Cognitive Computing and Big Data Analytics
By Judith Hurwitz, Marcia Kaufman and Adrian Bowles

CHAPTER
4

The Relationship Between Big Data and Cognitive Computing

A cognitive computing environment requires sufficient amount of data to discover patterns or anomalies within that data. In many situations a large data set is required. Within a cognitive system it is important to have enough data that the results of analytics are trustworthy and consistent. A cognitive system requires the ingestion and mapping of data so that the system can begin to discover where there are connections between data sources to begin discovering insights. To accomplish the goal of finding insights in data, a cognitive system includes both structured and unstructured data. Structured data, such as data in a relational database, is created for processing by a computer. In contrast, unstructured data in the form of written material, video, and images, is designed for human consumption and interpretation. This chapter explains the role that big data plays in creating cognitive computing systems.

Dealing with Human-Generated Data

There is nothing new about dealing with large data sets. In normal form database records, the content and structure are intended to minimize redundancy and to preconfigure the relationships between fields. Therefore, a relational database is optimized for the way *systems* interact and interpret data. Originally, data within a cognitive system was intended for *humans* to process. Such data includes everything from journal articles and other documents to videos, audio,

and images, to disparate streams of sensor and machine data. This type of data requires a level of processing beyond the capabilities of relational database systems because the goal is to interpret the meaning and create human readable data.

However, until the last few years, it has been both technically and financially difficult to manage terabytes, let alone petabytes of data. In the past, the best that most organizations could do was to capture samples of data and hope that the right data was sampled. However, there were limitations to how much analysis could be done when major data elements might be missing. In addition, the scope of the data needed to gain a deep understanding of business and technical issues has expanded dramatically. Companies want to look into the future and want to predict what will happen next, and they then want to understand the best actions to take. Without big data techniques, cognitive computing would not be nearly as useful.

Defining Big Data

Big data requires the capability to manage huge amounts of structured and unstructured data at the appropriate speed and within the right time frame to allow insightful analysis. Big data typically consists of data from a variety of related and unrelated sources that can be quite complex. This can result in huge datasets that may be difficult to manage and analyze. The architectural underpinnings of big data environments have to be designed in a highly distributed manner so that data can be managed and processed quickly and efficiently. This requires highly abstracted and open application programming interfaces (APIs) that provide the capability for a variety of data sources to be ingested, integrated, and evaluated. Big data solutions require a sophisticated infrastructure including security, physical infrastructure, and analytics tools.

Volume, Variety, Velocity, and Veracity

Before delving into the nuances of big data, you need to understand the four foundational characteristics that define the scope and dimension of the issue:

- *Volume* is the characteristic of big data that gets the most attention. Simply put, volume is the quantity of information that needs to be stored and managed. However, volume can vary significantly. For example, the amount of data generated from a Point of Sale (PoS) system is extremely large. However, the data itself is not complex. In contrast, a single medical image has a huge volume of very complex data. This data is semi-structured since its information consists of images that are well defined but do not have the structure of a database.
- *Variety* of data is instrumental in cognitive computing. As mentioned in the introduction to this chapter, data can be structured (traditional database),

unstructured (text), or semi-structured (images and sensor data). The variety of data can range from images to sensor data to text files.

- *Velocity* is the speed of data transmission, processing, and delivery. In some situations, a data source needs to be ingested in periodic batches so that it can be analyzed in context with other data elements. In other situations, data needs to be moved in real time with little or no delay. For example, data from sensors may need to move in real time to react to and repair an anomaly.
- *Veracity* is the requirement that data be accurate. Often if an unstructured source such as social media data is ingested, it will include many inaccuracies and confusing language. However, after an initial data analysis is complete, it will be important to analyze content to make sure that the data that is being used is meaningful.

The Architectural Foundation for Big Data

Because big data is one of the key foundations of a cognitive system, it is imperative to understand the components of a big data technology stack, as illustrated in Figure 4-1. Without a well-designed set of services, the cognitive environment cannot meet enterprise scaling, security, and compliance requirements. Many of the early cognitive systems designs focus on critical areas such as healthcare that require both scale and security to be viable.

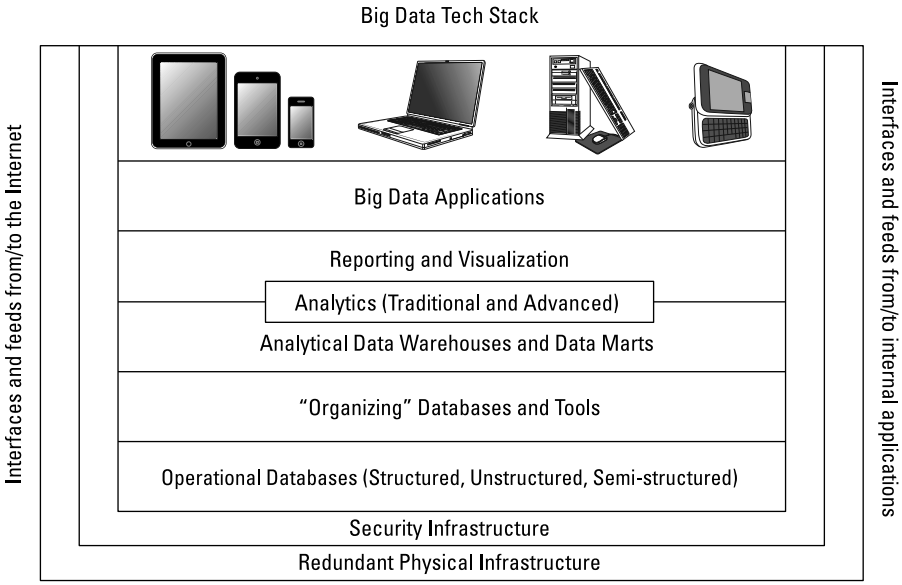

Figure 4-1: Big data technology stack

The Physical Foundation for Big Data

Although most of the discussion in this book focuses on the software enablers for cognitive computing, you need to understand that both big data and cognitive computing require a strong foundation of systems that perform without excessive latency. This physical infrastructure incorporates networks, hardware, and cloud services. Increasingly, to gain the performance required to support big data in the context of cognitive computing, big data infrastructure and hardware needs to be brought together. The underlying physical environment of a big data infrastructure needs to be parallelized and redundant (the system is designed to continue working if one component fails). Networking has to be designed so that it can move data quickly. Likewise, storage has to be implemented and configured so that it can move or connect to information at the right speed. Due to the relatively unlimited scale, capacity, and security of private and public clouds, it is likely that they will become a primary delivery and deployment model for data services.

Security Infrastructure

Security has to be built in to cognitive applications, but that is not enough given the expansive nature of big data and the speed that data sources have to be deployed in many situations. There are many situations, especially when data comes from real-time devices such as sensors and medical devices, when additional security is required. Therefore, the security infrastructure must ensure security when data is in motion and when data is distributed. Often data is culled from a variety of sources and used for different purposes than originally intended. For example, patient information moved to a big data application might not have the right level of protection for private patient data. Therefore, the security infrastructure needs to include the capability to anonymize data so that data such as Social Security numbers and other personal data are hidden. Techniques such as tokenization can be deployed so that unauthorized users cannot access sensitive data.

Operational Databases

What makes big data complicated is the requirement to use and integrate many different types of databases and data structures. This is also fundamental to creating a cognitive computing system. Although a lot of the data that will be important to a cognitive system will be unstructured, these data may also need to be stored and managed in structured SQL databases. For example, there might be a travel planning application that requires structured data about hotel room availability hosted in a SQL database; likewise in a healthcare application, there

might be the need to gain access to databases of drugs implemented in SQL. Therefore, you need to understand the role of both structured and unstructured data as they come together in big data environments.

Role of Structured and Unstructured Data

Structured data refers to data that has a defined length and format and whose semantics are explicitly defined in metadata, schemas, and glossaries. Much of structured data is stored in traditional relational databases and data warehouses. In addition, even more structured data is machine-generated from devices such as sensors, smart meters, medical devices, and Global Positioning Systems (GPS). These data sources are instrumental in creating cognitive systems.

Unlike structured data, unstructured or semi-structured data does not follow a specified format, and the semantics of these data types are not explicitly defined. Rather, the semantics must be discovered and extracted through techniques such as natural language processing, text analytics, and machine learning. The need to find ways to collect, store, manage, and analyze unstructured data has become increasingly urgent. As much as 80 percent of all data is unstructured, with the amount of unstructured data growing at a rapid pace. These unstructured data sources include data from documents, journal articles and books, clinical trials, customer support systems, satellite images, scientific data (seismic imagery, atmospheric data, and high-energy physics), radar or sonar data, mobile data, website content, and social media sites. All these types of sources are important elements of a cognitive system because they may provide context for understanding a specific issue.

Unlike most relational databases, unstructured or semi-structured data sources are typically not transactional in nature. Unstructured data follows a variety of structures and may be large. These unstructured data are typically used with nonrelational databases such as NoSQL databases and include the following structures:

- *Key-Value Pair (KVP)* databases rely on an abstraction that provides a combination of an identifier or pointer (Key) and an associated data set (Value). KVP are used in lookup tables, hash tables, and configuration files. It is often used with semi-structured data from XML documents and EDI systems. A commonly used KVP database is an open source database called Riak that is used in high-performance situations such as media-rich data sources and mobile applications.
- *Document databases* provide a technique for managing repositories of unstructured and semi-structured data such as text documents, web pages, complete books, and such. These databases are important in cognitive

systems because they effectively manage unstructured data either as static entities or as components that can be dynamically assembled. The JSON data-exchange format supports the ability to manage these types of databases. There are a number of important document databases including MongoDB, CouchDB, Cloudant, Cassandra, and MarkLogic.

- *Columnar databases* are an efficient database structure that stores data in columns rather than rows. This enables a more efficient technique for writing to and reading data from hard disk storage. The objective is to improve the speed of returning the results of a query. Therefore, it is useful when there is a huge amount of data that needs to be analyzed for query purposes. HBase is one of the most popular columnar databases. Based on Google's BigTable (a scalable storage system that supports sparse data sets), it is particularly useful in cognitive computing use cases because it scales easily and is designed to work with sparse and highly distributed data. This data structure is appropriate for high volumes of data that are updated frequently.
- *Graph databases* make use of graph structures with nodes and edges to manage and represent data. Unlike a relational database, a graph database does not rely on joins to connect data sources. Rather, graph databases maintain a single structure—the graph. The elements of the graph directly refer to each other so that they track relationships even in sparse data sets. Graph databases are used frequently when dependencies between elements need to be maintained in a dynamic fashion. Common applications include biological model interaction, language relationships, and network connectivity. Therefore, they are well suited for cognitive applications. Neo4J is a commonly used open source graph database.
- *Spatial databases* are those that are optimized to store and query geometric objects that can include points, lines, and polygons. Spatial data is used in Global Positioning Systems (GPS) to manage, monitor, and track positions and locations. It is useful in cognitive systems that often incorporate GPS data into a solution such as robotics or applications that require an understanding of the impact of weather on a situation. The amount of data required in these types of applications is huge.
- *PostGIS/OpenGEO* is a relational database that includes a specialized layer to support spatial applications such as 3-D modeling and gathering and analyzing data from sensor networks.
- *Polyglot Persistence* is a specialized case that brings together different database models for specialized situations. This model is especially important for organizations that need to leverage traditional lines of business applications and databases with text and image data sources.

Table 4-1 provides a comparison of the characteristics of SQL and NoSQL databases.

Table 4-1: Important Characteristics of SQL and NoSQL Databases

ENGINE	QUERY LANGUAGE	MAPREDUCE	DATA TYPES	TRANSACTIONS	EXAMPLES
Key-value	Lucene, Commands	JavaScript	BLOB, semityped	No	Riak, Redis
Document	Commands	JavaScript	Typed	No	MongoDB, CouchDB
Columnar	Ruby	Hadoop	Predefined and typed	Yes, if enabled	HBase
Graph	Walking, Search, Cypher	No	Untyped	ACID	Neo4J
Relational	SQL, Python, C	No	Typed	ACID	PostgreSQL, Oracle, DB2

Data Services and Tools

The underlying data services are critical to operationalizing big data. The supporting tool sets are intended to gather and assemble data so that the data can be processed in the most efficient way. There are a set of services that are needed to support integration, data translation, and normalization as well as scaling. These services include the following:

- A distributed filesystem that is needed to manage the decomposition of structured and unstructured data streams. A distributed filesystem is often a requirement for doing complex data analytics when data comes from a variety of sources.
- Serialized services are required to support persistent data storage as well as supporting remote procedure calls.
- Coordination services are essential for building an application that leverages highly distributed data.
- Extract, transform, and load (ETL) services are required to both load and convert structured and unstructured data to support Hadoop (a key technique for organizing big data).
- Workflow services are the technique for synchronizing processing elements across a big data environment.

Analytical Data Warehouses

Although a considerable amount of big data begins as unstructured sources, there is also a significant amount of information derived from transactional systems and corporate applications built on relational databases. These structured

data sources emanate from systems of record including accounting, customer resource management systems, and other industry-specific applications. The data is usually stored in analytical data warehouses or data marts that are typically a subset of the larger corporate relational database systems. They are useful in creating context when combined with largely unstructured data sources.

Big Data Analytics

Although business intelligence tools have been around for decades, they typically do not provide the type of sophisticated algorithms needed for big data analytics. Chapter 6, "Applying Advanced Analytics to Cognitive Computing," provides an in-depth perspective on advanced analytics. This chapter provides an overview of how analytics helps businesses to improve business knowledge, anticipate change, and predict outcomes. You see how companies are experiencing a progression in analytics maturity levels ranging from descriptive analytics to predictive analytics to machine learning and cognitive computing. One of the foundational principles of a cognitive computing environment is that there will be a variety of data types that need to be brought together to get a full understanding of the field being analyzed. For example, in medical diagnosis it is helpful to understand the environment of the patient (that is, is he a smoker or overweight?) in addition to analyzing test results. In addition, the diagnostician must compare this one case against new research and outcomes for patients with similar diagnoses and treatment plans. There are additional instances in which there is so much data that it is imperative to use visualization techniques.

In general, advanced analytics in a big data and cognitive computing environment requires the use of sophisticated algorithms because in most cases, there is too much data and too much complex analysis required to use a simple query. As discussed earlier in this chapter, big data typically is too massive to fit into a single machine or into the main memory of a single system. Despite this physical limitation, in order to be effective, the implementation of the algorithm must have the right speed or velocity. Fortunately, there are a number of available and emerging algorithms that support big data analytics including:

- **Sketching and streaming**: These algorithms are used when analyzing streaming data from sensors. Data elements are small but must be moved at a fast speed and require frequent updating.
- **Dimensionality reduction**: These algorithms help to convert data that is highly dimensional into much simpler data. This type of reduction is necessary so it will be easier to solve machine learning problems for classification and regression tasks.
- **Numerical linear algebra**: These algorithms are used when data includes large matrices. For example, retailers use numerical linear algebra to identify customer preferences for a large variety of products and services.

- **Compressed sensing**: These algorithms are useful when the data is sparse or signal data from a streaming sensor are limited to a few linear or time-based measurements. These algorithms enable the system to identify the key elements present in this limited data.

In some situations the large volumes of data included in the analytics process may overwhelm the memory capacity of single machine. Due to the nature of distributed systems it is often necessary to decompose problems and process them in separate physical machines. Techniques like Non-Uniform Memory Access (NUMA) help to overcome these limitations by minimizing thrashing and I/O overhead. NUMA allows for discontinuous pools of memory to be treated as one pool of memory. For example, this technique would allow for an algorithm running on one machine to use memory from another computing device. This additional memory would be treated as an extension of the memory on the first device.

Hadoop

Hadoop has emerged as one of the most important technologies for managing large amounts of unstructured data in an efficient manner because it uses distributed computing techniques. Hadoop enables you to use parallelization techniques to improve efficiency. It is an open-source community, codebase, and market for a big data environment that is designed, among other things, to parallel-execute code written to MapReduce. Text documents, ontologies, social media data, sensor data, and other forms of nontraditional data types can be efficiently managed in Hadoop. As a result, this technology is critical to the development of corpora for cognitive computing systems. The benefit of using Hadoop is that you can quickly transform massive amounts of nontraditional data from raw data to structured data so that you can search for patterns in that data.

Hadoop is particularly useful for managing big data in cognitive computing because it is easy to dynamically scale and make changes quickly. Hadoop provides a way to efficiently handle the problem of taking highly unstructured data and break it up into component parts to then solve the problem and produce results. Hadoop can be implemented on racks of commodity servers or included in a pre-optimized appliance on vendor-specific hardware. Two key components to Hadoop are described here:

- **Hadoop Distributed File System (HDFS):** A data storage cluster that is both highly reliable and low cost used to make it easy to manage related files across different machines.
- **MapReduce engine:** Provides a way to distribute the processing of the analytics algorithms across a large number of systems. After the distributed

computation is complete, all the elements are aggregated back together to provide a result.

Why is HDFS useful in big data and cognitive computing environments? HDFS provides a data service that is particularly well-suited to support large data volumes and high-velocity data. One capability of HDFS that speeds up the processing of large data volumes is based on the data that is written. In HDFS, data is written once and then read many times thereafter, rather than the constant read-writes of other filesystems. HDFS works by breaking large files into smaller pieces called *blocks*. Figure 4-2 illustrates an example of a Hadoop cluster.

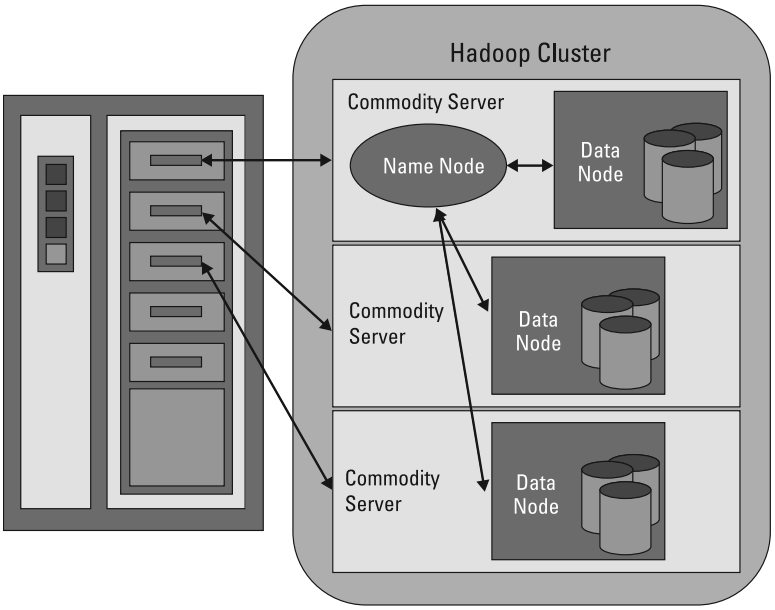

Figure 4-2: Example of a Hadoop cluster

These architectural elements are described here:

- **NameNodes:** The role of the NameNode is to keep track of where data is physically stored in the cluster. To maintain this knowledge, the NameNode needs to understand which blocks on which data nodes make up the complete file. The NameNode manages all access to the files, including reads, writes, creates, deletes, and replication of data blocks on the data nodes. In addition, NameNode has the important responsibility of telling the Data Nodes if there is anything for them to do. Because NameNode

is critical to keep the HDFS working, it should be replicated to protect against a single point of failure.

- **Data Nodes:** Data Nodes act as servers that contain the blocks for a set of files. Of the two components, NameNodes have some intelligence, whereas Data Nodes are more simplistic. However, they are also resilient and have multiple roles to play. They store and retrieve the data blocks in the local filesystem of the server. They also store the metadata of a block in the filesystem. In addition, Data Nodes send reports to the NameNode about what blocks are available for file operations. Blocks are stored on Data Nodes.

Hadoop MapReduce is the heart of the Hadoop system. It provides all the capabilities you need to break big data into manageable chunks, process the data in parallel on your distributed cluster, and then make the data available for user consumption or additional processing. And it does all this work in a highly resilient, fault-tolerant manner. You provide input and the MapReduce Engine converts the input into output quickly and efficiently, providing you with the answers you need. Hadoop MapReduce includes several stages or functions that you can implement to help reach your goal of getting the answers you need from big data. The stages include getting data ready, mapping the data, and reducing and combining the data. Figure 4-3 illustrates how Hadoop MapReduce performs its tasks.

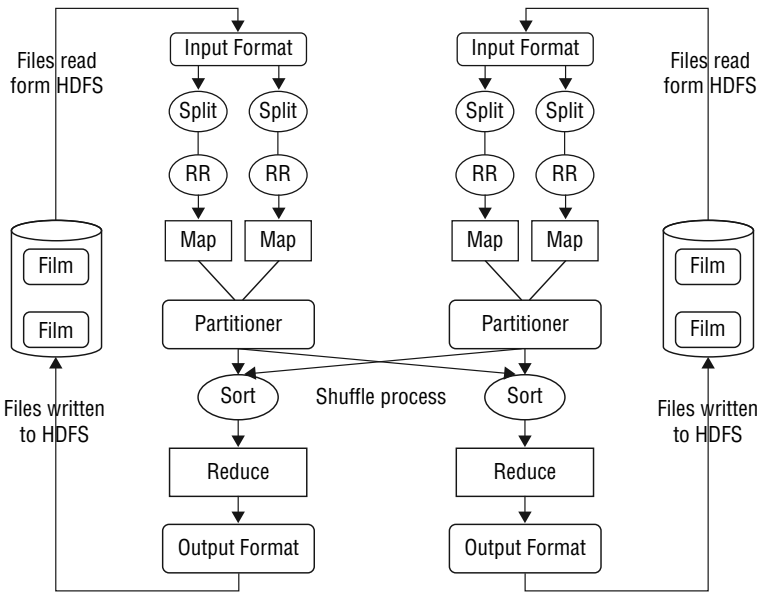

Figure 4-3: Workflow and data movement in a small Hadoop cluster

A large and growing ecosystem has developed around Hadoop. This has been extremely advantageous to companies wanting to implement big data initiatives because the technologies in the ecosystem make Hadoop much easier to use. The key tools in the Hadoop ecosystem are described here:

- **Hadoop Yet Another Resource Negotiator (YARN):** Acts as a distributed operating system for big data applications. YARN manages the resources and helps to provide efficient job scheduling and tracking for Hadoop using two services: The ResourceManager (RM) and the Application Master (AM). The RM acts as the arbitrator responsible for distributing resources among all the applications in the system. Each node in the system has a NodeManager that monitors the application's usage of CPU, disk, network, and memory and reports back to the RM. The AM negotiates with the RM regarding resource allocation and works with NodeManagers to execute and monitor tasks.
- **HBase:** A distributed, nonrelational (columnar) database (discussed earlier in this chapter). This means that all data is stored in tables with rows and columns similar to relational database management systems (RDBMSs). It is modeled after Google BigTable and can host large tables (billions of columns/rows) because it is layered on Hadoop clusters of commodity hardware. HBase provides random, real-time read/write access to big data. HBase is highly configurable, providing a great deal of flexibility to address huge amounts of data efficiently. Although the schema must be defined and created before any data can be stored, tables can be altered after the database is up and running. This factor is helpful in big data environments because you don't always know all the details of your data streams in advance.
- **Hive:** A batch-oriented, data-warehousing layer built on the core elements of Hadoop. Hive provides both SQL-like access to structured data and big data analysis with MapReduce. Hive is different from a typical data warehouse in that it is not designed for quick responses to queries. Therefore, it is not appropriate for real-time analytics as complex queries can take hours to complete. Hive is useful for data mining and deeper analytics that do not demand a rapid response.
- **Avro:** A data serialization system.
- **Cassandra**: A scalable multi-master database with no single points of failure.
- **Chukwa:** A data collection system for managing large distributed systems.
- **Mahout:** A scalable machine learning and data mining library.
- **Pig:** A high-level data-flow language and execution framework for parallel computation.

- **Spark:** A fast and general compute engine for Hadoop data. Spark provides a simple and expressive programming model that supports a wide range of applications, including ETL, machine learning, stream processing, and graph computation.
- **Tez:** A generalized data-flow programming framework, built on Hadoop YARN, which provides a powerful and flexible engine to execute an arbitrary DAG of tasks to process data for both batch and interactive use-cases. Tez is being adopted by Hive, Pig and other frameworks in the Hadoop ecosystem, and also by other commercial software (e.g. ETL tools), to replace Hadoop MapReduce as the underlying execution engine.
- **ZooKeeper:** A high-performance coordination service for distributed applications.

Data in Motion and Streaming Data

Cognitive computing can help businesses to gain value from many types of data that have been hard to interpret and analyze. One of the most important types of data that companies are beginning to work with is data in motion or streaming data. *Streaming data* is a continuous sequence of data that is moving at fast speeds. There are many examples of streaming data ranging from data coming from equipment sensors to medical devices to temperature sensors to stock market financial data and video streams. Streaming data platforms are designed to process this data at high speeds. Speed is of the highest priority when processing streaming data, and it can't be compromised or the results will not be useful. Streaming data is useful when analytics need to be done in real time while the data is in motion. In fact, the value of the analysis (and often the data) decreases with time. For example, if you can't analyze and act immediately, a sales opportunity might be lost or a threat might go undetected.

Many industries are finding ways to gain value from data in motion. In some situations, these companies can take data they already have and begin to use it more effectively. In other situations, they are collecting data that they could not collect before. Sometimes organizations can collect much more of the data that they had been only collecting snapshots of in the past. These organizations use streaming data to improve outcomes for customers, patients, city residents, or perhaps for mankind. Businesses use streaming data to influence customer decision making at the point of sale.

There are some important uses of data streaming today. There will be many more uses as organizations begin to understand the value of leveraging the data created by sensors and actuators. The uses for streaming data include the following:

- In power plant management, there is the need for a highly secure environment so that unauthorized individuals do not interfere with the delivery of

power to customers. Companies often place sensors around the perimeter of a site to detect movement. But not all forms of movement represent a threat. For example, the system needs to be able to detect if an unauthorized person is accessing a secure area versus an animal walking around. Clearly, the innocent rabbit does not pose a security risk. Therefore, the vast amount of data coming from these sensors needs to be analyzed in real time so that an alarm is sounded only when an actual threat exists.

- In manufacturing, it will be important to use the data coming from sensors to monitor the purity of chemicals being mixed in the production process. This is a concrete reason to leverage the streaming data. However, in other situations, it may be possible to capture a lot of data, but no overriding business requirement exists. In other words, just because you can stream data doesn't mean that you always should.
- In medical applications, sensors are connected to highly sensitive medical equipment to monitor performance and alert technicians of any deviations from expected performance. The recorded data is continuously in motion to ensure that technicians receive information about potential faults with enough lead time to make a correction to the equipment and avoid potential harm to patients.
- In the telecommunications industry it is critical to monitor large volumes of communications data to ensure that service levels meet customer expectations.
- In the retail industry, point-of-sale data is analyzed as it is created to try to influence customer decision making. Data is processed and analyzed at the point of engagement and maybe used in combination with location data or social media data.
- Understanding the context of data collected is critical in at-risk physical locations. The system has to be able to detect the context of the incident and determine if there is a problem.
- Medical organizations can analyze complex data from medical devices. The resulting analysis of this streaming data can determine different aspects of a patient's condition and then match results against known conditions or other abnormal indicators.

Analyzing Dark Data

Although the focus has been on the data that is well known and often used by organizations, there is a considerable amount of it that is stored but has never been analyzed or viewed. Called *dark data*, this information is often log data from equipment or security systems. There are often mandates for organizations to store this data. Before the advent of big data approaches such as Hadoop and MapReduce, it was prohibitively expensive to even attempt to analyze the data.

However, there are enormous amounts of valuable data that can help organizations begin to understand patterns that were unknown. For example, machine data stored in these logs may predict when a typical machine will fail based on patterns of temperature, moisture, or other repeated conditions. Having this data available for predictive analysis can help companies know the precise conditions when a machine will fail or when to change a traffic pattern.

Integration of Big Data with Traditional Data

Although much of the attention in big data has been focused on accessing and analyzing complex unstructured data, it is important to understand that the results of analysis of this data has to be integrated with traditional relational databases, data warehouses, and line of business applications. To create a cognitive system requires that an organization have a holistic view of the required data so that the context is correct.

Therefore, building a cognitive system requires that the massive amounts of data be managed and analyzed. It also requires that there are the right data integration tools and techniques in place to effectively create the corpus. This is not a static process. To be effective, all types of big data must be moved, integrated, and managed based on the problem being addressed.

Summary

Big data is at the heart of creating an effective cognitive system. There are a variety of different types of big data including structured and unstructured sources. The data is not all the same. There will be important differences in the volume of information, the types of information, and whether it needs to be moved quickly from one place to another. Organizations will have to ensure that when planning to use big data to create the corpus for a cognitive system that the underlying data is accurate and in the right context.