Issue 9 vol 3 ISSN: 2321-8134

IJFEAT

INTERNATIONAL JOURNAL FOR ENGINEERING APPLICATIONS AND TECHNOLOGY

A REVIEW ON APACHE SPARK-BASED DEEP LEARNING FOR MOBILE BIG DATA ANALYTICS

Divya J. Sodha[1], Akanksha G. Karnewar[2], Priya Kochar[3], Prof.JitendraSaturwar[4]

[1]*Student, Computer Science &Engg. Dept., JDIET, Yavatmal, Maharashtra, India,* **divyasodha10@gmail.com**

[2]*Student, Computer Science &Engg. Dept., JDIET, Yavatmal, Maharashtra, India,* **akankshakarnewar97@gmail.com**

[3]*Student, Computer Science &Engg. Dept., JDIET, Yavatmal, Maharashtra, India,* **pkochar100@gmail.com**

[4]*AssociateProffessor, Computer Science &Engg. Dept., JDIET, Yavatmal, Maharashtra, India,***jsaturwar@gmail.com**

Abstract

The proliferation of mobile devices, such as smartphones and Internet of Things gadgets, has resulted in the recent mobile big data era. Collecting mobile big data is unprofitable unless suitable analytics and learning methods are utilized to extract meaningful information and hidden patterns from data. This paper gives a review on Apache Spark-based Deep Learning for Mobile Big Data Analytics. Specifically, distributed deep learning is executed as an iterative MapReduce computing on many Spark workers. Each Spark worker learns a partial deep model on a partition of the overall mobile, and a master deep model is then built by averaging the parameters of all partial models. This Spark-based work speeds up the learning of deep models consisting of many hidden layers and millions of parameters. The various frameworks for Spark and Deep Learning have also been discussed in the paper and various comparisons have been shown among these.

Index Terms: *Mobile Big Data, Deep Learning, Apache Spark, Frameworks.*

--- *** ---

1. INTRODUCTION

The rapid expansion of broadband mobile networks by Telecom Operators, has introduced a versatile global infrastructure that is bringing unprecedented communication possibilities to the human mankind. Mobile telecoms have traditionally generated vast amounts of spatio-temporal mobile broadband data about their customers (e.g., user id, location, device type, etc.), but this data has been kept internal to the telecoms for a variety of reasons, including economic advantage, privacy and scale. The advent of mobile apps on the other hand, executing on powerful computational devices equipped with multitude of sensors that are capable of generating vast amounts of data (geo-location, audio, video, etc.), has brought data collection and crowdsourcing to the fingertips of virtually any of millions of Mobile App Vendors. [1]

Mobile devices have matured as a reliable and cheap platform for collecting data in pervasive and ubiquitous sensing systems. Specifically, mobile devices are:

• Sold in mass market chains
• Connected to daily human activities
•Supported with embedded communication and sensing modules [6]

According to the latest traffic forecast report by Cisco Systems, Annual global IP traffic will reach 3.3 ZB per year by 2021, or 278 exabytes (EB) per month. In 2016, the annual run rate for global IP traffic was 1.2 ZB per year, or 96 EB per month. [8] Mobile big data (MBD) is a concept that describes a massive amount of mobile data that cannot be processed using a single machine. MBD contains useful information for solving many problems such as fraud detection, marketing and targeted advertising, context-aware computing, and healthcare. Therefore, MBD analytics is currently a high-focus topic aimed at extracting meaningful information and patterns from raw mobile data.

Deep learning is a solid tool in MBD analytics. Specifically, deep learning:

• Provides highly accurate results in MBD analytics
• Avoids the expensive design of handcrafted features
• Utilizes the massive unlabelled mobile data for unsupervised feature extraction [1]

In machine learning community, dealing with Big Data can be supported by Deep Learning according to its ability to extract complex abstractions. It provides high-level data representation from large-scale data especially unlabelled data, which are collected abundantly in Big Data. Big Data analytics problems can be summarized as extracting hidden patterns from massive volumes of data, fast information retrieval, data indexing/tagging, and simplifying discriminative tasks. These problems can be better solved with the aid of Deep Learning. Deep Learning together with Big Data is reflected as the ‘‘big deals and the bases for an American innovation and economic revolution’’. [5]

Due to the curse of dimensionality and size of MBD, learning deep models in MBD analytics is slow and takes anywhere from a few hours to several days when performed on conventional computing systems. Conversely, most mobile systems are not delay-tolerant, and decisions should be made as quickly as possible to attain high user satisfaction.

To cope with the increased demand on scalable and adaptive mobile systems, this paper presents a solution on developing a framework that enables time-efficient MBD analytics using deep models with millions of modelling parameters. The framework can be built over Apache Spark, which provides an open source cluster computing platform. This enables distributed learning using many computing cores on a cluster where continuously accessed data is cached to running

ISSN: 2321-8134

memory, thus speeding up the learning of deep models several-fold. Moreover, the learning time of deep models is decreased as a result of the paralleled Spark-based implementation compared to a single-machine computation. [1]

2. CHARACTERISTICS AND CHALLENGES OF MOBILE BIG DATA

2.1 Characteristics

Big data refers to data sets or streams that strain the ability of commonly used relational DBMSs to capture, manage, and process the data within a tolerable elapsed time. Big data sizes commonly range from a few dozen terabytes to many petabytes in a single database and their underlying data model might be anything from structured (relational or tabular) to semi structured (XML or JSON) or even unstructured (Web text and log files). Mobile big data analytics refers to the discovery of previously unknown meaningful patterns and knowledge from data collected from mobile users. In many applications and use cases, the aim is typically to understand deeply the following aspects:

• who is the user (e.g., demographics and aggregate / summarization queries)?

• what are they likely to do at that location and time (e.g., activity)?

• where are they likely to go next (e.g., prediction of next place / place-of-interest)?

Example mobile big data analytics range from high-level metrics and summaries (e.g., through clustering, classification and association rule mining) useful to executive managers to alert-based analytics (e.g., anomaly detection) useful to frontline engineers and users. A typical constraint associated with such mobile user analytics is that it needs to be near realtime since the focus is typically on delivering location-based and highly personalized information and services to a mobile user. Mobile app vendors (social, location, games, marketing enterprises like flurry, etc.) might nowadays know much more about the customers of a telecom than the telecom itself, as they have access to finer-grain data collected right on the app-level rather than at the network-level. [6]

Figure 1a shows a typical architecture of large-scale mobile systems used to connect various types of portable devices such as smartphones, wearable computers, and Internet of Things (IoT) gadgets. The widespread installation of various types of sensors, such as accelerometers, gyroscopes, compasses, and GPS sensors, in modern mobile devices allows many new applications. Essentially, each mobile device encapsulates its service request and own sensory data in a stateless data-interchange structure like Javascript object notation (JSON) format. The stateless format is important as mobile devices operate on different mobile operating systems (e.g., Android, iOS, and Tizen). Based on the collected MBD, a service server utilizes MBD analytics to discover hidden patterns and information. The importance of MBD analytics stems from its role in building complex mobile systems that could not be assembled and configured on small datasets.

MBD analytics is more versatile than conventional big data problems as data sources are portable and data traffic is crowdsourced. MBD analytics deals with massive amounts of data collected by millions of mobile devices. Next, we discuss the main characteristics of MBD that complicate data analytics and learning on MBD compared to small datasets. [1]

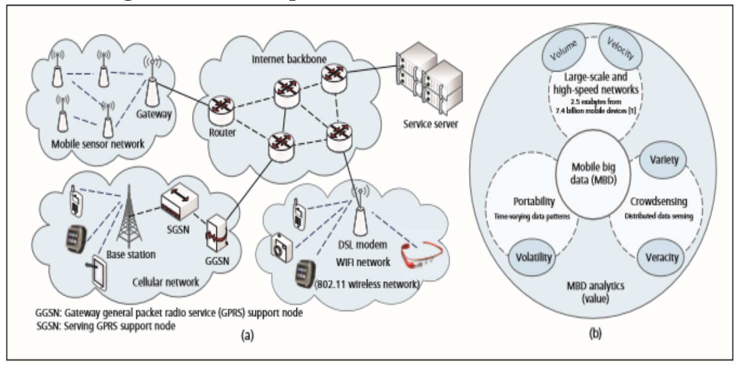

Figure 1: Illustration of the MBD era: a) typical architecture of a modern mobile network connecting smartphones, wearable computers, and IoT gadgets; b) main technological advances behind the MBD era.

2.2 Challenges of MBD Analytics

Figure 1b shows the main recent technologies that have produced the challenging MBD era: large-scale and high-speed mobile networks, portability, and crowdsourcing. Each technology contributes to forming the MBD characteristics in the following ways.

Large-scale and high-speed mobile networks: The growth of mobile devices and high-speed mobile networks (e.g., Wi-Fi and cellular networks) introduces massive and increasingly contentious mobile data traffic. This is reflected in the following MBD aspects:

• *MBD is massive (volume):* In 2015, 3.7 exabytes of mobile data was generated per month, which is expected to increase through the coming years.

• *MBD is generated at increasing rates (velocity):* MBD flows at a high rate, which impacts the latency in serving mobile users. Long queuing time of requests results in less satisfied users and increased cost of late decisions.

Portability: Each mobile device is free to move independently among many locations. Therefore, MBD is non-stationary (volatility). Due to portability, the time duration in which the collected data is valid for decision making can be relatively short. MBD analytics should be frequently executed to cope with the newly collected data samples.

Crowdsourcing: A remarkable trend of mobile applications is crowdsourcing for pervasive sensing, which includes massive data collection from many participating users. Crowd-sensing differs from conventional mobile sensing systems as the sensing devices are not owned by one institution but instead by many individuals from different places. This has introduced the following MBD challenges:

• *MBD quality is not guaranteed (veracity):* This aspect is critical for assessing the quality uncertainty of MBD as mobile systems do not directly manage the sensing process of mobile devices. Since most mobile data is crowd-sourced, MBD can contain low-quality and missing data samples due to noise, malfunctioning or uncalibrated sensors of mobile devices, and even intruders (e.g., badly labelled crowd-sourced data). These low-quality data points affect the analytical accuracy of MBD.

• *MBD is heterogeneous (variety):* The variety of MBD arises because the data traffic comes from many spatially distributed data sources (i.e., mobile devices). Also, MBD comes in different data types due to the many sensors that mobile devices support. For example, a triaxial accelerometer generates proper acceleration measurements, while a light sensor generates illumination values.

MBD analytics (value) is mainly about extracting knowledge and patterns from MBD. In this way, MBD can be utilized to provide better services to mobile users and create revenue for mobile businesses. The next section discusses deep learning as a solid tool in MBD analytics. [1]

3. DEEP LEARNING IN MBD

Deep learning is a new branch of machine learning that can solve a broad set of complex problems in MBD analytics (e.g., classification and regression). [1] DL involves ANNs like Deep Neural Networks (DNNs), Convolution Neural Networks (CNNs), Deep Belief Networks (DBNs) and Stacked AutoEncoder (SAE). Recently DL has gained popularity due to its vast application in computer field. From many research work concluded that in many areas of applications, DL is best method compared to past methods. [2] The objective of Deep Learning is to learn a complex and abstract data representation hierarchically, through passing the data over multiple transformation layers. [5]

A deep learning model consists of simulated neurons and synapses that can be trained to learn hierarchical features from existing MBD samples. The resulting deep model can generalize and process unseen streaming MBD samples. A deep model can be scaled to contain many hidden layers and millions of parameters, which are difficult to train at once. Instead, greedy layer-by-layer learning algorithms [3, 8] have been proposed that basically work as follows.

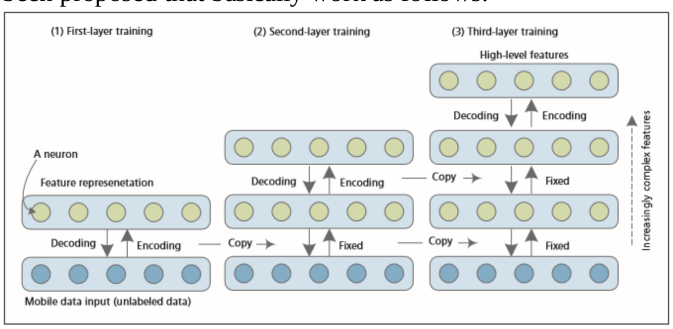

Figure 2: Generative layer-wise training of a deep model. Each layer applies nonlinear transformation to its input vector and produces intrinsic features at its output.

Generative layer-wise pre-training: This stage requires only unlabelled data, which is often abundant and cheap to collect in mobile systems using crowdsourcing. Figure 2 shows the layer-wise tuning of a deep model. First, one layer of neurons is trained using the unlabelled data samples. To learn the input data structure, each layer includes encoding and decoding functions. The encoding function uses the input data and the layer parameters to generate a set of new features. Then the decoding function uses the features and the layer parameters to produce a reconstruction of the input data. As a result, a first set of features is generated at the output of the first layer. Then a second layer of neurons is added on top of the first layer, where the output of the first layer is fed as input of the second layer. This process is repeated by adding more layers until a suitable deep model is formed. Accordingly, more complex features are learned at each layer based on the features that were generated at its lower layer.

Discriminative fine-tuning: The model's parameters, which are initialized in the first step, are then slightly fine-tuned using the available set of labelled data to solve the problem at hand.

4. DEEP LEARNING ADVANTAGES IN MBD

Deep learning provides solid learning models for MBD analytics. This argument can be supported with the following advantages of using deep learning in MBD analytics.

1) Deep learning scores highly accurate results, which are a top priority for growing mobile systems.

2) Deep learning generates intrinsic features that are required in MBD analytics.

3) Deep learning can learn from unlabelled mobile data, which minimizes the data labelling effort.

4) The "variety" aspect of MBD leads to multiple data modalities of multiple sensors (e.g., accelerometer samples, audio, and images). Multimodal deep learning can learn from multiple modalities and heterogeneous input signals.

5.DEEP LEARNING CHALLENGES IN MBD

Collecting MBD is unprofitable unless suitable learning methods and analytics are utilized to extract meaningful information and patterns. Deep learning in MBD analytics is slow and can take a few days of processing time, which does not meet the operation requirements of most modern mobile systems. This is due to the following challenges.

Challenge	Description
Required Learning for non-stationary data	This is the biggest problem in BDA, which deals with fast moving and streaming data. This DA helps in fraud detection. Thus, there is requirement of DL adoption for streaming data handle, as continuous input data is necessary.
High dimensional data	Few DLA is computationally-expensive, while dealing with high-dimensional data like images, due to slow learning process is linked with a DL of layered hierarchy of data representations and abstractions from a lower level layer to a higher level layer. i.e., DLA is stymied, while using for huge data with BDA. A high-dimensional data source favours heavily to the volume of the raw data.
Large scale models	In analytics and computation point of view, with an enormous number of model parameters, that are able in extraction of highly complicated features and representations.

6. SPARK-BASED DEEP LEARNING FOR MBD ANALYTICS

Learning deep models in MBD analytics is slow and computationally demanding due to the large number of parameters of deep models and the large number of MBD samples. Apache Spark is the proposed architecture for the work as shown in Figure 3. Apache Spark is an open source platform for scalable MapReduce computing on clusters. The main goal of the proposed work is speeding up MBD decision making by parallelizing the learning of deep models to a high-performance computing cluster. In short, the parallelization of a deep model is performed by slicing the MBD into many partitions. Each partition is contained in a resilient distributed dataset (RDD) that provides an abstraction for data distribution by the Spark engine. Besides data caching, RDDs of a Spark program also support fault-tolerant executions and recover the program operations at worker nodes.

In short, the Spark-based framework consists of two main components:

• A Spark master

• One or more Spark workers

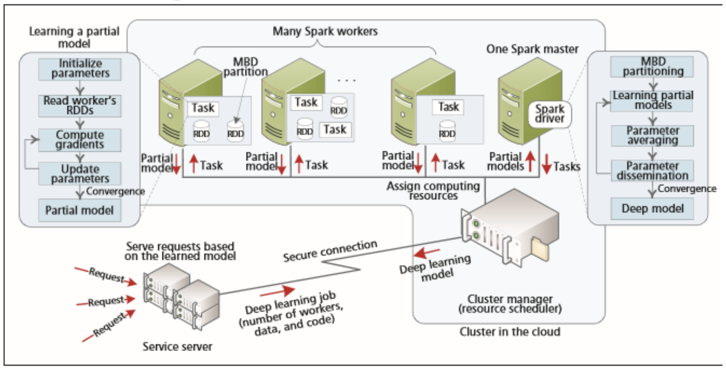

Figure 3: A Spark-based framework for distributed deep learning in MBD analytics.

The master machine initializes an instance of the Spark driver that manages the execution of many partial models in a group of Spark workers. At each iteration of the deep learning algorithm (Fig. 2), each worker node learns a partial deep model on a small partition of the MBD and sends the computed parameters back to the master node. Then, the master node reconstructs a master deep model by averaging the computed partial models of all executor nodes.

Parallelized Learning Collections

Learning deep models can be performed in two main steps:

• Gradient computation

• Parameter update (for the mathematical derivation)

In the first step, the learning algorithm iterates through all data batches independently to compute gradient updates (i.e., the rate of change) of the model's parameters. In the second step, the model's parameters are updated by averaging the computed gradient updates on all data batches. These two steps fit the learning of deep models in the Map-Reduce programming model. In particular, the parallel gradient computation is realized as a Map procedure, while the parameter update step reflects the Reduce procedure. The iterative Map-Reduce computing of deep learning on Apache Spark is performed as follows.

MBD partitioning: The overall MBD is first split into many partitions using the parallelize() application programming interface (API) of Spark. The resulting MBD partitions are stored into RDDs and distributed to the worker nodes. These RDDs are crucial to speed up the learning of deep models as the memory data access latency is significantly shorter than the disk data operations.

Deep learning parallelism: The solution of a deep learning problem depends on the solution of smaller instances of the same learning problem with smaller datasets. In particular, the deep learning job is divided into learning stages. Each learning stage contains a set of independent Map-Reduce iterations where the solution of one iteration is the input for the next iteration. During each Map--Reduce iteration, a partial model is trained on a separate partition of the available MBD as follows:

• *Learning partial models*: Each worker node computes the gradient updates of its partitions of the MBD (the Map procedure). During this step, all Spark workers execute the same Map task in parallel but on different partitions of the MBD. In this way, the expensive gradient computation task of the deep model learning is divided into many parallel sub-tasks.

• *Parameter averaging:* Parameters of the partial models are sent to the master machine to build a master deep model by averaging the calculated parameters of all Spark workers (the Reduce procedure).

•*Parameter dissemination:* The resulting master model after the Reduce procedure is disseminated to all worker nodes. A new Map-Reduce iteration is then started based on the updated parameters. This process is continued until the learning convergence criterion is satisfied.

As a result, a well-tuned deep learning model is generated that can be used to infer information and patterns from streaming requests. [1]

7. DIFFERENT FRAMEWORKS FOR SPARK-BASED DEEP LEARNING

Here, we are going to see the different kinds of frameworks available for Spark based Deep Learning. Finally, we are going to provide the comparisons among these various frameworks.

7.1 SparkNet

SparkNet is an open source software developed by AMPLab. Along with the core concept of a scalable, distributed deep neural network training algorithm, SparkNet also includes an interface for reading from Spark's data abstraction, known as the Resilient Distributed Dataset (RDD), a Scala interface for interacting with the Caffe deep learning framework (which is written in C++), and a lightweight tensor library.

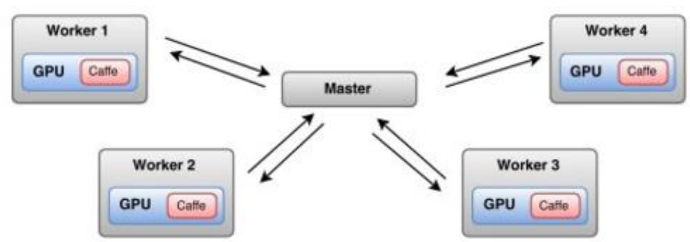

Figure 4: SpakNet Framework

7.2 DeepLearning4J

Deeplearning4j is the first commercial-grade, open-source, distributed deep-learning library written for Java and Scala. Integrated with Hadoop and Spark, DL4J is designed to be used in business environments on distributed GPUs and CPUs. Skymind is its commercial support arm. Some of its features are as follows:-

- Distributed CPUs and GPUs
- Java, Scala and Python APIs
- Adapted for micro-service architecture
- Parallel training via iterative reduce
- Scalable on Hadoop

- GPU support for scaling on AWS

7.3 CaffeOnSpark

CaffeOnSpark is designed to be a Spark deep learning package. Spark MLlib supported a variety of non-deep learning algorithms for classification, regression, clustering, recommendation, and so on. Deep learning is a key capacity that Spark MLlib lacks currently, and CaffeOnSpark is designed to fill that gap. CaffeOnSpark API supports dataframes so that you can easily interface with a training dataset that was prepared using a Spark application, and extract the predictions from the model or features from intermediate layers for results and data analysis using MLLib or SQL.

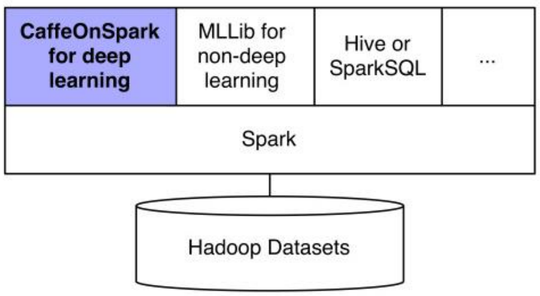

Figure 5: CaffeOnSpark Framework Architecture

7.4 TensorflowOnSpark

TensorFlowOnSpark (TFoS), enables distributed TensorFlow execution on Spark and Hadoop clusters. TensorFlowOnSpark is designed to work along with SparkSQL, MLlib, and other Spark libraries in a single pipeline or program.TensorFlowOnSpark supports all types of TensorFlow programs, enabling both asynchronous and synchronous training and inferencing. It supports model parallelism and data parallelism, as well as TensorFlow tools such as TensorBoard on Spark clusters. Any TensorFlow program can be easily modified to work with TensorFlowOnSpark. Typically, changing fewer than 10 lines of Python code are needed. Many developers at Yahoo who use TensorFlow have easily migrated TensorFlow programs for execution with TensorFlowOnSpark.

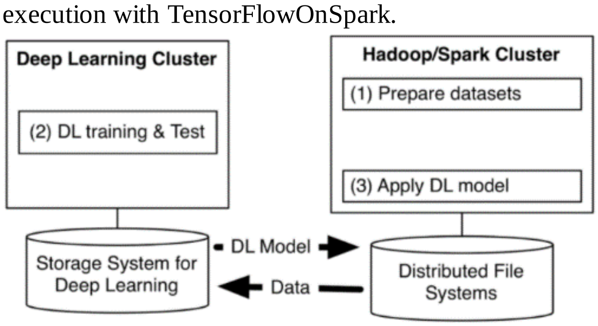

Figure 6: TensorflowOnSpark Framework Architecture

Now, let us compare these 4 frameworks. The evaluation criterion for the comparisons is as shown below:-

Evaluation Criteria	Dimensions	For Example
Ease of Getting Started	Documentation	Are there detailed, well-organized, up-to-date documents?
	Installation	How automatic it is?
	Built-in Examples	Examples available for quick warming up?
Ease of Use	Interface	Programming language support
	Model Encapsulation	Model/Layer/Node
Functionality	Built-in Models	Which NN models have been implemented?
	Parallelism	Model parallelism or data parallelism
Performance	Performance	MNIST benchmark results
Status Quo	Community Vitality	Github project statistics
	Enterprise Support	Contributions from organizations?

Table 1: Evaluation Criterion

On the basis of these evaluation criterions, the following comparisons was found as per in the Spark Summit 2016.

Evaluation Criteria	Dimensions	SparkNet	DL4J	CaffeOnSpark	Tensorflow on Spark
Ease of Getting Started	Documentation	★★★☆	★★★★	★★★☆	★★★☆
	Installation	★★★★☆	★★★★★	★★★	★★★★
	Built-in Examples	★★★★	★★★★	★★★	★★★★
Ease of Use	Interface	★★★★	★★★★	★★★★☆	★★★★
	Model Encapsulation	★★★★☆	★★★☆	★★★★☆	★★★★☆
Functionality	Built-in Models	★★★★☆	★★★★★	★★★☆	★★★★
	Parallelism	★★★	★★★	★★★	★★★
Performance	Performance	★★★☆	★★★	★★★★☆	★★★★
Status Quo	Community Vitality	★★★	★★★★	★★★☆	★★☆
	Enterprise Support	★★★	★★★★	★★★★	★★★☆

Table 2: Comparisons among different Frameworks [03]

8.APPLICATIONS

Below given are some of the applications of Deep Learning built over Apache Spark for Mobile Big Data Analytics:

1.Information Retrieval and Semantic Indexing

One of the prominent areas of Big Data Analytics is information retrieval [7]. Effective storage and retrieval of information is a growing problem of Big Data, essentially as extremely huge amounts of data like text, image, video, and audio are being gathered and processed for various purposes across various domains, for instance, social networks, web crawling, data aggregation, fraud detection, security systems, online shopping and internet traffic monitoring. Conventional approaches and solutions for information repository and retrieval face challenges with the large amounts of data and various data sources, both linked with Big Data. In these platforms, large amounts of data could be processed with semantic indexing rather than being resided as raw data. Semantic indexing defines the data in a more sufficient fashion and becomes useful as a source for information discovery and understanding, for instance by impacting search engines operate faster.

2.Conducting Discriminative Tasks

While performing discriminative tasks in Big Data Analytics, Learning algorithms allow users to extract complicated nonlinear features from the raw data. It also facilitates the use of linear models to perform discriminative tasks using extracted features as input. This approach has two advantages: Firstly, by extracting features with Deep Learning adds nonlinearity to the data analysis, thereby associating discriminative tasks closely to AI, and secondly applying linear analytical models on extracted features is more efficient computationally. These two benefits are important for Big Data because it allows practitioners in accomplishing complicated tasks related to Artificial Intelligence like object recognition in images, image comprehension, etc. [9]

3.Semantic Tagging

With the Internet exploding with online users, digital content has been on an exponential rise. This is especially true of images and videos uploaded from multiple sources.When talking of such massive repositories of images, you cannot afford to stick with textual relationships of images, for storage and retrieval. For improved image searches, the process of browsing and retrieval should be lightening quick and broad based. This needs an automated system of tagging images and videos.Deep learning prepares complicated representations for image/video data in the form of high level abstractions. These can then be used for image tagging that is more suitable for huge data. [10]

4.Sentiment Analysis

With the rise of social media such as blogs and social networks, reviews, ratings and recommendations are rapidly proliferating; being able to automatically filter them is a current key challenge for businesses looking to sell their wares and identify new market opportunities. This has created a surge of research in sentiment classification (or sentiment analysis), which aims to determine the judgment of a writer with respect to a given topic based on a given textual comment. Sentiment analysis is now a mature machine learning research topic [4].

9. CONCLUSION

In this paper, we have given a review of Spark based deep learning in mobile big data analytics. The work enables the tuning of deep models with many hidden layers and millions of parameters on a computing cluster. Typically, deep learning provides a promising learning tool for adding value by learning intrinsic features from raw mobile big data. Also, the different frameworks for Spark-based Deep Learning are presented and the comparisons have been shown among these.

REFERENCES

[1] Mohammad Abu Alsheikh, DusitNiyato, Shaowei Lin, Hwee-Pink Tan, and Zhu Han. Mobile Big Data Analytics Using Deep Learning and Apache Spark. IEEE Network, May/June 2016.

[2] Leelavathi MV, Sahana Devi K J. Efficient Deep Learning for Big Data: A Review. International Journal of Computer Sciences and Engineering, Volume-4, Special Issue-3, May 2016.

[3] Yu Cao, Zhe Dong, “Which is Deeper Comparison of Deep Learning frameworks on Spark”, Spark Summit - 2016.

[4] Vishal A. Kharde, S.S. Sonawane, "Sentiment Analysis of Twitter Data: A Survey of Techniques", International Journal of Computer Applications (0975 - 8887) Volume 139 - No.ll, April 2016.

[5] Nagwa M. Elaraby, Mohammed Elmogy, ShereifBarakat. Deep Learning: Effective Tool for Big Data Analytics. Nagwa M. Elaraby et al. / International Journal of Computer Science Engineering (IJCSE).

[6] DemetriosZeinalipour-Yazt, DemetriosZeinalipour-Yazt. Mobile Big Data Analytics: Research, Practice and Opportunities, 2014 IEEE 15th International Conference on Mobile Data Management.

[7] National Research Council. Frontiers in Massive Data Analysis. The National Academies Press, Washington, DC; 2013.

[8]https://www.cisco.com/c/en/us/solutions/collateral/service-provider/visual-networking-index-vni/vni-hyperconnectivity-wp.html

[9]https://www.allerin.com/blog/3-applications-of-deep-learning-in-big-data-analytics

[10] https://www.allerin.com/blog/applications-of-deep-learning-in-big-data-analytics