Transforming Data With Intelligence™

Previews of TDWI course books offer an opportunity to see the quality of our material and help you to select the courses that best fit your needs. The previews cannot be printed.

TDWI strives to provide course books that are content-rich and that serve as useful reference documents after a class has ended.

This preview shows selected pages that are representative of the entire course book; pages are not consecutive. The page numbers shown at the bottom of each page indicate their actual position in the course book. All table-of-contents pages are included to illustrate all of the topics covered by the course.

This page intentionally left blank.

Machine & Deep Learning: Delivering Insights from Big Data

1

You Will Learn

- The foundations of machine learning in chaos theory, game theory, and algorithms
- What “machine talk” is and how we architect for it
- How to bring search implementation theory to data with deep learning
- Introduction to TensorFlow, a library of open source algorithms
- Implementing machine and deep learning, including what processing techniques are involved
- Available analytics insights, outcomes, mashups, and KPIs
- How to visualize the results of machine and deep learning

Chaos Theory

3

Chaos Theory

- Chaos theory has been defined as the science that determines order in the randomness and unpredictability that exist in the natural and social systems
- Ex: coastlines, mountains, clouds, galaxy clusters, leaves, heart rhythms, etc.
- Chaos theory underlying principles
  - Non-linearity
  - Determinism
  - Sensitivity to initial conditions
  - Sustained irregularity in the behavior of the system
  - Unpredictable long-term behavior
  - Self-organization
  - Positive feedback loops
  - Qualitative character of systems rather than numerical predictions of future states
  - Unstable
  - Aperiodic – no periodic repetition of values

Characteristics of Systems

System	Order	Chaos	Randomness
Paradigmatic Example	Clocks, Planets	Clouds, Weather	Snow on TV Screen
Predictability	Very High	Finite, Short Term	None, Simple Laws
Effect of Small Errors	Very Small	Explosive	Nothing BUT Errors
Spectrum	Pure	Yes!	Noisy, Broad
Dimension	Finite	Low	Infinite
Control	Easy	Tricky, Very Effective	Poor
Attractor	Point, Cycle, Torus	Strange, Fractal	No!

5

Fractals

Fractals are geometric shapes that are very complex and infinitely detailed. You can zoom in on a section and it will have just as much detail as the whole fractal. They are recursively defined and small sections of them are similar to large ones. One way to think of fractals for a function f(x) is to consider x, f(x), f(f(x)), f(f(f(x))), f(f(f(f(x)))), etc. Fractals are related to chaos because they are complex systems that have definite properties.

Game Theory

Game Theory

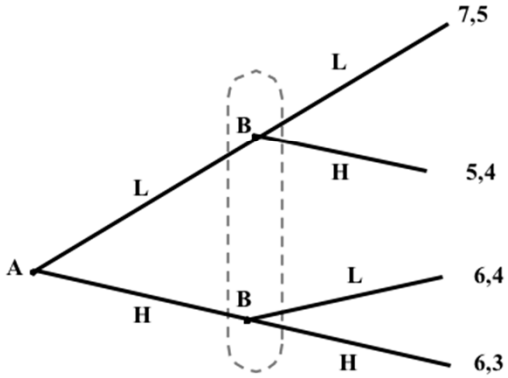
- Any situation in which individuals must make strategic choices and in which the final outcome will depend on what each person chooses to do can be viewed as a game.
- Game theory models seek to portray complex strategic situations in a highly simplified setting.
- All games have three basic elements:
  - Players
  - Strategies
  - Payoffs
- Players can make binding agreements in *cooperative* games, but can not in *noncooperative* games, which are studied in this chapter.

Equilibrium Concepts

- In the theory of markets an equilibrium occurred when all parties to the market had no incentive to change his or her behavior.
- When strategies are chosen, an equilibrium would also provide no incentives for the players to alter their behavior further.
- The most frequently used equilibrium concept is a **Nash equilibrium.**

An Illustrative Advertising Game

- Two firms (A and B) must decide how much to spend on advertising
- Each firm may adopt either a higher (H) budget or a low (L) budget.
- The game is shown in extensive (tree) form.
- A makes the first move by choosing either H or L at the first decision "node."
- Next, B chooses either H or L, but the large oval surrounding B's two decision nodes indicates that B does not know what choice A made.



14

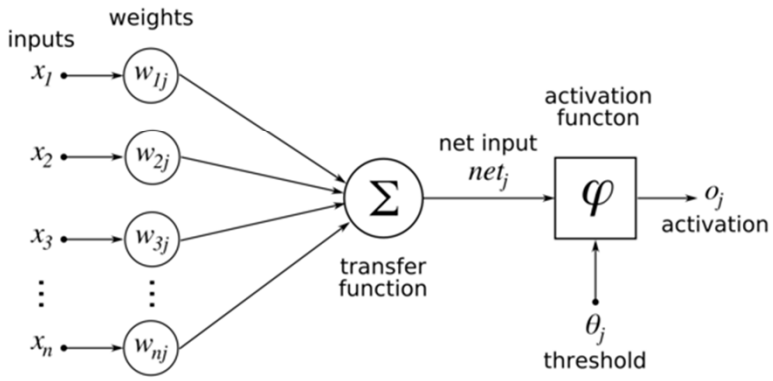
Machine Talk

What techniques we will see

- kNN algorithm
- Winnow algorithm
- Naïve Bayes classifier
- Decision trees
- Reinforcement learning (Rocchio algorithm)
- Genetic algorithm

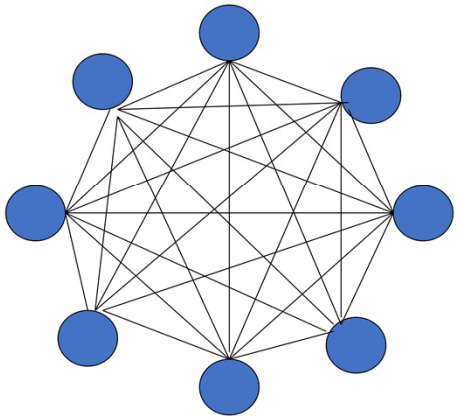
22

Neural Networks - Forward Pass



Hopfield Network

- The Hopfield network consists of a set of N interconnected neurons which update their activation values asynchronously and independently of other neurons.
- All neurons are both input and output neurons. The activation values are binary (+1, -1)



32

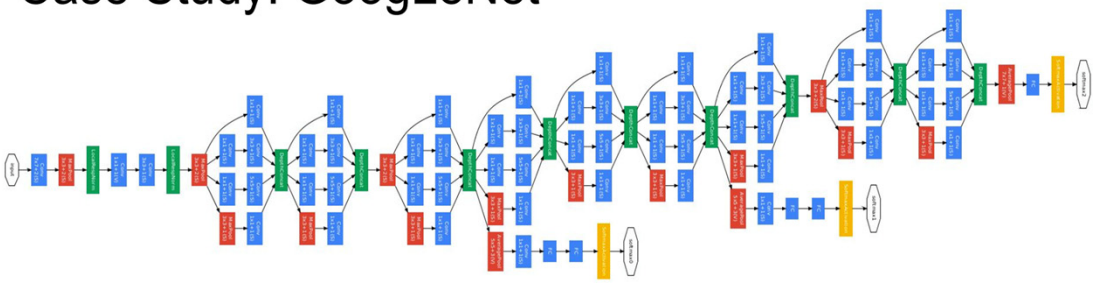
Self Organization

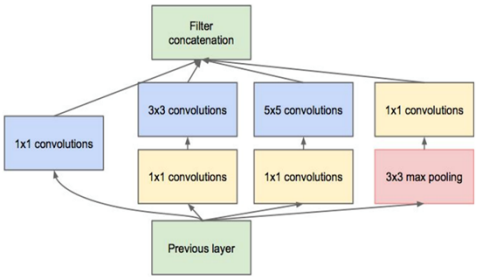
- The unsupervised weight adapting algorithms are usually based on some form of global competition between the neurons.
- Applications of self-organizing networks are:
  - **clustering:** the input data may be grouped in `clusters' and the data processing system has to find these inherent clusters in the input data.
  - **vector quantisation**: this problem occurs when a continuous space has to be discretised. The input of the system is the n-dimensional vector x, the output is a discrete representation of the input space. The system has to find optimal discretisation of the input space.
  - **dimensionality reduction**: the input data are grouped in a subspace which has lower dimensionality than the dimensionality of the data. The system has to learn an “optimal” mapping.
  - **feature extraction:** the system has to extract features from the input signal. This often means a dimensionality reduction as described above.

33

Case Study: GoogLeNet

[Szegedy et al., 2014]





Inception module

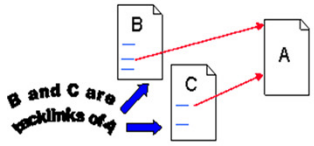
ILSVRC 2014 winner (6.7% top 5 error)

Search Algorithm

Types of Search Algorithm (Google Driven)

- Page-Rank Algorithm
- Penguin Algorithm
- Panda Algorithm
- Humming-Bird Algorithm

Page-Rank Algorithm(contd...)



Every page has some number of forward links(Outedges) and back links(Inedges)

Two cases Page-Rank is interesting:

1. Web pages vary greatly in terms of the number of backlinks they have. For example, the Netscape home page has 62,804 backlinks compared to most pages which have just a few backlinks. Generally, highly linked pages are more “important” than pages with few links.

43

TensorFlow

Google TensorFlow

- Second generation Machine Learning system, followed by DistBelief
- TensorFlow grew out of a project at Google, called Google Brain, aimed at applying various kinds of neural network machine learning to products and services across the company.
- An open source software library for numerical computation using data flow graphs
- Used in following projects at Google
  1. DeepDream
  2. RankBrain
  3. Smart Reply

  And many more..

Sequence Tensors

- Sequences
  - `tf.linspace(start, stop, num, name=None)`
  - `tf.range(start, limit=None, delta=1, name=range)`

Random Tensors

- Random Tensors
  - Examples:
  - `tf.random_normal(shape, mean=0.0, stddev=1.0, dtype=tf.float32, seed=None, name=None)`
  - `tf.truncated_normal(shape, mean=0.0, stddev=1.0, dtype=tf.float32, seed=None, name=None)`
  - `tf.random_uniform(shape, minval=0, maxval=None, dtype=tf.float32, seed=None, name=None)`
  - `tf.random_shuffle(value, seed=None, name=None)`
  - `tf.random_crop(value, size, seed=None, name=None)`
  - `tf.set_random_seed(seed)`

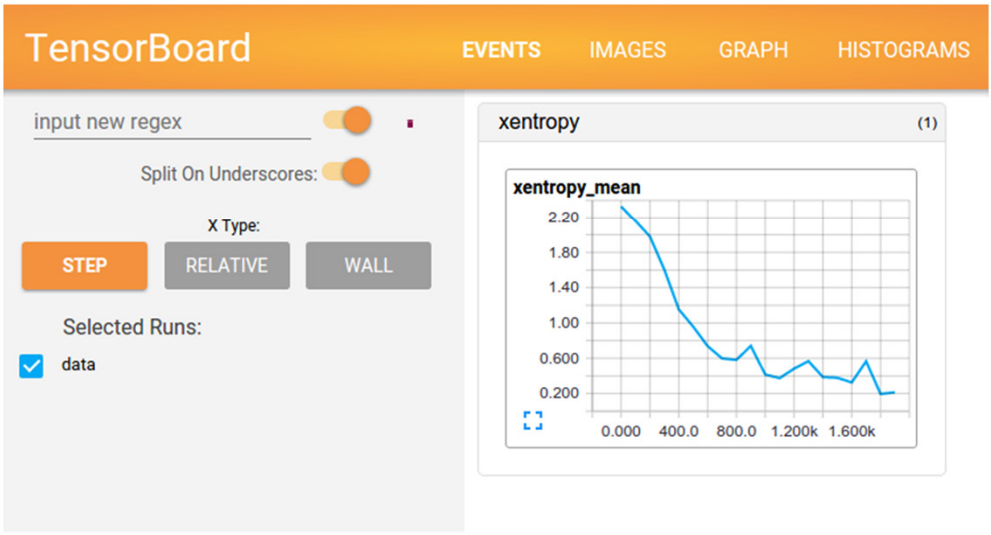
60

Placeholder

- A value that we'll input when we ask TensorFlow to run a computation.

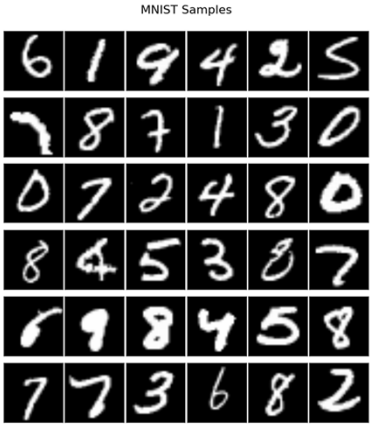
```
x = tf.placeholder(tf.float32, shape=[None, 784])
y_ = tf.placeholder(tf.float32, shape=[None, 10])
```

TensorBoard : Visual Learning

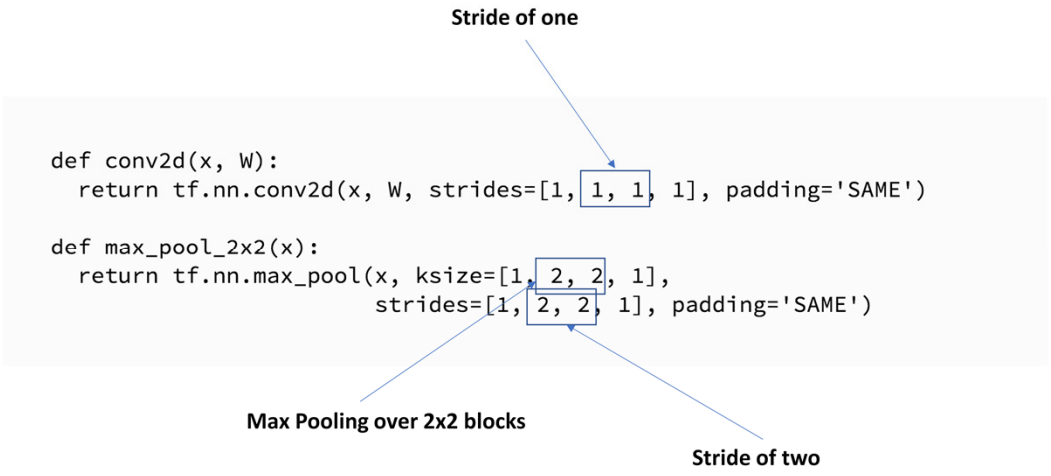


65

MNIST Dataset



Convolution and Pooling



69

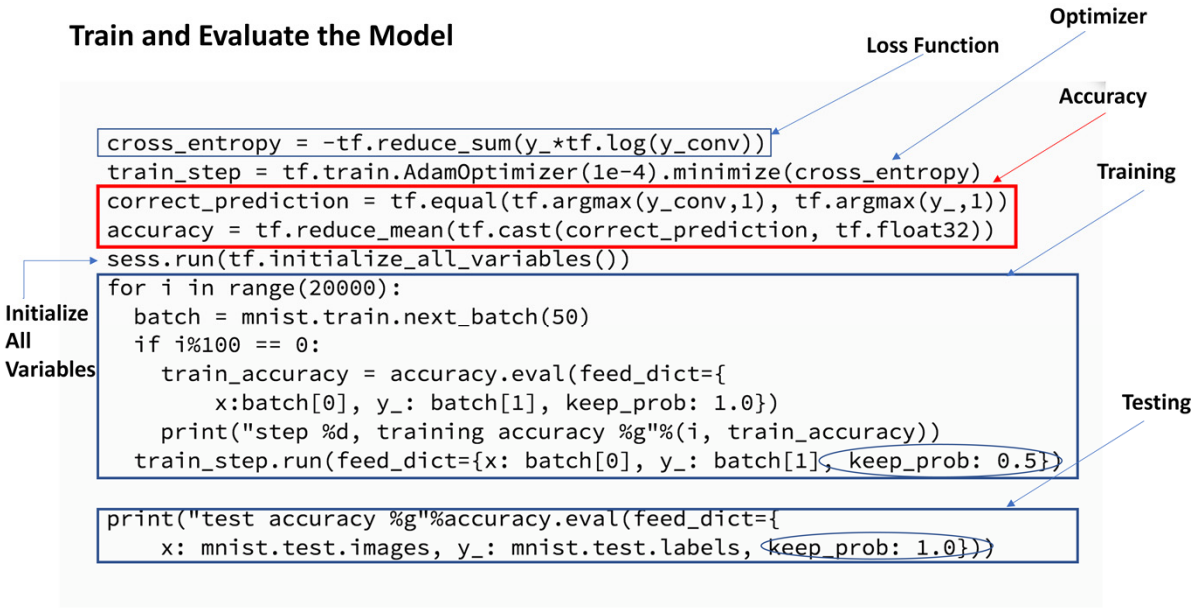
Readout Layer

- Finally, we add a softmax layer, just like for the one layer softmax regression.

```
W_fc2 = weight_variable([1024, 10])
b_fc2 = bias_variable([10])

y_conv=tf.nn.softmax(tf.matmul(h_fc1_drop, W_fc2) + b_fc2)
```

75

Train and Evaluate the Model



76

TensorBoard

- TensorBoard operates by reading TensorFlow events files, which contain summary data that you can generate when running TensorFlow.
- First, create the TensorFlow graph that we'd like to collect summary data from, and decide which nodes should be annotated with summary operation.
  - For example,
    - For MNIST digits CNNs, we'd like to record how the learning rate varies over time, and how the objective function is changing
    - We'd like to record distribution of gradients or weights

77

TensorBoard

Graph Representation

```
# Create the model
x = tf.placeholder(tf.float32, [None, 784], name="x-input")
W = tf.Variable(tf.zeros([784,10]), name="weights")
b = tf.Variable(tf.zeros([10], name="bias"))

# use a name scope to organize nodes in the graph visualizer
with tf.name_scope("Wx_b") as scope:
  y = tf.nn.softmax(tf.matmul(x,W) + b)

# Add summary ops to collect data
w_hist = tf.histogram_summary("weights", W)
b_hist = tf.histogram_summary("biases", b)
y_hist = tf.histogram_summary("y", y)
```

Graph Representation

Histogram Summary

78

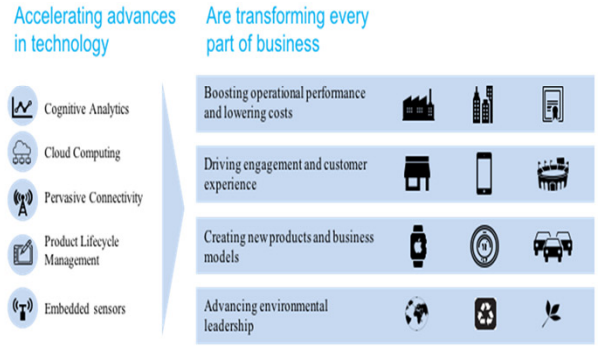
TensorBoard

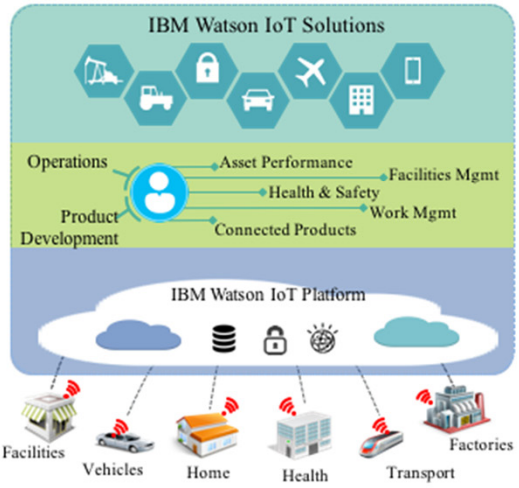
```
# Define loss and optimizer
y_ = tf.placeholder(tf.float32, [None,10], name="y-input")
# More name scopes will clean up the graph representation
with tf.name_scope("xent") as scope:
  cross_entropy = -tf.reduce_sum(y_*tf.log(y))
  ce_summ = tf.scalar_summary("cross entropy", cross_entropy)
with tf.name_scope("train") as scope:
  train_step = tf.train.GradientDescentOptimizer(0.01).minimize(cross_entropy)

with tf.name_scope("test") as scope:
  correct_prediction = tf.equal(tf.argmax(y,1), tf.argmax(y_,1))
  accuracy = tf.reduce_mean(tf.cast(correct_prediction, tf.float32))
  accuracy_summary = tf.scalar_summary("accuracy", accuracy)

# Merge all the summaries and write them out to /tmp/mnist_logs
merged = tf.merge_all_summaries()
writer = tf.train.SummaryWriter("/tmp/mnist_logs", sess.graph_def)
tf.initialize_all_variables().run()
```

79

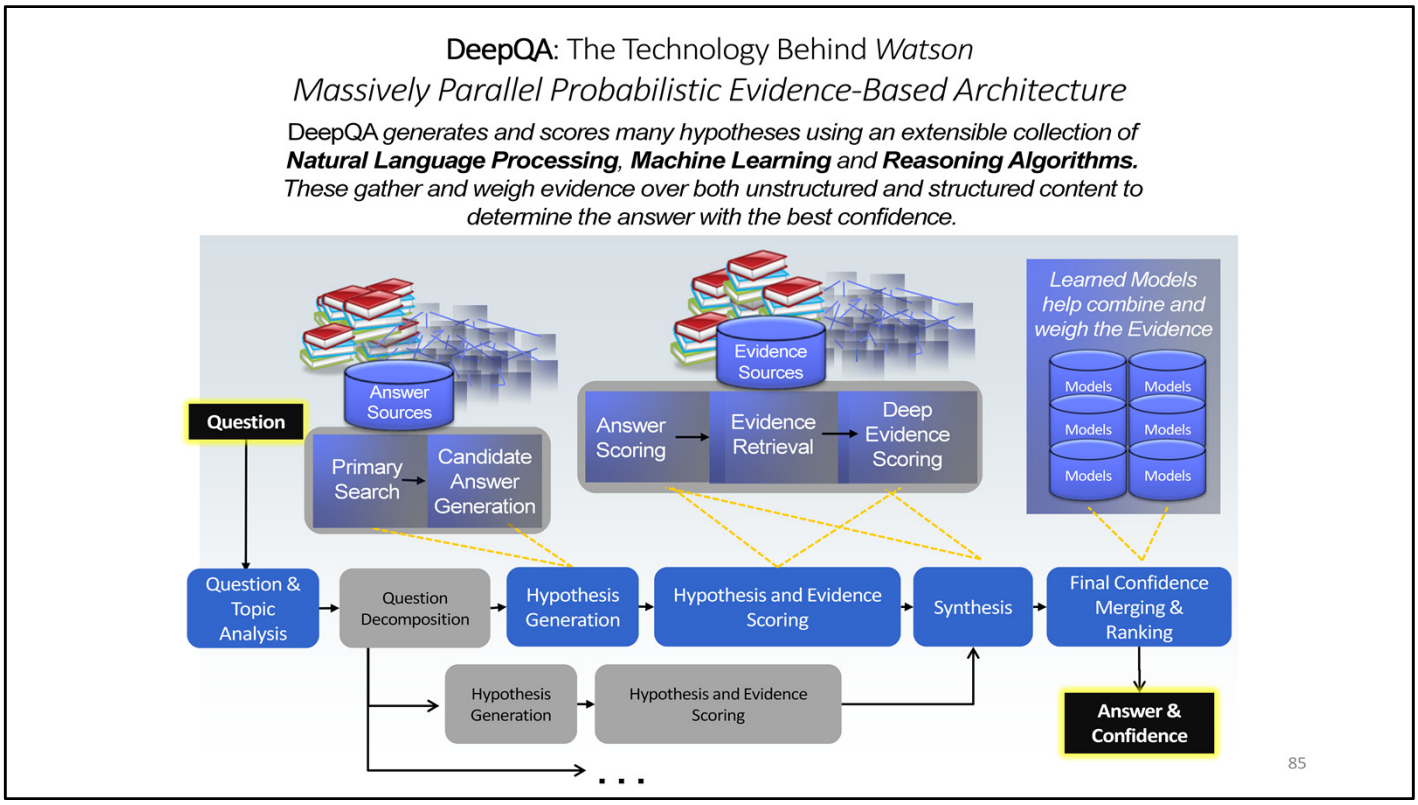
IBM Watson

IoT is driving digital disruption of the physical world





84



Watson – the computer system we developed to play Jeopardy! is based on the DeepQA softate archtiecture. Here is a look at the **DeepQA** architecture. This is like looking inside the brain of the Watson system from about 30,000 feet high.

Remember, the intended meaning of natural language is ambiguous, tacit and highly contextual. The computer needs to consider many possible meanings, attempting to find the evidence and inference paths that are most confidently supported by the data.

So, the primary computational principle supported by the DeepQA architecture is to assume and pursue multiple interpretations of the question, to generate many plausible answers or **hypotheses** and to collect and evaluate many different competing evidence paths that might support or refute those hypotheses.

Each component in the system adds assumptions about what the question might means or what the content means or what the answer might be or why it might be correct.

DeepQA is implemented as an extensible architecture and was designed at the outset to support interoperability.

<UIMA Mention>

For this reason it was implemented using UIMA, a framework and OASIS standard for interoperable text and multi-modal analysis contributed by IBM to the open-source community.

Over 100 different algorithms, implemented as UIMA components, were integrated into this architecture to build *Watson*.

Microsoft

- Computational Neuroscience
- Hebbian learning
- Hopfield Net
- Bolzmann machines
- Memory models
- Bio-inspired AI
- RNN
- Computer vision NLP
- IR

88

ML and Hadoop: The State of the World

Tools for Data Preparation/Feature Engineering

- Languages/Environments
  - PigLatin
  - HiveQL
  - Need to deal with mismatch between offline/online feature generation
- Java/Scala APIs
  - Crunch (Cloudera)
  - Scoobi (NICTA)
  - Cascading (Concurrent)
  - Jaql (IBM)

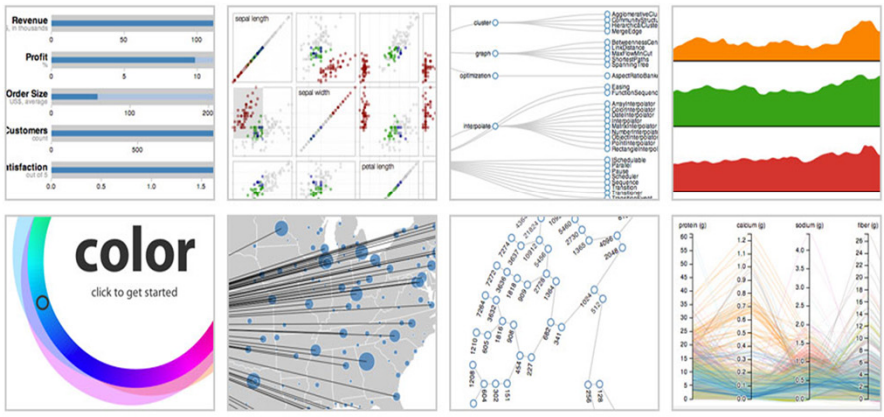
92

Apache Mahout

- The starting place for MapReduce-based machine learning algorithms
  - Not machine-learning-in-a-box
  - Custom tweaks/modifications are the rule
- A disparate collection of algorithms for:
  - Recommendations
  - Clustering
  - Classification
  - Frequent Itemset Mining

Visualization

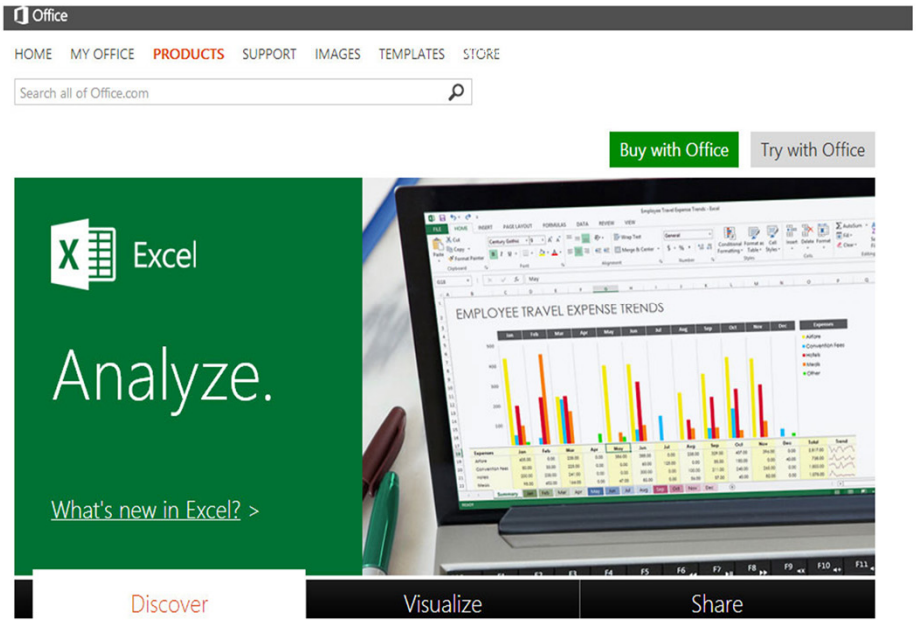
Data Driven Documents



D3.js is a JavaScript library for manipulating documents based on data. **D3** helps you bring data to life using HTML, SVG and CSS. D3's emphasis on web standards gives you the full capabilities of modern

See more examples.

100


Office
HOME
MY OFFICE
PRODUCTS
SUPPORT
IMAGES
TEMPLATES
STORE
Search all of Office.com
Buy with Office
Try with Office
Excel
Analyze.
What's new in Excel? >
EMPLOYEE TRAVEL EXPENSE TRENDS
Discover
Visualize
Share

101

Apache Tajo Storage and Data Format Support

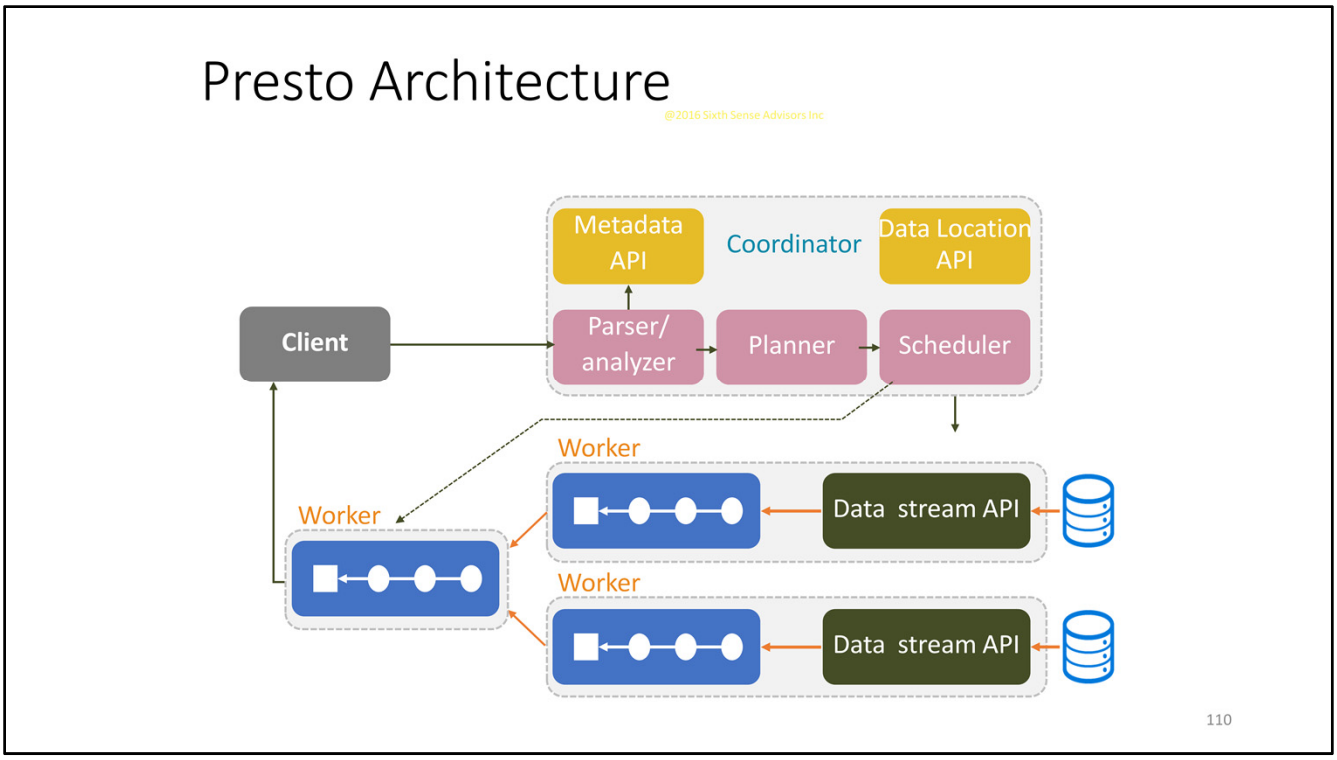
106

Presto

- 100% open source SQL on Hadoop query engine
- Presto offers a modern code base, proven scalability, interactive querying, and cross platform query capability
- Licensed by Apache
- Only major vendor support, Teradata
- Used by a community of well known, well respected technolo[illegible]nies

108

Interactive performance of execution engine
- Code generation for operators (similarly to Impala)
- Data is pipelined MPP-style
- Runs at Facebook scale
- *Capable of querying other non-HDFS data stores as well*



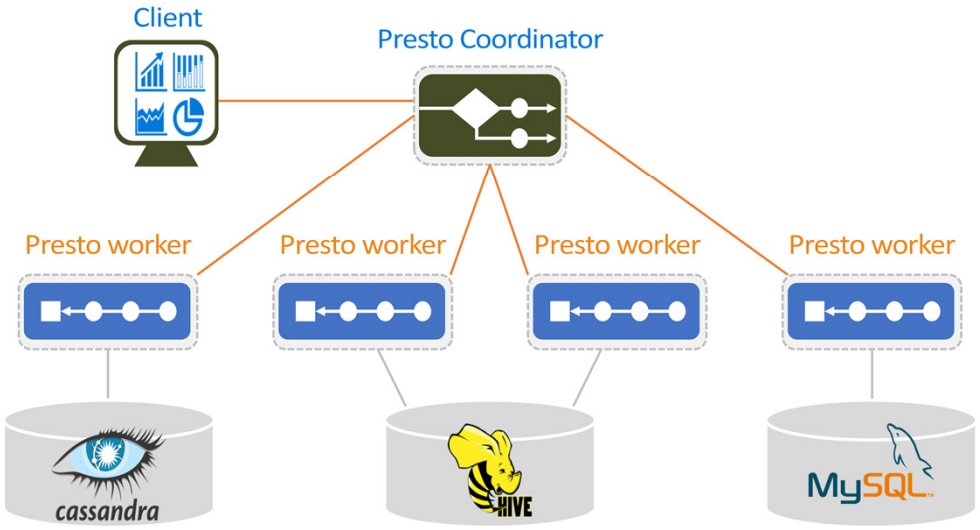
Notes from https://www.facebook.com/notes/facebook-engineering/presto-interacting-with-petabytes-of-data-at-facebook/10151786197628920

The diagram below shows the simplified system architecture of Presto. The client sends SQL to the Presto coordinator. The coordinator parses, analyzes, and plans the query execution. The scheduler wires together the execution pipeline, assigns work to nodes closest to the data, and monitors progress. The client pulls data from output stage, which in turn pulls data from underlying stages.

The execution model of Presto is fundamentally different from Hive/MapReduce. Hive translates queries into multiple stages of MapReduce tasks that execute one after another. Each task reads inputs from disk and writes intermediate output back to disk. In contrast, the Presto engine does not use MapReduce. It employs a custom query and execution engine with operators designed to support SQL semantics. In addition to improved scheduling, all processing is in memory and pipelined across the network between stages. This avoids unnecessary I/O and associated latency overhead. The pipelined execution model runs multiple stages at once, and streams data from one stage to the next as it becomes available. This significantly reduces end-to-end latency for many types of queries.

The Presto system is implemented in Java because it's fast to develop, has a great ecosystem, and is easy to integrate with the rest of the data infrastructure components at Facebook that are primarily built in Java. Presto dynamically compiles certain portions of the query plan down to byte code which lets the JVM optimize and generate native machine code. Through careful use of memory and data structures, Presto avoids typical issues of Java code related to memory allocation and garbage collection.

Presto Connectors



111

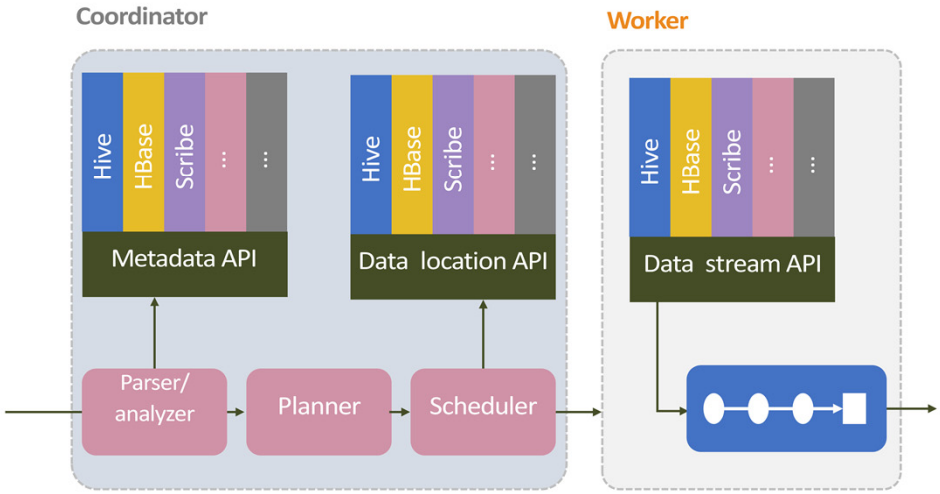
The client sends SQL to the Presto coordinator node. The coordinator in this case determines there are operations needed from more than just Hive data.

•A coordinator (a master daemon) uses connectors to get metadata (such as table schema) that is needed to build a query plan. Workers use connectors to get actual data that will be processed by them.

Presto supports pluggable connectors that provide metadata and data for queries. The Hive connector supports Text, SequenceFile, RCFile, ORC and Parquet (?) formats.

Presto does NOT access the Hive server or Hive. It accesses Hive tables in HDFS. The graphic is kept simple but the details are not so simple.

Presto Extensibility – plug-ins



112

Hive tables and HCatalog

Apache Cassandra

Apache Kafka

Kafka topics = Presto tables,

messages = rows

MySQL

Single node access only -- no sharding

Postgres

Single node access only