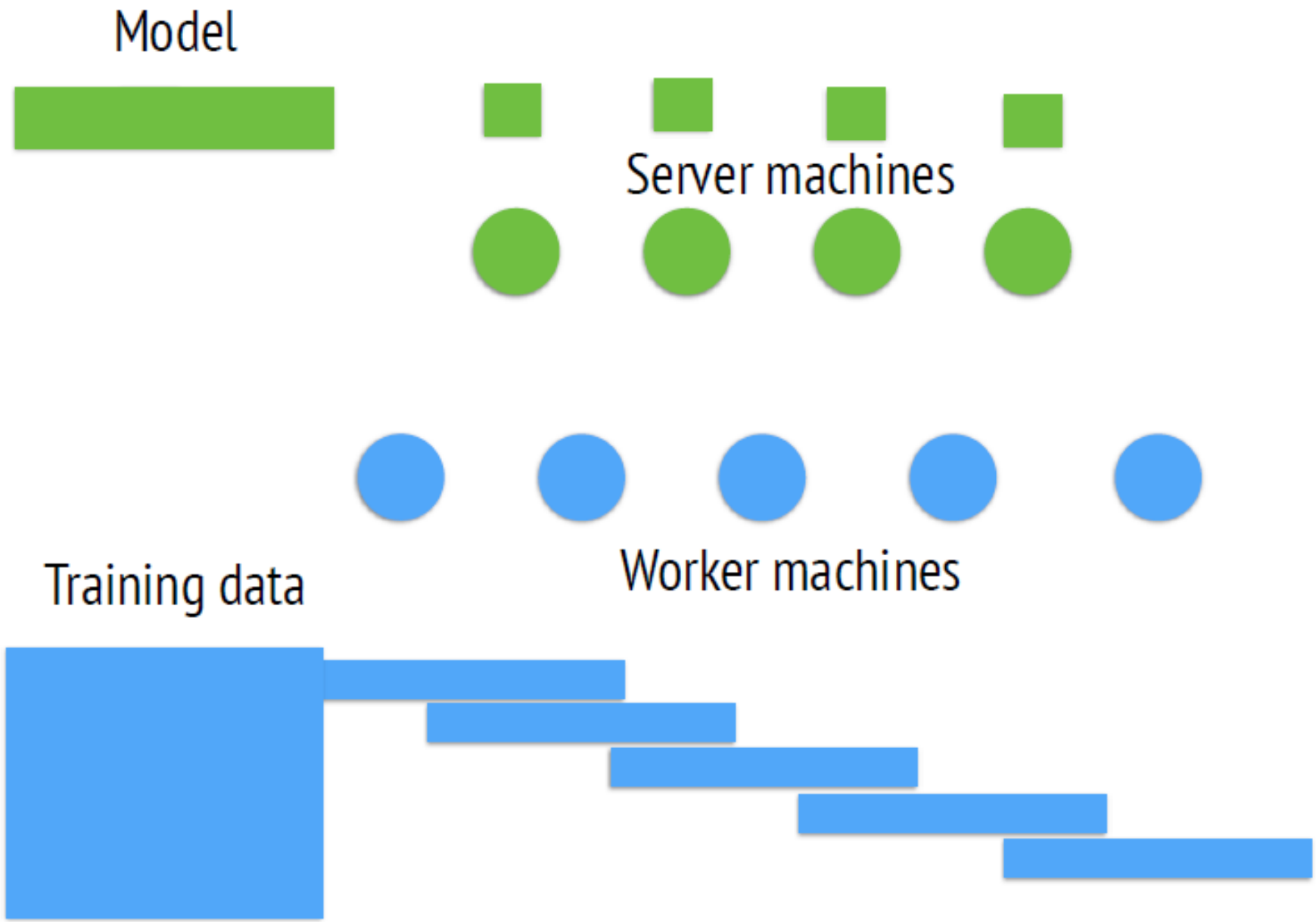
- New systems like TensorFlow combine this idea *with* dataflow

- Large Scale Deep Learning

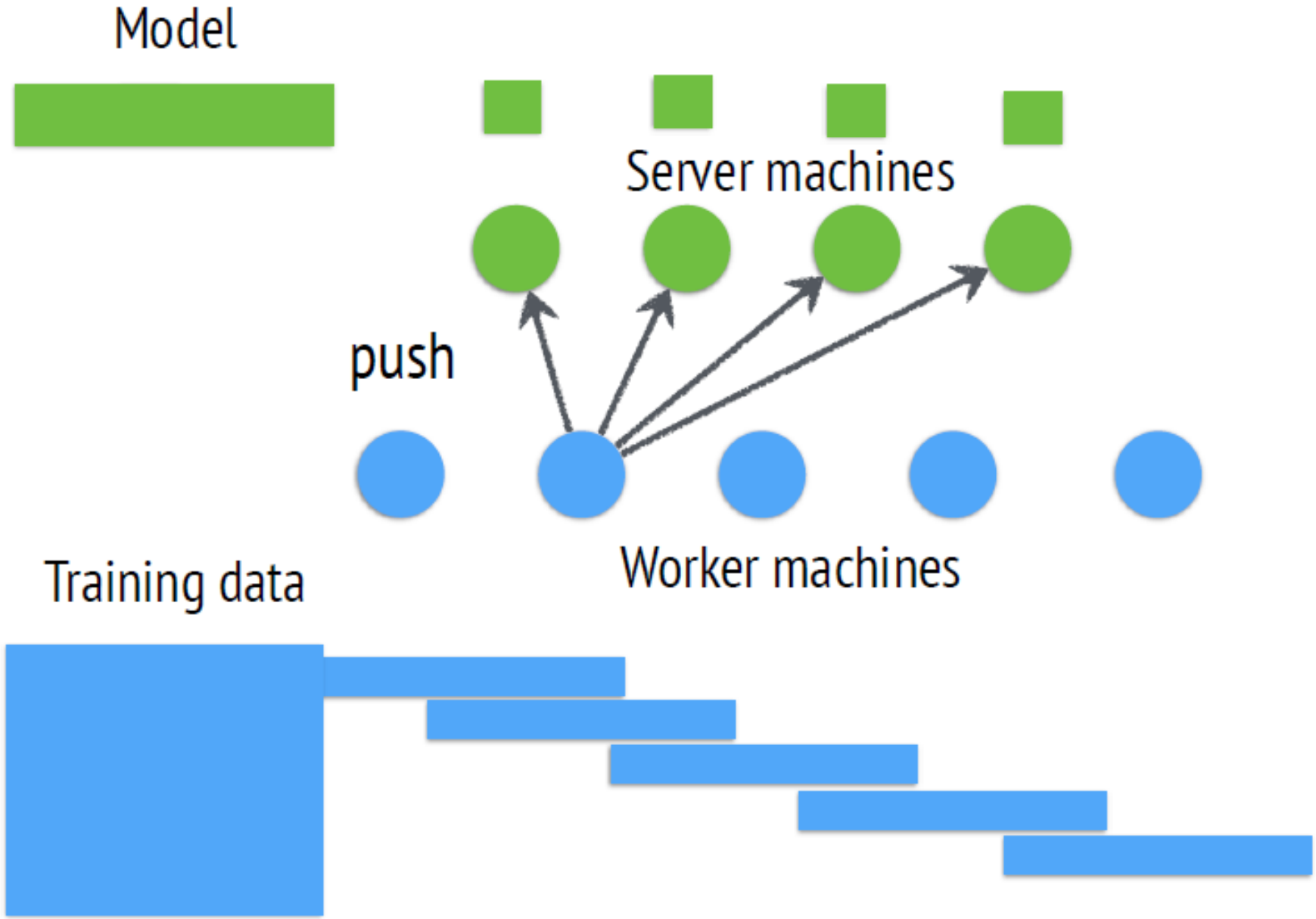
Data and model partition

Training data

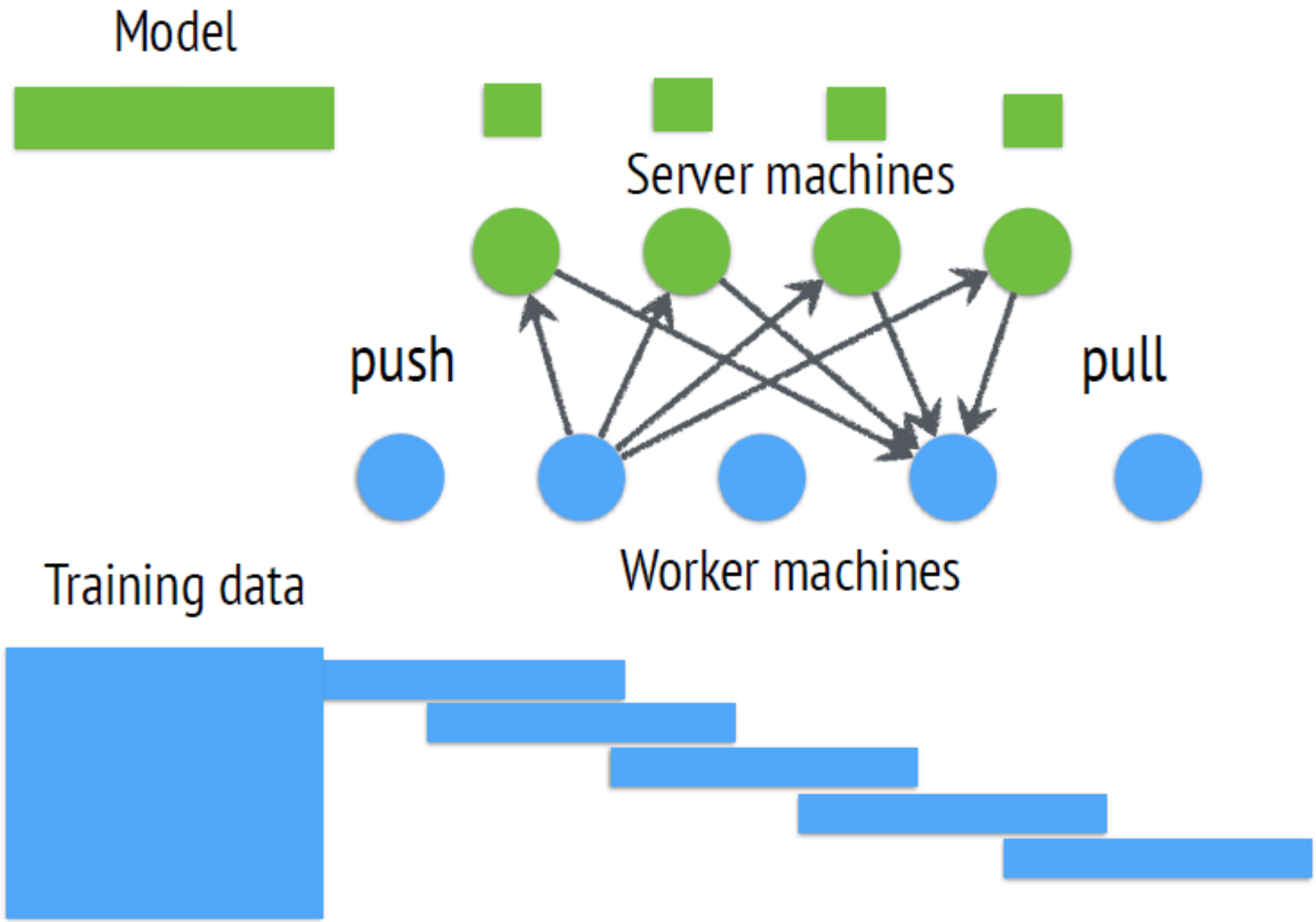
Data and model partition



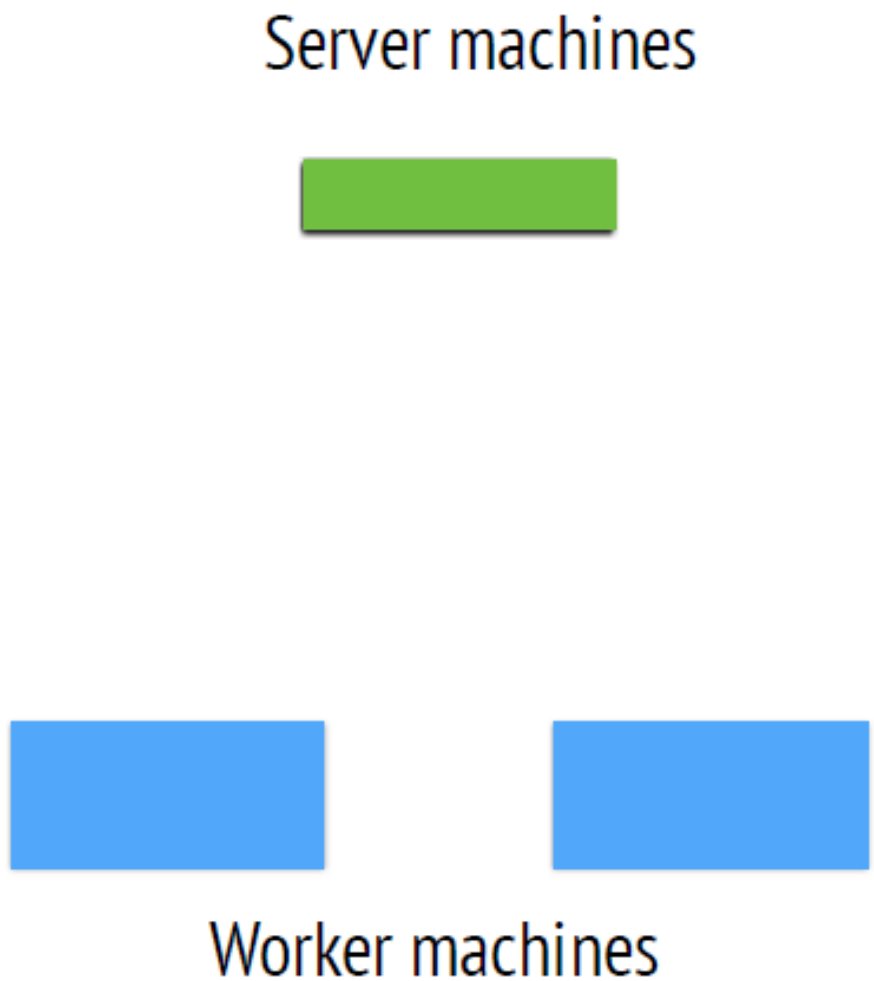
Data and model partition



Data and model partition

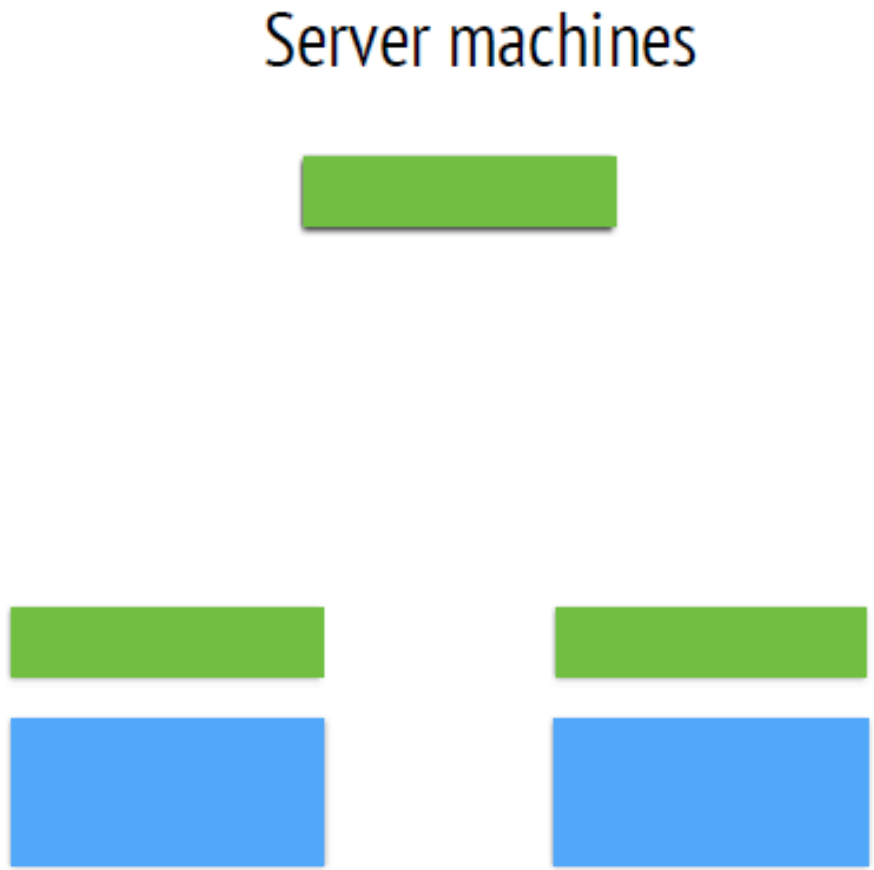


Example: distributed gradient descent



Example: distributed gradient descent

Workers **pull** the working set of model



Example: distributed gradient descent

Workers **pull** the working set of model

Iterate until stop

workers compute gradients

Server machines

Worker machines

Example: distributed gradient descent

Workers **pull** the working set of model

Iterate until stop

workers compute gradients

workers **push** gradients

Server machines

Worker machines

Example: distributed gradient descent

Workers **pull** the working set of model

Iterate until stop

workers compute gradients

workers **push** gradients

update model

Server machines

Worker machines

Example: distributed gradient descent

Workers **pull** the working set of model

Iterate until stop

workers compute gradients

workers **push** gradients

update model

workers **pull** updated model

Server machines

Worker machines

TensorFlow Architecture

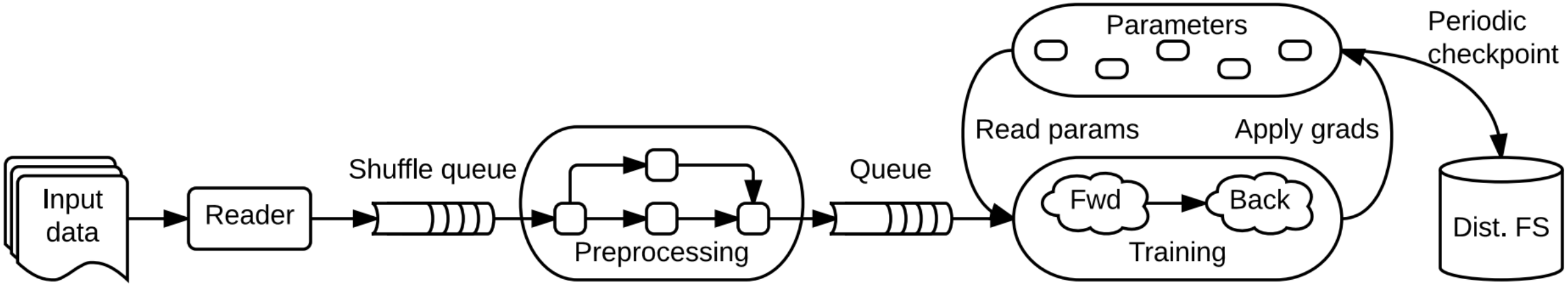


Figure 2: A schematic TensorFlow dataflow graph for a training pipeline, containing subgraphs for reading input data, preprocessing, training, and checkpointing state.

https://tensorflow.google.cn/guide/extend/architecture

Punchlines

- Parameter server architecture is useful for training large models
  - Increasingly popular in “deep networks”
- Lots of noise about new systems (Graphs, Deep Learning)
  - Often need special adaptation for workloads (e.g., special joins, operators)
  - But basic computational patterns (dataflow with some shared state) same
- Asynchrony can help training time in distributed environment

- Is training all we care about?

Machine Learning System and Data System: Integration

机器学习系统与数据系统：融合

Hidden Technical Debt in Machine Learning Systems

D. Sculley, Gary Holt, Daniel Golovin, Eugene Davydov, Todd Phillips
{dsculley,gholt,dgg,edavydov,toddphillips}@google.com
Google, Inc.

Dietmar Ebner, Vinay Chaudhary, Michael Young, Jean-François Crespo, Dan Dennison
{ebner,vchaudhary,mwyoung,jfcrespo,dennison}@google.com
Google, Inc.

Hidden Technical Debt in Machine Learning Systems

Machine learning offers a fantastically powerful toolkit for building useful complex prediction systems quickly. This paper argues it is dangerous to think of these quick wins as coming for free.

Using the software engineering framework of technical debt, we find it is common to incur massive ongoing maintenance costs in real-world ML systems.

We explore several ML-specific risk factors to account for in system design. These include boundary erosion, entanglement, hidden feedback loops, undeclared consumers, data dependencies, configuration issues, changes in the external world, and a variety of system-level anti-patterns.

D. Sculley et al., Hidden Technical Debt in Machine Learning Systems，nips 2015.

Next generation of systems

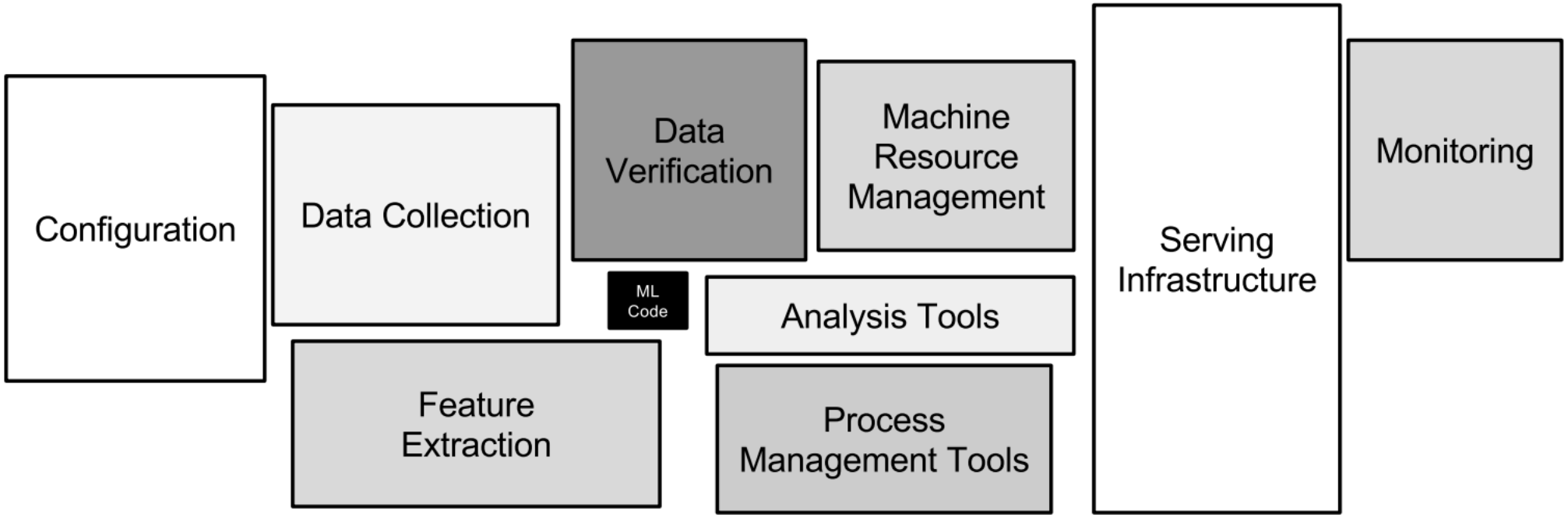


Figure 1: Only a small fraction of real-world ML systems is composed of the ML code, as shown by the small black box in the middle. The required surrounding infrastructure is vast and complex.

Post-database data management!!!

Machine Learning transform Computer System (including Data System)

SysML

SysML: The New Frontier of Machine Learning Systems

Machine learning (ML) techniques are enjoying rapidly increasing adoption. However, designing and implementing the systems that support ML models in real-world deployments remains a significant obstacle, in large part due to the radically different development and deployment profile of modern ML methods, and the range of practical concerns that come with broader adoption.

We propose to foster a new systems machine learning research community at the intersection of the traditional systems and ML communities, focused on topics such as hardware systems for ML, software systems for ML, and ML optimized for metrics beyond predictive accuracy.

To do this, we describe a new conference, SysML, that explicitly targets research at the intersection of systems and machine learning with a program committee split evenly between experts in systems and ML, and an explicit focus on topics at the intersection of the two.

Alexander Ratner, SysML: The New Frontier of Machine Learning Systems, 2019.
https://arxiv.org/abs/1904.03257v1

Learned Index

习得索引

B+ Tree 索引精确查找（Exact Search ）

- 查询K=30?

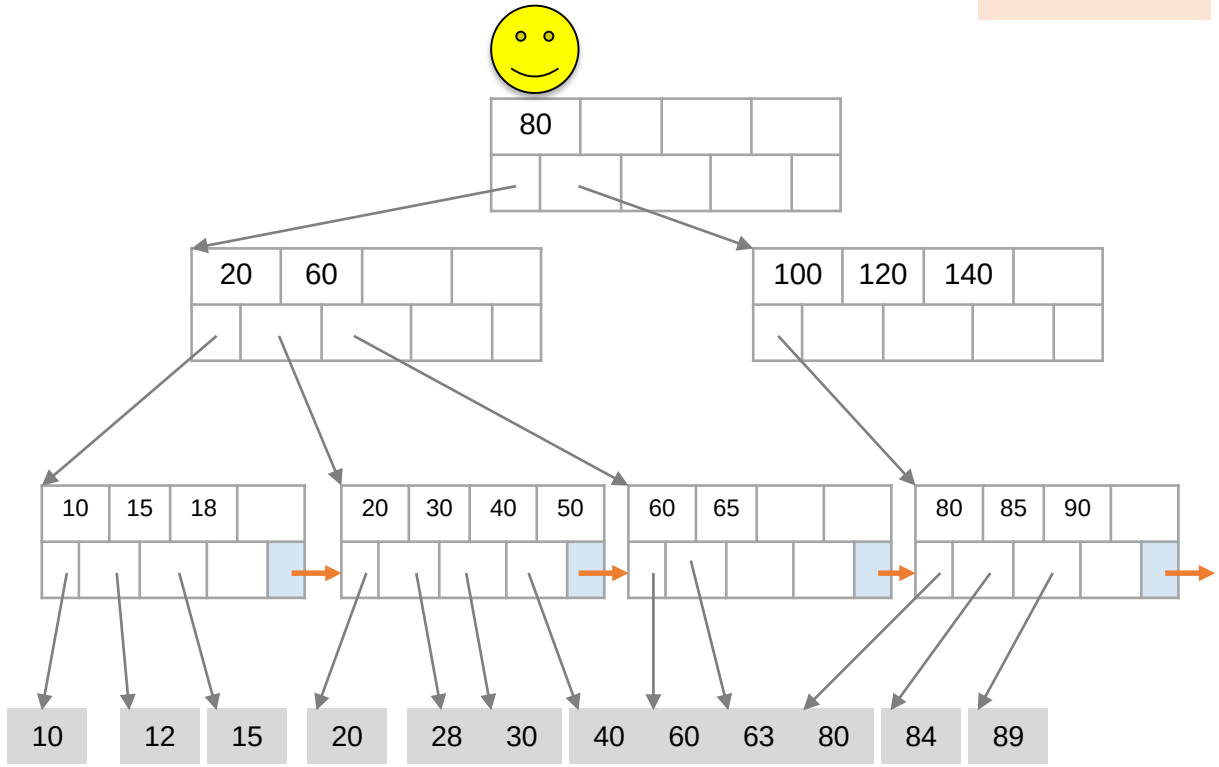
K = 30?

30 < 80

30 in [20,60)

30 in [30,40)

To the data!



B+ Tree 索引的范围查询（Range Search ）

- 查询
  K=[30,85]?

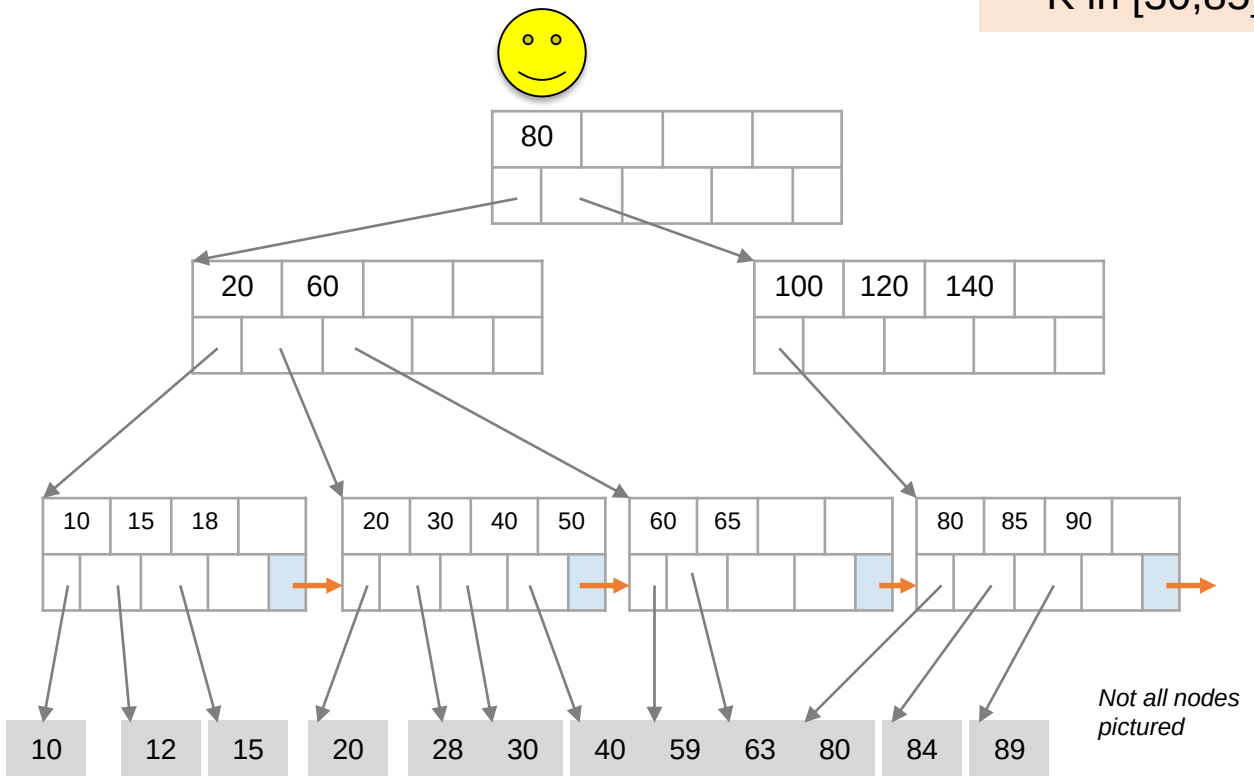
K in [30,85]?

30 < 80

30 in [20,60)

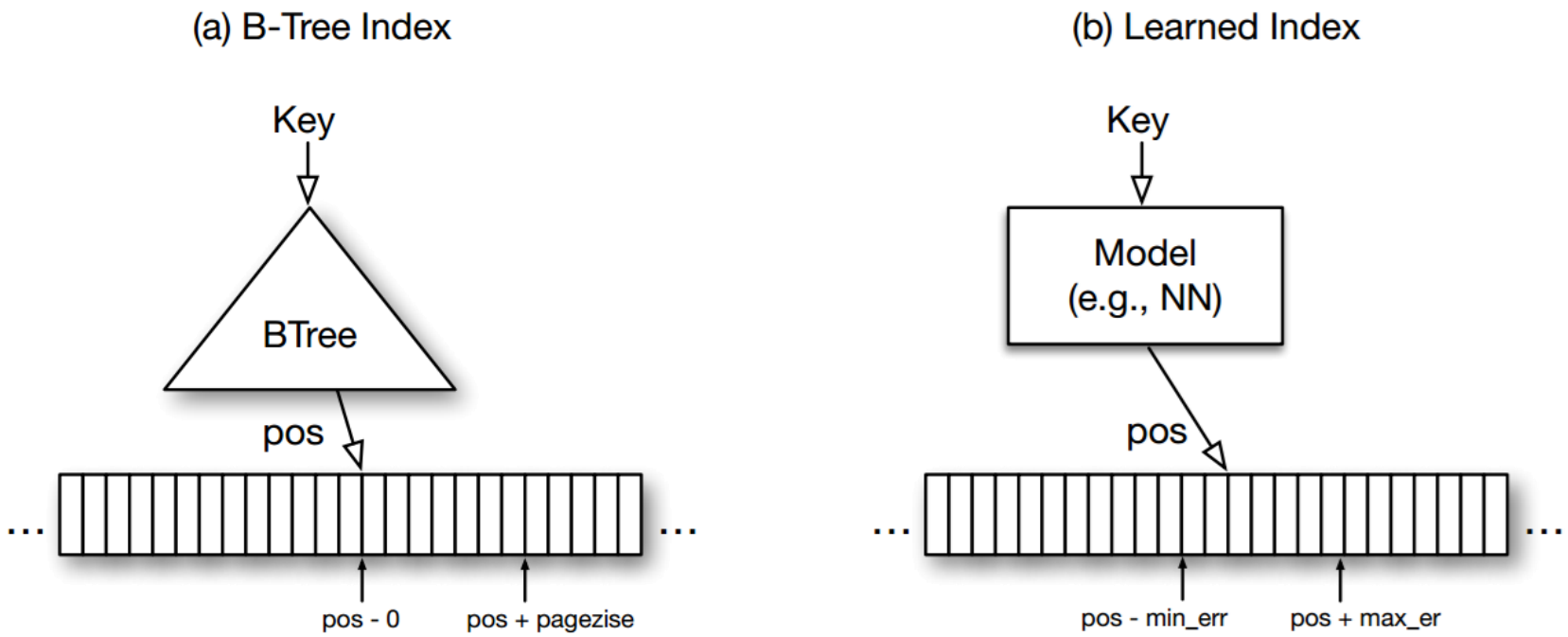
30 in [30,40)

To the data!



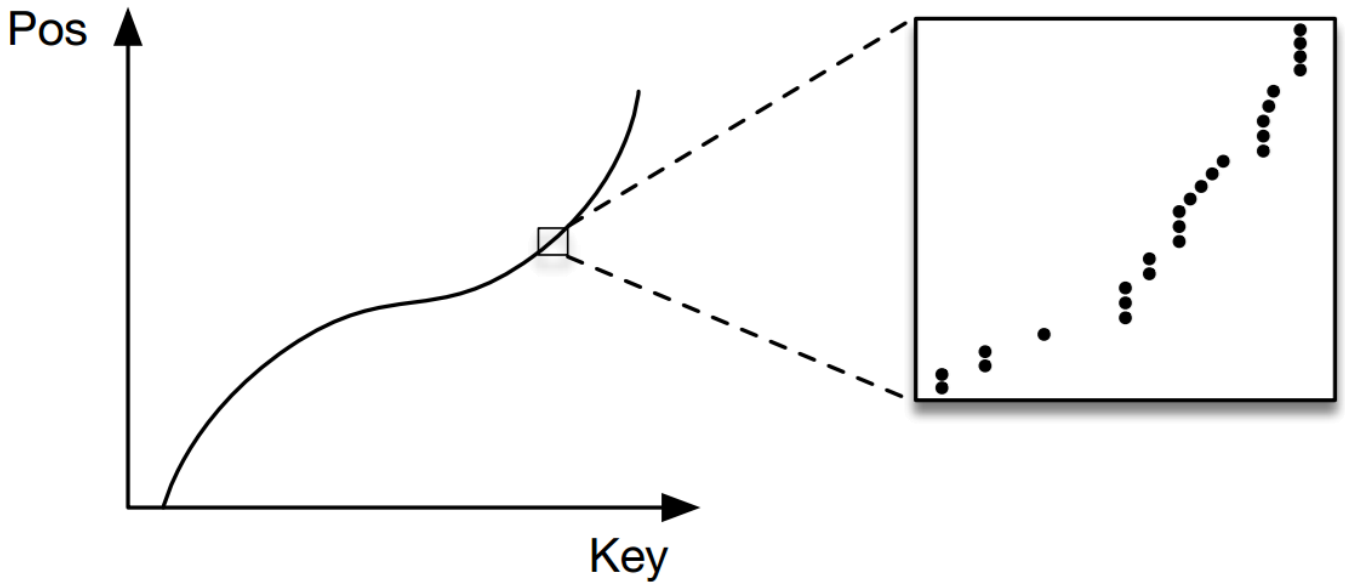
索引问题的抽象

- 有序数组的索引
- 映射：“键值”→ “位置”+ “置信区间”



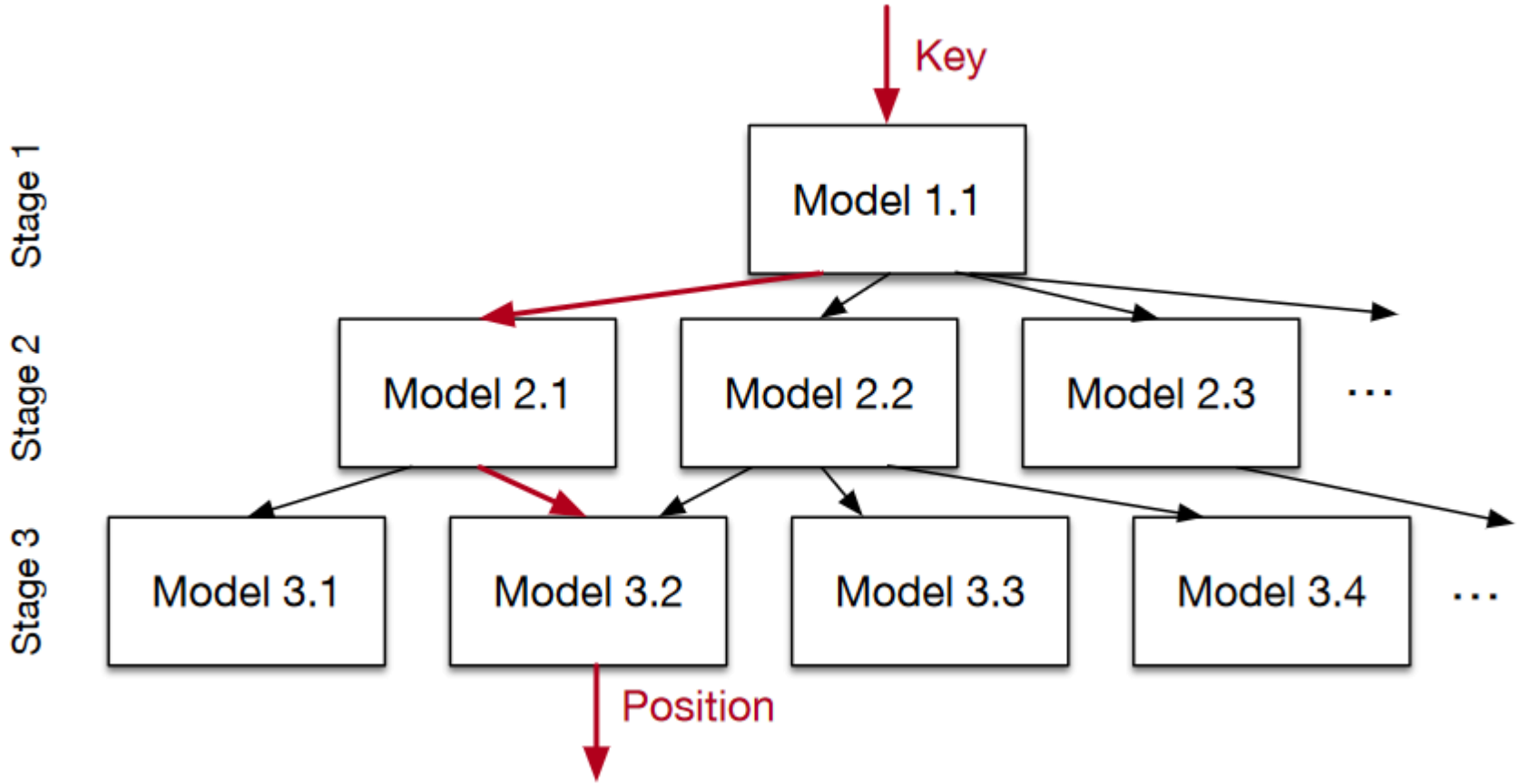
索引与CDF函数

- key的位置(pos) = 比key小的元素的个数
- key的百分位 = 比key小的元素的比例



习得索引框架

- 层层细分，节省参数



习得索引的效果

Type	Config	Search	Total (ns)	Model (ns)	Search (ns)	Speedup	Size (MB)	Size Savings	Model Err ± Err Var.
Btree	page size: 16	Binary	285	233	52	9%	99.66	700%	4 ± 0
	page size: 32	Binary	274	198	77	4%	49.83	300%	16 ± 0
	page size: 64	Binary	274	169	105	4%	24.92	100%	32 ± 0
	page size: 128	Binary	263	131	131	0%	12.46	0%	64 ± 0
	page size: 256	Binary	271	117	154	3%	6.23	-50%	128 ± 0
Learned Index	2nd stage size: 10,000	Binary	178	26	152	-32%	0.15	-99%	17060 ± 61072
		Quaternary	166	25	141	-37%	0.15	-99%	17060 ± 61072
	2nd stage size: 50,000	Binary	162	35	127	-38%	0.76	-94%	17013 ± 60972
		Quaternary	152	35	117	-42%	0.76	-94%	17013 ± 60972
	2nd stage size: 100,000	Binary	152	36	116	-42%	1.53	-88%	17005 ± 60959
		Quaternary	146	36	110	-45%	1.53	-88%	17005 ± 60959
	2nd stage size: 200,000	Binary	146	40	106	-44%	3.05	-76%	17001 ± 60954
		Quaternary	148	45	103	-44%	3.05	-76%	17001 ± 60954
Learned Index Complex	2nd stage size: 100,000	Binary	178	110	67	-32%	1.53	-88%	8 ± 33
		Quaternary	181	111	70	-31%	1.53	-88%	8 ± 33

Kraska, Beutel, Chi, et al. The case for learned index structures. 2017. https://arxiv.org/abs/1712.01208.

Learned System

Hardware

- A New Golden Age for Computer Architecture-cacm-2019

- RISC-V based open source CPU

Data Management

- Data Management Challenges in Production Machine Learning, sigmod 2017.

Abstract
The tutorial discusses data-management issues that arise in the context of machine learning pipelines deployed in production. Informed by our own experience with such large scale pipelines, we focus on issues related to understanding, validating, cleaning, and enriching training data. The goal of the tutorial is to bring forth these issues, draw connections to prior work in the database literature, and outline the open research questions that are not addressed by prior art.

Learned Database System

- Learning Sorting
- Learned Index
- Learned Cardinalities
- Learned Scheduling
- Learned Query Optimizer

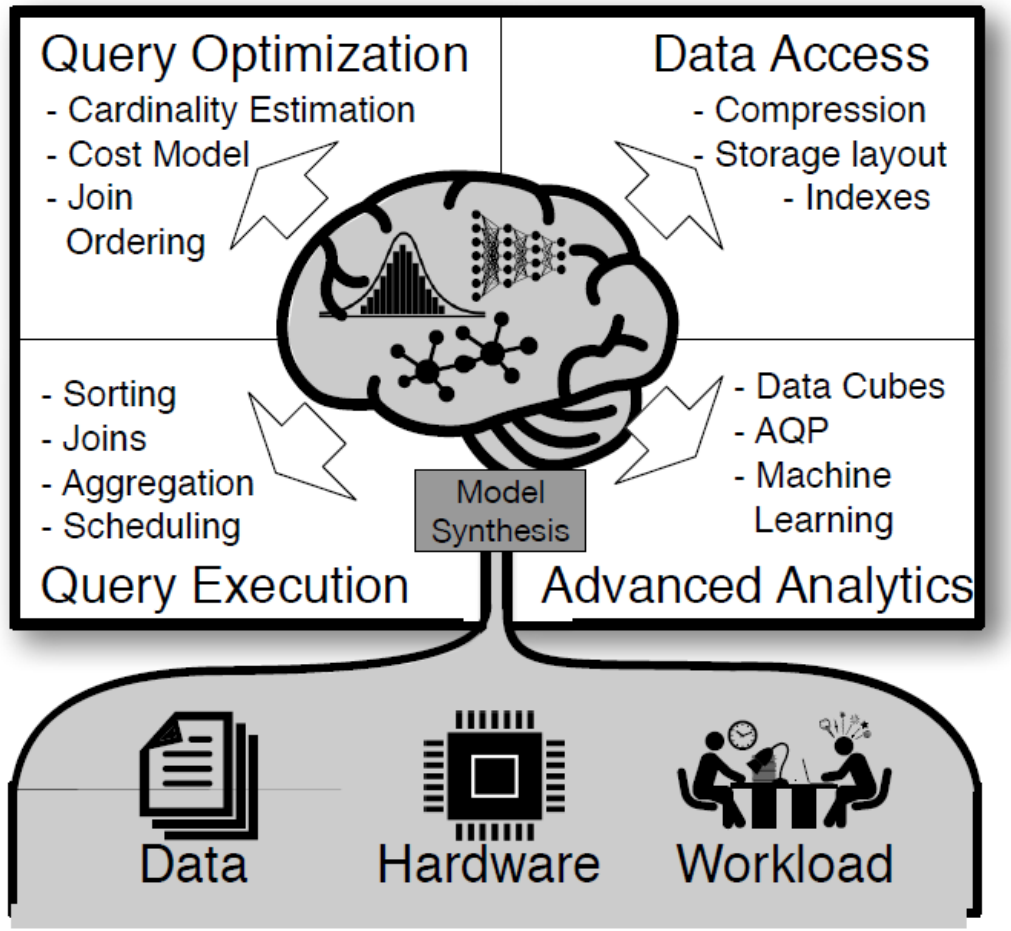


Figure 2: SageDB Overview

Frontier papers

- SageDB
  - A Learned Database System-CIDR-2019
- Self-Driving Database Management Systems-cidr-2017
- Learned Cardinalities
  - Estimating Correlated Joins with Deep Learning-cidr-2019
- Neo
  - A Learned Query Optimizer-arxiv-1904.03711
- Learning scheduling algorithms for data processing clusters-sigcomm-2019
- Machine Learning Sorting

谢谢指正！