高级大数据解析及应用简介

付彦伟 {yanweifu@fudan.edu.cn}

复旦大学 大数据学院

About Course

1. 10% class, 30% about the project, 60% final exam.
2. TA: Yu Xie (wechat: Y1314941 email: 15955038579@163.com); Shihan Ran (wechat: yishengyouyiyizuyi, email: 15307130424@fudan.edu.cn)
3. Office hours: Thursday afternoon 4:15-6:00pm, Xin Jinbo Building, Room 1307.
4. Homepage: http://yanweifu.github.io/courses/Data_analytics/index.html

More About the Course

1. Every three weeks one component: linux_shell, OS, Spark, GPU computing, etc.
2. We will also do some tutorial.
3. may invite friends from industry to talk about their commercial systems for Big data analysis;
4. Pr-required course: nope.

Academic Integrity (学术诚信)

- **Academic integrity** is the moral code or ethical policy of academia. This includes values such as avoidance of cheating or plagiarism; maintenance of academic standards; honesty and rigor in research and academic publishing. (https://en.wikipedia.org/wiki/Academic_integrity)
- No cheating and plagiarism,
  - How to define *Plagiarism*? We follow ACM Policy on Plagiarism.
  - 抄袭和被抄袭双方的成绩都将被取消.
  - 作业、报告、期末论文的署名原则：署你名字的工作必须由自己完成；允许讨论，但作业必须独立完成，并在作业中列出所有参与讨论的人。不允许其他任何形式的合作——尤其是与已经完成作业的同学“讨论”。
  - 这是学术底线。

Dene Plagiarism

Plagiarism manifests itself in a variety of forms, including

1. Verbatim copying, near-verbatim copying, or purposely paraphrasing portions of another author's paper;
2. Copying elements of another author's paper, such as equations or illustrations that are not common knowledge, or copying or purposely paraphrasing sentences without citing the source;
3. and Verbatim copying of portions of another author's paper with citing but not clearly differentiating what text has been copied (e.g., not applying quotation marks correctly) and/or not citing the source correctly.
4. Self-plagiarism is a related issue. In this document we define self-plagiarism as the verbatim or near-verbatim reuse of significant portions of one's own copyrighted work without citing the original source[2]. Note that self-plagiarism does not apply to publications based on the author's own previously copyrighted work (e.g., appearing in a conference proceedings) where an explicit reference is made to the prior publication[3]

The content of the courses

1. Something about OS and linux shell. We need to know how to use linux do some simple preprocessing and data analysis on single machine;
2. Something about cloud computing and Spark;
3. Something about other advanced Big Data Analytics.

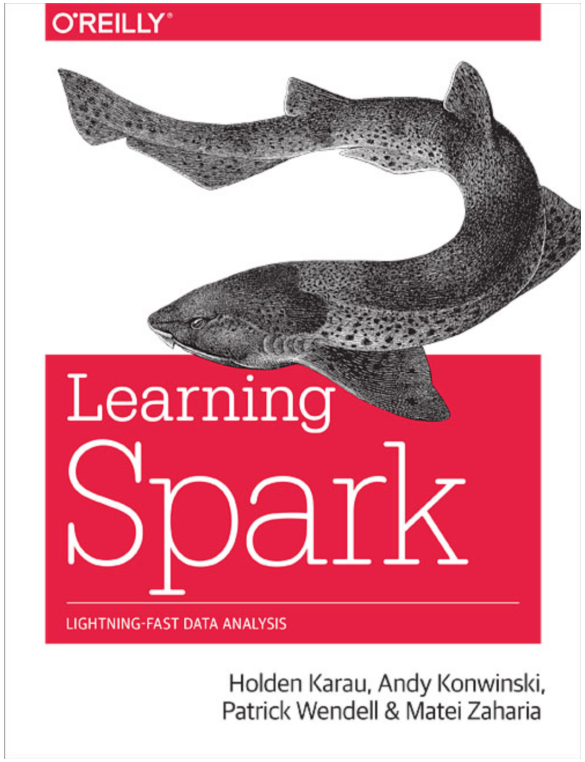
Enjoy

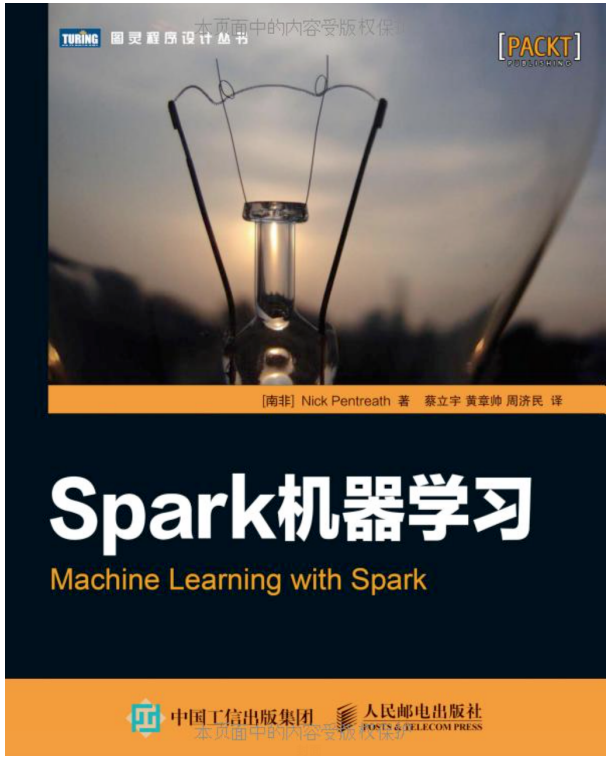
Yes, we need to know OS, linux commands and other stuffy, before really learning Spark; but not that difficulty.

Some Jokes

- 我找不到普通家庭也需要计算机的理由。 —Ken Olsen, co-founder of DEC Corp.
- 很多人预测1996年互联网产业将大规模增长。但我的预测是1996年互联网产业由于增长过于快速，将像超新星一样爆炸后而走向崩溃。 — Robert Metcalfe, co-founder of 3Com
- 全球垃圾邮件问题将在今后两年内得到解决。—Bill Gates, MS.
- 电视节目的流行时间不会超过半年，公众每晚会面对着一个小盒子，他们将对此感到厌倦。—Darryl Zanuck, 20 Century Fox
- 苹果已死。— Nathan Myhrvold, CTO of Microsoft
- 我觉得全球市场大概只需要5台计算机。—Thomas Watson, CEO of IBM

References about Spark





数据科学面临的挑战

数据科学界有几个硬道理是不能违背的。

第一，成功的分析中，绝大部份工作是数据预处理。

-- 需要对数据进行清洗、处理、融合、挖掘甚至其他操作

-- 一般在数据处理管道作业中，花在特征提取和选择上的时间比选择和实现算法时间长。

第二，迭代与数据科学紧密联系。

-- 建模、分析需要对一个数据集多次遍历。

第三，构建完表现卓越的模型不等于大功告成。

-- 比如数据推荐引擎、实时欺诈检测系统，需要定期甚至实时重建。

我们要区分：

试验环境：理解工作数据集的本质

生产环境：把模型打包成服务，用于现实世界的决策。

目录

背景简介： 机器学习、操作系统等

Background

A Recipe for Machine Learning

1. Given training data:

$$\{\boldsymbol{x}_i, \boldsymbol{y}_i\}_{i=1}^{N}$$

Face *Face* *Not a face*



2. Choose each of these:
   - Decision function

$$\hat{\boldsymbol{y}} = f_{\boldsymbol{\theta}}(\boldsymbol{x}_i)$$

   - Loss function

$$\ell(\hat{\boldsymbol{y}}, \boldsymbol{y}_i) \in \mathbb{R}$$

Examples: Linear regression, Logistic regression, Neural Network

Examples: Mean-squared error, Cross Entropy

Background

A Recipe for Machine Learning

1. Given training data:

$$\{\boldsymbol{x}_i, \boldsymbol{y}_i\}_{i=1}^{N}$$

2. Choose each of these:
   - Decision function

$$\hat{\boldsymbol{y}} = f_{\boldsymbol{\theta}}(\boldsymbol{x}_i)$$

   - Loss function

$$\ell(\hat{\boldsymbol{y}}, \boldsymbol{y}_i) \in \mathbb{R}$$

3. Define goal:

$$\boldsymbol{\theta}^* = \arg\min_{\boldsymbol{\theta}} \sum_{i=1}^{N} \ell(f_{\boldsymbol{\theta}}(\boldsymbol{x}_i), \boldsymbol{y}_i)$$

4. Train with SGD:
(take small steps opposite the gradient)

$$\boldsymbol{\theta}^{(t+1)} = \boldsymbol{\theta}^{(t)} - \eta_t \nabla \ell(f_{\boldsymbol{\theta}}(\boldsymbol{x}_i), \boldsymbol{y}_i)$$

A Recipe for Machine Learning

Background

Gradients

Backpropagation can compute this gradient!

And it's a **special case of a more general algorithm** called reverse-mode automatic differentiation that can compute the gradient of any differentiable function efficiently!

1. Given training da

$$\{\boldsymbol{x}_i, \boldsymbol{y}_i\}_{i=1}^{N}$$

2. Choose each of these:
   - Decision function

$$\hat{\boldsymbol{y}} = f_{\boldsymbol{\theta}}(\boldsymbol{x}_i)$$

   - Loss function

$$\ell(\hat{\boldsymbol{y}}, \boldsymbol{y}_i) \in \mathbb{R}$$

opposite the gradient)

$$\theta^{(t} \leftarrow \theta^{(t)} - \eta_t \nabla \ell(f_{\boldsymbol{\theta}}(\boldsymbol{x}_i), \boldsymbol{y}_i)$$

A Recipe for Machine Learning

Background

1. Explore a **new class of decision functions** (Neural Networks)
2. Consider **variants of this recipe** for training

2. Choose each of these:
   - Decision function

$$\hat{y} = f_{\boldsymbol{\theta}}(\boldsymbol{x}_i)$$

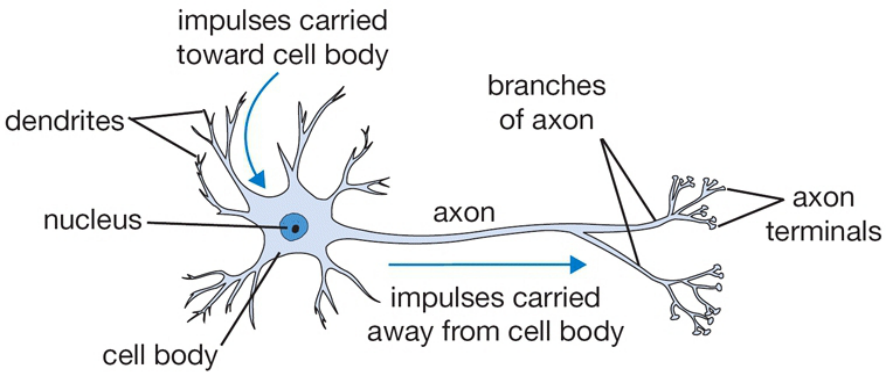
   - Loss function

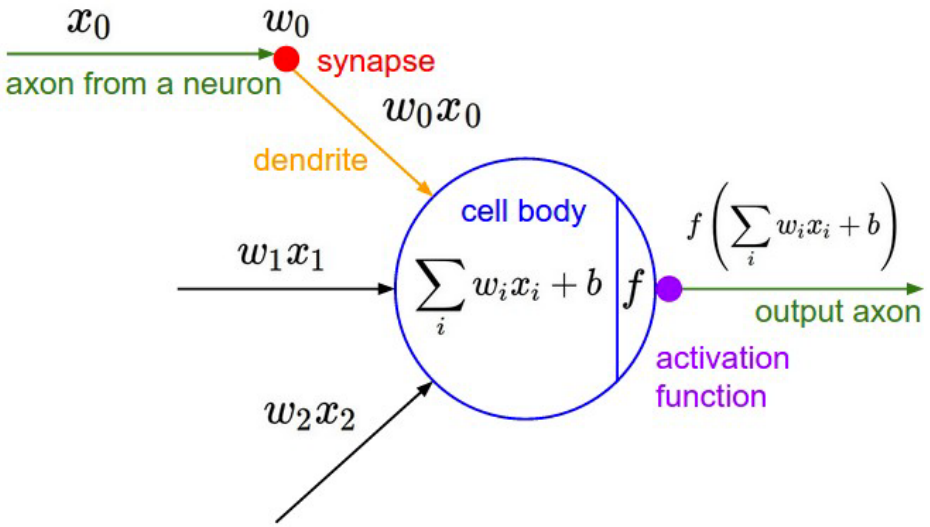
$$\ell(\hat{\boldsymbol{y}}, \boldsymbol{y}_i) \in \mathbb{R}$$

Train with SGD:
(Take small steps opposite the gradient)

$$\boldsymbol{\theta}^{(t+1)} = \boldsymbol{\theta}^{(t)} - \eta_t \nabla \ell(f_{\boldsymbol{\theta}}(\boldsymbol{x}_i), \boldsymbol{y}_i)$$

Human Neuron Vs. Math Neuron





The basic computational unit of the brain: Neuror

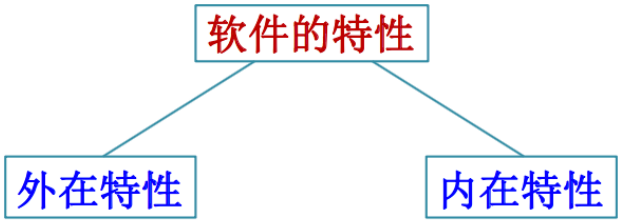
Many machine learning methods inspired by biology, e.g. the (human) brain

A mathematical model of the neuron in a neural network;

Neural networks define functions of the inputs (hidden features), computed by neurons

从不同角度认知操作系统

1.作为软件来看的观点



软件即是服务

界面/接口：使用方式
如：命令、系统调用等

软件的结构
- ✓ 由哪几个部分组成？
- ✓ 各部分之间的关系？

怎样管理资源？

- 跟踪记录资源使用状况
  - 如：哪些资源空闲，好坏与否，被谁使用，使用多长时间等
- 分配和回收资源（资源分配策略与算法）
  - 静态分配策略
  - 动态分配策略 √
- 提高资源利用率
- 保护
- 协调多个进程对资源请求的冲突

公平竞争
防止非法使用

2.资源管理的观点

自底向上 → 操作系统 是 资源的管理者

硬件资源：

CPU，内存，设备（I/O设备、磁盘、时钟、网络接口等）

软件资源：

磁盘上的文件、信息

从资源管理的角度— 五大基本功能

- 进程和线程管理（CPU管理、调度）
  进程控制、同步互斥、通信、调度
- 存储管理
  分配/回收、地址映射、存储保护、内存扩充
- 文件管理
  文件目录、磁盘空间、文件系统布局、存取控制
- 设备管理
  设备驱动、分配回收、缓冲技术
- 用户接口
  系统命令、编程接口

从不同角度认知操作系统

3.进程的观点

从操作系统运行的角度动态的观察操作系统

按照这一观点：

- 操作系统 是 由一些可同时、独立运行的进程 和 一个对这些进程进行协调的核心组成

进程：完成某一特定功能的程序
是程序的一次执行过程
动态的、有生命的，存在/消亡

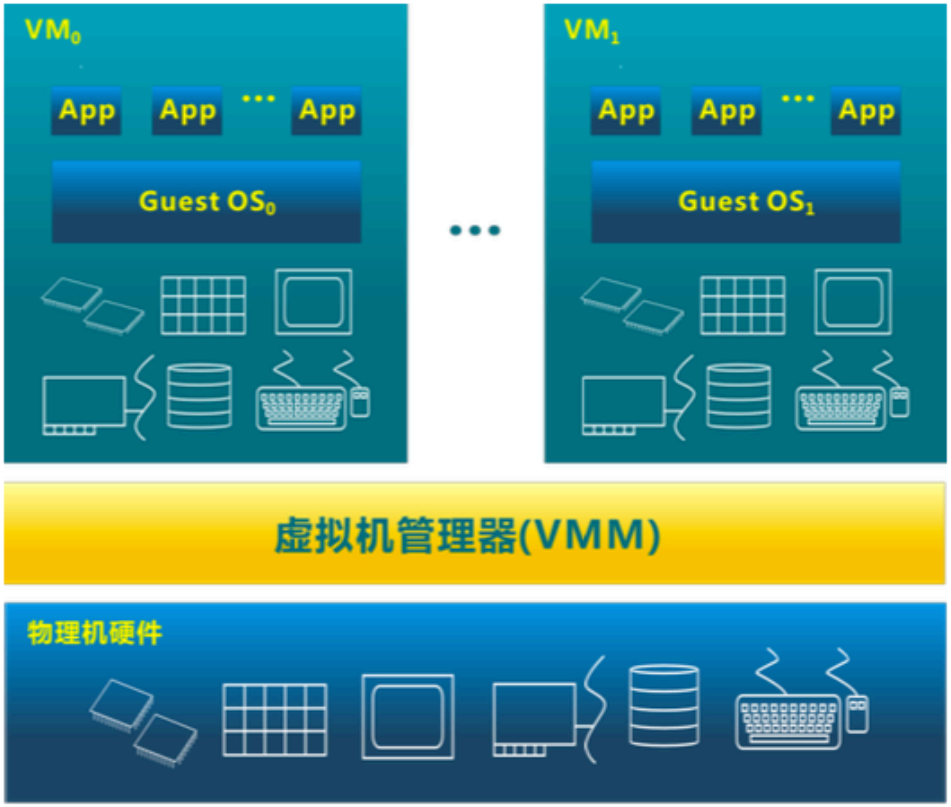
4.虚机器观点

从操作系统内部结构来看：

- ✓ 把操作系统分成若干层 分层结构
- ✓ 每一层完成其特定功从而构成一个虚机器，并对上一层提供支持
- ✓ 通过逐层功能扩充，最终完成整个操作系统虚机器
- ✓ 而操作系统虚机器向用户提供各种功能，完成用户请求

VMM(虚拟机管理器)

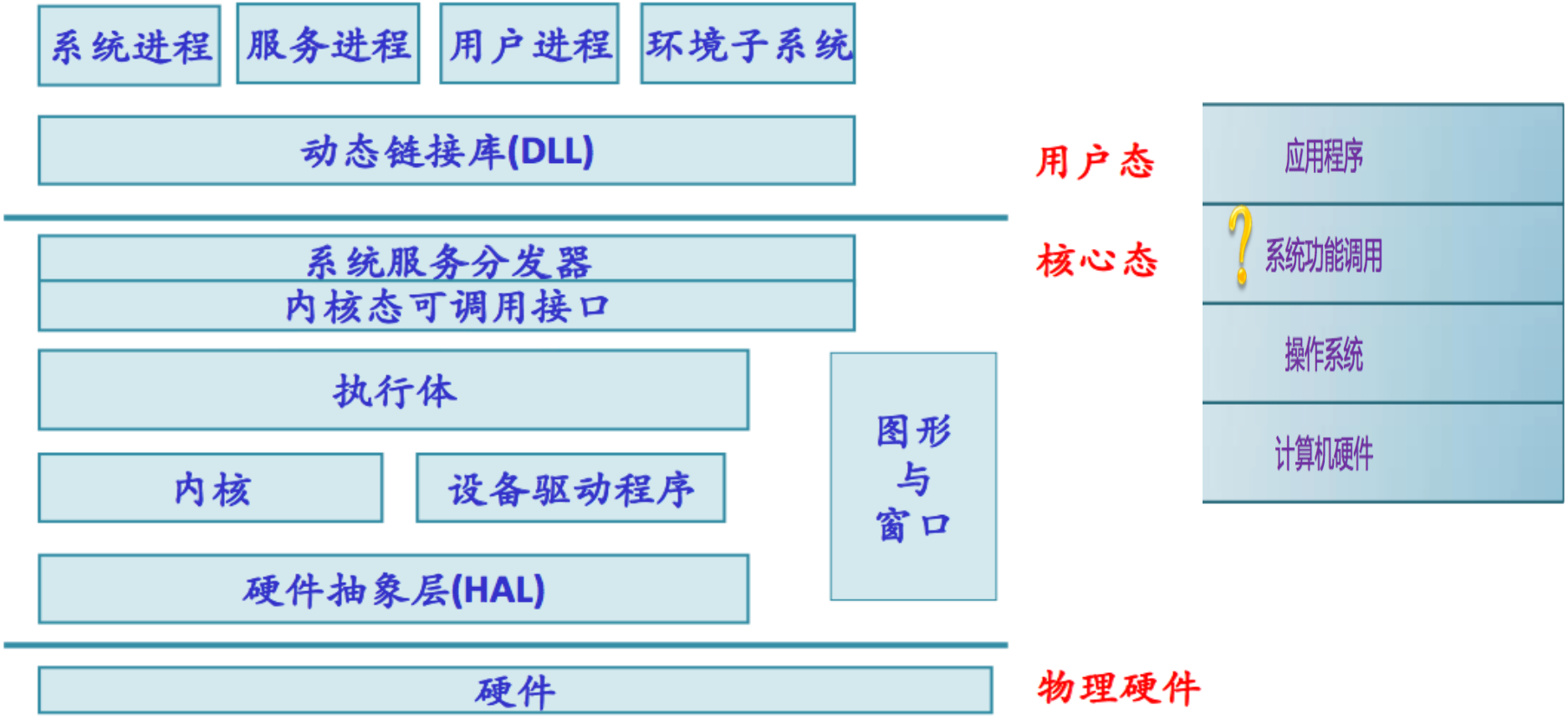
无虚拟机：单操作系统拥有所有硬件资源



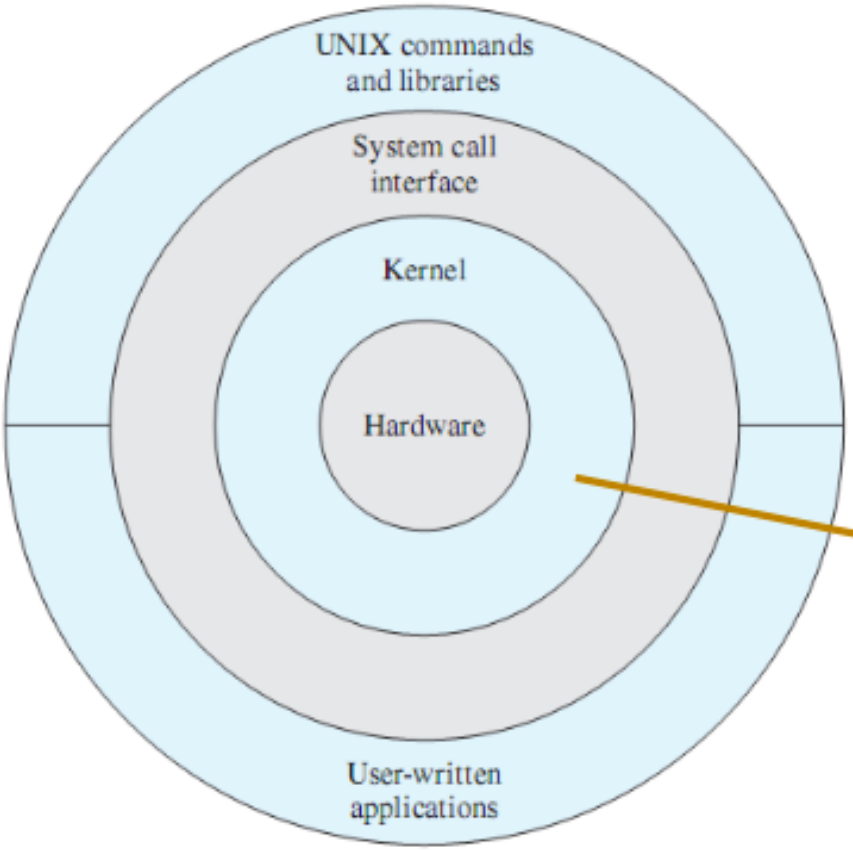
有虚拟机：多操作系统共享硬件资源

Acknowledgement: 清华大学 向勇、陈渝

Windows操作系统的体系结构

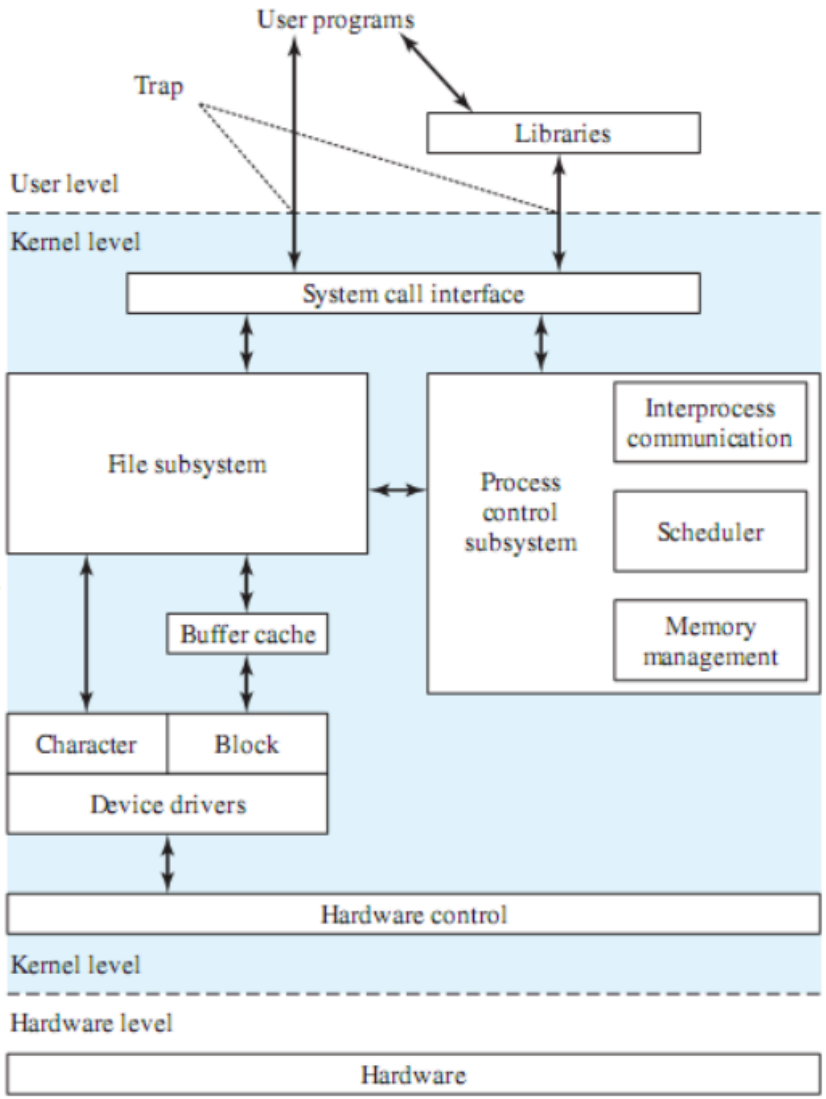


UNIX 操作系统的体系结构

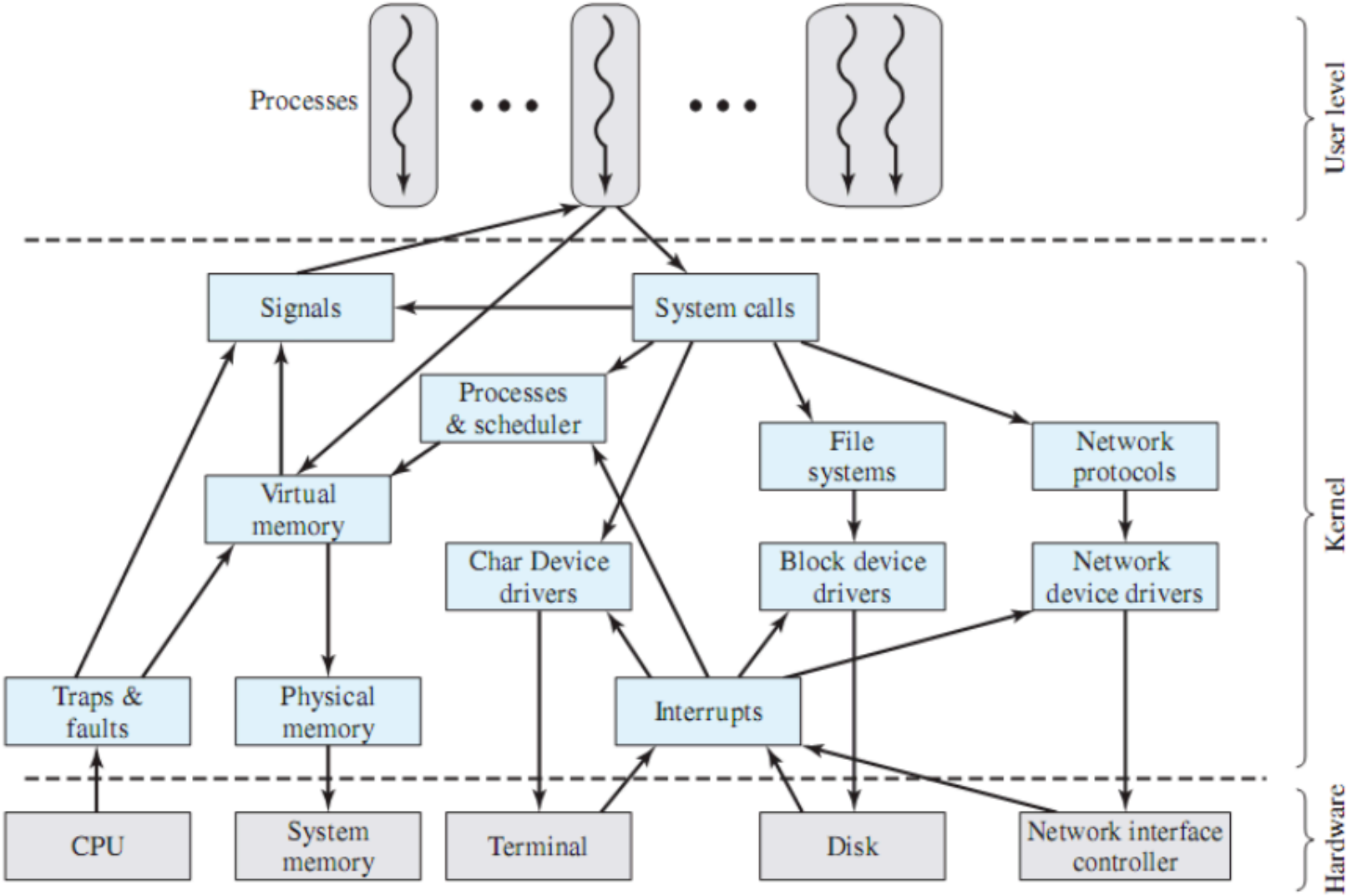


层次结构

内核结构



Linux 架构


Processes
User level
Signals
System calls
Processes & scheduler
Virtual memory
File systems
Network protocols
Char Device drivers
Block device drivers
Network device drivers
Traps & faults
Physical memory
Interrupts
Kernel
CPU
System memory
Terminal
Disk
Network interface controller
Hardware

Linux操作系统内核

Android 架构

Android应用程序

Email客户端，SMS短消息程序，日历，地图，浏览器，联系人管理等

应用程序框架

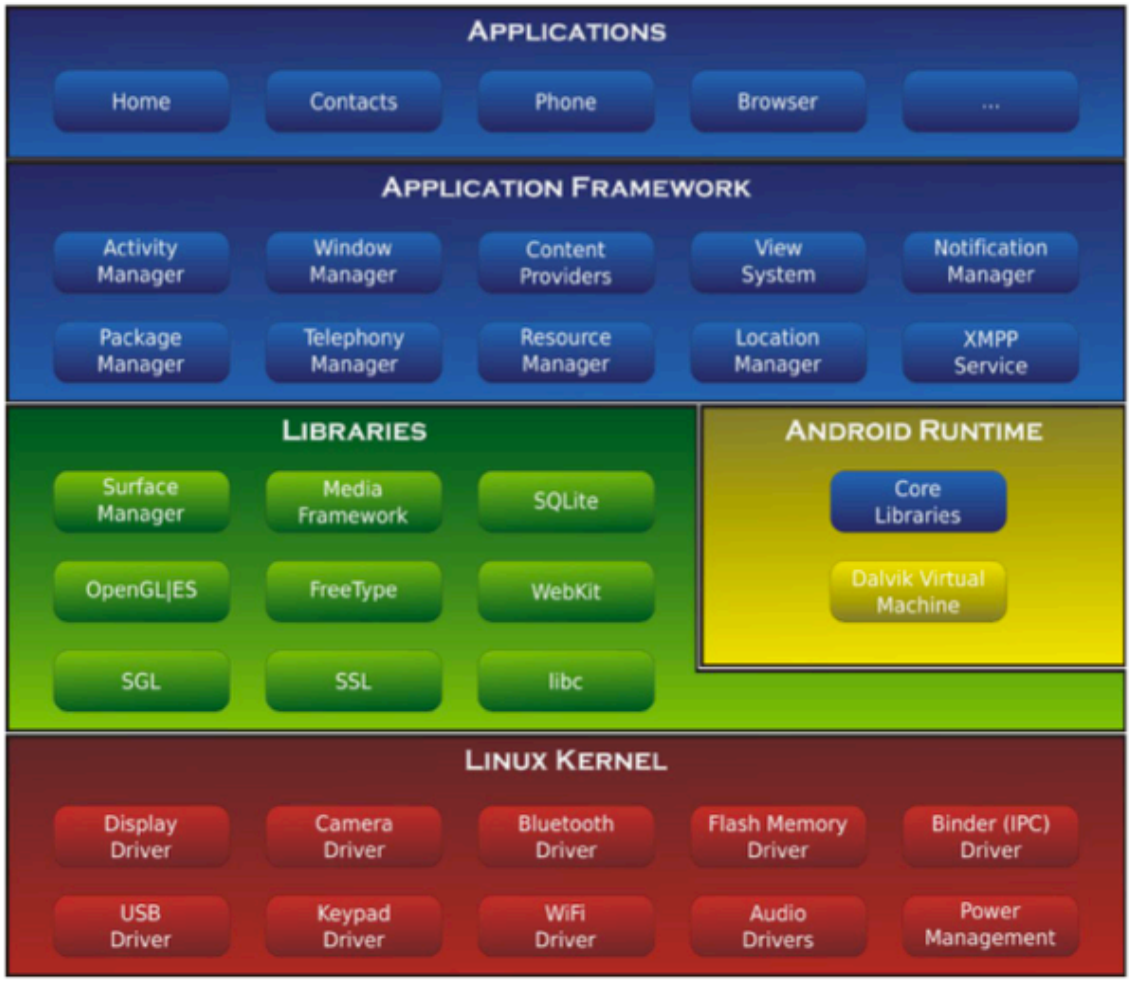
开发者可以完全使用核心应用程序所使用的框架APIs
视图、内容提供者、资源管理器等

库

Android包含一个C/C++库的集合，供Android系统的各个组件使用。
如：系统C库、3D库、SQLite、媒体库等

Linux内核

提供核心系统服务，例如：安全、内存管理、进程管理、网络堆栈、驱动模型

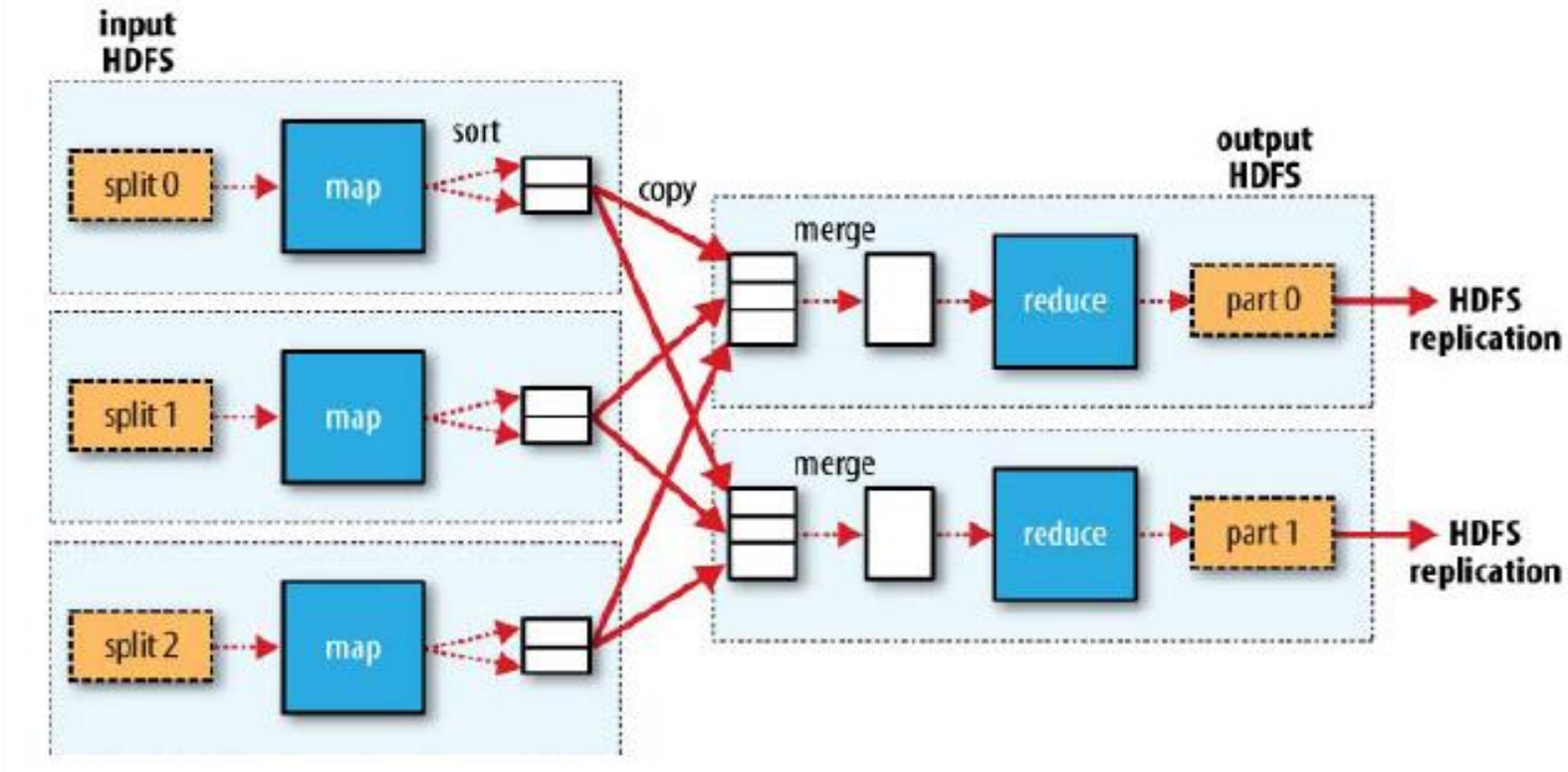


大数据计算框架 – Hadoop构架

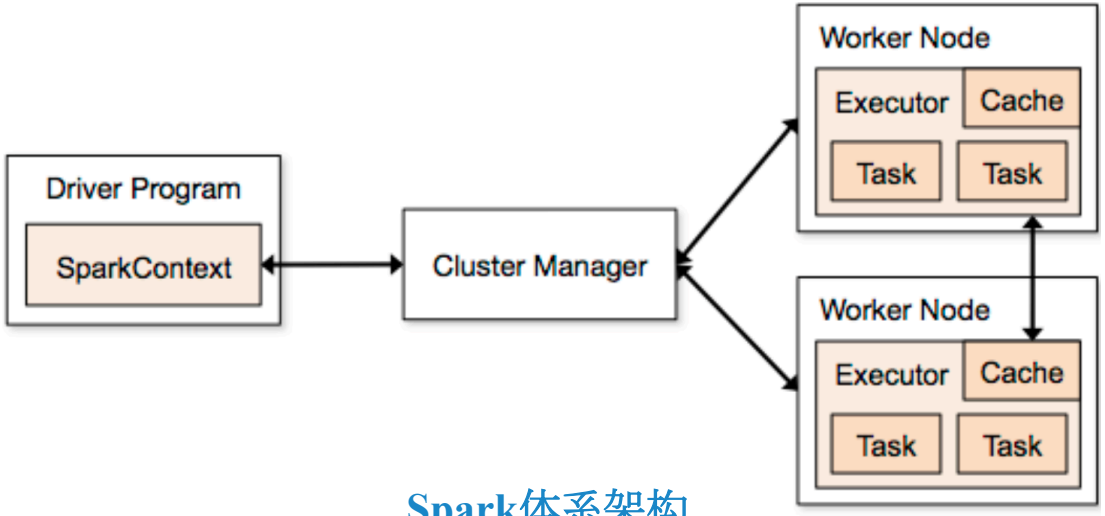
统一框架Hadoop，隐藏系统层细节

- 统一的计算框架无需考虑数据存储、划分、分发、结果收集、错误恢复等诸多细节

Hadoop

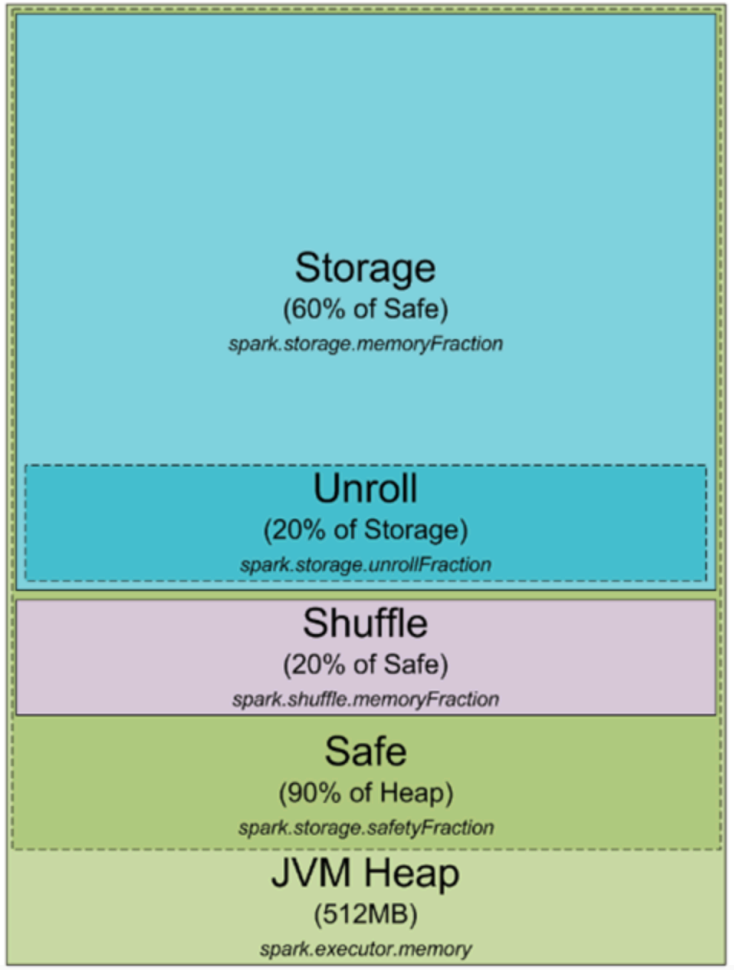


大数据计算框架 -- Spark 构架



Spark体系架构

	Heap(堆)	Stack(栈)
JVM中的功能	内存数据区	内存指令区
存储数据	对象实例(1)	基本数据类型, 指令代码,常量,对象的引用地址(2)



Spark Java Virtual Machine 堆内存分配图

Figs come from http://www.csuldw.com/2016/07/23/2016-07-23-spark-architecture/

1 人工智能与大数据简介

一些有意思的“断言”

- 我找不到普通家庭也需要计算机的理由。 —Ken Olsen, co-founder of DEC Corp.
- 很多人预测1996年互联网产业将大规模增长。但我的预测是1996年互联网产业由于增长过于快速，将像超新星一样爆炸后而走向崩溃。 — Robert Metcalfe, co-founder of 3Com
- 全球垃圾邮件问题将在今后两年内得到解决。—Bill Gates, MS.
- 电视节目的流行时间不会超过半年，公众每晚会面对着一个小盒子，他们将对此感到厌倦。—Darryl Zanuck, 21 Century Fox
- 苹果已死。 — Nathan Myhrvold, CTO of Microsoft
- 我觉得全球市场大概只需要5台计算机。—Thomas Watson, CEO of IBM


GARRY
KASPAROV
DEEP
BLUE


$24,000
Who is Stoker?
(I FOR ONE WELCOME OUR NEW COMPUTER OVERLORDS)
$1,000
$77,147
Who is Bram Stoker?
$ 17,973
$21,600
WHO IS BRAM STOKER?
$5600


ALPHAGO
01:59:50
AlphaGo
Lee Sedol

1.1 大数据研究的历史

起源：1880年人口普查、打孔机

发展：1957年国际地球物理年、“大科学”

成熟：2005年Hadoop项目诞生、“大商业”

SOMETHING TO TELL YOU ABOUT THIS

1.2 “大”数据

1 Byte = 8 bit

1 KB = 1,024 Bytes 一则短篇故事内容

1 MB = 1,024 KB 一则短篇小说文字内容

1 GB = 1,024 MB 贝多芬第五乐章交响曲的乐谱内容

1 TB = 1,024 GB 一家大型医院中所有X光图片资讯量

1 PB = 1,024 TB 50%全美学术研究图书馆藏书资讯内容

1 EB = 1,024 PB 5EB相当于至今全世界人类所讲过的话语

1 ZB = 1,024 EB 全球数据总量2013年4.4ZB，2020年40ZB

1 YB = 1,024 ZB 相当于7000位人体内的微细胞总和

1 BB = 1,024 YB

1 NB = 1,024 BB

1 DB = 1,024 NB

不可说 10^{24}

不可说转 $10^{930459597049441111032664942196241 2032}$

不可说不可说 $10^{186091919409888222065329884392482 4064}$

不可说不可说转 $10^{372183838819776444413065976878496 48128}$

不可说不可说不可说 $10^{744367677639552888826131953756992 96256}$

1.3 大数据应用举例

社会大数据

- 洛杉矶警察局和加利福尼亚大学合作利用大数据**预测犯罪**的发生

商业大数据

- 梅西百货基于SAS的系统对多达7300万种货品进行**实时调价**

城市大数据

- 麻省理工学院利用手机定位数据和交通数据建立**城市规划**

电信大数据

- 美国Verizon分析观众来源地，对棒球、篮球队提供观众对**赞助商喜好**
- 日本NTT Docomo建立**用户资料库**实现活用
- 德国Vodafone开放API，向合作方提供部分用户匿名**地理位置数据**
- 法国Orange分析评估用户消费数据，改善法国电信**服务质量**
- 西班牙电信推出智慧足迹，基于完全匿名和聚合的移动网络数据，提供政企客户**人流量**的关键影响因素

Volume

数据量巨大

全球在2010 年正式进入ZB 时代，IDC预计到2020 年，全球将总共拥有35ZB 的数据量

Variety

结构化数据、半结构化数据和非结构化数据

如今的数据类型早已不是单一的文本形式，订单、日志、音频，能力提出了更高的要求

Value

沙里淘金，价值密度低

以视频为例，一部一小时的视频，在连续不间断监控过程中，可能有用的数据仅仅只有一两秒。如何通过强大的机器学习、人工智能算法更迅速地完成数据的价值“提纯”是目前大数据汹涌背景下亟待解决的难题

Velocity

实时获取需要的信息

大数据区分于传统数据最显著的特征。如今已是ZB时代，在如此海量的数据面前，处理数据的效率就是企业的生命

Veracity

准确性

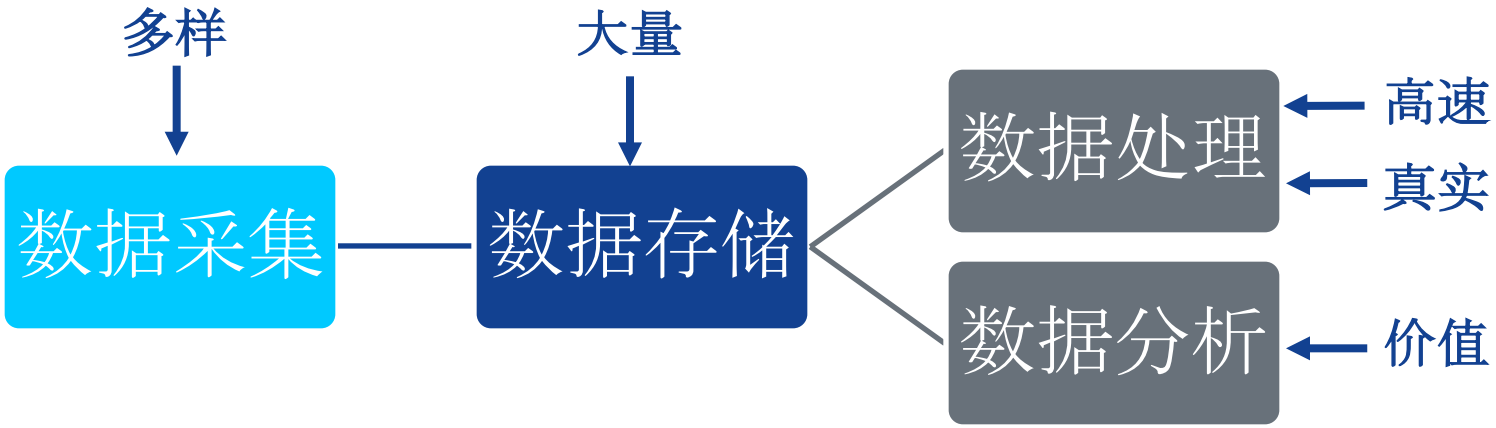
大数据处理结果要有准确性

1.4 大数据技术

大数据的特点

- Volume（大量）- 规模急剧膨胀
- Velocity（高速）- 变化快速瞬息
- Variety（多样）- 来源多种多样
- Value（价值）- 价值超乎想象
- Veracity（真实性） - 关系错综复杂

大数据技术流程

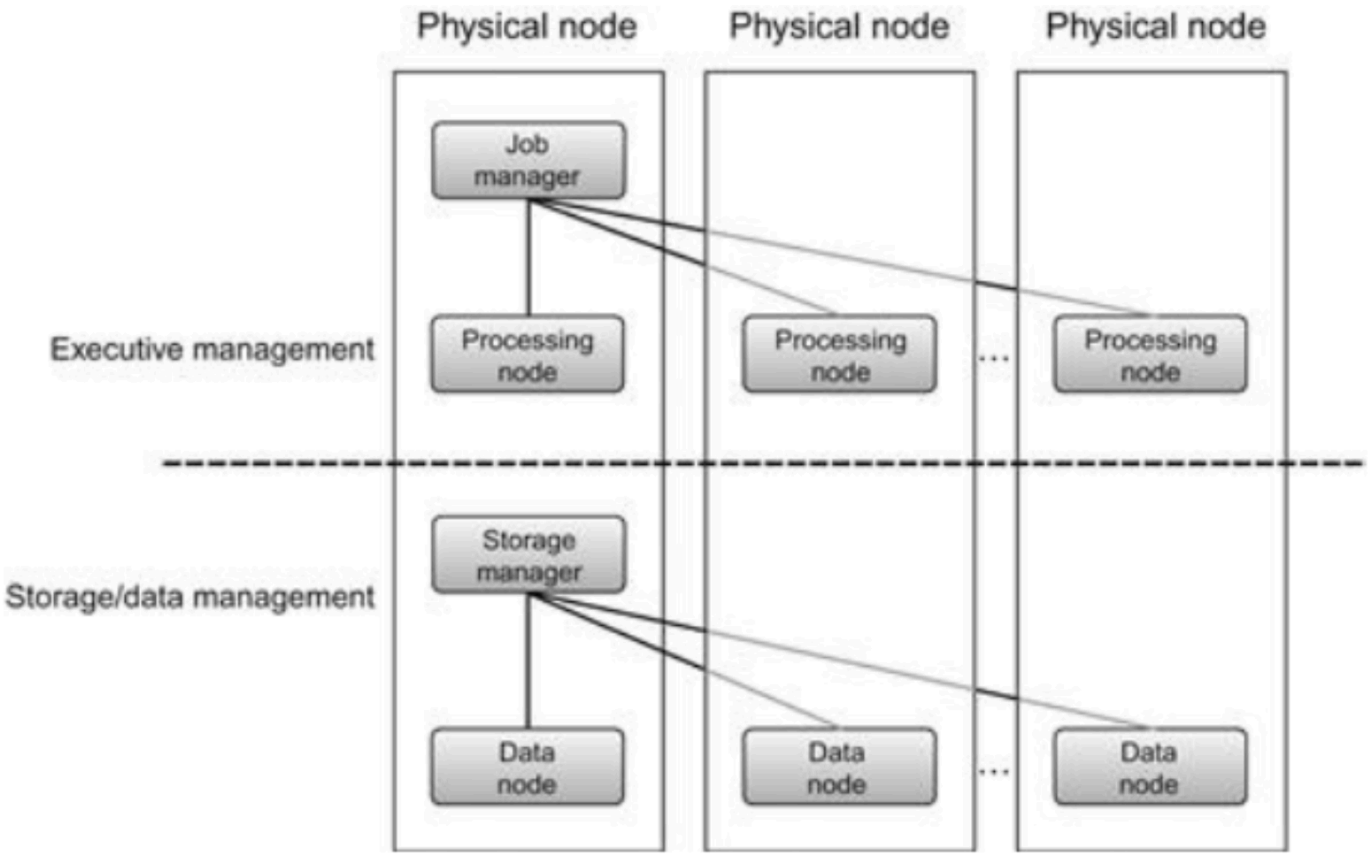


1.5 哪些地方需要大数据？

“Big Data Analytics”, David Loshin, 2013

大数据的计算资源

Processing capability: CPU, processor, or node.
Memory
Storage
Network



高性能计算 V.S. 分布式计算

Aspect	Typical Scenario	Big Data
Application development	Applications that take advantage of massive parallelism developed by specialized developers skilled in high-performance computing, performance optimization, and code tuning	A simplified application execution model encompassing a distributed file system, application programming model, distributed database, and program scheduling is packaged within Hadoop, an open source framework for reliable, scalable, distributed, and parallel computing
Platform	Uses high-cost massively parallel processing (MPP) computers, utilizing high-bandwidth networks, and massive I/O devices	Innovative methods of creating scalable and yet elastic virtualized platforms take advantage of clusters of commodity hardware components (either cycle harvesting from local resources or through cloud-based utility computing services) coupled with open source tools and technology
Data management	Limited to file-based or relational database management systems (RDBMS) using standard row-oriented data layouts	Alternate models for data management (often referred to as NoSQL or “Not Only SQL”) provide a variety of methods for managing information to best suit specific business process needs, such as in-memory data management (for rapid access), columnar layouts to speed query response, and graph databases (for social network analytics)
Resources	Requires large capital investment in purchasing high-end hardware to be installed and managed in-house	The ability to deploy systems like Hadoop on virtualized platforms allows small and medium businesses to utilize cloud-based environments that, from both a cost accounting and a practical perspective, are much friendlier to the bottom line

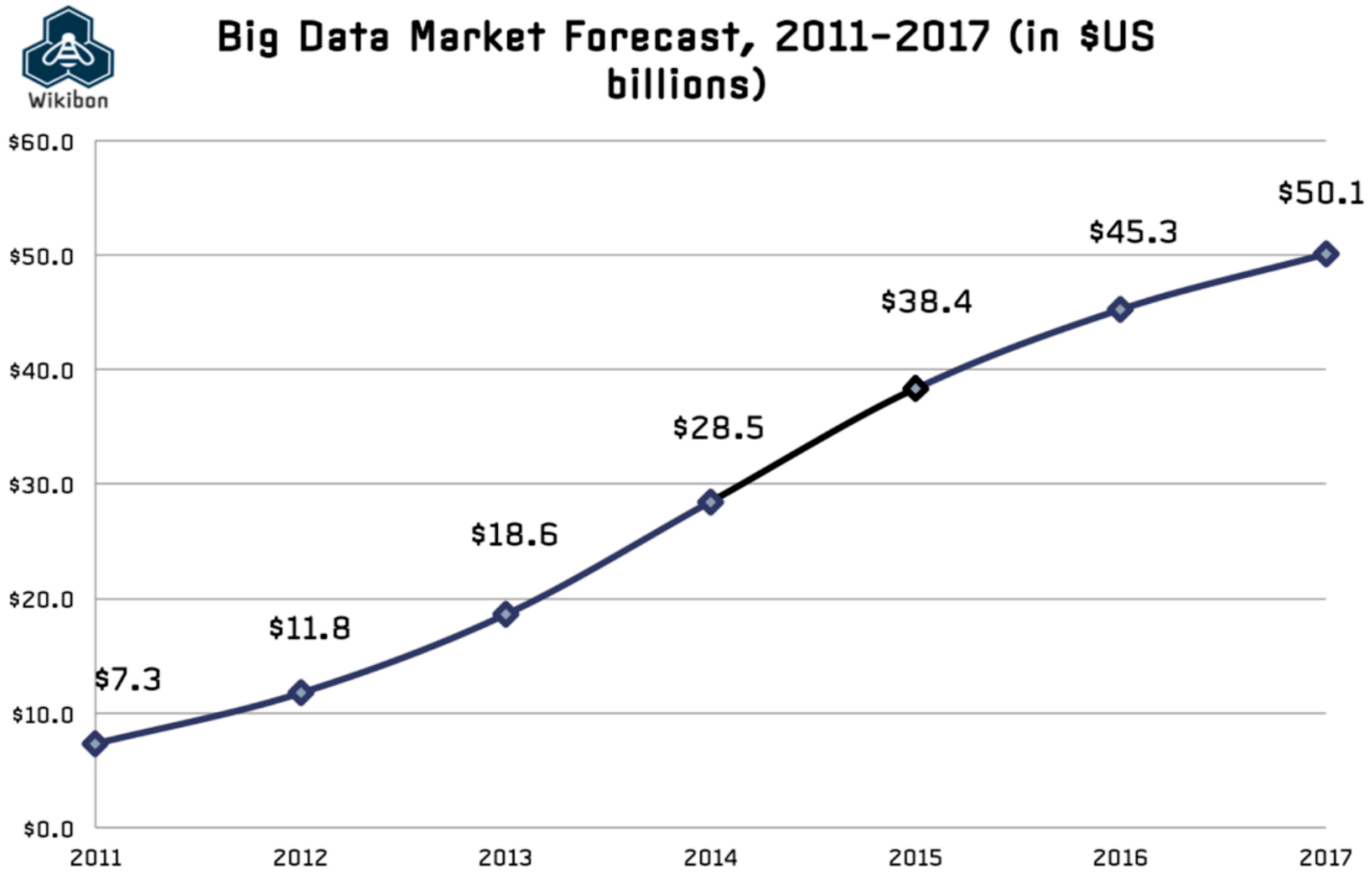
大数据计算需要的技术

- Massive Parallelism
- Huge Data Volumes Storage
- Data Distribution
- High-Speed Networks
- High-Performance Computing
- Task and Thread Management
- Data Mining and Analytics
- Data Retrieval
- Machine Learning
- Data Visualization

➔ Techniques exist for years to decades. Why did Big Data become hot now?

- More data are being collected and stored
- Open source code
- Commodity hardware

大数据市场



http://wikibon.org/wiki/v/Big_Data_Vendor_Revenue_and_Market_Forecast_2013-2017

大数据市场的细分(Breakdown)

USD: billions	2014	2015	2016	2017
Big Data XaaS Revenue	$1.71	$2.43	$2.87	$3.19
Big Data Professional Services Revenue	$9.24	$12.31	$14.06	$15.30
Big Data Application (Analytic and Transactional) Revenue	$3.24	$4.94	$6.05	$6.89
Big Data NoSQL Database Revenue	$0.73	$1.14	$1.41	$1.62
Big Data SQL Database Revenue	$2.00	$2.48	$2.74	$2.91
Big Data Infrastructure Revenue	$0.67	$0.93	$1.08	$1.19
Big Data Networking Revenue	$0.67	$0.89	$1.02	$1.11
Big Data Storage Revenue	$4.39	$5.85	$6.68	$7.27
Big Data Compute Revenue	$5.23	$6.70	$7.50	$8.06
Total Big Data Revenue	$27.9	$37.7	$43.4	$47.5

http://wikibon.org/wiki/v/Big_Data_Database_Revenue_and_Market_Forecast_2012-2017

2 大数据商业应用案例

几类大数据应用的类别

Big Data Exploration

Find, visualize, understand all big data to improve decision making

Enhanced 360º View of the Customer

Extend existing customer views (MDM, CRM, etc) by incorporating additional internal and external information sources

Security/Intelligence Extension

Lower risk, detect fraud and monitor cyber security in real-time

Operations Analysis

Analyze a variety of machine data for improved business results

Data Warehouse Augmentation

Integrate big data and data warehouse capabilities to increase operational efficiency

Big Data Analytics Example Use Cases

1. Expertise Location
2. Recommendation
3. Commerce
4. Financial Analysis
5. Social Media Monitoring
6. Telco Customer Analysis
7. Watson
8. Data Exploration and Visualization
9. Personalized Search
10. Anomaly Detection (Espionage, Sabotage, etc.)
11. Fraud Detection
12. Cybersecurity
13. Sensor Monitoring (Smarter another Planet)
14. Celluar Network Monitoring
15. Cloud Monitoring
16. Code Life Cycle Management
17. Traffic Navigation
18. Image and Video Semantic Understanding
19. Genomic Medicine
20. Brain Network Analysis
21. Data Curation
22. Near Earth Object Analysis

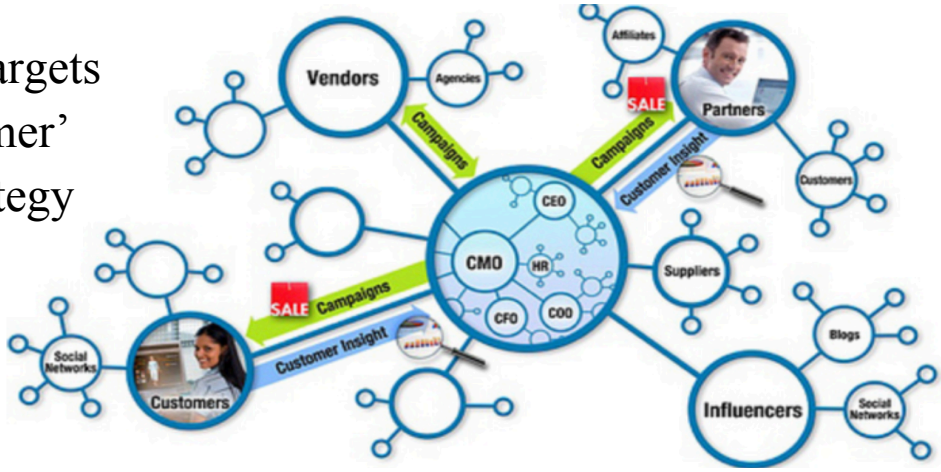
Financial Anomaly Detection

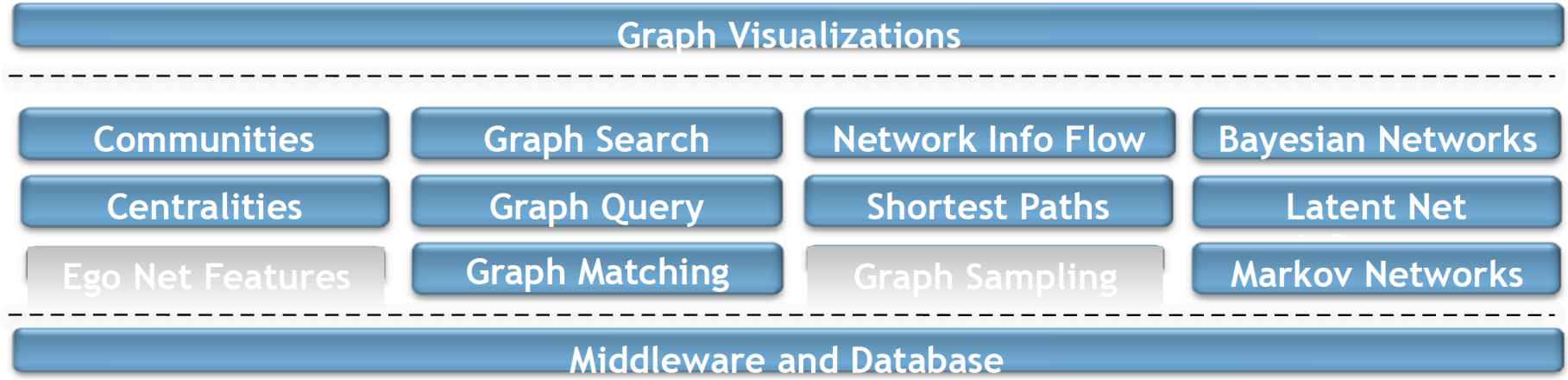
Model validation	*Entity process*	*Social net analysis*	**Analytic Engines**			*Predictive analysis*	*Peer group analysis*	*Sim. what/if analysis*
AML	**Anti-fraud**	**Non-Performing Loans**	**ABC**	**Trade Surveillance**	**PRG**	**Control room**	**Employee compliance**	**GRC**
Transaction monitoring	Internal fraud	Payment screening	Gifts & entertainment	Communication monitoring	Position disclosure	Conflict management	Employee trading	Risk assessment
Client screening	External fraud	List management	Bus dev consultants	Digital surveillance		List management	G&E	Control backtesting
CRE (list rating)	Unauthorized trading		Hiring practices	Information barrier monitoring		Research clearance	Annual attestations	Monitor testing
List management				Market abuse			Business interests	Regulatory charge

Alerting Engine	**Anomaly Detection**	**Machine Learning**	**Case Mgmt,& Workflow**
Natural Language Proc	**Text Analytics**	**Sentiment Analytics**	**Discover, Search & Export**
Cognitive Reasoning	**Data Lake**	**Entity/Link Analytics**	**Visualization / UI**

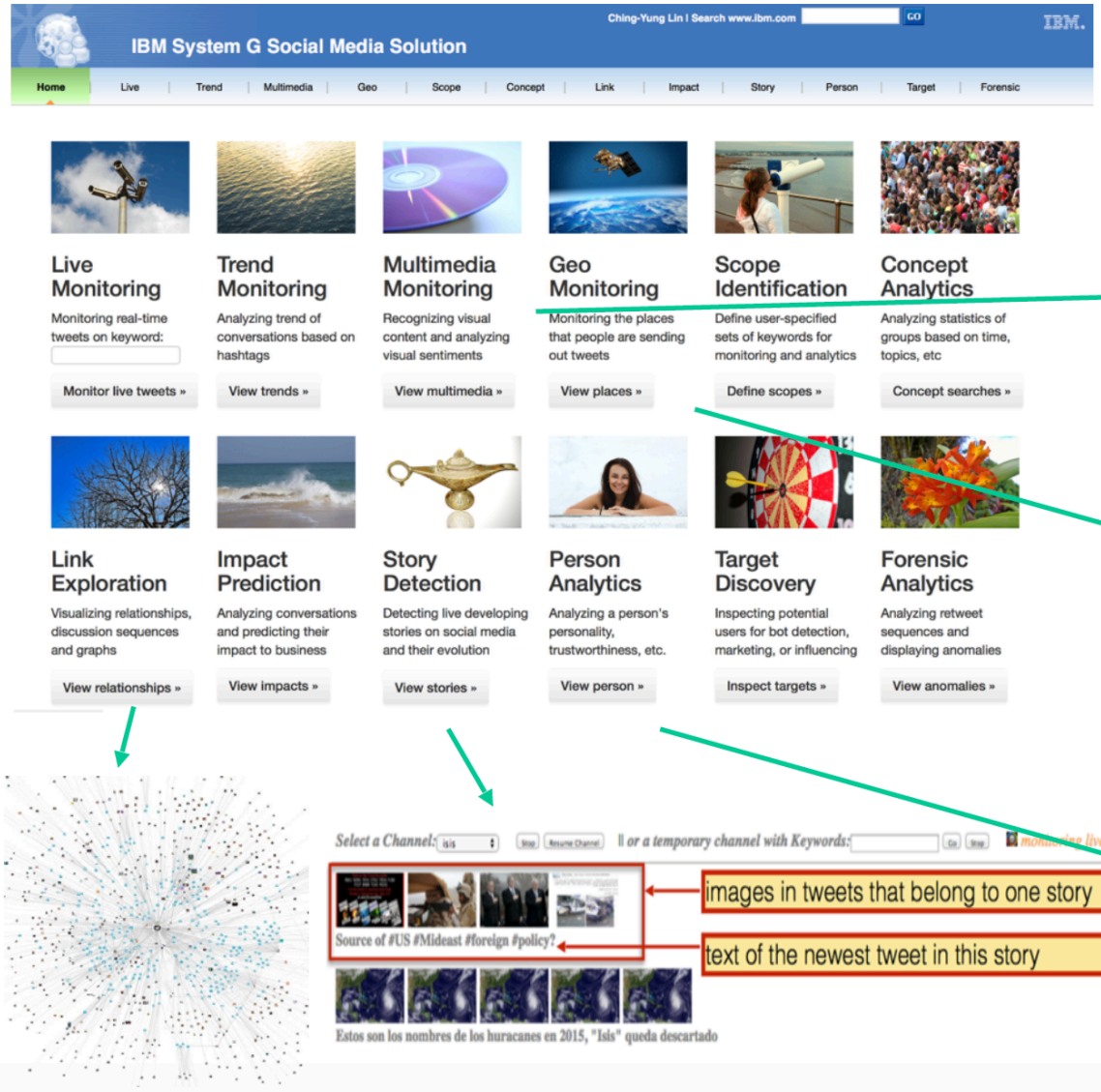
Investment Advisory

Task 1. Market Data Analysis and Investment Targets
Task 2. Advanced Dynamic 'Know Your Customer'
Task 3. Optimized Personalized Investment Strategy
Task 4. Bank-Customer Interaction Strategy





Social Media Solution



Sun Jan 25 02:00:04 +0000 2015 GMT

RT @IraqLiveUpdate: Pics - Large convoy of KH heading to undisclosed location (3 of 3) #Iraq http://t.co/147ZjPkmsC

Retweet Count: *3*
Follower Count: *207*
Tweet Sender Location: *N/A*

Automatic Tagging: *deadly_attack, horizontal_text, excellent_book, annoying_reflection, busy_bridge*

Visual Sentiment Detection: ***Negative***

案例：海量规模的视频理解

YouTube vimeo Dailymotion twitch ...

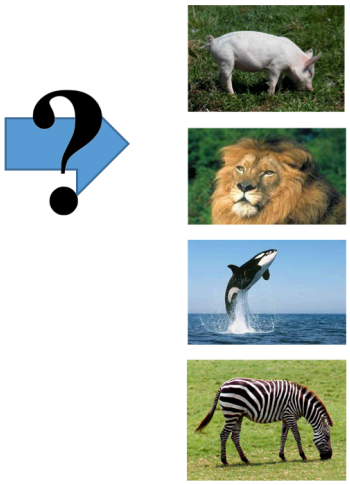
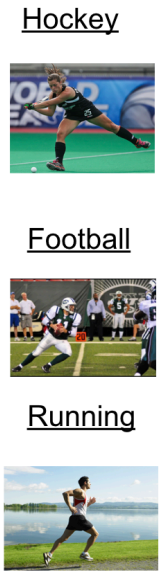
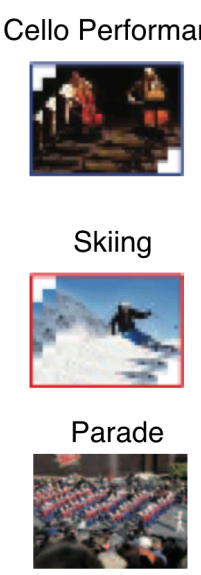


Image Classes



Action



Activity

Sadness
Fear
Anger
Joy
Disgust

Emotion

大规模视频内容理解

Harnessing Object and Scene Semantics for Large-Scale Video Understanding

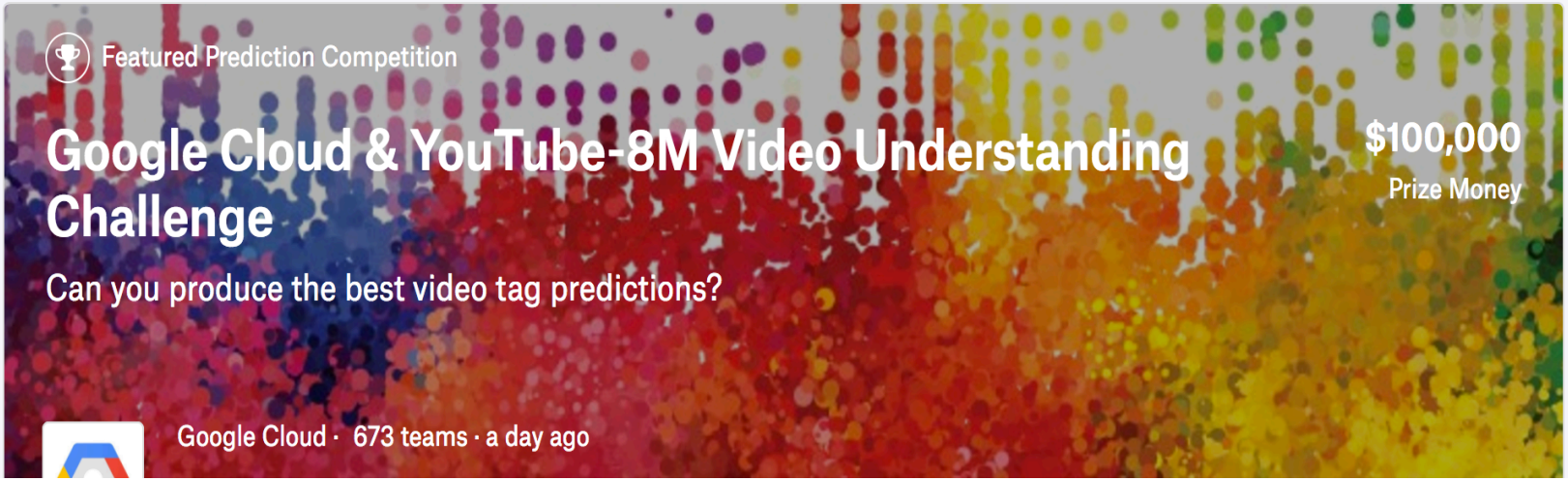
Zuxuan Wu[1], Yanwei Fu[2], Yu-Gang Jiang[1], Leonid Sigal[2]

[1] School of Computer Science, Fudan University
[2] Disney Research Pittsburgh

暴恐视频识别

复旦大学团地搭建了一个暴恐视频识别系统，已成功部署于国家网信办的“***系统”中，7×24小时线上运行；已接入秒拍、360云盘、115网盘等企业的视频数据

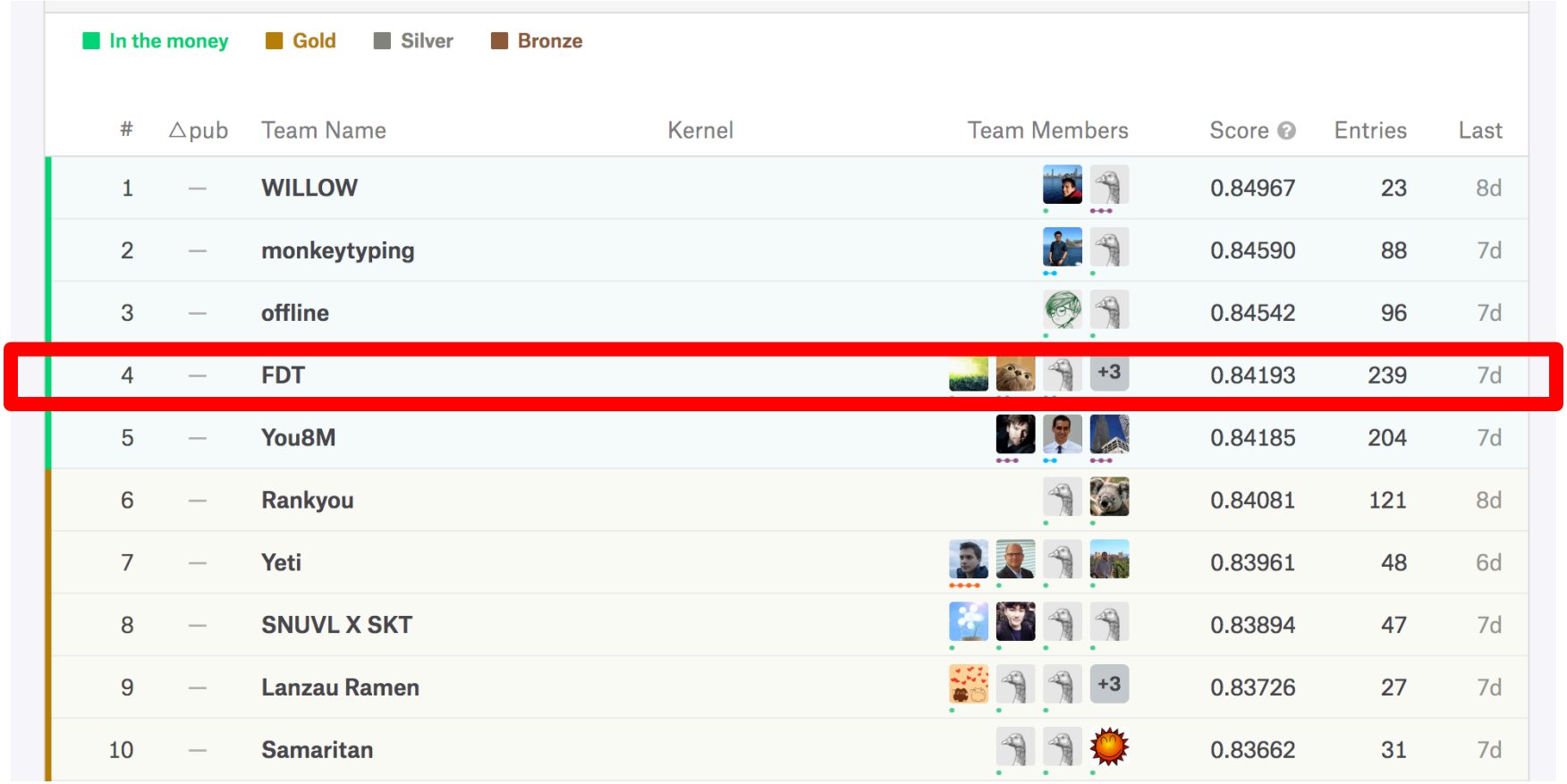
谷歌800万视频分类大赛



- 800万网络视频
- 4000余个类别
- 基于每秒采样的视频帧特征，完成分类任务
- 最终有效参赛团队数量：650个

谷歌800万视频分类大赛

- 在全球650支队伍中排名第4



In the money Gold Silver Bronze

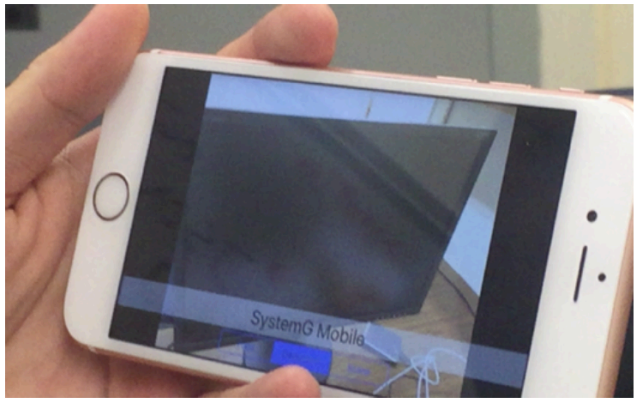
#	△pub	Team Name	Kernel	Team Members	Score	Entries	Last
1	—	WILLOW			0.84967	23	8d
2	—	monkeytyping			0.84590	88	7d
3	—	offline			0.84542	96	7d
4	—	FDT		+3	0.84193	239	7d
5	—	You8M			0.84185	204	7d
6	—	Rankyou			0.84081	121	8d
7	—	Yeti			0.83961	48	6d
8	—	SNUVL X SKT			0.83894	47	7d
9	—	Lanzau Ramen		+3	0.83726	27	7d
10	—	Samaritan			0.83662	31	7d

案例："类脑"机器人系统

Mobile Cognition -- Enabling AI right on the Edge

- Created novel graph computing and deep learning framework on iOS devices and NAOqi robots including:
- generic object recognition, event recognition, face recognition, visual sentiment recognition, and document recognition
- graph database

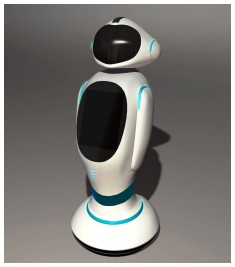
IBM System G



Novel Deep Learning works that Speed Up image computation utilizing the GPUs on iOS devices: 195x or 1657x faster

	iPad Pro	iPhone 6s
Classification rate (on ~1000 classes)	~13 frames/sec	~7 frames/sec

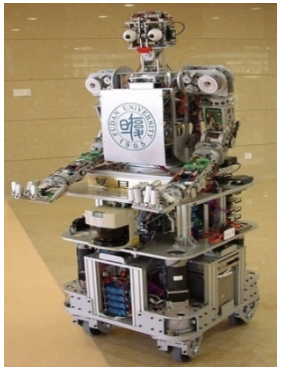
复旦大数据学院、计算机学院的类脑方面研究团队



- **研制类脑智能系统**
  - 可运行于云端，成为大数据智能分析引擎和服务机器人大脑
  - 可完成多种认知任务，具备以下认知能力：听觉视觉信息识别理解、基于自然语言的人机交互、情感识别与表达、自主行走与路径规划、自主学习与适应环境变化、机器人与机器人以及人与机器人协同工作等

- **研制机器人的本体**
  - 集成多种传感器（听觉、视觉、距离、气味等）和效应器（移动、发声、显示等），具有强大的计算、存储和联网能力，具有吸引力和亲和力的外形，支持云计算等。

机器人本体：持续开展服务机器人研制工作



复旦I号

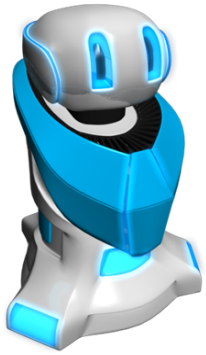
2005年

复娃

2007年

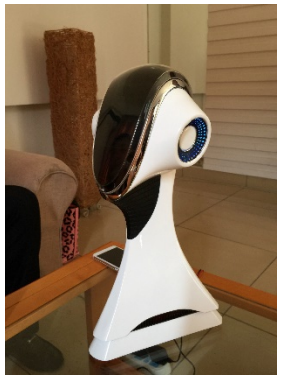
海宝

2010年



爱家I号

2011年



中医机器人

2016年

爱家II号

2017年

案例:电力大数据系统

电力大数据（1）

背景：

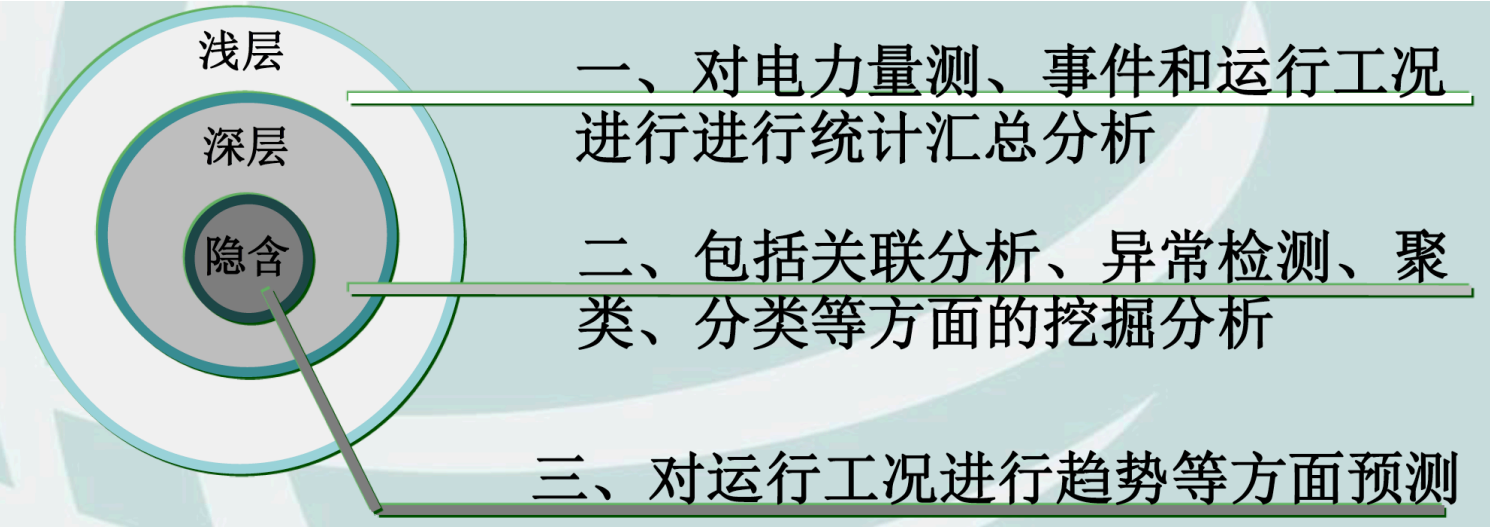
1、到2020年预计我国直流输电工程约达30项，目前我国投运直流工程20余项，有的已经 运行 10余年。

2、换流站持续存储了海量的历史数据，由于缺乏相应的技术手段，没有充分的挖掘分析，其价值没有得到体现。

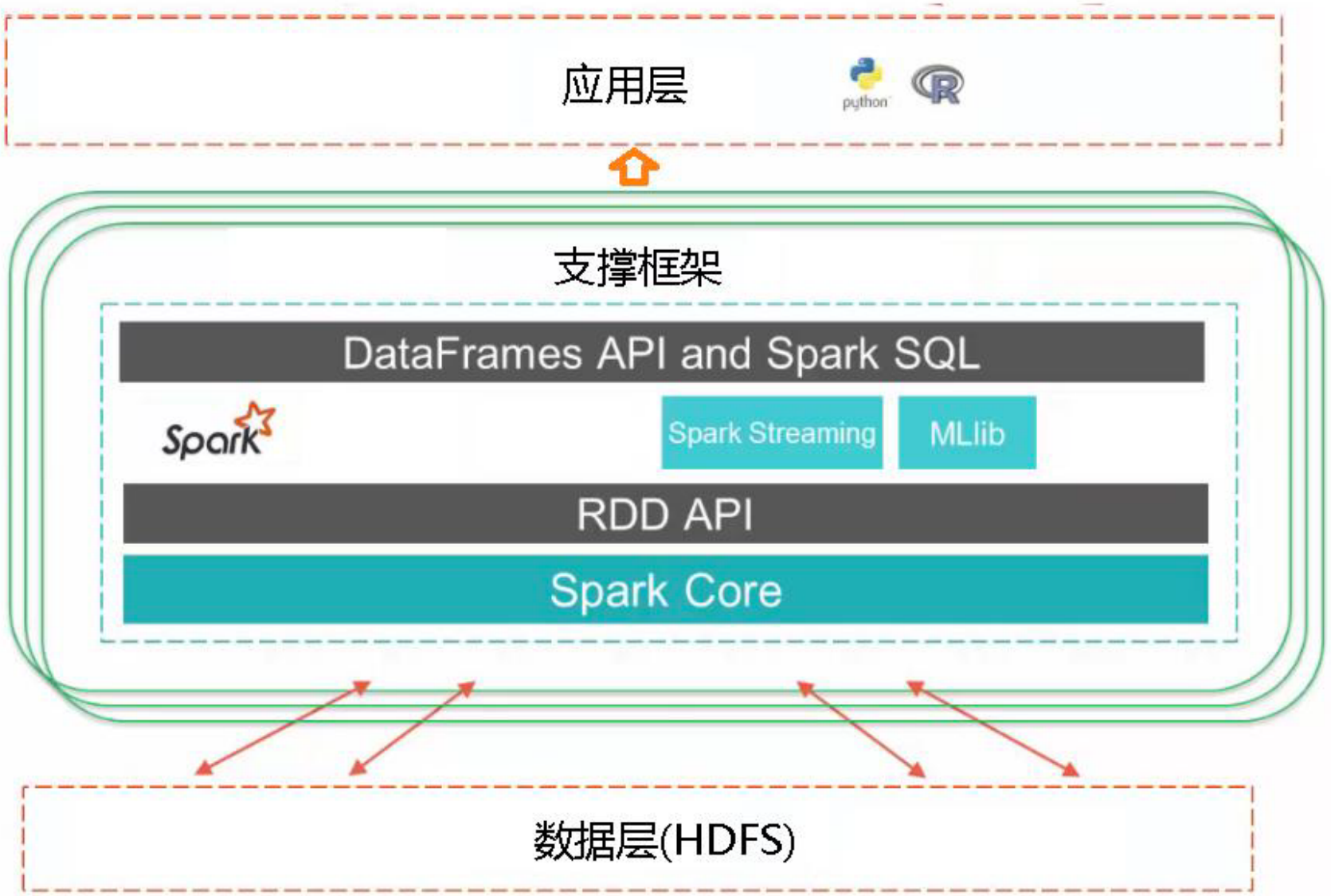
目标：

1、换流站智能化的需求：研究将数据挖掘分析技术、以及与直流控制保护技术相结合进行电力量测、事件和运行工况进行关联分析预测？

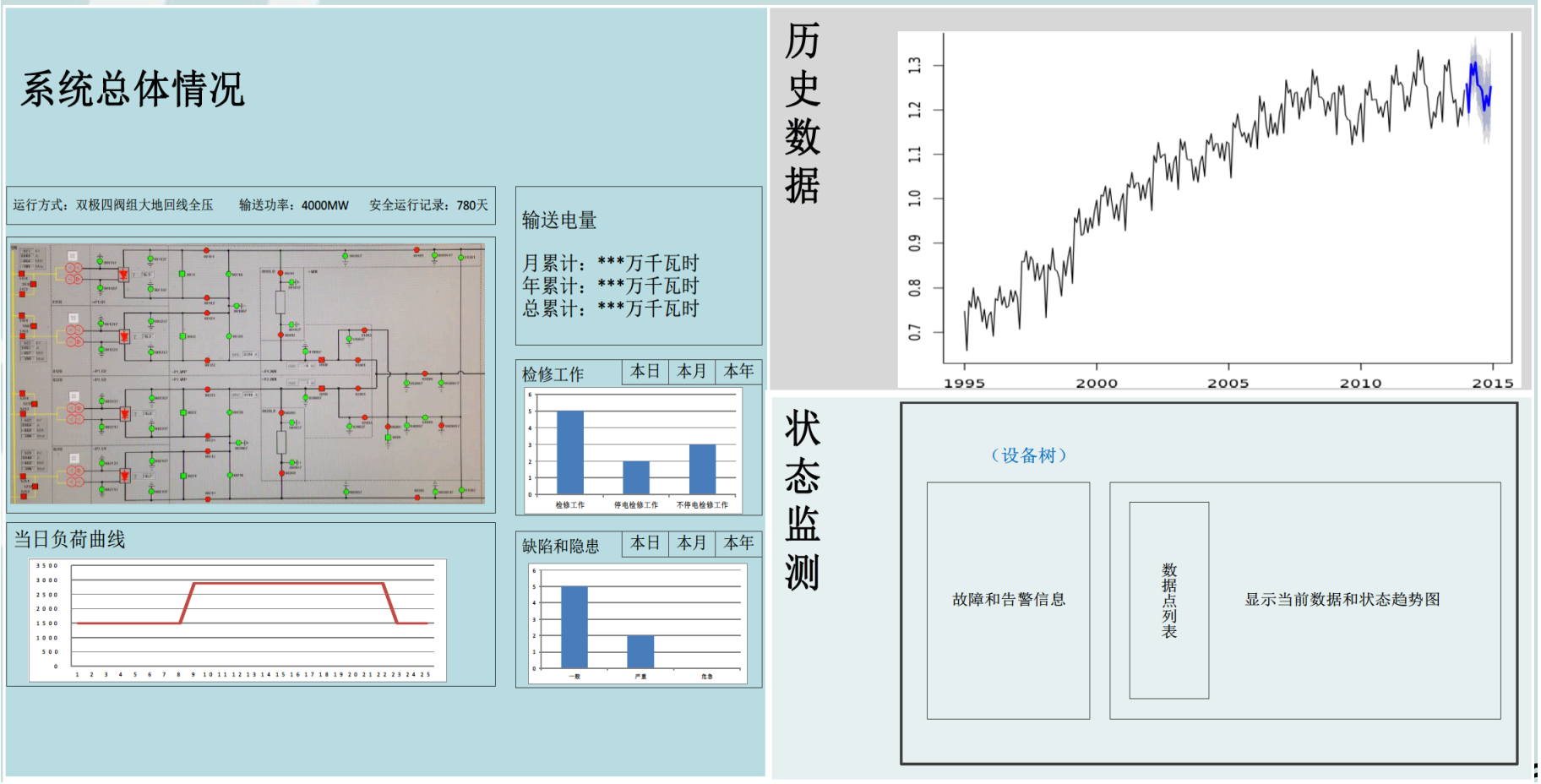
2、从一个新角度，提供一种输电系统稳定可靠运行的新的手段和技术支撑



电力大数据（2）

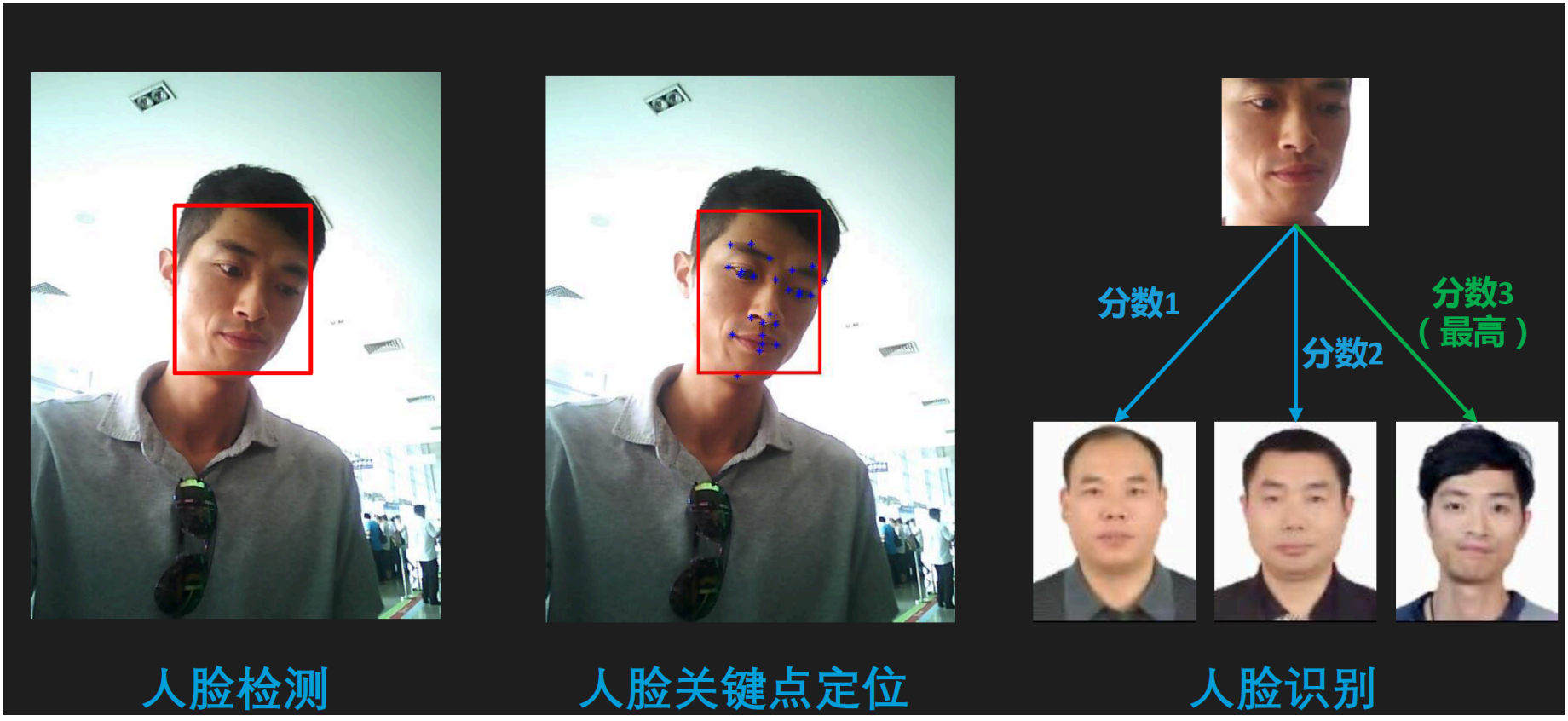

应用层
python
支撑框架
DataFrames API and Spark SQL
Spark
Spark Streaming
MLlib
RDD API
Spark Core
数据层(HDFS)

电力大数据（3）

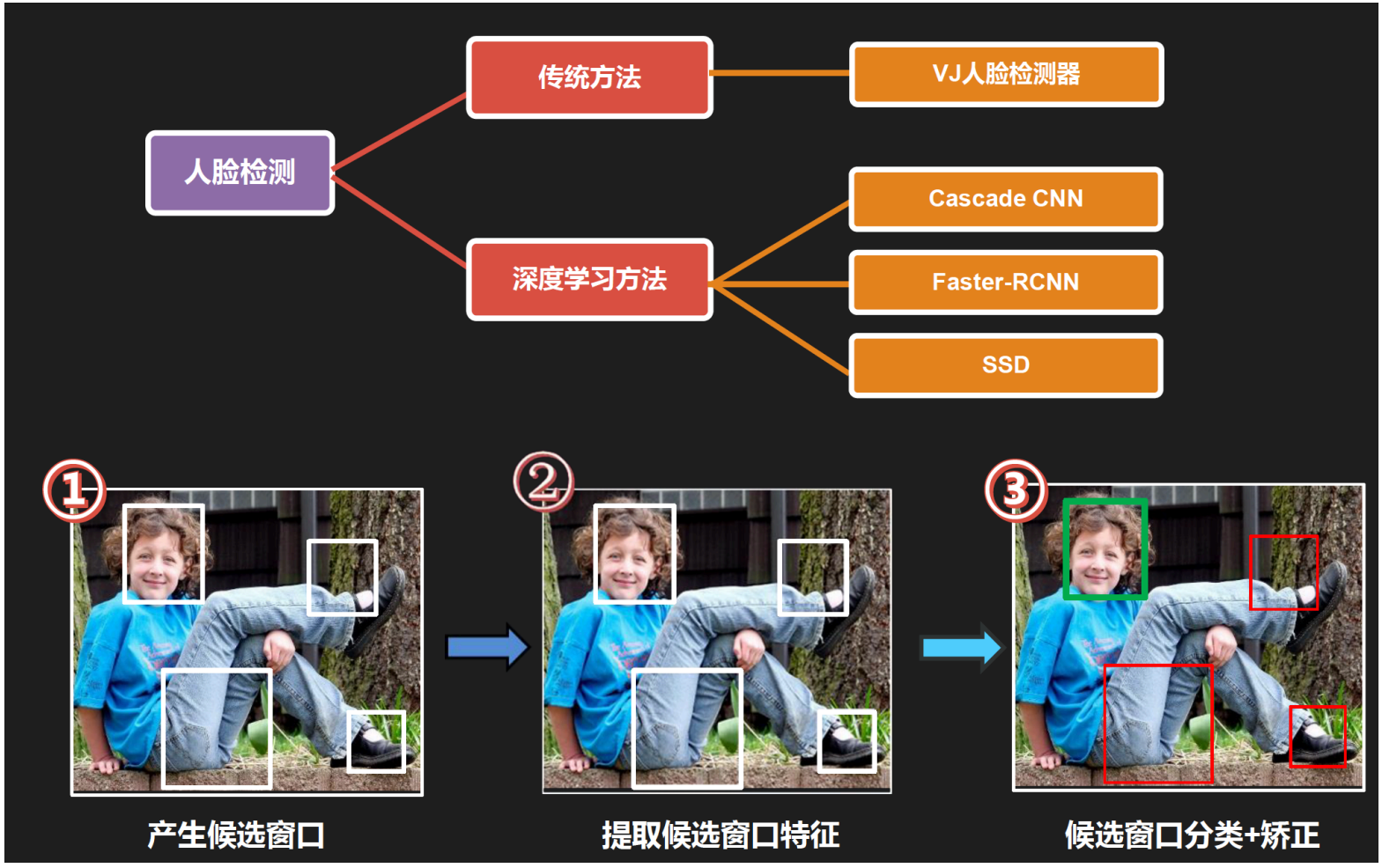

系统总体情况
运行方式：双极四阀组大地回线全压
输送功率：4000MW
安全运行记录：780天
输送电量
月累计：***万千瓦时
年累计：***万千瓦时
总累计：***万千瓦时
检修工作
本日
本月
本年
检修工作
停电检修工作
不停电检修工作
当日负荷曲线
缺陷和隐患
本日
本月
本年
历史数据
状态监测
（设备树）
故障和告警信息
数据点列表
显示当前数据和状态趋势图

案例:大数据的人脸识别

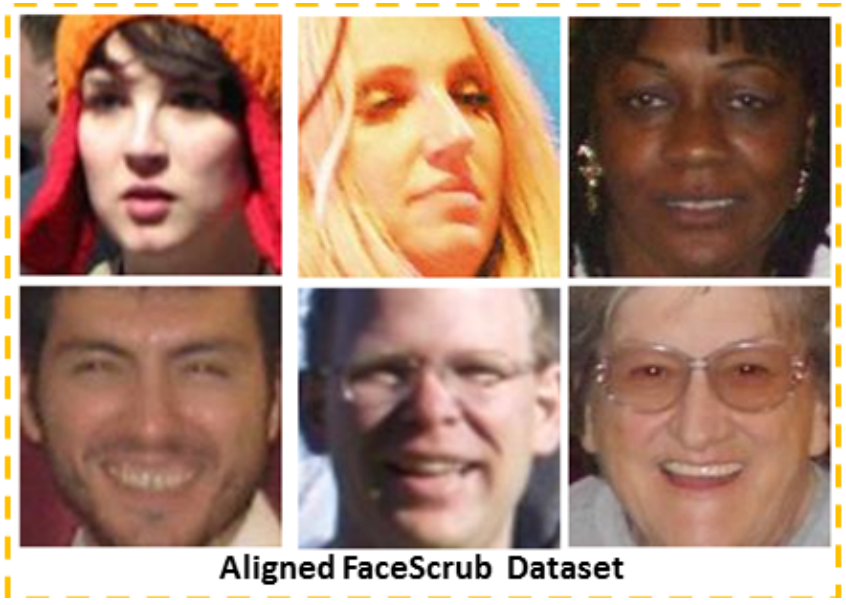
人脸识别流程



人脸检测流程


人脸检测
传统方法
VJ人脸检测器
深度学习方法
Cascade CNN
Faster-RCNN
SSD
1
2
3
产生候选窗口
提取候选窗口特征
候选窗口分类+矫正

Some face images in MegaFace dataset



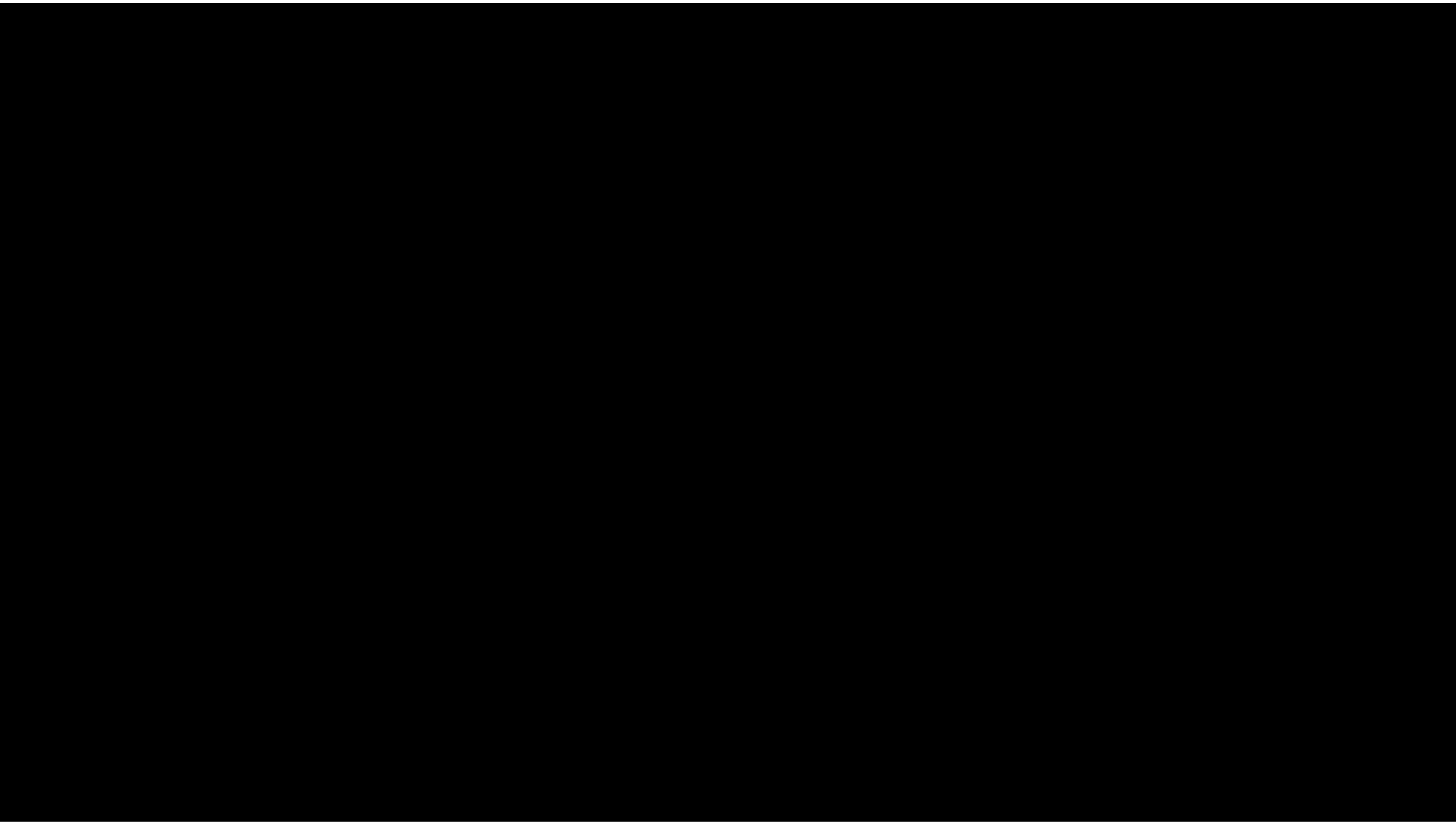
MegaFace is a very challenging dataset to evaluate the performance of face recognition algorithms. It includes gallery set and probe set. The gallery set consists of 690K different individuals with more than 1 million images.

On Mega-Face Challenge, our method can achieve the best Rank-1 identification performance under the small protocol :

Large is >500K photos trained

Algorithm	Date Submitted	Set 1	Set 2	Set 3	Data Set Size
Vocord - deepVo V3	04/27/2017	91.763%	91.711%	91.704%	Large
YouTu Lab (Tencent Best-Image)	04/08/2017	83.290%	83.267%	83.295%	Large
DeepSense V2	1/22/2017	81.298%	81.298%	81.298%	Large
Vocord-deepVo1.2	12/1/2016	80.258%	80.195%	80.241%	Large
Fudan University - FUDAN-CS_SDS	1/29/2017	77.982%	78.006%	77.990%	Small
GRCCV	12/1/2016	77.677%	77.021%	77.147%	Small
Beijing Faceall Co. - FaceAll V2	04/28/2017	76.661%	76.643%	76.607%	Small
SphereFace - Small	12/1/2016	75.766%	75.765%	75.770%	Small
Vocord - DeepVo1	08/3/2016	75.127%	75.093%	75.125%	Large
DeepSense - Large	07/31/2016	74.799%	74.780%	74.813%	Large
SIATMMLAB TencentVision	12/1/2016	74.207%	74.213%	74.195%	Large
Shanghai Tech	08/13/2016	74.049%	74.032%	74.020%	Large
NTechLAB - facenx_large	10/20/2015	73.300%	73.309%	73.287%	Large
ForceInfo	04/7/2017	72.11%	72.084%	72.121%	Large
3DiVi Company - tdvm V2	04/15/2017	71.742%	71.727%	71.703%	Large
DeepSense - Small	07/31/2016	70.983%	70.948%	70.962%	Small
Google - FaceNet v8	10/23/2015	70.496%	70.492%	70.551%	Large


YOLO v2
http://pjreddie.com/yolo

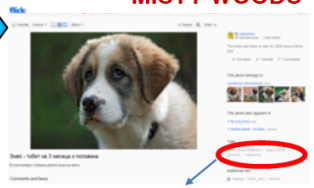


案例:大数据图像视频情感分析

Visual Sentiment and Semantic Analysis

First work in the literature on automatic visual sentiment analysis

"*For content to go viral, it needs to be emotional,*" Dan Jones, 2012

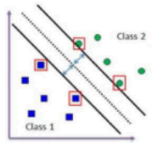


Training from 6 million tags

Select Adj-Noun Pairs

Build Sentiment Ontology

Train Classifiers



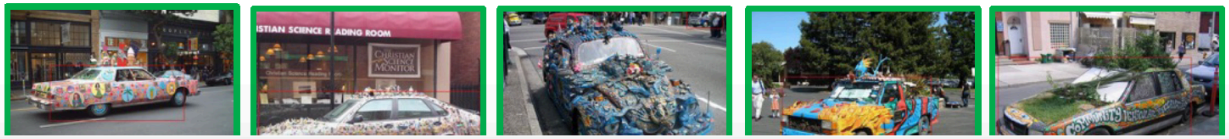
Performance Filtering

SentiBank (1200 Detectors)

Sentiment Prediction

Detection results of "lonely dog" (80% accuracy, 4 out of 5 correct)

Detection results of "crazy car" (100% accuracy, 5 out of 5 correct)

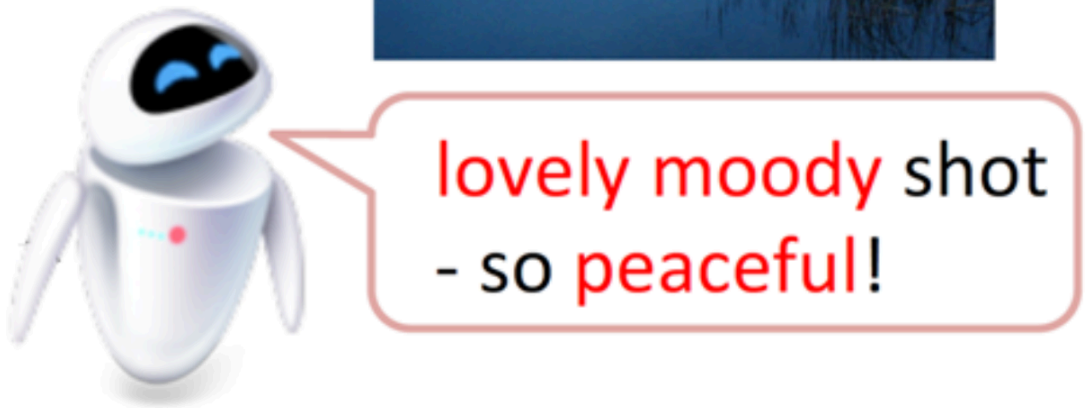


Experiment on Sentiment Detection Accuracy on Twitter

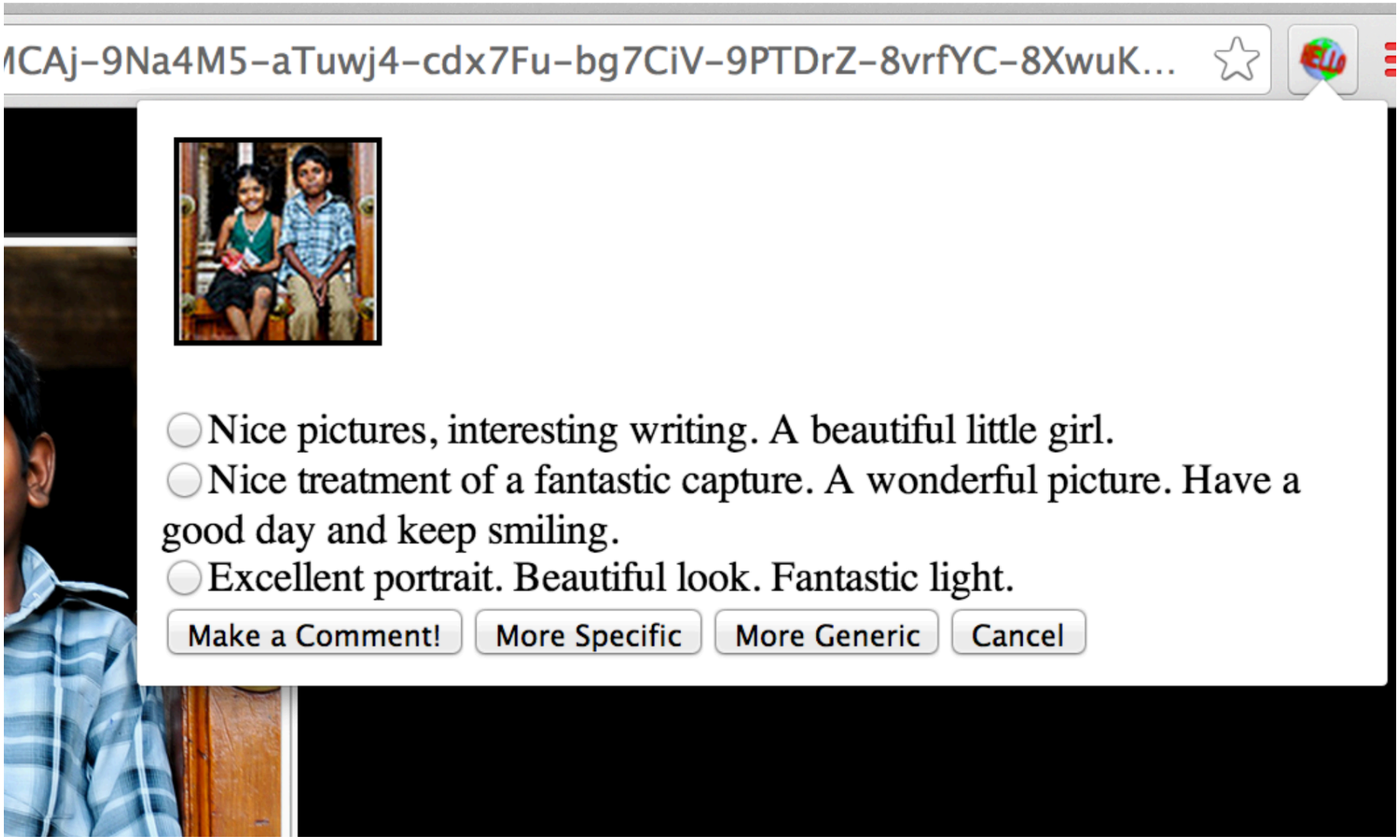
Text	0.43
Visual	0.70
T+V	0.72

Cognitive Feeling Detection on Images

Cognitive Feeling Detection on Images



Automatic Comments on Images



Video Emotion Recognition


Probability
Time
Emotion
Energy Probability
Time
Energy

案例: 大规模的图像识别

Open /Users/yanwei/projects/latest_gui/ImageNet.fig

ImageNet

Directory: /Users/yanwei/projects/latest_gui/wiki_good/Bacholer/PRBlonde-Bachelor-Juan-Pablo-Finale-7.jpg Browse

File List:

04-bachelor.w529.h352.jpg
141420_0494.jpg
1423514721_chris-soules-bachelor-article.jpg
160229-news-bachelor-hp-lg.jpg
PRBlonde-Bachelor-Juan-Pablo-Finale-7.jpg
aaron-and-helene-abc__iphone_640.jpg
bachelor-20-ben-higgins-caila.jpg
bachelor_clare.jpg.CROP.promovar-mediumlarge.jpg
the-bachelor-2-800.jpg

Model: imagenet-vgg-verydeep-19

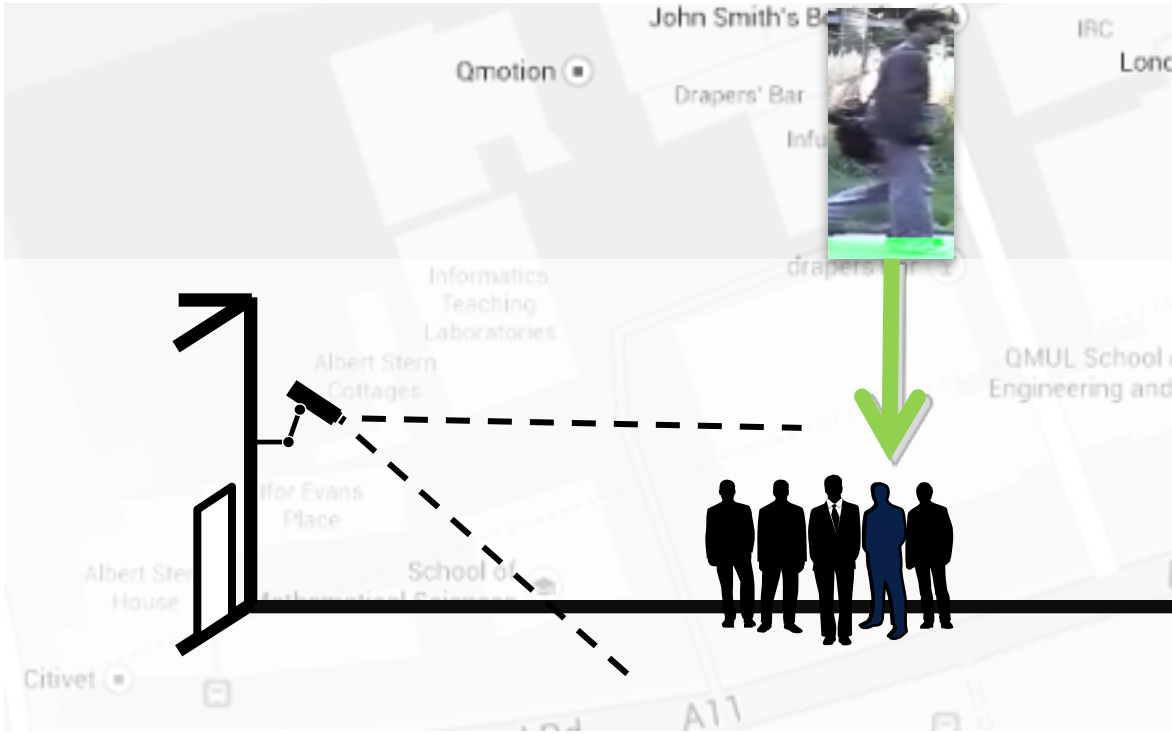
Classify

suit, suit of clothes: 0.74172
groom, bridegroom: 0.062447
Loafer: 0.03274
jean, blue jean, denim: 0.024774
bow tie, bow-tie, bowtie: 0.02112

案例:监控场景下的行人再识别

What is re-identification?

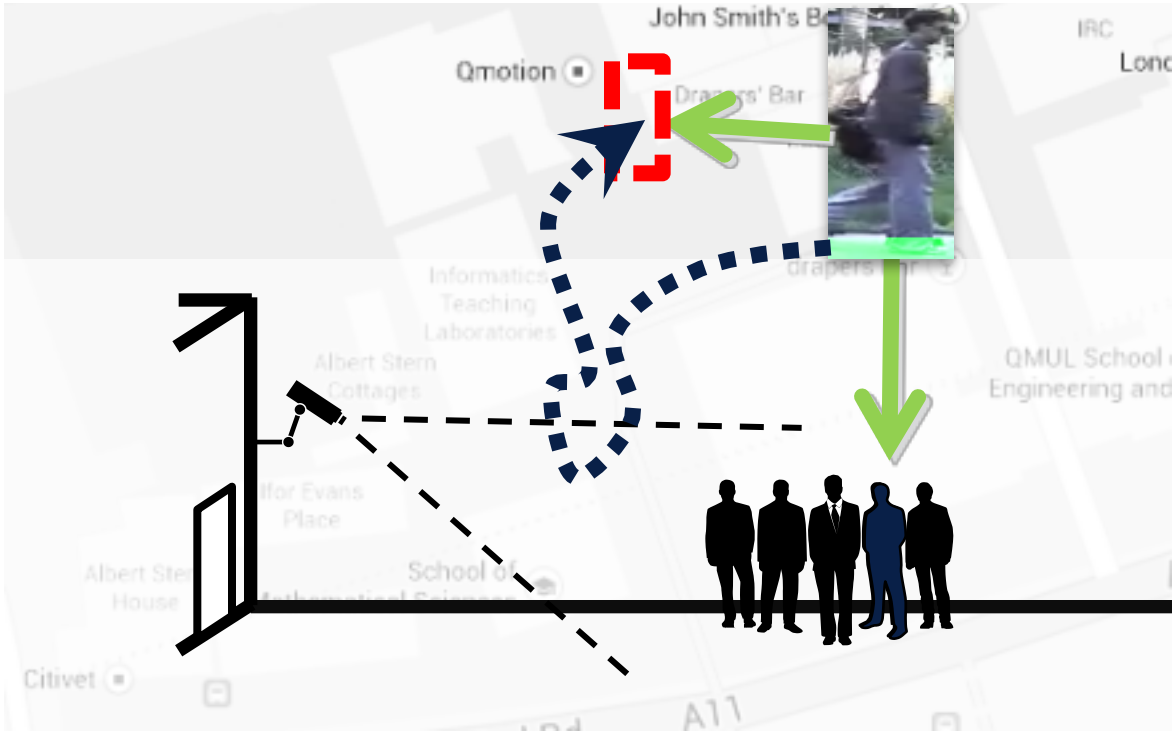
Given an image or description of a person observed in one camera view



What is re-identification?

Given an image or description of a person observed in one camera view

Locate all other images of the same person from any other camera view



What is re-identification?

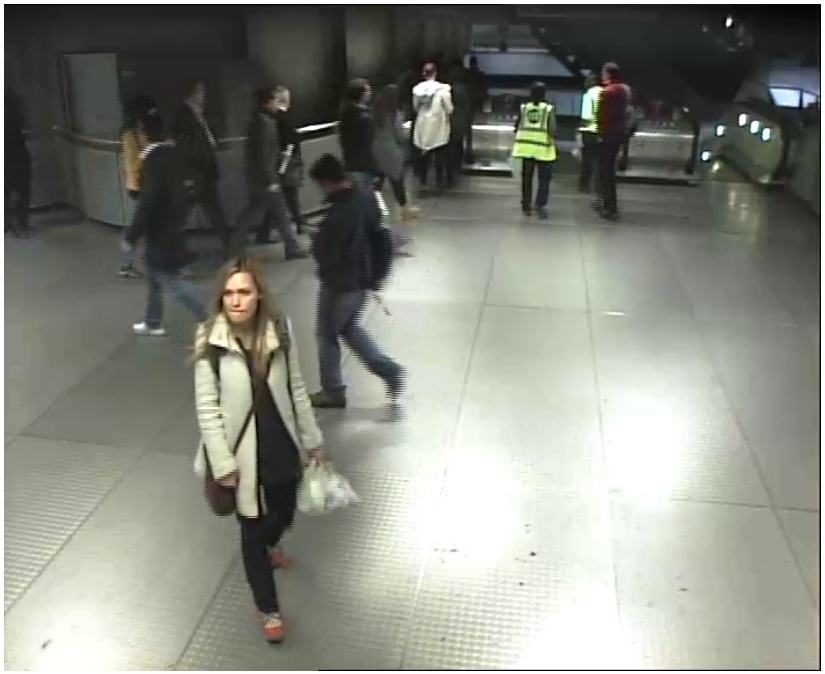
Given an image or description of a person observed in one camera view

Locate all other images of the same person from any other camera view

81

Why is re-id important?

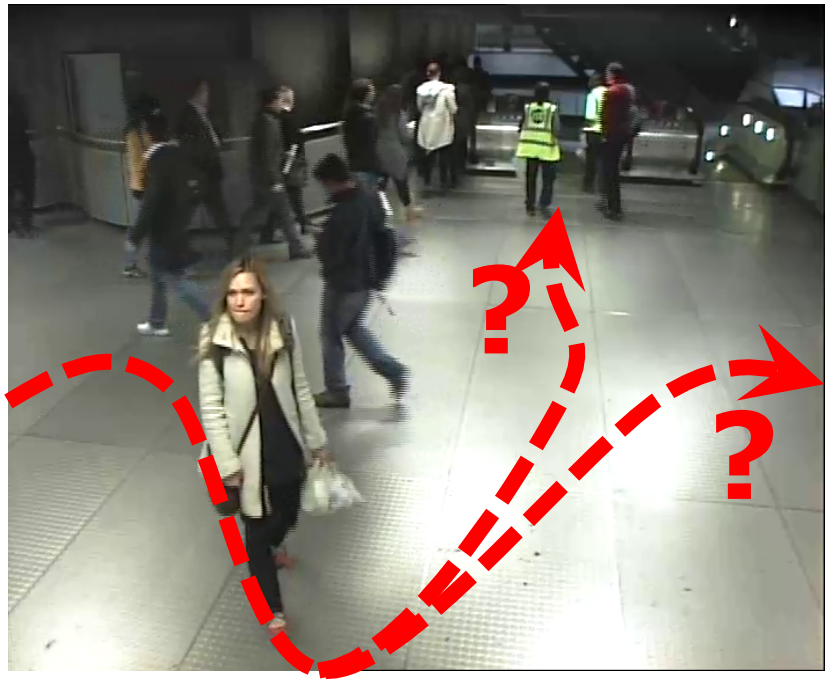
Underpins visual automated surveillance tasks



Why is re-id important?

Underpins visual automated surveillance tasks:

- **Tracking between cameras**



Why is re-id important?

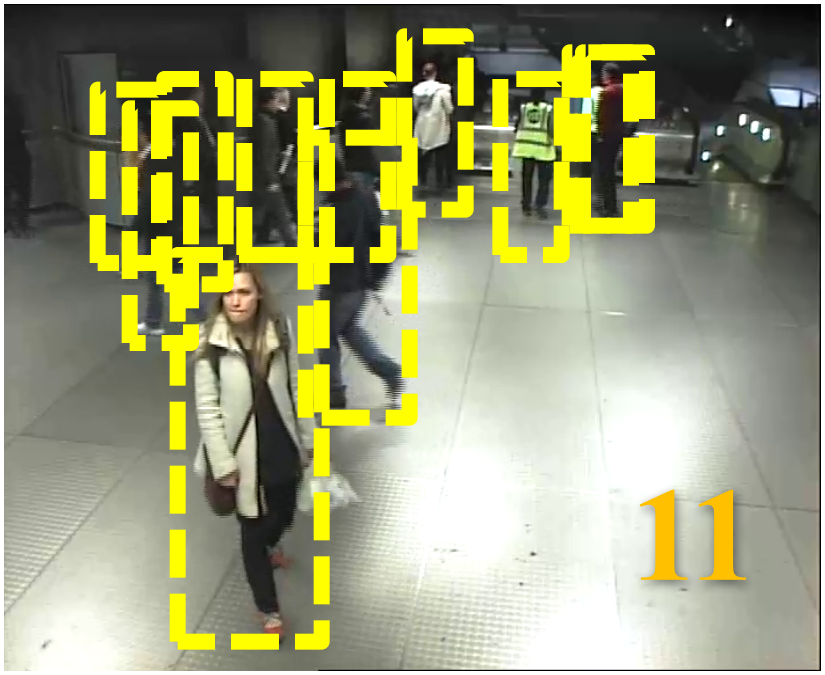
Underpins visual automated surveillance tasks:

- Tracking between cameras
- **Locate specific suspects**

Why is re-id important?

Underpins visual automated surveillance tasks:

- Tracking between cameras
- Locate specific suspects
- **Person counting**



Why is re-id important?

Underpins visual automated surveillance tasks:

- Tracking between cameras
- Locate specific suspects
- Person counting
- **Human traffic analysis**

Why is re-id important?

Underpins visual automated surveillance tasks:

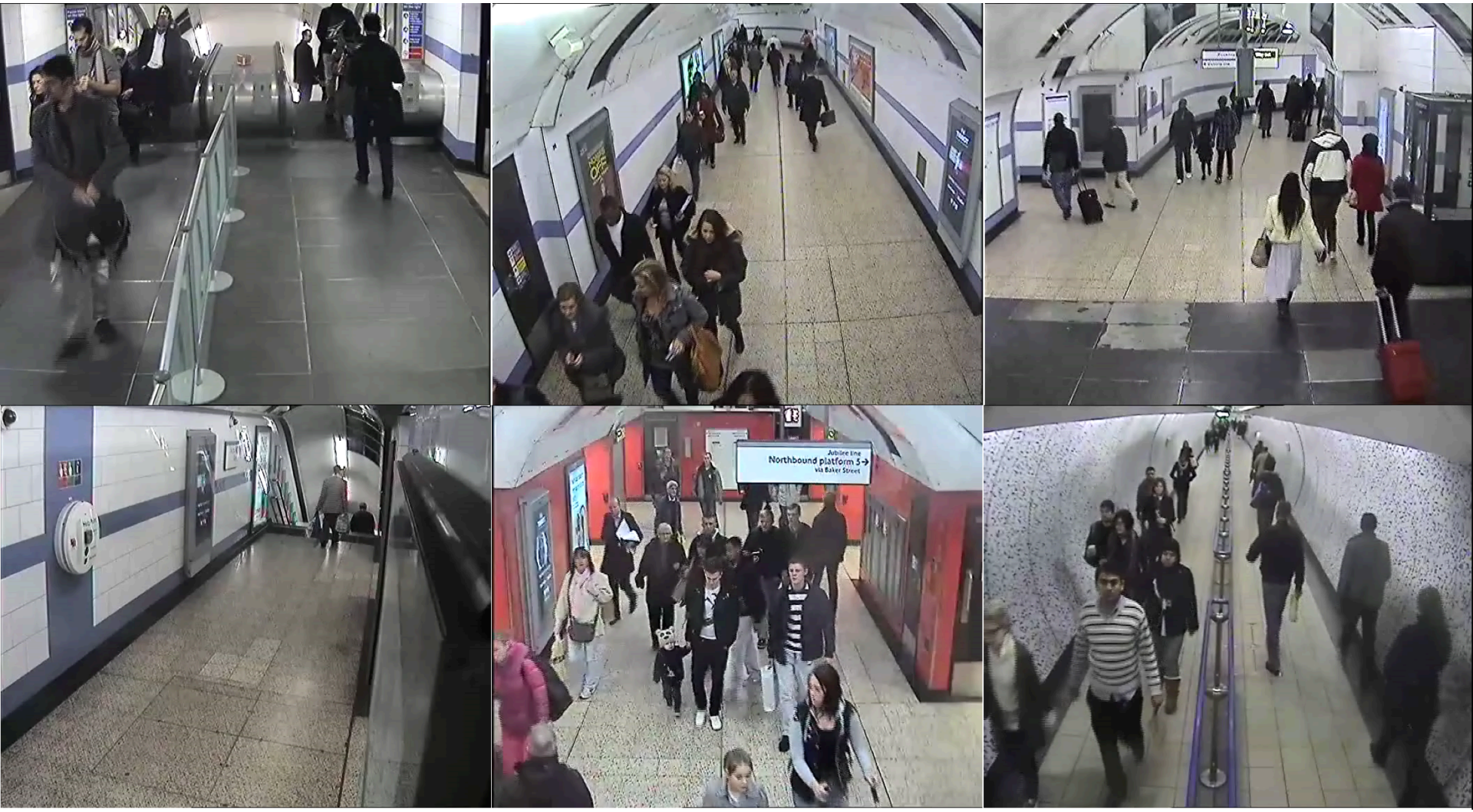
- Tracking between cameras
- Locate specific suspects
- Person counting
- Human traffic analysis

- **Incident / Intrusion detection**
- **… many more**

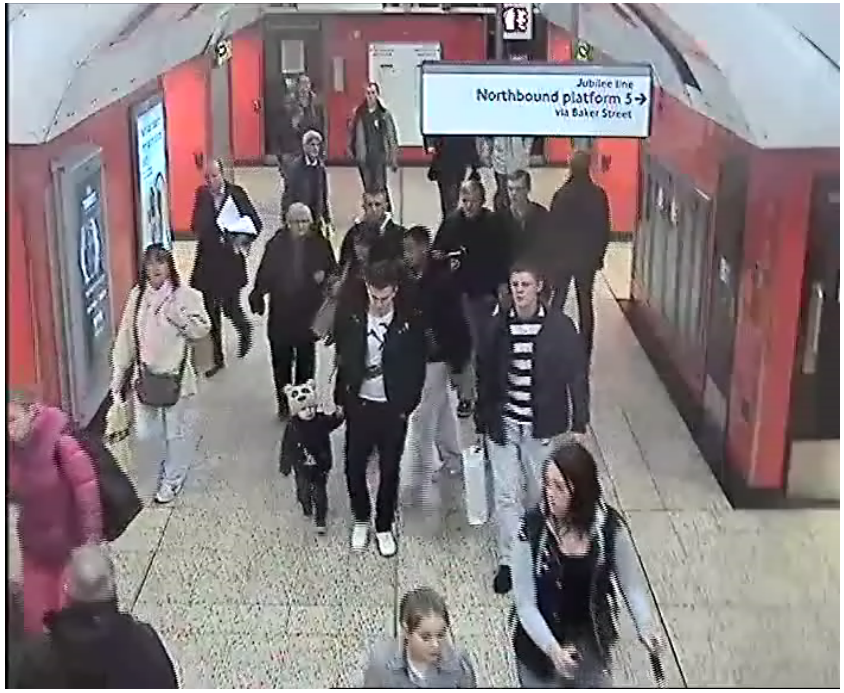
Why is re-id important?

Relying on human operator manual re-identification in large camera networks is prohibitively costly and inaccurate.

Operators are often assigned more cameras than they can feasibly monitor simultaneously, and even within a single camera, manual matching is vulnerable to inevitable attentional gaps.


Jubilee line
Northbound platform 5
via Baker Street

Human Re-id from surveillance is hard:



Heavy clutter
Inter-person occlusion
Self-occlusion
Camera calibration issues
Resolution
Lighting condition
Dynamic lighting
Human pose
Unconstrained camera pose

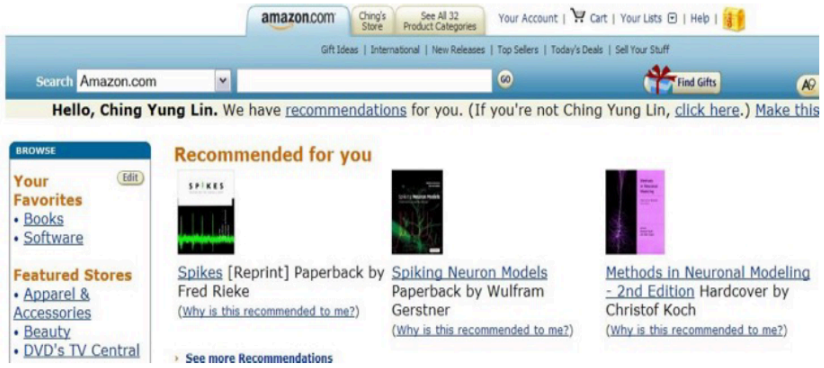
案例:大数据卫星图像的监控

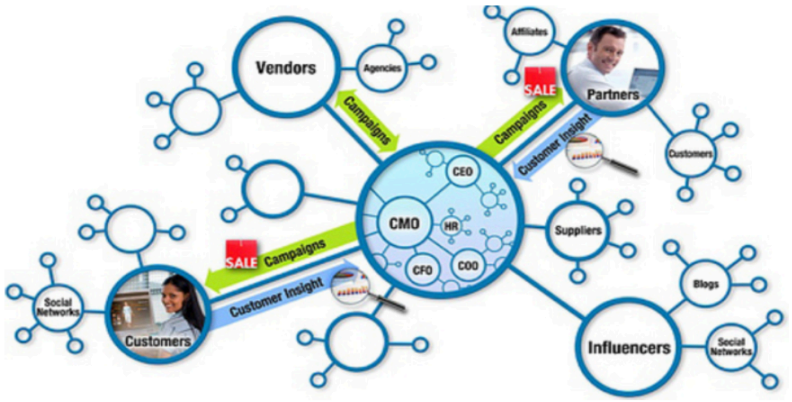
Raw videos: /Users/yanwei/其他项目/长广所卫星/video_1.avi

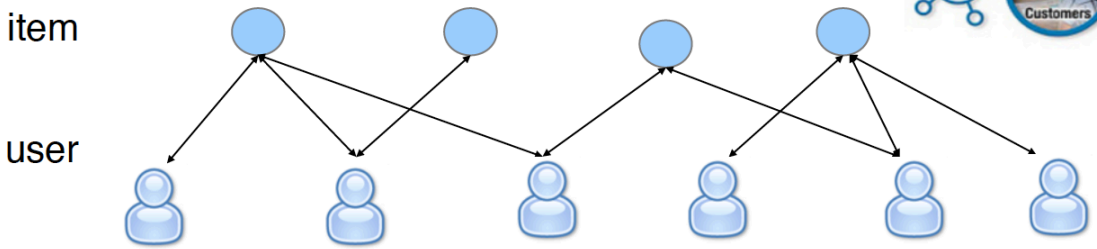
案例:360度的观察消费者

360 View

Recommendation



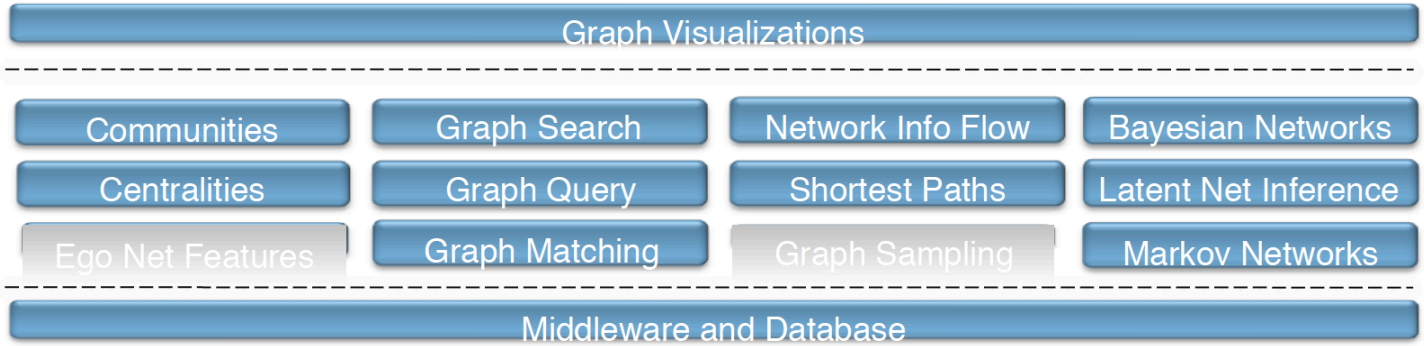




Enhancing:

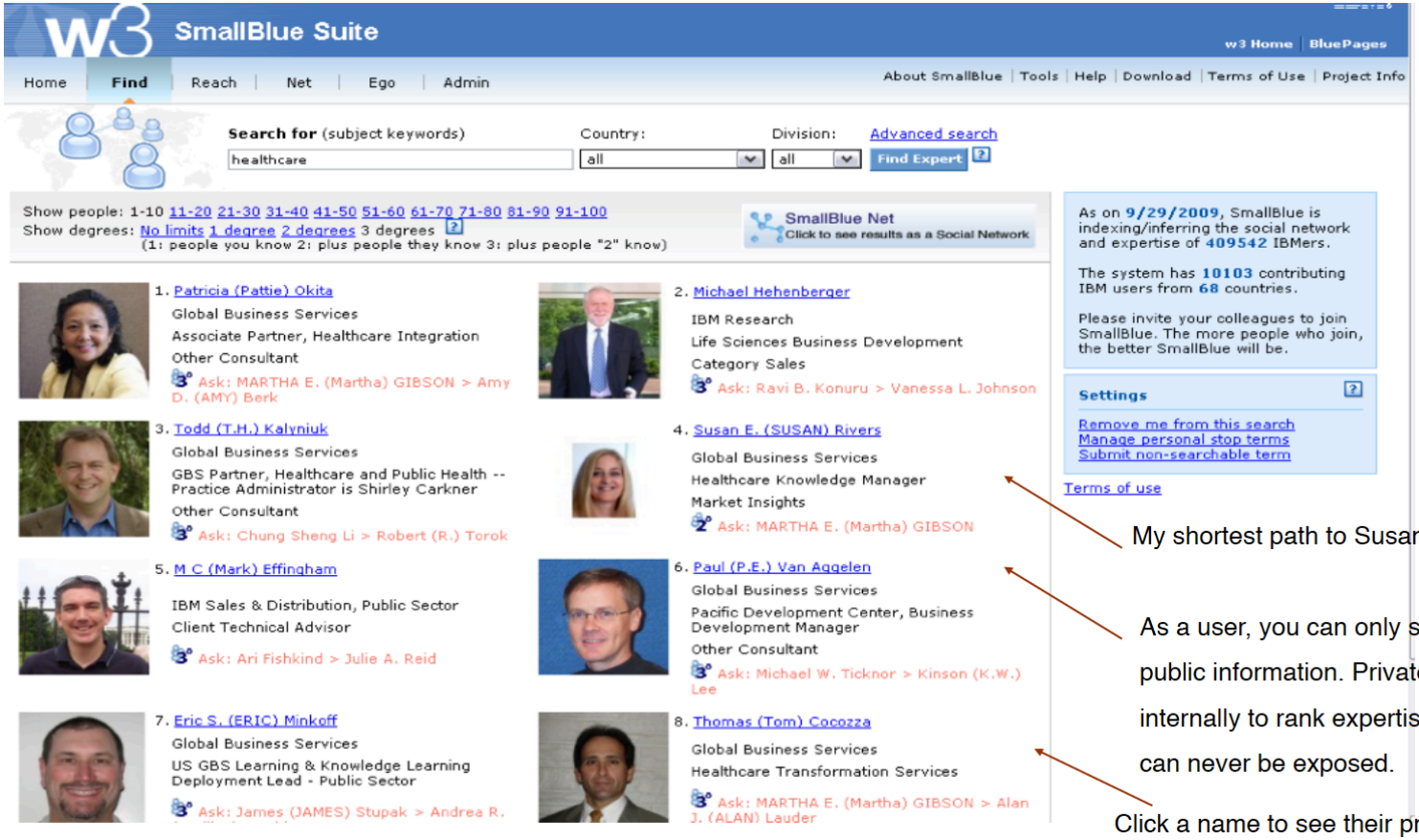
Coremetrics An IBM Company

unica

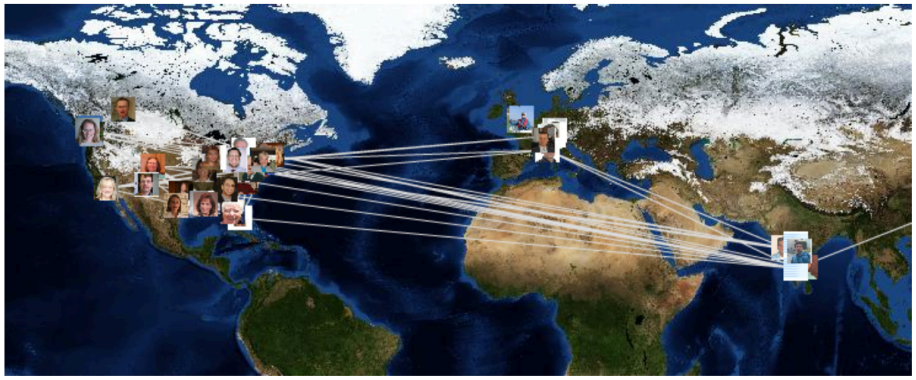


Finding and Ranking Expertise -- Social Network Analysis

Search for the most knowledgeable colleagues within organization or my 3-degree network for who knows topic XYZ (or within a country, a division, a job role, or any group/community)



Visualize social roles of individuals in company



Example: Healthcare experts in the world

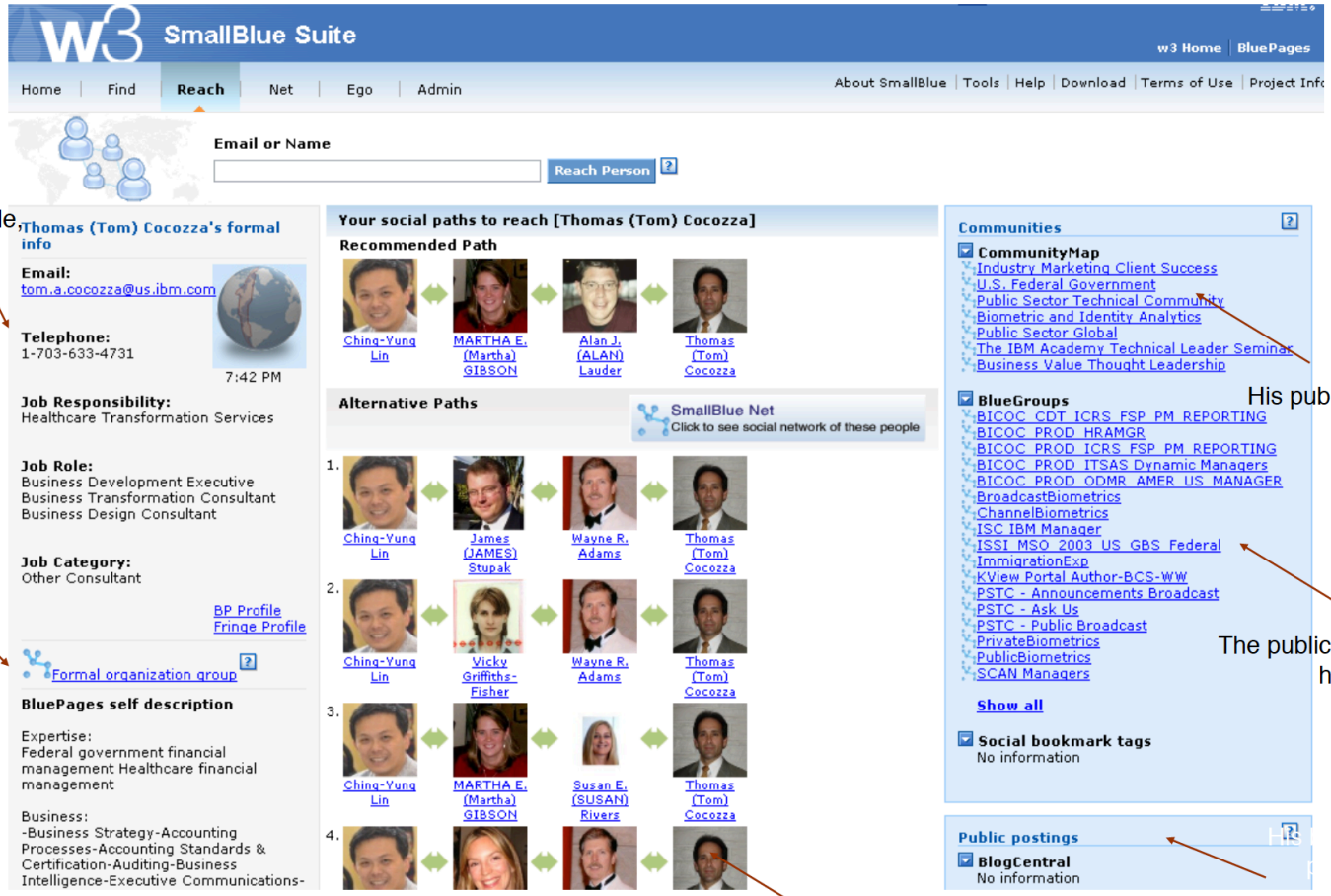
Example: Healthcare experts in the U.S.

Connections between different divisions

Key social bridges

Shortest Paths between two people in enterprise

- **Example: Is Tom a right person to me?**



His official job role, title, contact info

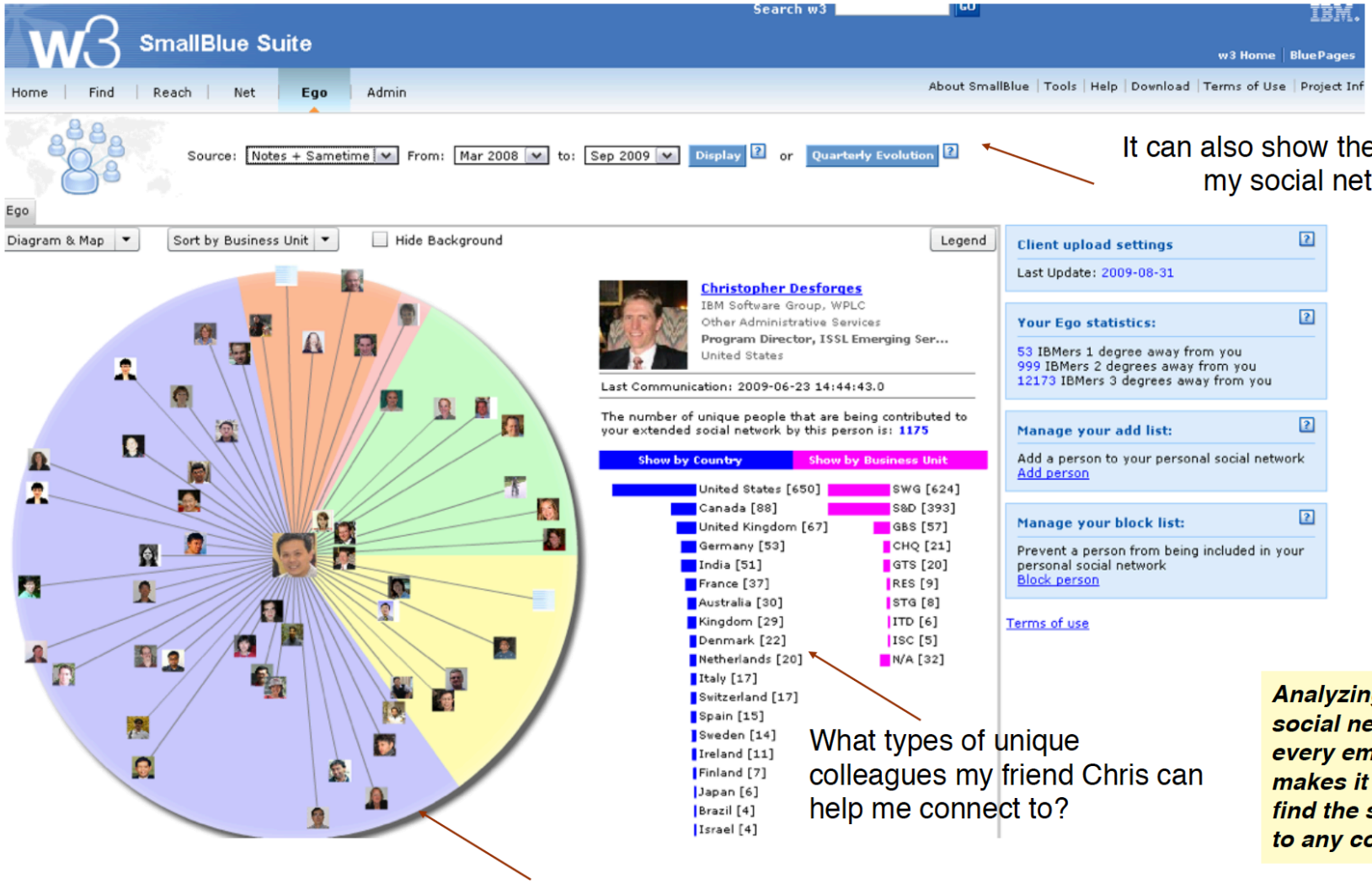
His self-described expertise

His public communities

The public interest groups he is in

My various paths to Tom. SmallBlue can show the paths to any colleagues up to 6-degree away

Personal social network capital management



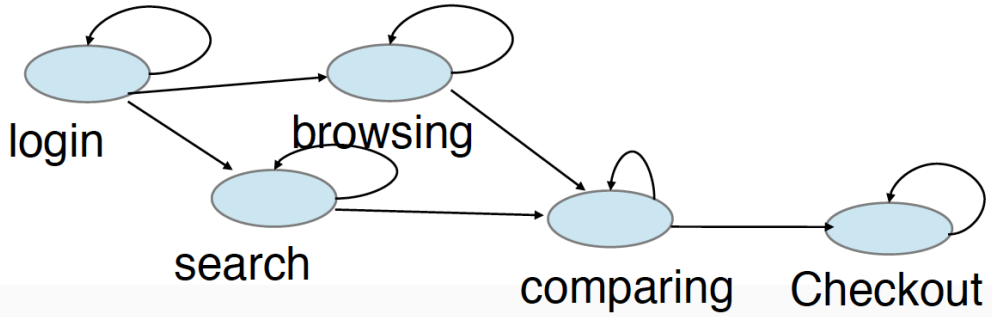
It can also show the evolution of my social network..

What types of unique colleagues my friend Chris can help me connect to?

Evolutionalry personal social network

Analyzing existing social networks of every employee That makes it possible to find the shortest path to any colleague..

Customer Behavior Sequence Analytics



- Behavior Pattern Detection
- Help Needed Detection

Dynamics of Information Graphs in Social Media

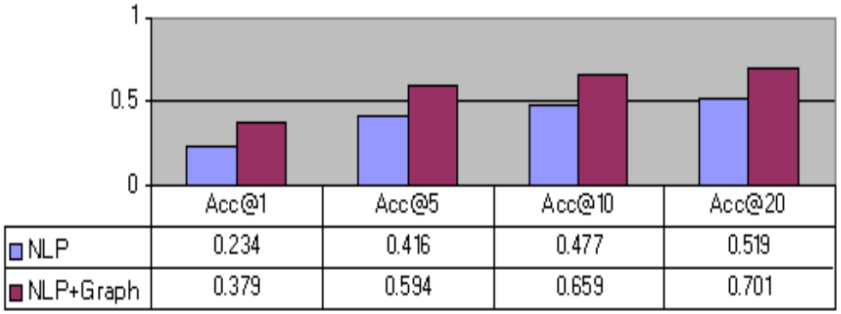
- **Motivation:**
  - –Info morph: new links keep emerging to give new meaning to existing phrases
- **Approach:**
  - –Compare characteristics of meta-paths between nodes in heterogeneous networks

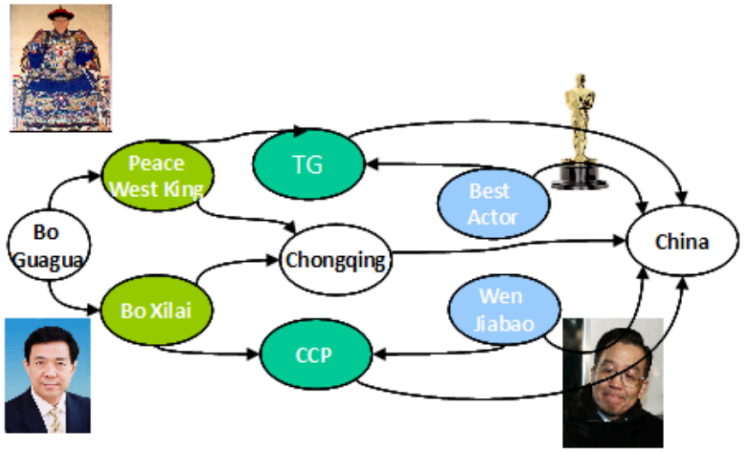
Peace West King from ***Chongqing fell from power***, still need to ***sing red songs***?

中国日报 CHINA DAILY
THE NATIONAL ENGLISH LANGUAGE NEWSPAPER

- ***Bo Xilai*** led ***Chongqing*** city leaders and 40 district and county party and government leaders to ***sing red songs***

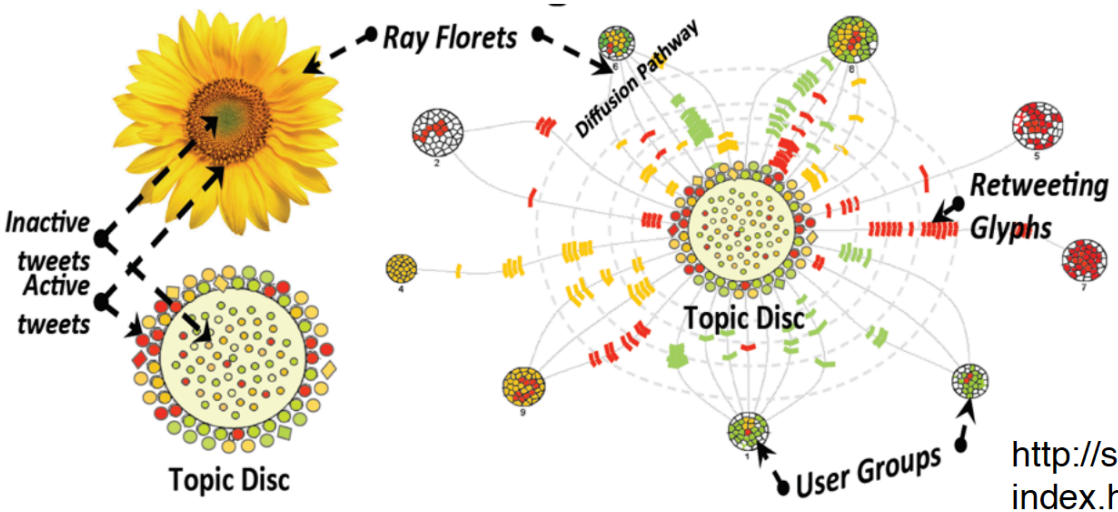
Entity morph resolution accuracy
(ACL 2013)





$$\sum_{i=1}^{N} p_m(x_i) \log \frac{p_m(x_i)}{p_e(x_i)} + p_e(x_i) \log \frac{p_e(x_i)}{p_m(x_i)}$$

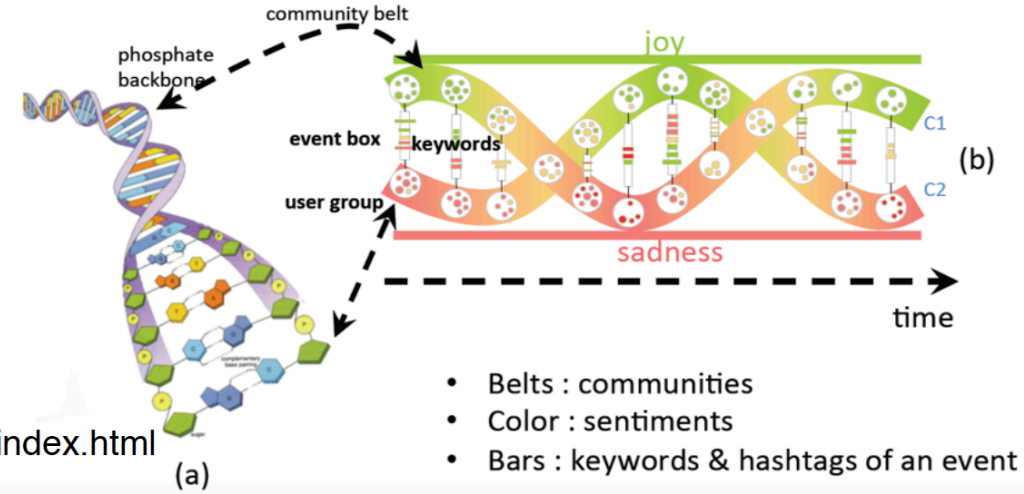
Visualizing Information Diffusion and Divergence



Whisper : Tracing the information diffusion in Social Media

http://systemg.ibm.com/apps/whisper/index.html

SocialHelix: Visualizaiton of Sentiment Divergence in Social Media



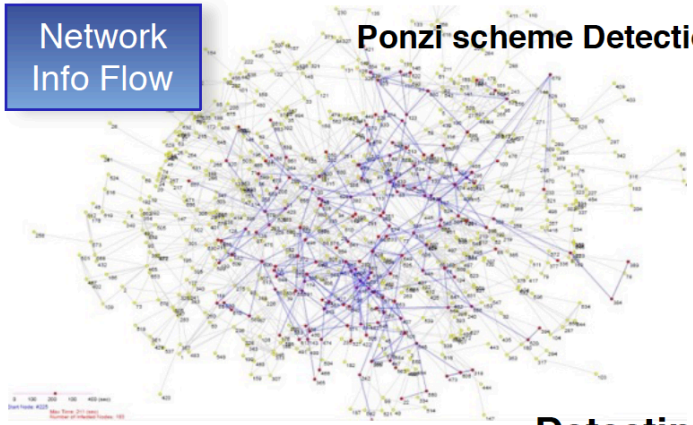
- Belts : communities
- Color : sentiments
- Bars : keywords & hashtags of an event

http://systemg.ibm.com/apps/socialhelix/index.html

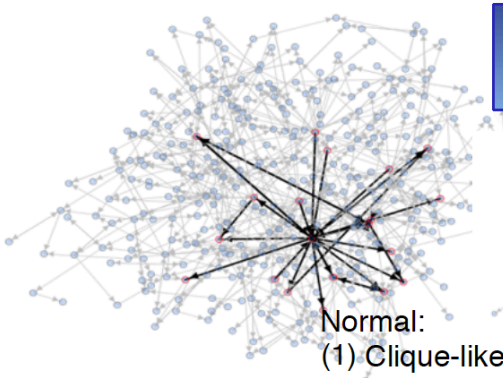
Security

Network Info Flow

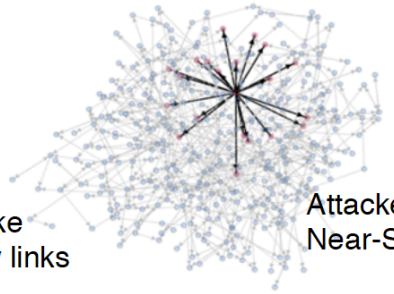
Ponzi scheme Detection



Ego Net Features

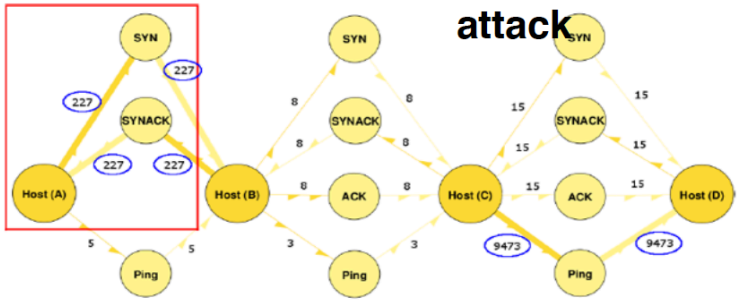


Normal:
(1) Clique-like
(2) Two-way links

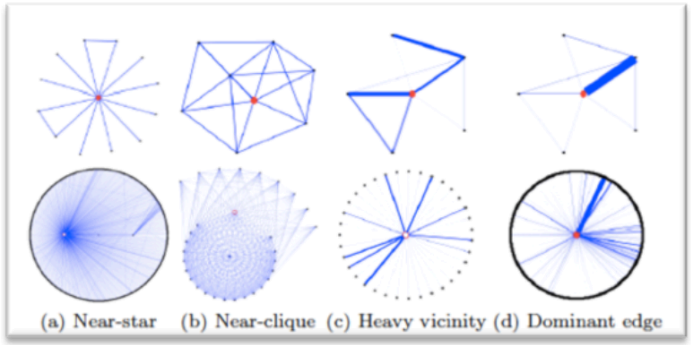


Attacker:
Near-Star

Detecting DoS attack



(a) Single large graph representing TCP SYN and ICMP PING network traffic, with two Denial of Service (DoS) attacks taking place.



(a) Near-star (b) Near-clique (c) Heavy vicinity (d) Dominant edge

Graph Visualizations

Communities	Graph Search	Network Info Flow	Bayesian Networks
Centralities	Graph Query	Shortest Paths	Latent Net Inference
Ego Net Features	Graph Matching	Graph Sampling	Markov Networks

Middleware and Database

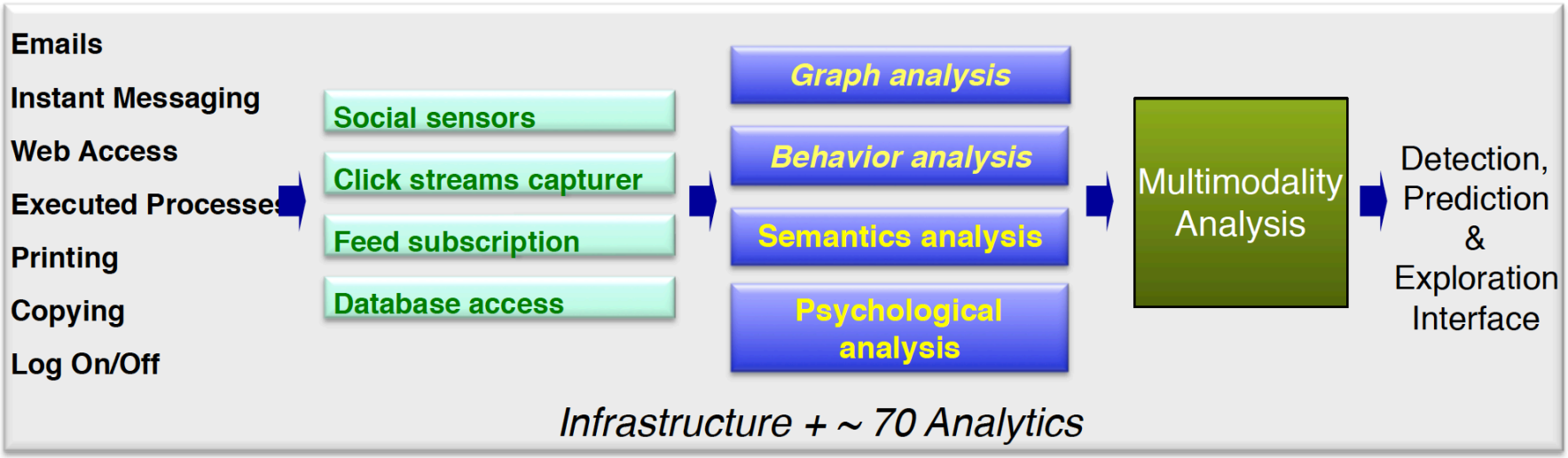
Anomaly Detection at Multiple Scales

Based on President Executive Order 13587

Goal: System for Detecting and Predicting Abnormal Behaviors in Organization, through large-scale social network & cognitive analytics and data mining, to decrease insider threats such as espionage, sabotage, colleague-shooting, suicide, etc.

"Enterprise Information Leakage Impacted economy and jobs" Feb 2013

"What's emerged is a multibillion dollar detective industry"
npr Jan 10, 2013



Story --- Espionage Example

(1)Personal stress:

(1)Gender identity confusion

(2)Family change (termination of a stable relationship)

(2)Job stress:

– Dissatisfaction with work

- Job roles and location (sent to Iraq)
- long work hours (14/7)

(1)Unstable Mental Status:

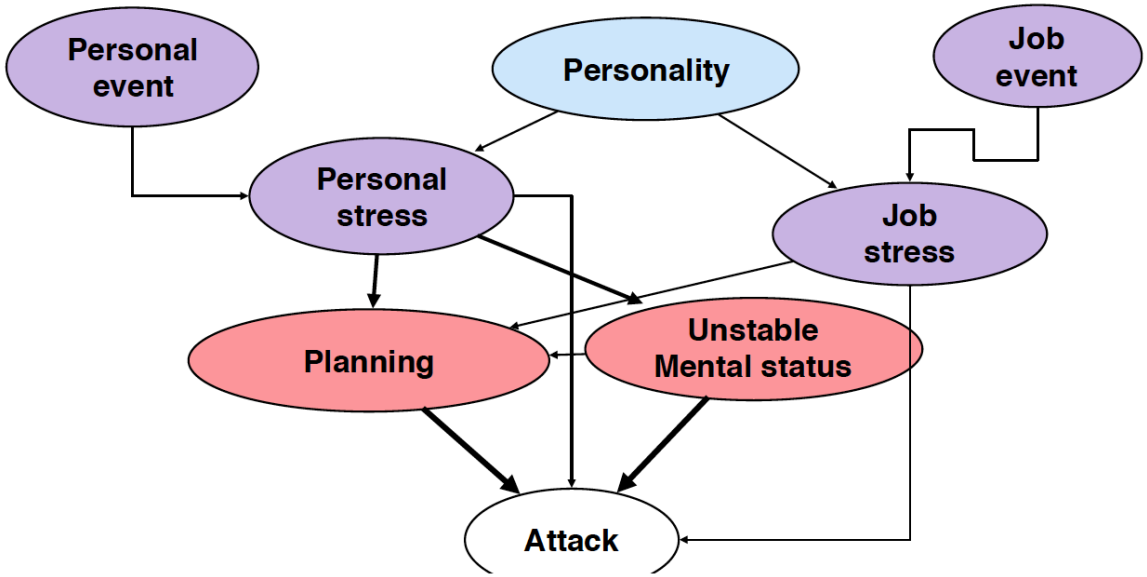
(1)Fight with colleagues, write complaining emails to colleagues

(2)Emotional collapse in workspace (crying, violence against objects)

(3)Large number of unhappy Facebook posts (work-related and emotional)

(2)Planning:

– Online chat with a hacker confiding his first attempt of leaking the information



(1) Attack:

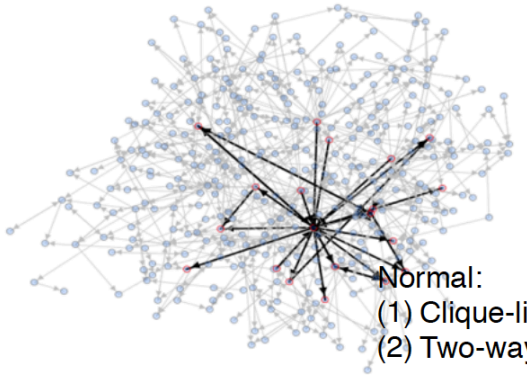
– Brought music CD to work and downloaded/ copied documents onto it with his own account

Fraud Detection for Bank

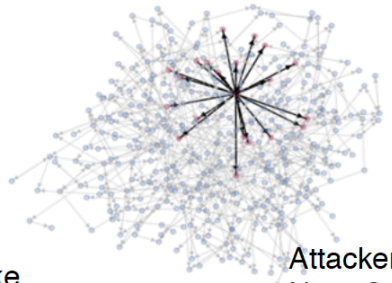
Network Info Flow

Ego Net Features

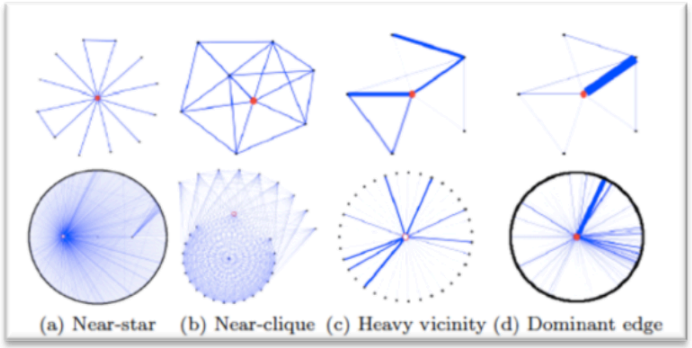
Ponzi scheme Detection



Normal:
(1) Clique-like
(2) Two-way links



Attacker:
Near-Star



3 大数据存储

2.1 数据获取、收集

系统日志采集

- 系统日志采集，如Hadoop的Chukwa，Cloudera的Flume，Facebook的Scribe，~100MB/s

用户数据采集

- 用户使用数据获取

其他企业级数据获取

- 机器数据：由系统、为现代企业助力的技术和架构创建的数字描述

机器数据源

数据类型	位置	它可以告诉您什么
应用日志	本地日志文件、log4j、log4net、Weblogic、WebSphere、JBoss、.NET、PHP	用户活动、欺诈检测、应用性能
业务流程日志	业务流程管理日志	跨渠道客户活动、购买、帐户变更以及问题报表
呼叫详细信息记录	呼叫详细信息记录 (CDR)、计费数据记录、事件数据记录均由电信和网络交换机所记录。	计费、收入保证、客户保证、合作伙伴结算，营销智能
点击流数据	Web 服务器、路由器、代理服务器和广告服务器	可用性分析、数字市场营销和一般调查
配置文件	系统配置文件	如何设置基础设施、调试故障、后门攻击、"定时炸弹"病毒
数据库审计日志	数据库日志文件、审计表	如何根据时间修改数据库数据以及如何确定修改人
文件系统审计日志	敏感数据存储在共享文件系统中	监测并审计敏感数据读取权限
管理并记录 API	通过 OPSEC Log Export API (OPSEC LEA) 和其他 VMware 和 Citrix 供应商特定 API 的 Checkpoint 防火墙	管理数据和日志事件
消息队列	JMS、RabbitMQ 和 AquaLogic	调试复杂应用中的问题，并作为记录应用架构基础
操作系统度量、状态和诊断命令	通过命令行实用程序（例如 Unix 和 Linux 上的 ps 与 iostat 以及 Windows 上的性能监视器）显示的 CPU、内存利用率和状态信息	故障排除、分析趋势以发现潜在问题并调查安全事件
数据包/流量数据	tcpdump 和 tcpflow 可生成 pcap 或流量数据以及其他有用的数据包级和会话级信息	性能降级、超时、瓶颈或可疑活动可表明网络被入侵或者受到远程攻击
SCADA 数据	监视控制与数据采集 (SCADA)	识别 SCADA 基础结构中的趋势、模式和异常情况，并用于实现客户价值

2.2 云存储

公有云

- 由商业机构等管理和运营（如Amazon、Google和阿里云）
- 优点：成本低可扩展性好；缺点：缺乏控制安全性相对较差

私有云

- 由组织自行部署管理和运营，可以在机构内部或外部（如HDFS）
- 优点：安全性强；缺点：增加管理维护成本

混合云

- 由两种不同模式组成
- 优点：利用各自优点；缺点：需要协调管理相对复杂

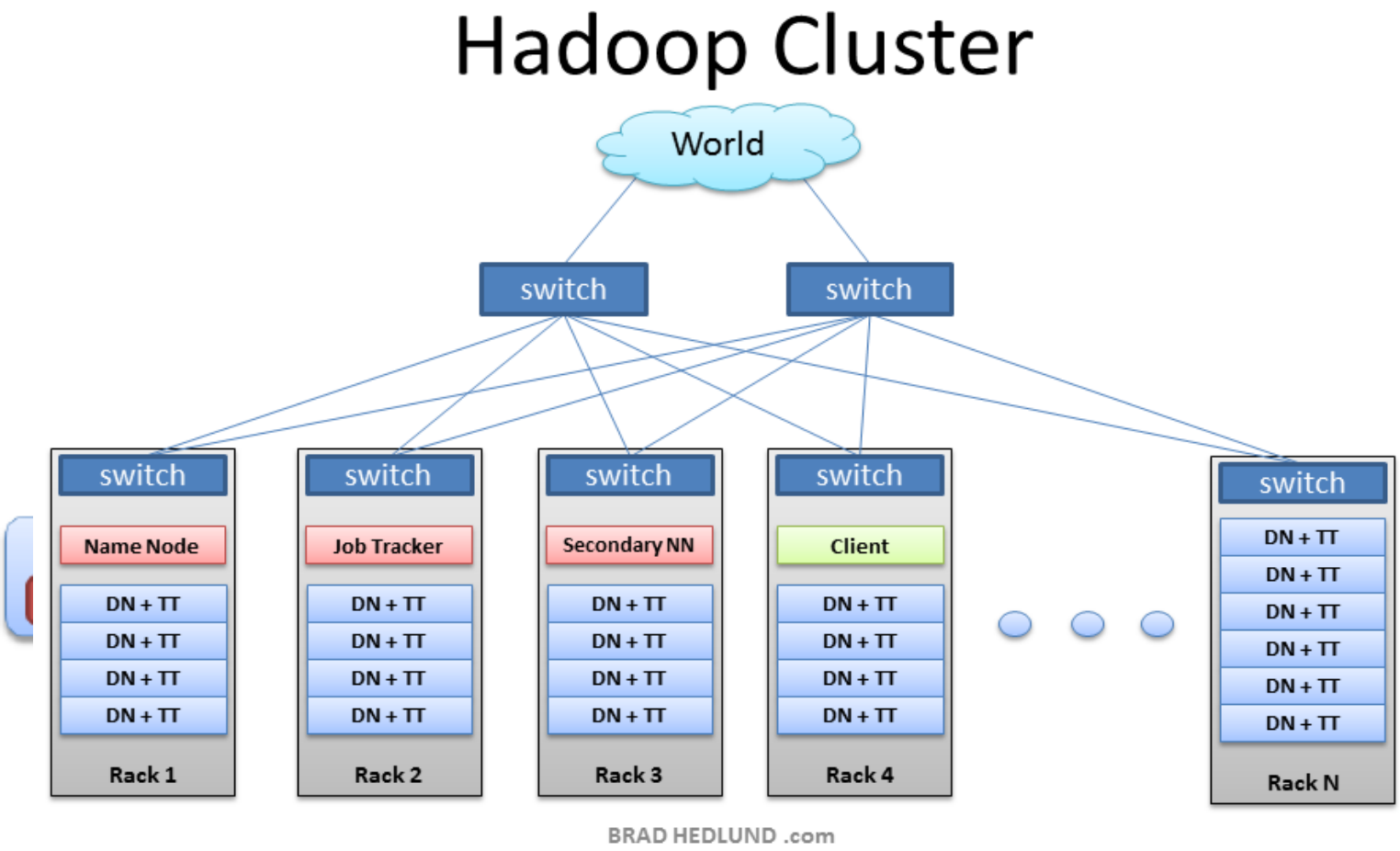
Ucloud: https://www.ucloud.cn

2.3 分布式文件系统：HDFS

Hadoop Distributed File System (HDFS)

大数据技术根基 - HDFS

2.3 分布式文件系统：HDFS[续]



2.3 分布式文件系统：HDFS [续]

Hadoop Distributed File System (HDFS)

优点：

1. 高容错性
- 多个副本；丢失后自动恢复
2. 适合批处理
- 数据位置暴露给计算框架；移动计算而非数据
3. 适合大数据处理
- TB、甚至PB级数据；百万规模以上的文件数量；万计节点规模
4. 流式文件访问
- 一次性写入，多次读取；保证数据一致性
5. 可构建在廉价机器上
- 通过多副本提高可靠性；提供了容错和恢复机制

缺点：

1. 高延迟、低吞吐率数据访问
2. 单个结点存储所有信息
- 大量小文件占用NameNode大量内存；寻道时间超过读取时间
3. 串行写入
- 一个文件同一个时间只能有一个写者；仅支持append

2.4 分布式数据库：HBase

HDFS：仅支持文件追加写（append）、数据非结构化

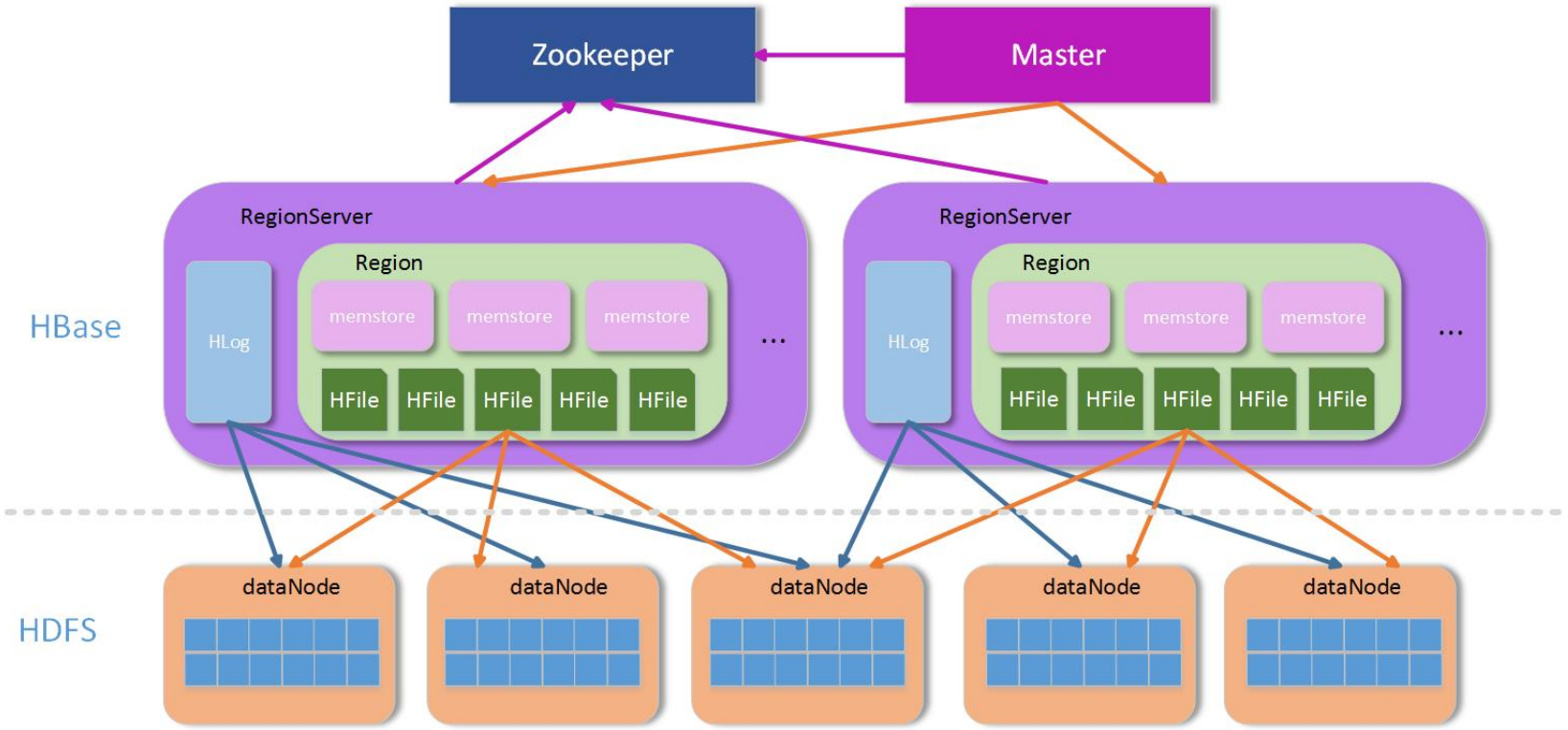
分布式数据库 - HBase

谁在用HBase？

2.4 分布式数据库：HBase [续]

架构

- Zookeeper群、Master群和RegionServer群

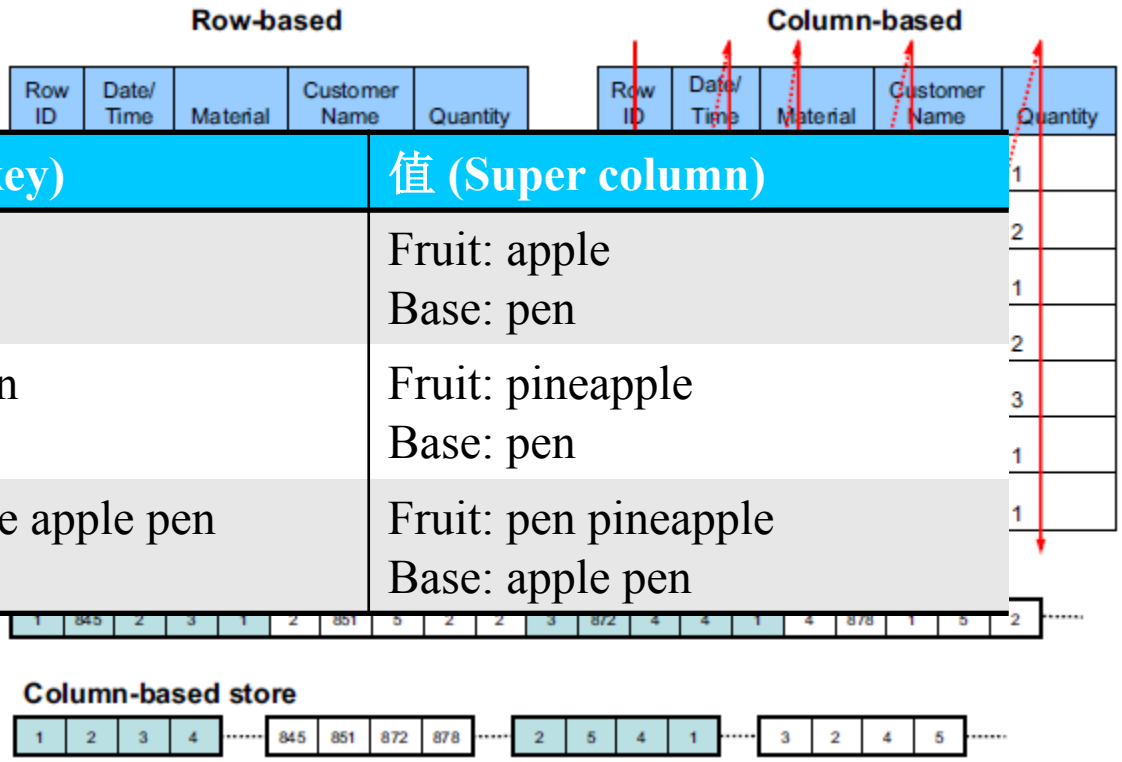


2.4 分布式数据库：HBase [续]

Hbase的核心：NoSQL

- Not Only SQL (NoSQL)
- SQL关系型数据库 – [X] 超高并发读写、 [X] 海量数据的并发查询、 [X] 数据库的横向扩展
- 行式存储
- Key-Value键值对

行键 (Row key)	值 (Super column)
Apple pen	Fruit: apple Base: pen
Pineapple pen	Fruit: pineapple Base: pen
Pen pineapple apple pen	Fruit: pen pineapple Base: apple pen



4 大数据处理

3.1 分布式数据处理

如何对付大数据处理：**分而治之**

- 并行处理相互间不具有计算依赖关系的大数据 -> 分而治之

上升到抽象模型：**Mapper与Reducer**

- 用Map和Reduce提供了高层的并行编程抽象模型（MPI等没有）

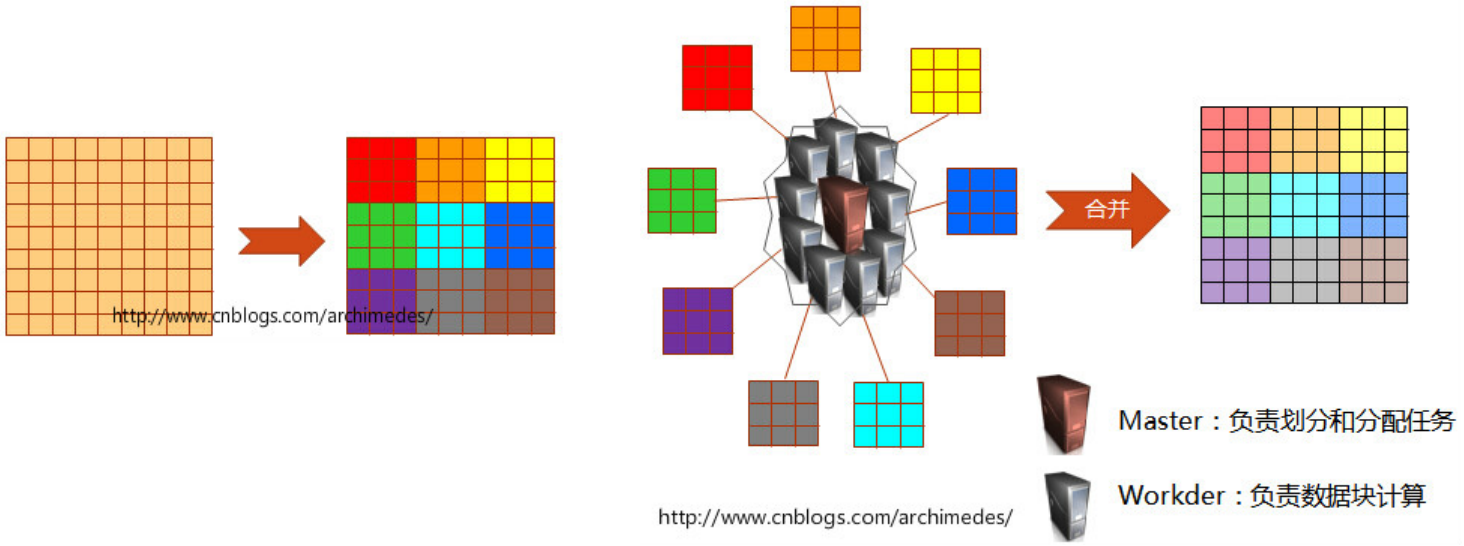
上升到框架：**统一框架，隐藏系统层细节**

- 统一的计算框架无需考虑数据存储、划分、分发、结果收集、错误恢复等诸多细节

3.2 分布式数据处理 – 分而治之

如何对付大数据处理：分而治之

- 并行处理相互间不具有计算依赖关系的大数据 -> 分而治之



3.2 分布式数据处理 – 分而治之 [续]

3.3 分布式数据处理 - MapReduce

映射（Mapping）

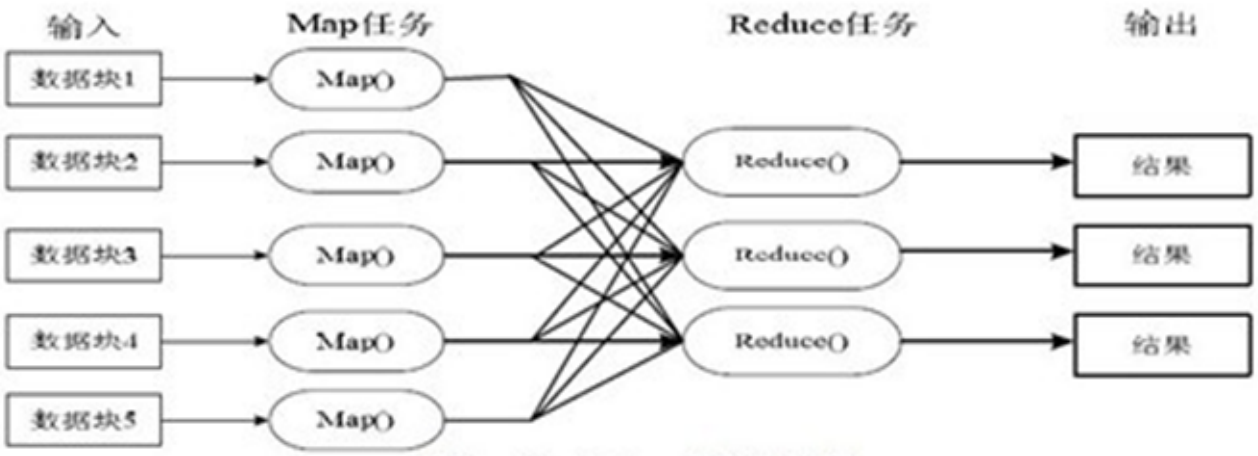
- 对集合里的每个目标应用同一个操作。如把每个对象乘2

化简（Reducing）

- 遍历集合中的元素来返回一个综合的结果。如输出所有对象的和

举例

- 扑克牌：计数每种花色张数



3.3 分布式数据处理 – MapReduce [续]

上升到抽象模型：**Mapper与Reducer**

- 用Map和Reduce提供了高层的并行编程抽象模型（MPI等没有）
- 扑克牌：分到每人手里的<k1,v1> -> 一堆随机牌，每人汇总的List(<k2,v2>) -> 每种花色张数

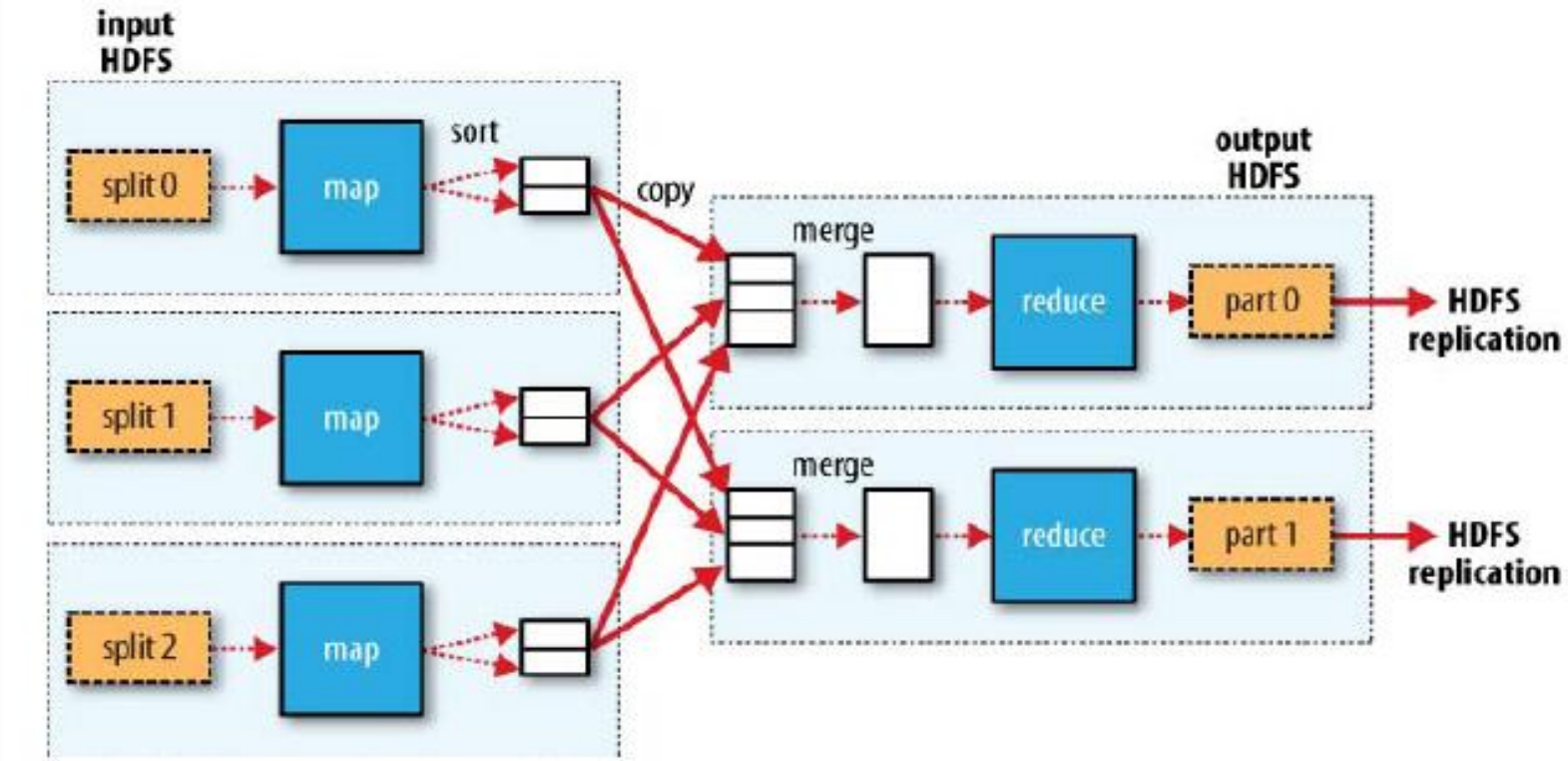
函数	输入	输出	说明
Map	$<k_1,v_1>$	List($<k_2,v_2>$)	1. 将数据集解析成一批<k, v>对作为输入； 2. 每一个输入$<k_1,v_1>$会输出一批$<k_2,v_2>$为中间结果
Reduce	$<k_2, List(v_2)>$	$<k_3,v_3>$	$<k_2, List(v_2)>$中$List(v_2)$是属于同个k_2的值

3.4 分布式数据处理 – Hadoop框架

上升到框架：**统一框架Hadoop，隐藏系统层细节**

- 统一的计算框架无需考虑数据存储、划分、分发、结果收集、错误恢复等诸多细节

Hadoop



3.4 分布式数据处理 – Hadoop框架 [续]

谁在用Hadoop？

- 第一梯队：这类公司已经将Hadoop当作大数据战略武器
- 第二梯队：这类公司将Hadoop 产品化
- 第三梯队：这类公司创造对Hadoop整体生态系统产生附加价值的产品
- 第四梯队：这类公司消费Hadoop，并给规模比第一类和第二类小的公司提供基于Hadoop的服务

3.4 分布式数据处理 – Hadoop框架 [续]

Hadoop经典案例

- Last.fm – 图表生成、用户使用统计等
- Facebook – 用户浏览情况分析、广告点击数、成本分析
- Nutch – 可扩展的网络爬虫
- Rackspace – 汇总多个邮件代理服务器的消息日志分析
- Wukong – Infochimps项目处理图数据，不超过一页脚本处理TB级图数据

3.4 分布式数据处理 – Hadoop框架 [续]

Hadoop实例

- WordCount部分代码示例

```
//Mapper<keyin,valuein,keyout,valueout>
public static class WordCountMapper extends Mapper<Object, Text, Text, IntWritable>{
    private final static IntWritable one = new IntWritable(1);
    private Text word = new Text();

    @Override
    protected
            th
        String
        while(
            wo
            co
        }
    }
}
```

```
    //Reduce<keyin,valuein,keyout,valueout>
    //Reducer的valuein类型要和Mapper的va lueout类型一致,Reducer的valuein是Mapper的valueout经过
shuffle之后的值
    public static class WordCountReducer extends Reducer<Text, IntWritable, Text, IntWritable>{
        private IntWritable result = new IntWritable();

        @Override
        protected void reduce(Text key, Iterable<IntWritable> values,
                Context context)
                throws IOException, InterruptedException {
            int sum  = 0;
            for(IntWritable i:values){
                sum += i.get();
            }
            result.set(sum);
            context.write(key,result);//Context机制
        }
```

3.5 分布式内存计算 - Spark

从MapReduce说起

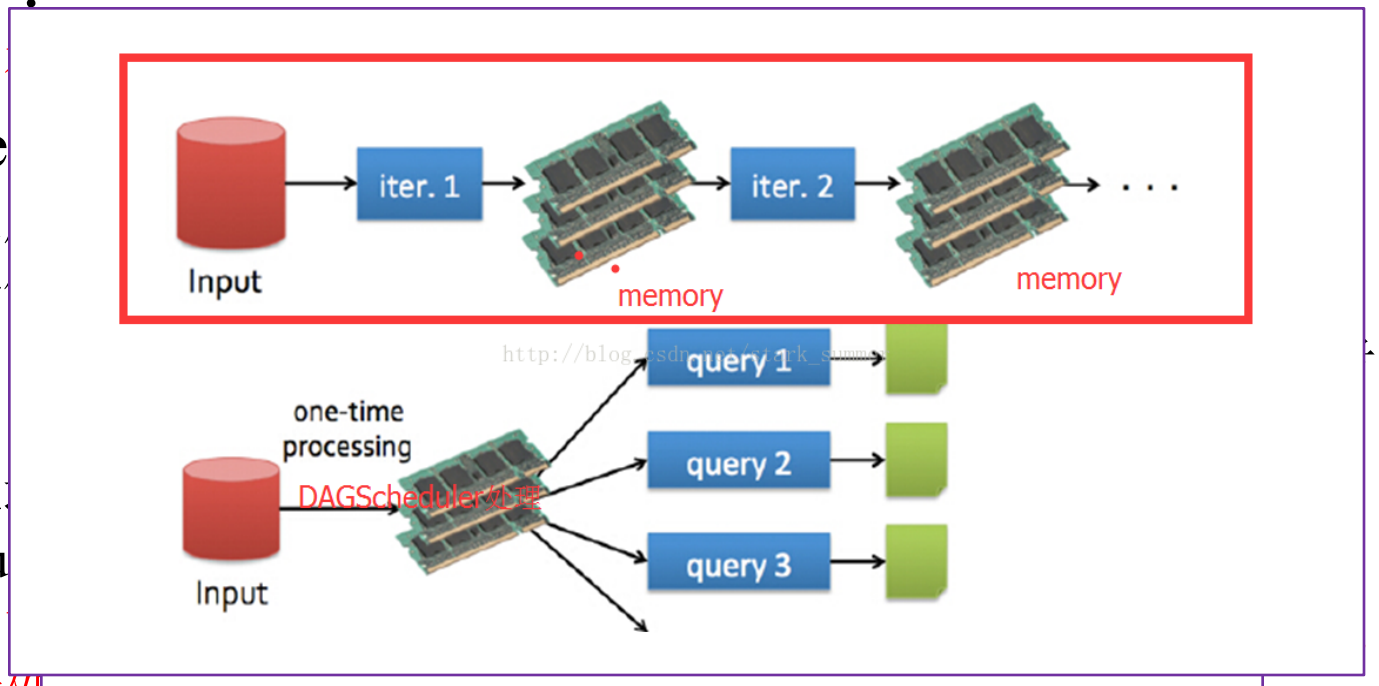
- 最大缺陷：网络、磁盘I/O

怎么办？

- 放内

MapRe

- 抽象
- 只提
- 复杂 自己管理的
- 处理
- Redu
- 时延 迭代式数据处理性能比较差。



3.5 分布式内存计算 - Spark [续]

Spark的崛起

谁在用Spark？

3.5 分布式内存计算 - Spark [续]

Spark经典案例

- 腾讯 – 广点通pCTR投放系统。基于日志的快速查询系统构建于Spark之上的Shark，比Hive快2-100倍
- Yahoo – 定向广告业务。基于广告者提供的观看广告并购买产品的样本客户，学习并寻找更多可能转化的用户
- 淘宝 – 阿里搜索和广告业务。从Mahout或自己写的MR解决复杂机器学习到使用Spark、Graphx运用于相关推荐算法
- 优酷土豆 – 视频推荐(图计算)、广告业务等。Spark相比于Hadoop交互查询响应快；模拟广告投放计算效率高、延迟小；机器学习、图计算等迭代计算，大大减少了网络传输、数据落地等

3.5 分布式内存计算 - Spark [续]

[What] 什么是Resilient Distributed Datasets （RDD）？

- 不变的、容错的、并行的数据结构
- 显式地将数据存储到磁盘和内存中，并能控制数据的分区

[Why] 为什么要RDD？

- 数据处理常见模型：1. Iterative Algorithms（迭代算法）; 2.Relational Queries（关联查询）; 3.MapReduce; 4. Stream Processing（流式处理）
- RDD支持四种模型

[How] 如何构建RDD？

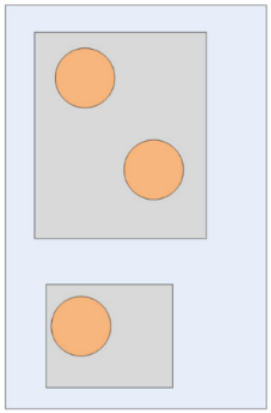
- 从文件系统
- 通过Scala集合对象并行化生成
- 通过对已存在的RDD transform生成
- 通过改变其他RDD的持久化状态

3.5 分布式内存计算 - Spark [续]

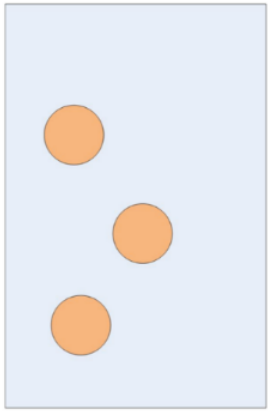
RDD Transformations和Actions

FLATTEN运算

Transforma



flatten



Seq[W]))]

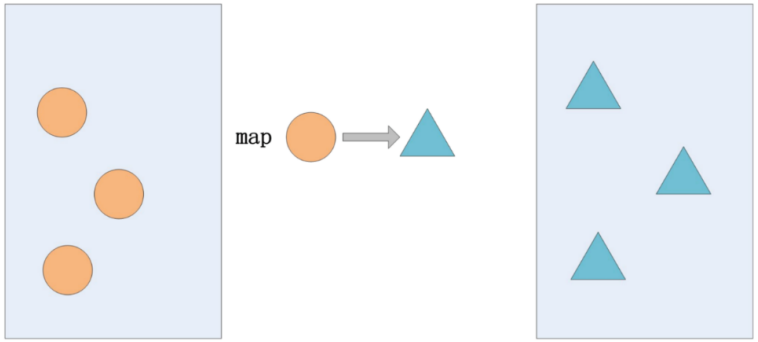
ng)

Action

DDs)

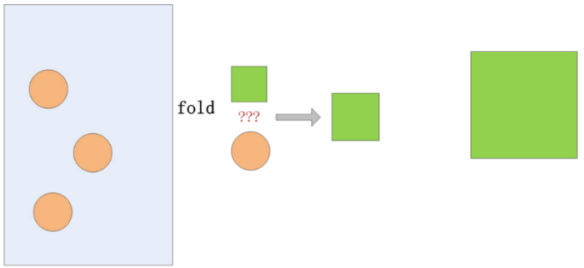
```
scala> List(List(1,2),Nil,List(3)).flatten
res10: List[Int] = List(1, 2, 3)
```

MAP运算




```
scala> List(1,2,3).map(_.toString)
res1: List[String] = List(1, 2, 3)
```

FOLD/REDUCE运算

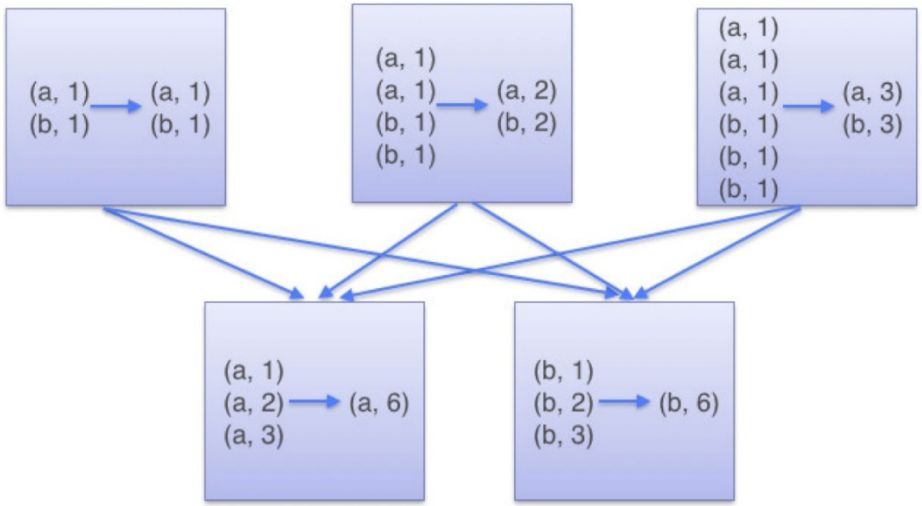



```
scala> List(1,2,3).foldLeft(0)(_ + _)
res3: Int = 6
```

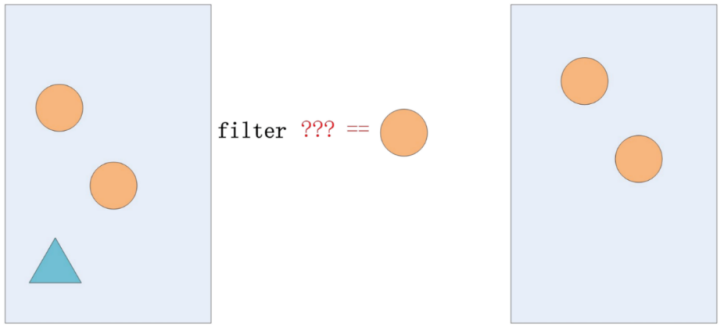
```
scala> List(1,2,3).foldLeft(List[Int]())((acc,i) => i :: acc)
res8: List[Int] = List(3, 2, 1)
```

```
scala> List(1,2,3).reduce(_ * _)
res11: Int = 6
```

ReduceByKey



FILTER运算




```
scala> List(1,2,2).filter(_ == 2)
res2: List[Int] = List(2, 2)
```

FLATTEN运算


```
scala> List(List(1,2),Nil,List(3)).flatten
res10: List[Int] = List(1, 2, 3)
```

3.5 分布式内存计算 - Spark [续]

Spark代码实例

- WordCount部分代码示例

```
val textFile = sc.
val counts = textF
                .
                .
counts.saveAsTextF
```

Java

```
JavaRDD<String> textFile = sc.textFile("hdfs://...");
JavaRDD<String> words = textFile.flatMap(new FlatMapFunction<String, String>() {
  public Iterable<String> call(String s) { return Arrays.asList(s.split(" ")); }
});
JavaPairRDD<String, Integer> pairs = words.mapToPair(new PairFunction<String,
String, Integer>() {
  public Tuple2<String, Integer> call(String s) { return new Tuple2<String, Integer>
(s, 1); }
});
JavaPairRDD<String, Integer> counts = pairs.reduceByKey(new Function2<Integer, Integ
er, Integer>() {
  public Integer call(Integer a, Integer b) { return a + b; }
});
counts.saveAsTextFile("hdfs://...");
```

Scala

```
text_file = sc.textF
counts = text_file.
             .map(l
             .reduceByKey(lambda a, b: a + b)
counts.saveAsTextFile("hdfs://...")
```

Python

3.6 Hadoop vs. Spark

应用场景

Hadoop – 极大数据量 > TB - PB级

- 单次海量数据的离线分析处理
- 大规模Web信息搜索
- 数据密集型并行计算

Spark – 内存可容纳 ≈TB级

- 多次操作特定数据集，迭代运算
- 搜索引擎 – PageRank
- 计算相似 – Single Source Shortest Path (单源最短路径)

3.6 Hadoop vs. Spark [续]

性能

- Hadoop：适合不能全部读入内存；单次读取、类似 ETL （抽取、转换、加载）操作的任务，比如数据转化、数据整合等时
- Spark：适合数据不太大内存放得下，重复读取同样数据迭代计算

上手

- Hadoop：Java编写，需要学习语法，有一些工具（Pig、Hive等）简化
- Spark：Scala、Java和Python，还支持交互式命令模式

兼容性

- Spark兼容Hadoop数据源

3.6 Hadoop vs. Spark [续]

容错

- Hadoop：硬盘静态数据，硬盘驱动失败处自动重启执行
- Spark：内存失效，需要手动设置checkpoint

数据处理

- Hadoop：批处理
- Spark：实时/批数据处理，迭代任务

5 大数据分析

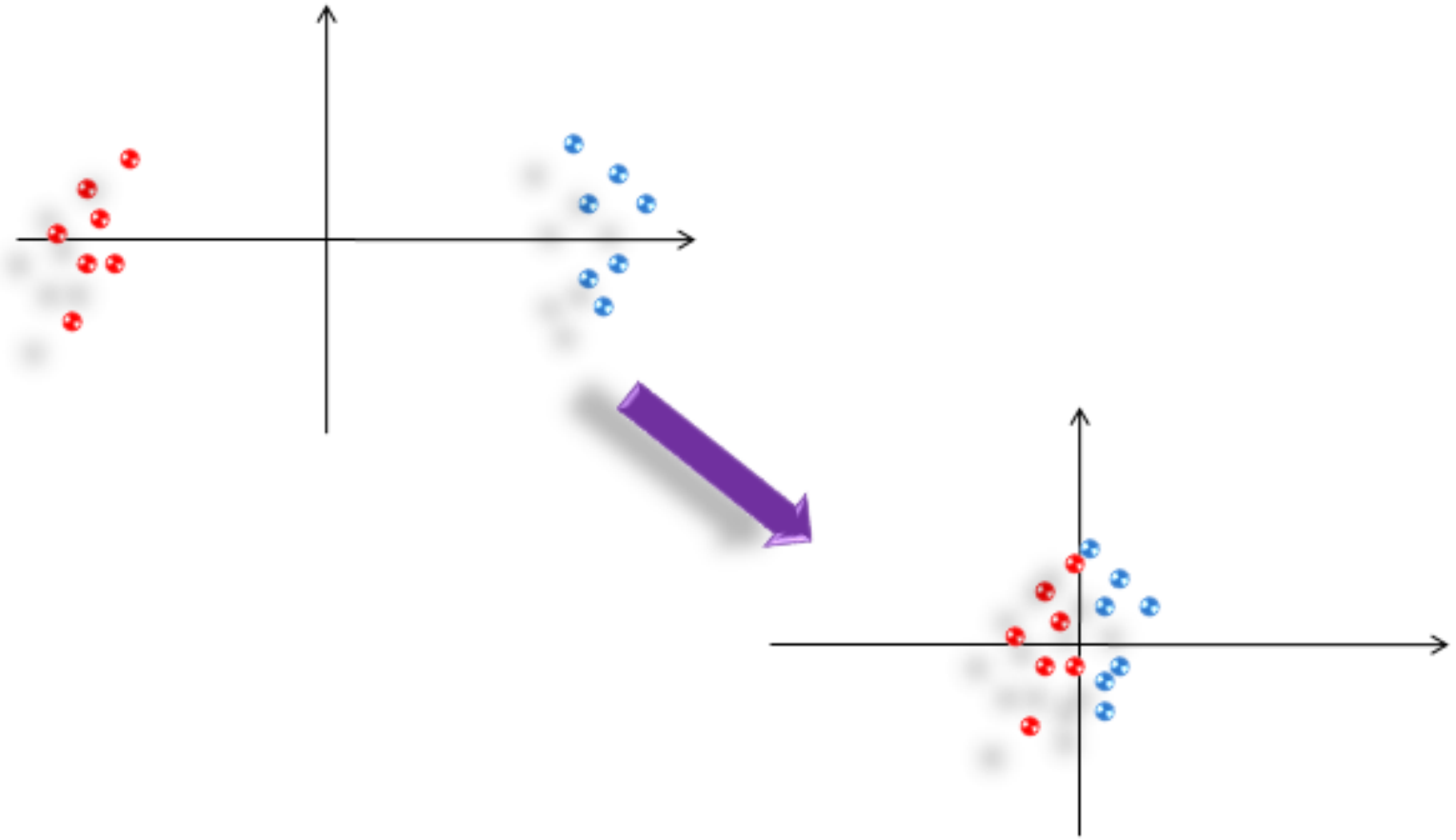
4.1 数据分析流程

数据分析整体流程

1. 数据采集
2. 数据预处理
3. 统计分析
4. 特征提取
5. 数据挖掘

4.2 数据预处理

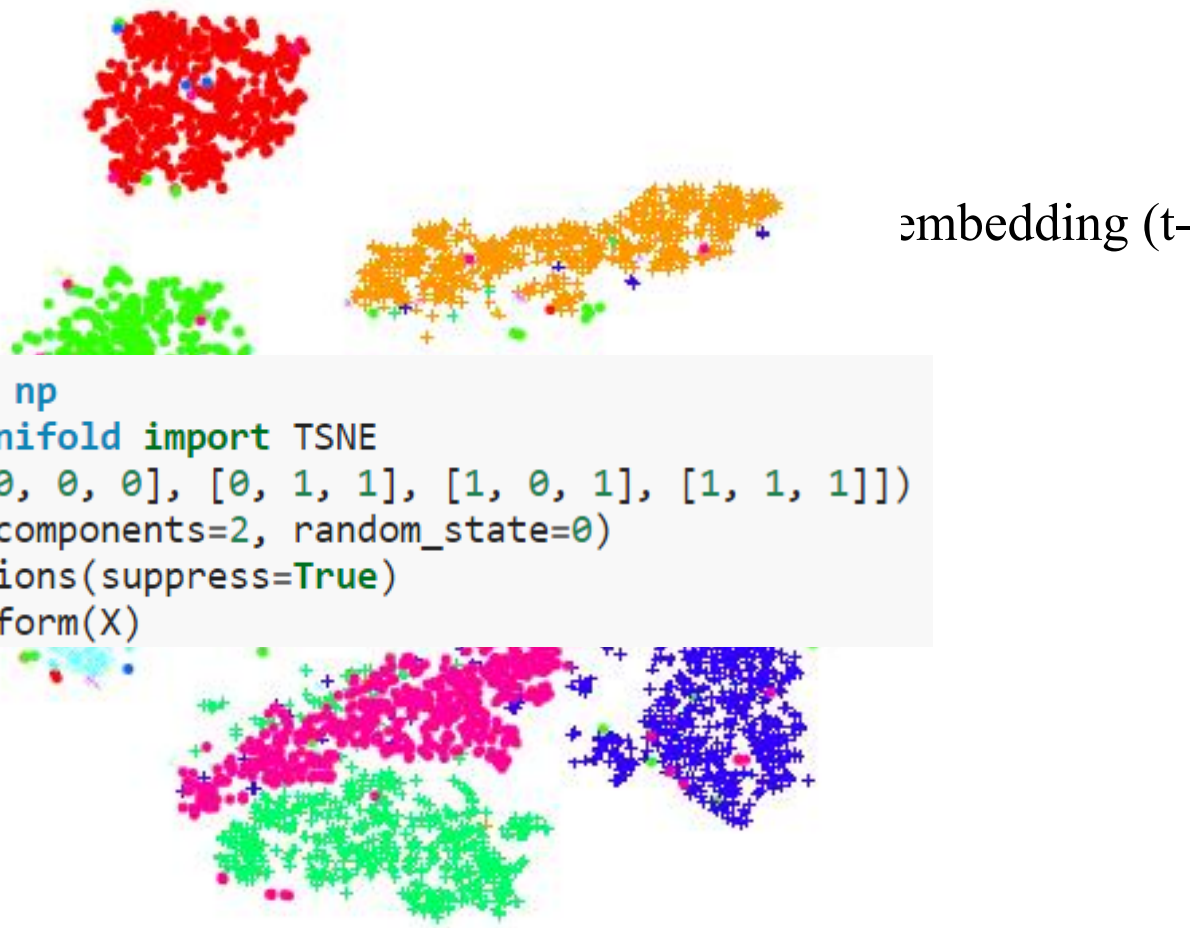
数据预处理



4.3 统计分析

可视化

- 库：m
- 平台：
- 算法：



embedding (t-SNE)

```
import numpy as np
from sklearn.manifold import TSNE
X = np.array([[0, 0, 0], [0, 1, 1], [1, 0, 1], [1, 1, 1]])
model = TSNE(n_components=2, random_state=0)
np.set_printoptions(suppress=True)
model.fit_transform(X)
```

4.4 数据挖掘

大数据分析理论基础（大数据情景基本结论）

- 大数据 + 简单模型 >> 小数据 + 复杂模型

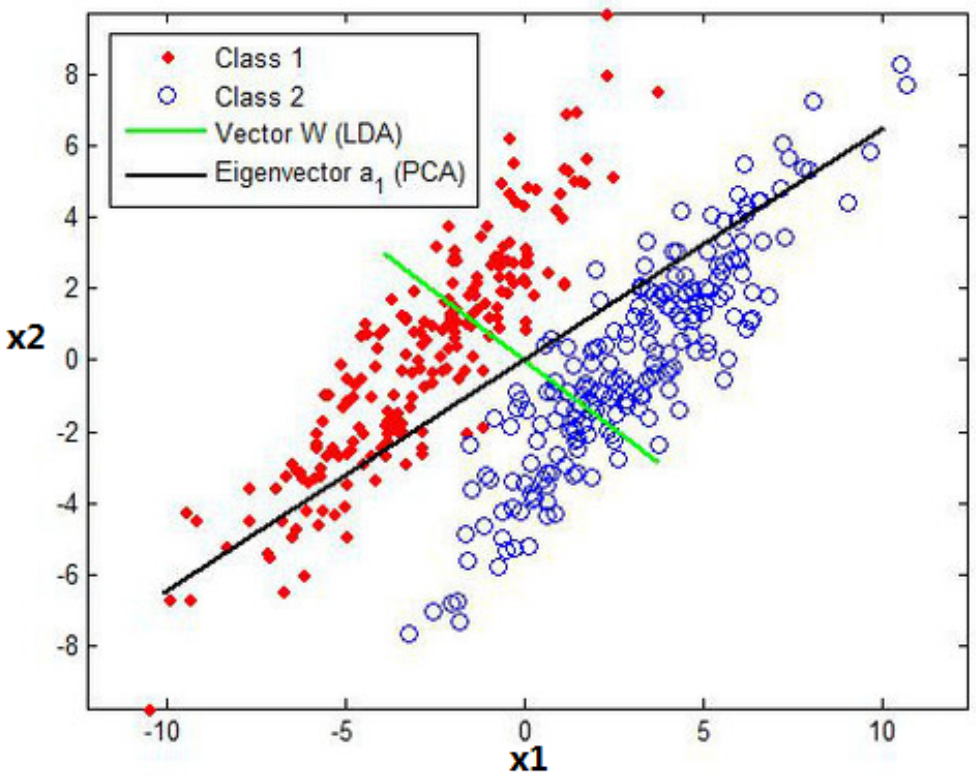
算法模型（Scikit-Learn）

- 数据降维
- 回归算法
- 分类算法
- 聚类算法
- 深度学习

4.4.1 数据挖掘算法 - 数据降维

经典算法

- Principal Component Analysis (PCA, 主成分分析)
- Linear discriminant analysis (LDA, 线性判别分析)
- ……

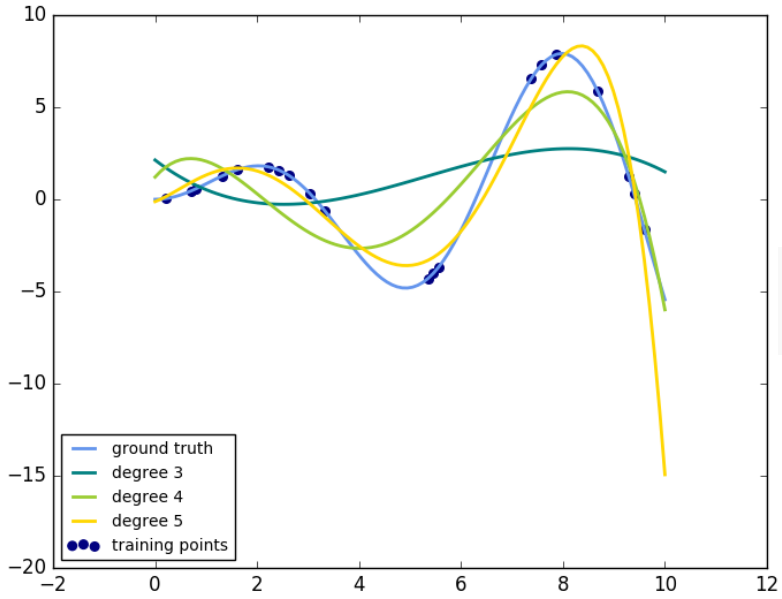



```
lda = LinearDiscriminantAnalysis(n_components=2)
X_r2 = lda.fit(X, y).transform(X)
```

4.4.2 数据挖掘算法 - 回归算法

经典算法

- Linear regression (LR, 线性回归)
- Polynomial regression (多项式回归)
- ……



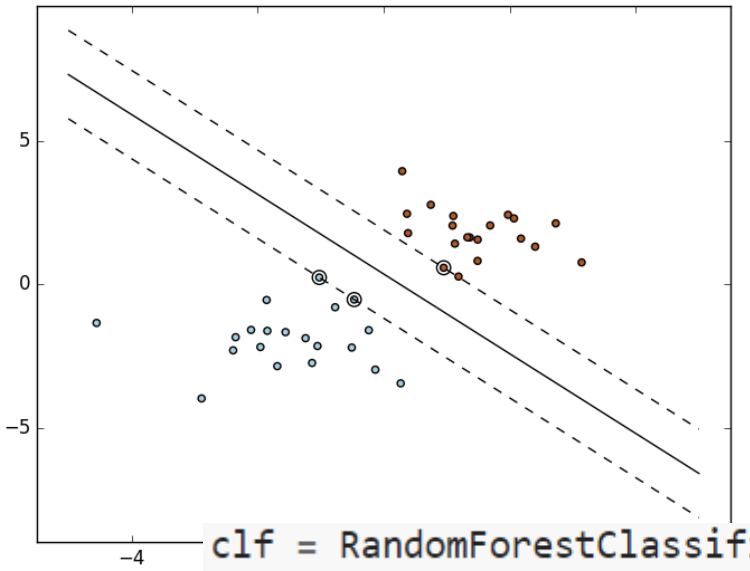

```
        # Create linear regression object
for count, degree in enumerate([3, 4, 5]):
    model = make_pipeline(PolynomialFeatures(degree), Ridge())
    model.fit(X, y)
        regr.fit(diabetes_X_train, diabetes_y_train)
```

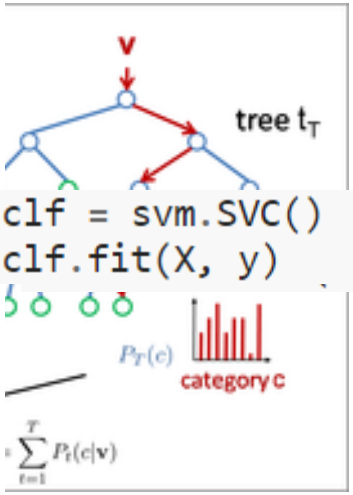
4.4.3 数据挖掘算法 - 分类算法

经典算法

- Decision tree (DT, 决策树)
- Random forest (RF, 随机森林)
- Support vector machine (SVM, 支持向量机)
- ……

```
clf.predict([[2., 2.]])
```






```
clf = svm.SVC()
clf.fit(X, y)
```

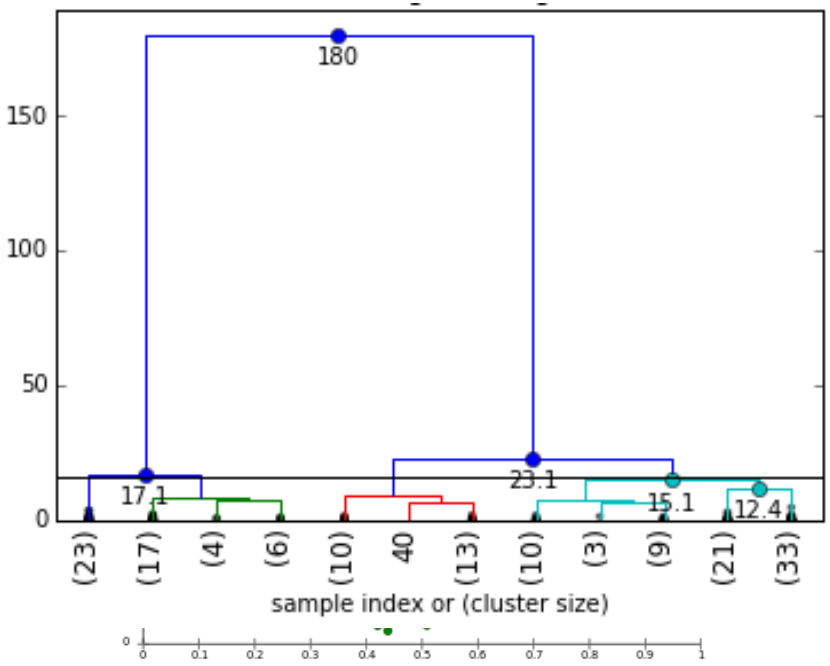
```
ionTreeClassifier()
 Y)
```

```
clf = RandomForestClassifier(n_estimators=10)
clf = clf.fit(X, Y)
```

4.4.3 数据挖掘算法 – 聚类算法

经典算法

- K-means (k均值)
- Hierarchical clustering (层级聚类)
- ……



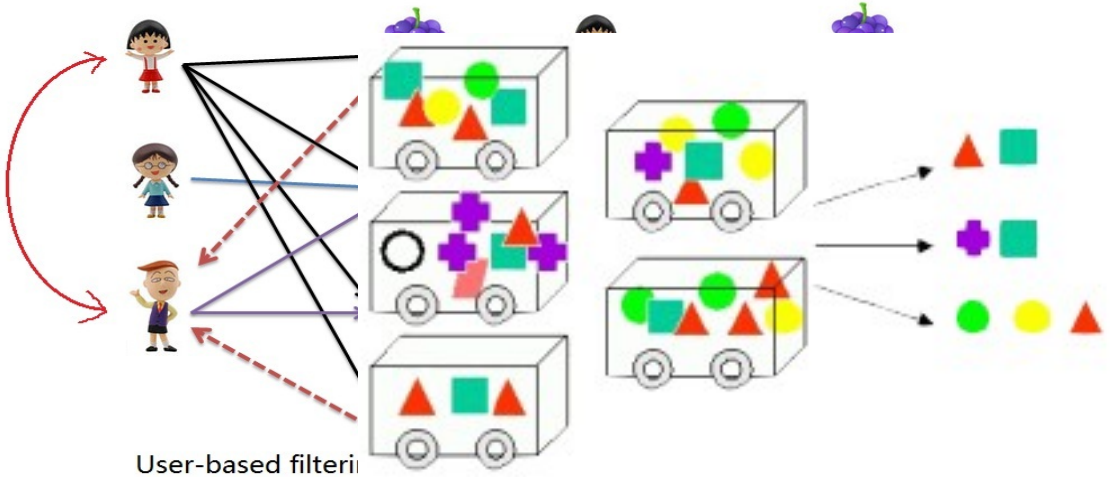

```
y_pred = KMeans(n_c...                                    ...ate).fit_predict(X_aniso)
```

```
Z = linkage(X, 'ward')
k=2
fcluster(Z, k, criterion='maxclust')
```

4.4.4 数据挖掘算法 - 推荐算法

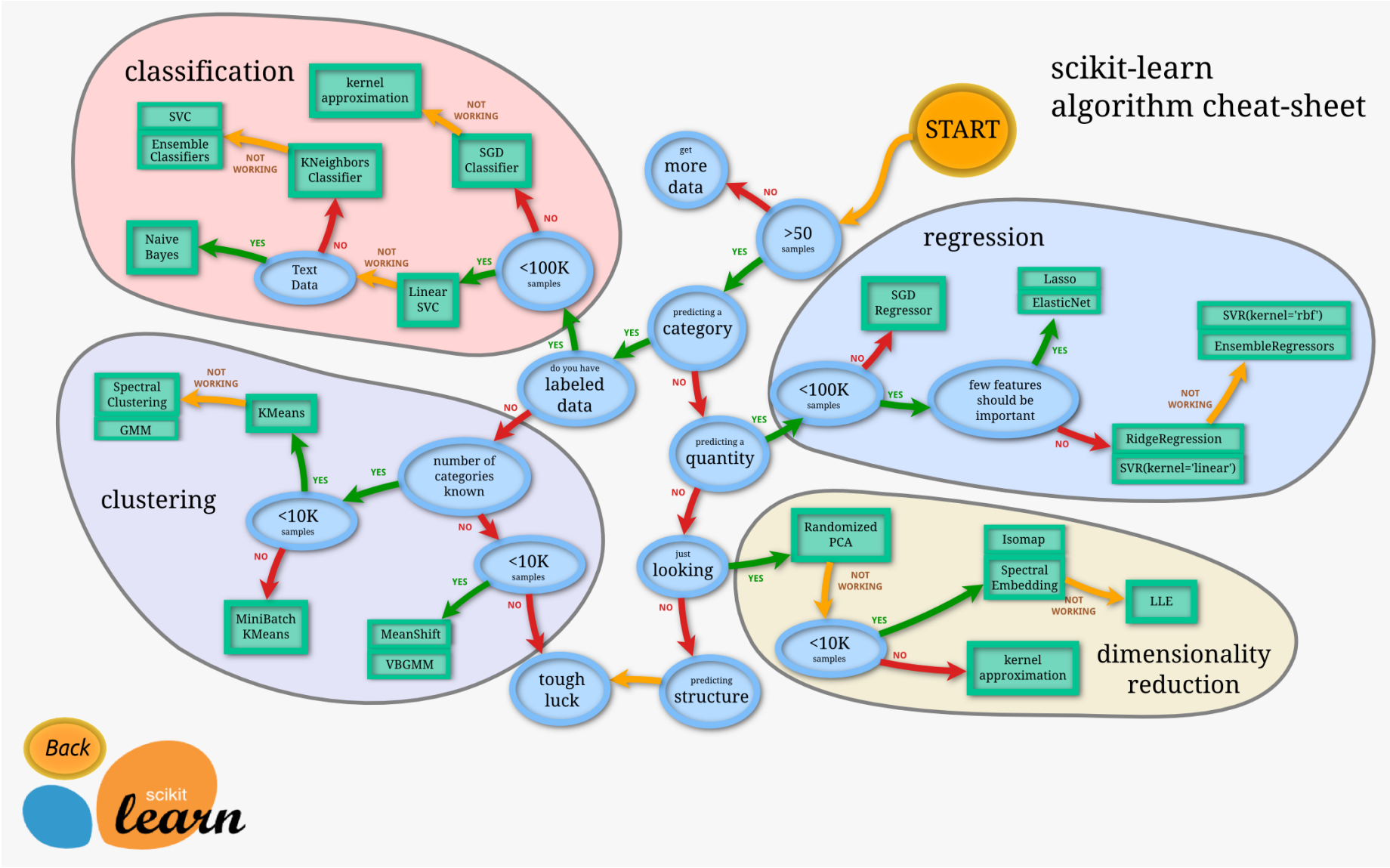
经典算法

- Collaborative filtering (CF, 协同过滤)
- Association rule-based Mining (ARM, 关联规则挖掘)
- ……



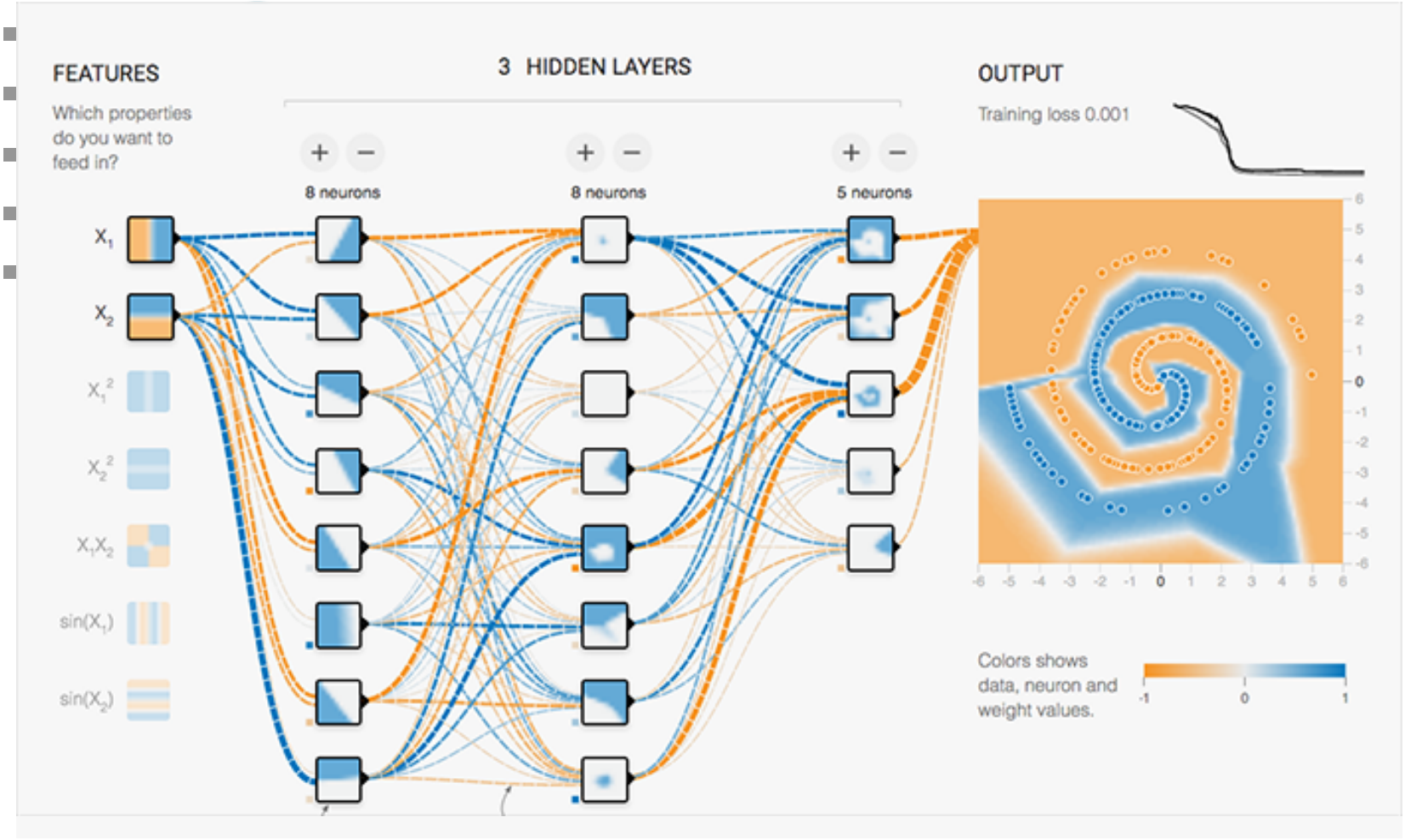

```
cosine similarities

:
column
ns)) :
similarities
germany.ix[:,i],data_germany.ix[:,j])

neighbours to an item
ta_ibs.columns,columns=[range(1,11)])

# Loop through our similarity dataframe and fill in neighbouring item names
for i in range(0,len(data_ibs.columns)):
    data_neighbours.ix[i,:10] = data_ibs.ix[0:,i].order(ascending=False)[:10]
```

4.4.5 数据挖掘算法 - 选择算法



4.4.6 数据挖掘算法 - 深度学习

深度学习简介



4.4.7 数据挖掘算法 - 深度学习平台Tensorflow

潜力无限的Google Tensorflow平台

- 玩一下？ http://playground.tensorflow.org/

谁在用Tensorflow？

4.4.7 数据挖掘算法 - 深度学习平台Tensorflow [续]

```
import tensorflow as tf
import tensorflow.examples.tutorials.mnist.input_data as input_data
mnist = input_data.read_data_sets("MNIST_data/", one_hot=True)
x = tf.placeholder(tf.float32, [None, 784])
y_actual = tf.placeholder(tf.float32, shape=[None, 10])
W = tf.Variable(tf.zeros([784,10]))          #初始化权值W
b = tf.Variable(tf.zeros([10]))              #初始化偏置项b
y_predict = tf.nn.softmax(tf.matmul(x,W) + b)     #加权变换并进行softmax回归，得到预测概率
cross_entropy = -tf.reduce_sum(y_actual*tf.log(y_predict))  #求交叉熵
train_step = tf.train.GradientDescentOptimizer(0.01).minimize(cross_entropy)    #用梯度下降
法使得残差最小

correct_prediction = tf.equal(tf.argmax(y_predict,1), tf.argmax(y_actual,1))    #在测试阶
段，测试准确度计算
accuracy = tf.reduce_mean(tf.cast(correct_prediction, "float"))                 #多个批次的
准确度均值

init = tf.initialize_all_variables()
with tf.Session() as sess:
    sess.run(init)
    for i in range(1000):                    #训练阶段，迭代1000次
        batch_xs, batch_ys = mnist.train.next_batch(100)          #按批次训练，每批100行数据
        sess.run(train_step, feed_dict={x: batch_xs, y_actual: batch_ys})    #执行训练
        if(i%100==0):                        #每训练100次，测试一次
            print "accuracy:",sess.run(accuracy, feed_dict={x: mnist.test.images,
y_actual: mnist.test.labels})
```

参数定义

模型公式

损失函数

选择解法

$Vx + b)$

6 小结

5 小结

大数据简介

大数据存储

- 分布式文件系统 - HDFS

大数据处理

- 分布式数据处理平台- Hadoop
- 分布式内存数据处理平台- Spark

大数据分析

- 高维数据可视化
- 各类数据挖掘算法简介 – Scikit-learn
- 深度学习Tensorflow平台

Q & A

Thank you!

缩写表

ARM = Association rule-based Mining (关联规则挖掘)
CF= Collaborative filtering (协同过滤)
CNN = Convolutional Neural Network (卷积神经网络)
DNN = Deep Neural Network (深度神经网络)
DT = Decision tree (决策树)
HBase = Hadoop Database (Hadoop数据库)
HDFS = Hadoop Distributed File System (Hadoop分布式文件系统)
LDA = Linear discriminant analysis (线性判别分析)
LR = Linear regression (线性回归)
NoSQL = Not Only SQL (非关系型的数据库)
PCA = Principal Component Analysis (主成分分析)
RDD = Resilient Distributed Datasets (弹性分布式数据集)
RF = Random forest (随机森林)
RNN = Recurrent Neural Network (循环神经网络)
SVM = Support vector machine (支持向量机)
t-SNE = t-distributed stochastic neighbor embedding (t-分布邻域嵌入)

云计算与大数据的区别

云和大数据，应该是近几年IT炒的最热的两个话题了。在我看来，这两者之间的不同就是： **云是做新的瓶，装旧的酒；大数据是找合适的瓶，酿新的酒**。

云说到底是一种基础架构的革命。原先用物理服务器的应用，在云中变成以各种虚拟服务器的形式交付出去，从而计算、存储、网络资源都能被更有效率的利用了。

大数据，是侧重于利用统计、机器学习、人工智能算法对数据进行有效的分析。关键在于有效利用的数据。