Jordanian Journal of Computers and Information Technology (JJCIT), Vol. 08, No. 01, March 2022.

A COMPARATIVE STUDY OF DIFFERENT SEARCH AND INDEXING TOOLS FOR BIG DATA

Ahmed Oussous[1] and Fatima Zahra Benjelloun[2]

(Received: 17-Nov.-2021, Revised: 15-Jan.-2022, Accepted: 26-Jan.-2022)

ABSTRACT

The exponential growth of data generated from the Moroccan court makes it difficult to search for valuable knowledge within multiple and huge datasets. Traditional searching methods are not adapted to Big Data context. Indeed, handling the search of specific information on Big Data requires advanced methods and powerful search systems. To contribute to the Court Digital Transformation Strategy, we aim to develop a solution that will leverage the technological advances in this field.The project we propose consists in developing new methods and techniques of artificial intelligence in order to automate the content of a large mass of data produced by the jurisdictions of the Kingdom of Morocco and to design a system capable of analyzing large volumes of complex judicial data. The aim is to discover and explain certain existing phenomena or to extrapolate new knowledge from the information analyzed, to recognize shapes, make predictions and make the necessary adjustments if necessary. For that, the purpose of this first study is to investigate and examine the existing search and indexing technologies for Big Data. It compares the leading solutions used for information retrieval in order to choose one that will serve as the base for our jurisprudential search engine.

KEYWORDS

Big data, Indexation, Search engines, Solr, ElasticSearch, Lucene.

1. INTRODUCTION

Nowadays, the potential of Big Data is recognized by many industries, research laboratories, governmental and private sectors. They exploit Big Data to extract valuable insight and knowledge. In fact, more than thousands of data gigabytes are rapidly generated every day, in different formats and from heterogeneous sources (ex., ICT applications, sensors, social media, mobile devices, logs and so on) [1].

Big Data is raising many challenges [2]. In fact, because of Big Data characteristics (velocity, variety and volume), experts need to process and analyze Big Data rapidly to extract valuable insights, find and analyze patterns within such large data, establish more accurate predictions and get a better understanding of the industrial changes. Big Data analysis is a powerful tool to maintain companies' agility and competitiveness [3].

Thus, to process such huge streams of data generated very rapidly and in different formats, experts need powerful solutions to stock, manage, process and analyze Big Data.

These tools are well explained in our paper as a review that surveys recent technologies developed for Big Data [4].This article offers a broad overview of the major Big Data technologies, as well as comparisons based on system components, such as data storage, data processing, data querying, data access and management. It classifies and examines the primary technological aspects, benefits, limitations and applications. Actually, experts need also advanced search and indexation tools to deal with Big Data that imposes huge volumes and high complexity. In fact, research efforts previously focused on finding efficient massive storage capabilities. But, there was a shift in research to innovate new and efficient solutions for advanced big-data analytics [5].

In fact, extracting useful information from such huge volumes of data requires adapted tools to perform advanced analytics and search operations on big data. Not only solutions should be scalable, efficient and powerful, but they should handle the complexity of the unstructured datasets and to retrieve data from distributed storage [6].

1. A. Oussous is with Department of Informatics, Faculty of Sciences and Techniques of Mohammedia (FSTM), Hassan II University, Casablanca, Morocco. Email: Ahmed.oussous@fstm.ac.ma
2. F. Benjelloun is with Laboratory of Engineering Sciences, National School of Applied Sciences, Ibn Tofail University, Kenitra, Morocco. Email: fatima.benjelloun1@gmail.com

Thanks to the advancement in information technology, it has become easier to collect and store Big Data. But, there are other challenges, such as how to find new and efficient ways to first index data, then to extract information and valuable knowledge from massive volumes of unstructured data.

To fill this gap, many advanced indexing solutions have been developed and incorporated in big data analysis. The goal is to enhance query execution and optimize operations as well as to improve the efficiency of searching information in large, complex and unstructured datasets.

The process of search engine indexing needs scalable and powerful solutions that can collect, parse and store tremendous data volumes. Such process includes also the creation of indexes to ensure fast, efficient and accurate information retrieval. Indeed, searching aims to process queries and retrieve information based on both queries and the indexes previously created [6].

Indexation and search tools are important, as they play a major role to access and rapidly search data items. These search tools help experts in many data analysis tasks; for instance, to investigate and analyze massive datasets in order to find hidden patterns and discover relationships and other useful information.Those extracted patterns and information enable experts to get a better understanding of the studied phenomenon as well as to monitor sector changes and evolutions (for instance, customer behaviours and preferences). Such cohesive understanding is needed to make timely strategic decisions to take advantage of opportunities, minimize risks and control costs [7].

Big-data search and analytics technologies are used in many sectors (e.g. finance, marketing, research, health, security) and for important applications such as : to monitor disease evolution, adapt medical prescriptions online, detect traffic congestion points, understand the decision process of drivers enhance customer services, predict citizen and customers behaviors, prevent terrorism, detect policy violations and better understand nature evolution [8]-[11].

1.1 Motivation and Methodology

Multiple solutions were designed to tackle the issues related to information retrieval in the case of Big Data. However, because each solution has its advantages and limits, users need experience and deep knowledge to select the best suitable solution for their user case.

As far as we know, there is no published state-of-the-art survey assessing the efficiency and performance of such technologies in the literature. Hence, we try to fill this gap. This article is a continuation of our previous article, which deals with big-data technologies [4].

Information management is one of the main axes on which the Ministry of Justice is committed to modernize and develop the judicial administration with a view to establish the foundations of the digital court. To this end, the Ministry has made several investments to modernize judicial administration methods by intensifying the use of new information and communication technologies.

The transition to digital court implies the generation of raw, semi-structured and information-rich data. It generates extremely large data, which makes the management of this data quite complicated and difficult to process and analyze with classic database management tools or traditional information management tools. This large amount of data motivated us to study new methods and technologies that can be applied in order to understand the structure of any type of data and integrate it into models that can be understood and used by everyone. One of the research axes that we deal with in this article is the indexing and search for judicial information.

Indeed, the objective of this paper is to investigate and examine the existing search and indexing technologies for Big Data. It compares between the leading solutions used for information retrieval. The goal is to offer a detailed overview about such solutions as well as their best use cases in order to choose one that will serve as the base for our jurisprudential search engine.

Several search engines have been compared in terms of multiple criteria, like functionality, productivity, efficiency in searching and indexing, ease of use, speed and safety and so on. Advantages and disadvantages of each search engine have been included.

This paper is structured as follows: An overview of important related works on different search tools for Big Data is discussed in Section 2. Big Data search technologies are presented in Section 3 and an advanced comparison analysis of these different tools is discussed in Section 4. A brief conclusion is given in the last section.

2. RELATED WORKS

Most of Big Data search surveys pertinent to this topic give an overview of Big Data search tools' applications, opportunities and indexing challenges. Others discuss also techniques and methodologies used in big-data search and how they can help improve performance and results' accuracy.

In light of the literature, [12] examined Solr and ElasticSearch in terms of query and indexing speeds, simplicity of use, configuration forms and architectures. [13] compared and contrasted the two most popular platforms for building information retrieval systems, Apache Solr and ElasticSearch. The writers looked at both systems to see what they have to offer and how they are used. They looked at expert comments on both systems as well as real-world examples. They conducted a comparative analysis that looked at a variety of factors, including usability and scalability. Finally, they came to the conclusion regarding whatever system is superior for which application. Solr and ElasticSearch were compared by [14] in terms of productivity, ease of use, speed and safety. In addition, the advantages and disadvantages of either search engines have been included.

[15] utilized Splunk to detect the attack of Distributed Denial of Service. The authors used the data generated from the attacks with the Splunk platform to conduct data analysis to quickly identify attacks and predict potential dangers that could arise. [16] developed SmallClient as an indexing system for huge text data to increase indexing and search performance for large datasets. SmallClient focuses on increasing the volume and speed with which large datasets are processed. As a result, their technology becomes a generic indexing framework with quicker data access by allowing users to choose block size and replication factor. SmallClient's performance improves with increasing data amount, according to tests on small and large datasets.

A short comparison of four search engines, Sphinx, Apache Solr, ElasticSearch and Xapian, is done in a research article released by a group of researchers from Moscow Technological University [17]. The authors recommended to use ElasticSearch to arrange the interface for working with Big Data (search and visualization). [18] presented a system that uses a customized ElasticSearch search engine to successfully solve the problems of real-time analysis. As a consequence, the authors discovered that a suitable configuration of ElasticSearch and Kibana enables real-time analysis of large-scale data and can assist policy makers in seeing the findings instantly to support the decision-making. [19] conducted a functional analysis of well-documented open source forensic tools and search engines. The authors presented also a literature study of publicly accessible forensic datasets. They compared through a benchmarking exercise both ElasticSearch and Solr's indexing as well as full text searching procedures in terms of memory and time usage. [20] outlined the fundamentals of developing and deploying a social-media monitoring and analysis system for cybersecurity. The system is the outcome of a systematic method of gathering, processing and analyzing publicly available data. It is based on the use of information retrieval, data analysis and information flow aggregation methodologies and tools. In their built-in system, Sphinx, a full-text search engine for massive data, is employed as a search engine. [21] described the design and deployment of a CLP tool that compresses unstructured text logs while allowing rapid searches on the compressed material. CLP allows more efficient search and analytics on historical logs as compared to ElasticSearch and Splunk enterprise. [22] examined how the academic infrastructure network SINET was hit by a coronavirus-based cyber assault. They built a data flow pipeline based on ElasticSearch and Splunk to handle massive session traffic data recorded on SINET in order to extract and evaluate the COVID-19 attacker group's traffic patterns. Table 1 covers some of the most current studies on search and indexing technologies.

Table 1. Recent works on search and indexing tools.

Article & Year	Objective	Indexing and Search Tools Used	Obtained Results
[12] 2016	Comparison of big-data tools Solr and ElasticSearch	ElasticSearch and Solr	They are similar tools in terms of technical features. Both tools are rapid search tools. ElasticSearch has a wider range of coding languages than Solr. ElasticSearch performs better with short data, whereas Solr performs better with long data. When compared to the amount of data after indexing, Solr utilizes less disk space.

[13] 2016	Providing an overview of the best options for constructing information retrieval systems to developers and members of the scientific community, as well as offering insight into the best use cases for both technologies	Apache, Solr and ElasticSearch	ElasticSearch's ease of use, flexibility and modular architecture make it an excellent candidate for prototype as well as big, scalable information retrieval applications. ElasticSearch provides considerably superior data analytics and the ELK stack, when paired with Logstash and Kibana, it outperforms Solr in several areas, including preprocessing, analytics and visualization. The present version of ElasticSearch has a drawback in that it lacks a centralized mechanism for managing cluster nodes. Teams with Solr expertise should think twice before switching to a new system, as both systems are almost comparable in most circumstances.
[14] 2016	Comparing and analyzing the security of Solr and ElasticSearch; two popular full-text search engines	Solr and ElasticSearch	It includes powerful filtering, highlighting, multi-dimensional searching, caching, Rest Api and a distributed architecture support engine. The Restful API is a very quick and useful tool. When compared to the Solr search engine, ElasticSearch is less complicated and detailed. It is both durable and adaptable. One of the most significant advantages is that it is distributed and real-time.
[15] 2016	Detecting distributed denial of service attacks	Splunk	During DDoS assaults against firewalls, researchers used Splunk big-data technologies to examine traffic characteristics. The experimental results did certainly aid in the knowledge of various attack types and a warning system might be used to identify security issues prior to an assault.
[16] 2017	Creating a huge text data indexing system to increase indexing and search efficiency for massive datasets	SmallClient	When compared to the Lucene indexing library, SmallClient outperforms in terms of index generation and has the shortest time between data upload and query execution. SmallClient indexes are also lower in size than Lucene indexes, in addition to being faster to create. To get complete records, Lucene requires that all attributes be indexed. SmallClient, on the other hand, is not one of them. Even when just one attribute is indexed, SmallClient allows you to obtain whole data records. With growing data amount, SmallClient improves.
[17] 2017	Analysis of software for full-text search and data visualization	Sphinx, Solr and ElasticSearch	Sphinx has a rapid search and indexing system, but it is slow to update. Only MySql and Postgres are supported by Sphinx. Overall, the ElasticSearch system is the best choice for full-text search and data visualization.
[18] 2018	Suggesting a solution to real-time analytical problems	ElasticSearch and Kibana	ElasticSearch is a real-time storage, pre-indexing, search and query solution for very big datasets. A correct ElasticSearch and Kibana configuration enables for real-time analysis of enormous amounts of data, allowing policy makers to view the results instantly and in a manner that allows for decision-making.
[19] 2018	Comparing the functionality, efficiency and effectiveness of open-source search engines for digital forensic search	Solr and ElasticSearch	Deduplication keyword recommendations and search result clustering are supported by Solr, whereas phonetic search is supported by ElasticSearch. Solr provides many unique capabilities that can help with large-scale dataset search. In terms of index building time, ElasticSearch outperformed Solr.

[20] 2020	Monitoring system for social-media content	Sphinx	The practical importance of the obtained results is to develop a functioning model of a social-media content monitoring and analysis system that can be used as part of information and cyber security decision support systems.
[21] 2021	Building a fast and scalable search tool for compressed text logs	CLP tool	ElasticSearch and Splunk enterprise are equivalent, if not better than the CLP tool when it comes to search performance. The CLP tool exceeds Elastic-search and Splunk enterprise in terms of log ingestion by almost 13 times.
[22] 2021	Extracting and analyzing the traffic patterns of the COVID-19 attacker group	ElasticSearch and Splunk	Some unveiled patterns are informative to handling security operations of the academic backbone network.

3. BIG DATA INDEXING TECHNOLOGIES

With big-data search and indexing technologies, data scientists and others can analyze huge volumes of data that conventional analytics and traditional business intelligence solutions cannot handle. The following sub-sections discuss the finest search tools that provide full featured search engines. Thanks to their scalable and high-performance indexing, these tools are designed for information retrieval in Big Data.

3.1 Apache Lucene

We opted to introduce Apache Lucene [23] before looking into Solr and ElasticSearch. This introduction constitutes the information retrieval library for both systems.

Apache Lucene [24] is developed to address big-data searching needs. Lucene is an open-source, high-performance and full-featured text search engine library that is built completely in Java.

Apache Lucene offers multiple query options and scalable indexing (it indexes almost 150 GB per hour on commodity hardware) with minimal memory requirements. The algorithm offers ranked searching, field searching, data-range searching as well as multiple-index searching [8].

Apache Lucene offers powerful features related to four main categories: analysis of incoming content and queries, indexing and storage, searching and ancillary modules (everything else) [25]. The first three items contribute to Lucene's core, while the last item consists of code libraries that have proven to be useful in solving search-related problems.

A high-level Lucene architecture is presented in Figure 1. Its main components are IndexSearcher, IndexReader, IndexWriter and Directory. The IndexWriter object is used to create the index and add new index entries (i.e., Documents). IndexReader reads the content of indexes in support of IndexSearcher. Directory abstracts out the implementation of index dataset access and provides APIs for manipulating them. Both IndexReader and IndexWriter leverage Directory for access to this data. The standard Lucene distribution contains several Directory implementations, such as filesystem-based and memory-based, Berkeley DB-based (in the Lucene contrib module) and several others [26]. Lucene is one of the most powerful and widely used search engines.

3.2 Apache Solr

Solr [27] is the popular, blazing fast open-source enterprise search platform from the Apache LuceneTM project. Apache Solr is more compatible, its major features include powerful full-text search, hit highlighting, faceted search, nearly real-time indexing, dynamic clustering, database integration, rich document (e.g. Word, PDF) handling and geospatial search.

Thanks to SolrCloud mode [28], Solr provides a highly available, scalable, replication and fault-tolerant environment for distributing the indexed content and requests across multiple servers with the help of ZooKeeper. In fact, SolrCloud uses the information in the ZooKeeper database to figure out which servers need to handle the request. In fact SolrCloud's integration with end-user applications is depicted in Figure 2.

As illustrated in Figure 2, there are four elements. SolrCould is an indexing and search service that runs

independently. Users can shop for items from the online store *via* an end-user application, such as an online store application. The Content Management System gives the shop's employees an internal access to update product information from various data sources. Solr will index the product metadata for end users to consume *via* a simple HTTP request and return format of JSON, XML or CSV.

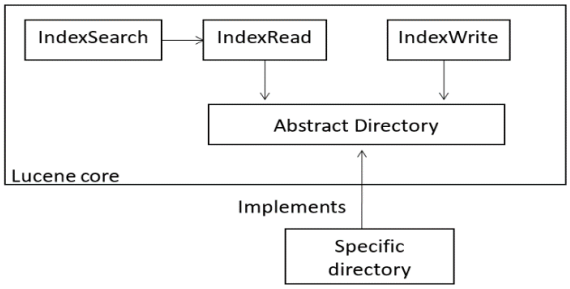

Figure 1. High-level Lucene architecture.

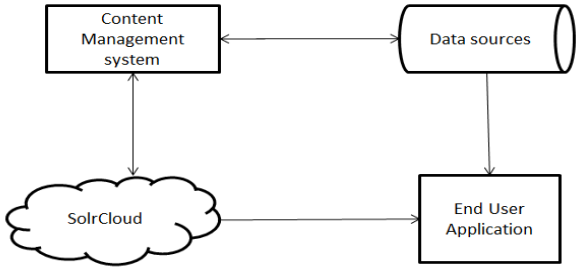

Figure 2. Solr integration with applications.

While Apache Solr can easily handle high-volume traffic, Apache Lucene is used by search-based sites in order to handle reverse index and the related issues.

Contrary to Lucene, Solr is easy to use. It can be installed and used by non-programmers. Solr is a web application (WAR) which can be deployed in any servlet container. Solr is integrated into major distribution of Hadoop (Cloudera, Hortonworks, MapR) as the search engine for their products marketed for Big Data [29].

Unlike Lucene, which is a Java library that can only be used from other Java programs, Solr on the other hand is a wrapper around Lucene that allows using the Lucene functionality from any programming language that can submit HTTP requests.

Solr is used by many largest internet sites across the world. This is because it enhances the search and navigation features. Indeed, it is capable of indexing, efficiently searching multiple websites and returning recommendations for related content. For that, it uses the search query's taxonomy. Furthermore, Solr is a mature solution that has a large user community.

3.3 ElasticSearch

ElasticSearch [30] is an Apache 2.0 licensed open source search solution that is based on JSON. It is an efficient solution used to store, search and analyze structured and unstructured data. Thus, it is suitable for many data types, including system logs, free text, time-series and NoSQL data. ElasticSearch is built on top of Apache Lucene.

ElasticSearch is a distributed, multi-tenant and document-oriented search engine [31]. It supports distributed deployments, by breaking down an index into shards and distributing the shards across the nodes in the cluster.

By integrating it to Logstash and Kibana tools, ElasticSearch can perform tasks for search, analysis and visualization operations. The integration of these three tools is called ELK stack [32].

While both ElasticSearch and Apache Solr use Apache Lucene as the core search engine, ElasticSearch aims to provide a more scalable and distributed solution that is better suited for cloud environments than Apache Solr.

3.4 Splunk

Splunk Enterprise [27] is one of the leading platforms for collecting, analyzing and visualizing machine-generated Big Data. It provides a unified way to organize and extract real-time insights from massive amounts of machine data generated by diverse sources. Spunk is equivalent to ELK Stack that includes ElasticSearch, Logstash and Kibana for storage, analysis and visualization. But, it is mainly used for big-data analysis and can analyze structured or semi-structured data. It is possible to get 15 days free trail of Splunk commercial solutions. The latter was released in 2003.

HunK: Splunk Analytics for Hadoop [33] is a platform for discovering, analyzing and visualizing Hadoop's historical data at rest. Hunk is a full-featured Hadoop exploration, analysis and visualization application. Hunk offers huge improvements in the speed and ease of gaining insights from large data at

rest in Hadoop, based on many years of expertise designing big-data solutions that have been implemented by thousands of Splunk customers. Hunk is compatible with Apache Hadoop and the majority of Hadoop distributions, including MapReduce.

Splunk has three main functionalities, including data collection, data indexing, as well as data search and analysis as follows [34]:

- Data collection: Splunk can gather static data as well as data created by real-time monitoring of modifications and additions to files and directories. Data can also be gathered directly from programs or scripts using network ports. Splunk can also gather, insert and update data from relational databases.
- Indexing: The acquired data is divided into events, which are essentially equal to database entries or simply lines of data. The data is then processed and a high-performance index that points to the stored data is built and updated.
- Search and analysis: Users may use the Splunk Processing Language to search for data and alter it to get the information they need, whether it is in the form of reports or alerts. Individual events, tables and charts can be used to show the findings [35].

3.5 Sphinx Search Server

Sphinx (SQL Phrase Index) [36] is a standalone full-text search engine that gives third-party programs, particularly SQL databases with efficient search capability. This search engine was created in 2001 by Andrew Aksyonoff, a Russian engineer, to ensure (1) good search quality, (2) fast speed and (3) low resource usage (Disk IO, CPU). It's compatible with scripting languages like Python and Java.

Sphinx [37] is an open-source full-text search server, designed from the ground up with performance, relevance (aka search quality) and integration simplicity in mind. It is written in C++ and works on Linux (RedHat, Ubuntu, …etc), Windows, MacOS, Solaris, FreeBSD and a few other systems. Sphinx clusters scale up to tens of billions of documents and hundreds of millions search queries per day, powering top websites, such as Craigslist, Living Social, MetaCafe and Groupon.

Sphinx [38] has been improved. Currently, it is able to handle nearly real-time search among huge volumes of files. In fact, if users need search functions without data visualization and analysis, then Sphinx is a good choice for fast indexing and querying. Sphinx can process 500 queries/sec against 1,000,000 documents with the biggest registered number of indexing estimated at 25+ billion documents. Table 2 provides a general overview for the features of big-data search tools.

Table 2. General overview for the features of big-data search tools.

Feature	**solr**	**ElasticSearch**	**Splunk**	**Sphinx**
Initial release	2004	2010	2003	2001
License	Open-source	Open-source	Commercial	Open-source
Developer	Apache Software Foundation	Elastic	Splunk, Inc.	Sphinx Technologies, Inc.
Format	XML, CSV, JSON	JSON		
Official client libraries	Java	Java, Groovy, PHP, Ruby, Perl, Python, .NET, Javascript		C++
Community client libraries	PHP, Ruby, Perl, Scala, Python, .NET, Javascript, Go, Erlang, Clojure	Clojure, Cold Fusion, Erlang, Go, Groovy, Haskell, Java, JavaScript, .NET, OCaml, Perl, PHP, Python, R, Ruby, Scala, Smalltalk, Vert.x	C# ,Java, JavaScript, PHP, Python Ruby	C++, Java, Perl, PHP, Python, Ruby
Server operating systems	All OS with a Java VM	All OS with a Java VM	Linux OS X Solaris Windows	FreeBSD Linux NetBSD OS X Solaris Windows

4. COMPARATIVE ANALYSIS OF BIG DATA INDEXING TECHNOLOGIES

This section compares the various tools we discussed in Section 3, notably Solr, ElastiSearch, Splunk and Sphinx, in terms of a variety of criteria.

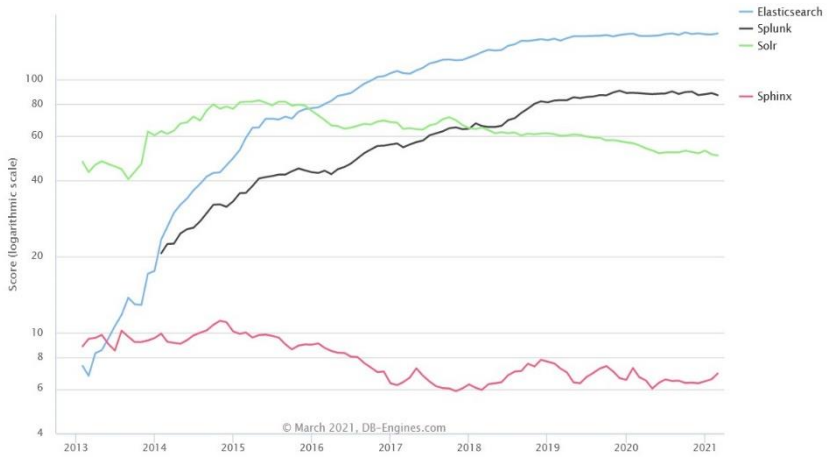

Figure 3. DB-engines ranking.

4.1 Ranking of Search Engines

Regarding the popularity (Figure 2), ElasticSearch is well ranked in comparison to other big-data tools, as it is considered the most popular search engine since 2016. In fact, while ElasticSearch is ranked number one, Sphinx is ranked number two and Solr is ranked number three, as confirmed by DB-Engines that ranks database management systems and search engines according to their popularity. On the contrary, Solr has become popular among the first ten years of its initial release.

4.2 Data Sources

Both ElasticSearch and Solr can handle various types of data sources. Since ElasticSearch is totally based on JSON, it supports data ingestion from different sources by using Logstach and Beasts family.

On the contrary, Solr is based on request handlers to ingest data from multiple sources, including CSV files, databases, XML files, Microsoft Word documents and PDFs. Solr is capable of supporting extraction and indexing from over one thousand file types. This is because of the native support for the Apache Tika library.

Splunk, on the other hand, provides tools for setting a variety of data sources, including those specific to application requirements. Splunk also has capabilities for configuring input forms for any type of data. Files and folders, data from system log files and any other application that uses the TCP protocol can all be used as Splunk inputs. Data may be indexed by Splunk Enterprise from any network port. It can also index data sent over UDP. Splunk software also supports a variety of data sources, including Windows Event Log data, Active Directory data and data from performance monitoring.

When it comes to Sphinx, the data to be indexed can originate from a variety of places, including SQL databases, plain text files, HTML files, e-mails and more. The data that Sphinx indexes is a collection of structured documents, each with the identical set of fields and characteristics. This is comparable to SQL, in which each row represents a document and each column represents a field or property. Different code is necessary to get the data and prepare it for indexing depending on what source Sphinx should obtain the data from. Data source driver is the name of the program (or simply driver or data source for brevity).

4.3 Use Cases

Both Solr and ElasticSearch are document-oriented search engines [13]. But, Solr is more focused on enterprise-directed text searches with advanced information retrieval (IR). It is a good choice for use cases that need to search within a massive volume of static data or to handle Rich Text Format (RTF) documents. Solr is also recommended for Enterprise applications that are based on Big Data ecosystem, like Spark or Hadoop. To be competitive, Solr has implemented new features, including Parallel SQL Interface and streaming expressions.

On the contrary, ElasticSearch is adapted for many use cases. For instance, it is a powerful and flexible solution for full text search. Indeed, it is a good choice not only for Enterprise search and E-commerce [39], but also for other use cases, such as fraud detection, security and collaboration [40]-[41].

ElasticSearch is known for its scalability and easiest way to implement powerful logging solutions. It is capable to grab and index different remote large data sources. It can handle easily time-series data, such as application events and metrics. ElasticSearch is more convenient for modern web applications where data is in JSON format.

Sphinx is a full-text search engine with the advantages of fast indexing and searching, as well as integration with existing database management systems (MySQL, PostgreSQL) and an API for common web programming languages (officially supports PHP, Python and Java; community-implemented APIs for Perl, Ruby, .NET and C++). For Russian and English languages, sophisticated search features, such as ranking and stemming, are supported. The Delta index technique can be used to accelerate indexing for huge amounts of data. Sphinx also offers Real-Time indexes, search result filtering and sorting and wildcard searching. In comparison to ElasticSearch, Sphinx uses fewer memory and compute resources.

Splunk began as a machine-generated data analytics platform, but it has now extended into a number of different domains, including the fields of IT operations and application delivery, security compliance, fraud management, business analytics and the internet of Things. It continues focused on becoming "the omnipresent machine data platform, the standard in every enterprise."This was accomplished through the development of a number of products and version updates, including Splunk Enterprise 8.2.4, a new version of Splunk Cloud Platform, Splunk Enterprise Security 7.0.0 and so on. Event sequencing, a new use case library to speed investigations, rules to strengthen insider threat detection models and solutions to give a threat intelligence-centered view for investigations are among the new features of the revised products.

Both ElasticSearch [3] and Splunk [5] are two of the industry's biggest players right now. Elastic claimed sales of $428 million with 11,300 clients in their most recent fiscal year [13], whereas Splunk reported a revenue of $2.359 billion with 19,400 customers [21]. Furthermore, the products ElasticSearch and Splunk Enterprise are employed by a number of major corporations, including eBay, Verizon and Netflix.

4.4 Searching

Currently, ElasticSearch and Solr support (nearly real-time) searches as well as JSON-based Query DSL. They both take advantage of Lucene's search capabilities.

Unlike ElasticSearch, Solr enables users to write complex search queries. Solr's Standard Query Parser allows users to create a variety of structured queries, but the probability of syntax errors is higher.

On one hand, Solr provides a search user interface named Velocity Search. The latter has robust features. In addition to searching, users can exploit faceting, highlighting, autocomplete and Geo Search. On the other hand, ElasticSearch has a native DSL and a robust aggregation framework with a better caching. It is noticed that the last releases of ElasticSearch ensure a better memory management.

Splunk Assistant [42] is a search function that appears as users input their search parameters into the Search application. The Search Assistant is similar to autocomplete, but with a lot of more features. Matching queries are also returned by the Search Assistant, which are based on recent searches. When users wish to rerun a search from yesterday or a week ago, they can use the Matching Searches list. When they log out, their search history is saved.

Splunk's Search Processing Language (SPL) [43] contains commands and functions for creating searches. Sphinx treats full-text searches as simple "bags of words" by default and all keywords in a document must match in order for the query to succeed. To put it another way, users may do a rigorous Boolean AND on all keywords by default. Text queries, on the other hand, are significantly more versatile and Sphinx has its own full-text query language to reveal that versatility.

4.5 Indexing Performance

Earlier, Solr was based on a defined schema. But currently, both ElasticSearch and Solr supports schemaless mode. As a result, both are flexible and can be used to index data and dynamic fields. So, users do not need to define in advance the schema of the index.

Both ElasticSearch and Solr write indexes in Lucene. But, they have different architecture, files and different mechanisms for sharding and replication. Moreover, while Solr has a powerful Standard Query Parser that is compatible with Lucene syntax, ElasticSearch has native DSL (Domain Specific Language)

support. Both solutions support synonym-based indexing, stemming, custom analyzers and various tokenization options.

Sphinx has a quick search and indexing system [44]; however, it is slow to update due to the lack of an automated index updating mechanism. It only works with MySql and Postgres, which is a huge limitation. It is incompatible with the work at hand, since it is unable to update or remove documents in the index.

Apache Solr features a fast indexing and searching performance, one of the lowest index sizes and a lot of adaptability. It can also be used as a storage facility. Solr comes with a slew of extra features, like imprecise search and the capacity to scale right out of the box. The drawback is that it is a Java server in a servlet container that has been turned into a web service with XML, JSON and CSV interfaces.

ElasticSearch, which is built on Apache Lucene, has somewhat slower indexing and searching speeds than Sphinx, but it also has other features in addition to search and storage (visualization, log collector, encryption system, …etc.). It has the ability to scale and can sample highly complicated forms, making it an excellent choice for an analytical platform. This engine is not the most user-friendly, but it has a lot of extra functions. The main benefit is that this engine consumes very little memory and incremental indexing is as quick as indexing several articles at once. ElasticSearch is substantially quicker than Solr for indexing, as demonstrated by [19].

As a result, ElasticSearch, a search engine and full-text search system, is ideally suited for searching and visualizing enormous volumes of clustered data resulting from users' interactions with diverse information resources.

On the other hand, ElasticSearch and Splunk Enterprise work by creating external indexes on log messages as they are being ingested. These tools may then swiftly search the indexes corresponding to the logs in response to a query, decompressing just the chunks of data that may include logs matching the search term. For example, ElasticSearch is based on Lucene, a general-purpose search engine. This strategy, however, comes with a high cost in terms of storage space and memory use. Despite the fact that these methods compress the logs lightly, the indexes typically take up the same amount of space as the raw logs; moreover, to be completely effective, these indexes must be maintained largely in memory or on fast random access storage.

Thus, users of Splunk Enterprise and ElasticSearch who have a lot of data may only afford to keep their indexed logs for a few weeks [21].

4.6 Clusters, Sharding and Rebalancing

Both search engine solutions support sharding. However, while SolrCloud enables further splitting of an existing shard, ElasticSearch does not offer this option. So, shards cannot increase once they've been created in ElastiSearch. But, shards of an index can be reduced in ElasticSearch based on a shrink API, but it is not possible using SolrCloud.

For cluster coordination, ElasticSearch provides built-in Zen Discovery module. Instead, SolrCloud needs an additional service that is Apache Zookeeper.

When there is a shard or node failure, Elastisearch rebalances clusters automatically. It is rare when manual intervention is required. But, SolrCloud has a complex rebalancing mechanism that is hard to manage [45].

Indexer clusters are groups of Splunk Enterprise indexers set to duplicate each other's data, allowing the system to store multiple copies of all data. Index replication is the name for this procedure. Clusters reduce data loss while enhancing data availability for searches by retaining several, identical copies of Splunk Enterprise data.

Automatic failover from one indexer to the next is a characteristic of indexer clusters. This implies that even if one or more indexers fail, incoming data is still indexed and searchable.

Sphinx offers distributed search capabilities, which helps it scale effectively. In multi-server, multi-CPU or multi-core setups, distributed searching can help reduce query latency (i.e., search time) and throughput (max queries/sec). This is critical for apps that must sift through large volumes of data (i.e., billions of records and terabytes of text). It also allows you to create an arbitrary cluster architecture, clustering and sharding over several agent servers.

4.7 Data Visualization

A user-friendly interface (Graphical user interface (GUI)) is essential for users. For that, Splunk has improved its GUI by integrating a new dashboard and its controls. It offers also the possibility to export the dashboards to pdf version *via* simple features [46].

On the contrary, ElasticSearch does not offer its own GUI. Therefore, users need to install Kibana for visualization [18]. Kibana has various cool background themes that Splunk does not offer. It offers also different controls to manipulate dashboards. Thus, the dashboard in Kibana is slightly better than in Splunk.

The Banana project [47], which was forked from Kibana and works with all types of time series (and non-time series) data saved in Apache Solr, has been integrated into Apache Solr's data visualization capabilities. It makes use of Kibana's extensive dashboard configuration capabilities, adapts important panels to work with Solr and adds a slew of new features. The objective is to offer a rich and flexible user interface that allows users to quickly design end-to-end applications that take advantage of Apache Solr's capability.

4.8 Machine Learning

Solr offers machine learning as a free module that runs on top of the streaming aggregations architecture [48]. Users may employ machine-learned ranking models and feature extraction on top of Solr with the help of the additional libraries in the contrib module, whilst the streaming aggregation-based machine learning is focused on text categorization using logistic regression.

ElasticSearch, on the other hand, offers a commercial solution called X-Pack [49], which includes a Kibana plugin that enables machine-learning techniques for anomaly and outlier identification in time-series data. It is a fantastic set of tools with professional services wrapped in, but it is rather costly.Through Splunkbase, users of Splunk Enterprise and Splunk Cloud Platform can use the Machine Learning Toolkit (MLTK).The Machine Learning Toolkit adds additional Search Processing Language (SPL) search commands, macros and visualizations to the Splunk platform. More than 30 algorithms are supported by MLTK, which are the most extensively used machine-learning algorithms. Anomaly Detection, Classifiers, Clustering Algorithms, Cross-validation, Feature Extraction, Preprocessing, Regressors, Time Series Analysis and Utility Algorithms are all categorized by algorithm type.

4.9 The Community

ElasticSearch is driven more by its company. Indeed, even though those contributors can access and change the code, the final changes are confirmed by the employee of the company. Most of the code is open-source, but there are non-open premium features. For Solr, users can contribute directly to its open-source code. New Solr developers or code committers are selected based on merit. It has a large community.

Splunkbase is a Splunk-hosted community where users can find applications and add-ons for Splunk that can enhance its capability and usefulness, as well as providing a quick and easy interface for certain use-cases and/or vendor products. There are currently over 2,512 applications on the framework [50].

4.10 Documentation

ElasticSearch improved its website and its documentation. As a result, user can find easily clear configuration instructions and multiple examples. Furthermore, because of the ElasticSearch popularity, the internet is full of its books and guides. On the contrary, Solr documentation is not well-maintained. In fact, following its release, it was easy to find well documentation about API's use cases and good examples. But currently, Solr documentation is not complete as many gaps were noticed by users. APIs coverage is not sufficient and it is not easy to find good technical examples and tutorials.Sphinx is the same way, with only a few pages of documentation and no technical examples.

Splunk documentation, on the other hand, comes in a number of formats and topic kinds. Step-by-step instructions, conceptual information, reference manuals, troubleshooting pages, use cases and product tutorials are all included in the Splunk documents collection. The easiest approach for users to achieve their goals using Splunk products is to read the documentation provided by Splunk.

4.11 Summary

After carefully analyzing all systems and reviewing relevant papers and publications, we have come to the conclusion that all of the systems given are viable options for document indexing and searching.

It is not easy to define a winner between those advanced big-data solutions. For that, it is necessary not only to understand their features, ease of maintenance, scaling options, but also to analyze their use cases.

To summarize the comparison, Solr and ElasticSearch have common advantages. For instance, both technologies are quite easy to install and to begin working with. Furthermore, both engines are documented and have matured codebase and large ecosystem. However, they are different. For instance, while Solr offers many functionalities in the field of information retrieval, ElasticSearch is easier for production and scalability.

Depending on their case study and requirements, users can select between the two.Solr is the ideal option for consumers who want a text-based search. ElasticSearch, on the other hand, is the perfect solution if they need distributed and scalable features with analytical queries.

Sphinx is an excellent tool for searching structured data (predefined fields and non-text attributes). Sphinx, on the other hand, needs a lot of effort and time to configure for unstructured material, like MP3s, PDFs and DOCs.As a result, compared to its competitors, it is more difficult to use.

ElasticStack (ELK Stack) and Splunk are the two most popular enterprise log analytics platforms. Splunk is a software tool for monitoring, analyzing and visualizing data. ElasticSearch is a database search engine, while Splunk is a software tool for monitoring, analyzing and visualizing data. Splunk is used to search, monitor and analyze machine data, whereas ElasticSearch stores and analyzes data. Splunk has a number of drawbacks, one of which is that it is a paid and pricey tool, whereas ElasticSearch is a free tool.

In terms of data transfer and user administration, Splunk is a simple and dependable solution, although ElasticSearch is rapidly gaining these capabilities.

Table 3 compares the performance and setup capabilities of several big-data search tools. The comparison is based on data processing techniques and direct or indirect access, data storage, data processing, distributed architectural features, search and indexing capabilities and tool performance.

Table 3. Technical comparison of big data-search tools.

Technical Specifications	**Sub-specifications**	**Solr**	**ElasticSearch**	**Splunk**	**Sphinx**
Access and Data Processing	SQL	Solr Parallel SQL Interface	SQL-like query language	No	SQL-like query language (SphinxQL)
	APIs and other access methods	Java API REST-ful HTTP/JSON API	Java API REST-ful HTTP/JSON API	HTTP REST	Proprietary protocol
	Data Import	DataImportHandler CSV, XML, Tika, URL, Flat File	Rivers modules, ActiveMQ, Amazon SQS, CouchDB, Drop-box, DynamoDB, FileSystem, Git, GitHub, , JDBC, JMS, Kafka, MongoDB, neo4j, Redis, RSS , Twitter, … etc.	Event logs, web logs, live application logs, network feeds, system metrics, archive files, …etc	SQL databases, plain text files, HTML files, mailboxes and so on
Distributed Architecture	Master-slave replication	Only in non-SolrCloud	Not an issue, because shards are replicated across nodes	Multi-source replication	None
	Partition Tolerance	Yes	No	Yes	Yes
	Shard replication	Yes	Yes	Yes	Yes
	Consistency	Eventual Consistency: Indexing requests that are synchronous with replication	By default consistent; Replication between nodes is set to synchronous	Eventual Consistency	
	Web Admin interface	Bundled with Solr	Marvel or Kibana apps	Splunk Web	JamDocs: a web interface

Indexing and Searching	Indexing and Searching	Text-oriented	Better performance of analytical queries	A scalable and reliable platform for investigating, monitoring, analyzing and acting on data	Better ranking relevance
	Real-time Sarch/Indexing	Yes	Yes	Yes	Yes
	Performance	High	High	High	High
	Visualization of data	Banana (Port of Kibana)	Kibana	Splunkbase	No
Characteristics		Highlighters, spell check, autocomplete, filter queries, geospatial, synonyms	index, cross-cluster search, Highlighters, Query DSL, Typeahead, corrections (spell check)	autocomplete suggestions, multifield	autocomplete suggestions, spell checker, faceted, synonyms, highlighting

5. CONCLUSIONS AND FUTURE WORK

Nowadays, large data volumes are daily generated at unprecedented rates from heterogeneous sources. However, traditional technologies lack scalability and performance needed in big-data context. Indeed, traditional indexing solutions are not adapted for big-data. This is because of the large and increasing size of indexes that requires more processing time and optimized index scheme.

This paper reviews and compares the main searching and indexing tools developed to handle big-data challenges. Those solutions are different, but most of them integrated advanced technologies to be scalable and powerful with high-performance indexing.

Furthermore, we compare their features. We notice that most of them optimize indexing and queries to ensure a real-time searching and indexing. They support full-text search by many ways. In addition, they offer shard replication, eventual consistency and methods to process data coming from distributed storage or in the Cloud with some differences. Some offer also the possibility to create plug-in APIs. For an easy usage, they also integrate options for visualization and other features. In addition to this comparison, users need experience to select among big-data searching and indexing solutions according to their needs, because each of them has advantages and limitations.

Overall, the ElasticSearch system is the best choice for full-text search and data visualization applications (free, open-source, simple interface, web-based data processing). It is suggested that the interface for working with big-data be organized using ElasticSearch's capabilities (search and visualization).

In the future work, we aim to build a legal search engine for the Moroccan Ministry of Justice, based on ElasticSearch, which was recommended after conducting this study.

ACKNOWLEDGEMENTS

This paper is part of a project of the AL KHAWARIZMI research program funded by CNRST and ADD: "Elaboration d'un système numérique robuste et intelligent dans le domaine de la justice".

REFERENCES

[1] T. J. Ma, R. J. Garcia, F. Danford, L. Patrizi, J. Galasso and J. Loyd, "Big Data Actionable Intelligence Architecture," Journal of Big Data, vol. 7, no. 1, pp. 1–19, 2020.

[2] V. V. Kolisetty and D. S. Rajput, "A Review on the Significance of Machine Learning for Data Analysis in Big Data," Jordan. Jou. of Comp. and Inf. Technol. (JJCIT), vol. 6, no. 01, pp.41-57, 2020.

[3] J. Wang, Y. Yang, T. Wang, R. S. Sherratt and J. Zhang, "Big Data Service Architecture: A Survey," Journal of Internet Technology, vol. 21, no. 2, pp. 393–405, 2020.

[4] A. Oussous, F.-Z. Benjelloun, A. A. Lahcen and S. Belfkih, "Big Data Technologies: A Survey," Journal of King Saud University-Computer and Information Sciences, vol. 30, no. 4, pp. 431–448, 2018.

[5] H. Hu, Y. Wen, T.-S. Chua and X. Li, "Toward Scalable Systems for Big Data Analytics: A Technology Tutorial," IEEE Access, vol. 2, pp. 652–687, 2014.

[6] A. Gani, A. Siddiqa, S. Shamshirband and F. Hanum, "A Survey on Indexing Techniques for Big Data: Taxonomy and Performance Evaluation," Knowledge and Inf. Systems, vol. 46, no. 2, pp. 241–284, 2016.

[7] V. Jatakia, S. Korlahalli and K. Deulkar, "A Survey of Different Search Techniques for Big Data," Proc.

of the IEEE International Conference on Innovations in Information, Embedded and Communication Systems (ICIIECS), pp. 1–4, Coimbatore, India, 2017.

[8] T. Lee, H. Lee, K.-H. Rhee and U. S. Shin, "The Efficient Implementation of Distributed Indexing with Hadoop for Digital Investigations on Big Data," Computer Science and Information Systems, vol. 11, no. 3, pp. 1037–1054, 2014.

[9] T. H. Davenport and J. Dyché, "Big Data in Big Companies," International Institute for Analytics, vol. 3, pp. 1–31, 2013.

[10] R. V. Zicari, "Big Data: Challenges and Opportunities," Big Data Computing, vol. 564, p. 103, 2014.

[11] H. Ma, W. Du, S. Xu and W. Li, "Searching Tourism Information by Using Vertical Search Engine Based on Nutch and Solr," Proc. of the 17th IEEE International Conference on Software Engineering Research, Management and Applications (SERA), pp. 128–132, Honolulu, HI, USA, 2019.

[12] M. A. AKCA, T. Aydoğan and M. İlkuçar, "An Analysis on the Comparison of the Performance and Configuration Features of Big Data Tools Solr and ElasticSearch," International Journal of Intelligent Systems and Applications in Engineering, vol. 6, no. Special Issue (2016), pp. 8–12, 2016.

[13] N. Luburić and D. Ivanović, "Comparing Apache Solr and ElasticSearch Search Servers," Proc. of the 6th International Conference on Information Society and Technology (ICIST 2016), pp. 287-291, 2016.

[14] U. Kılıç and K. Aksakalli, "Comparison of Solr and ElasticSearch among Popular Full Text Search Engines and Their Security Analysis," Proc. of 6th International Conference on Future Internet of Things and Cloud Workshops, pp. 163–168, DOI: 10.13140/RG.2.2.24563.32803, 2016.

[15] T.-J. Su, S.-M. Wang, Y.-F. Chen and C.-L. Liu, "Attack Detection of Distributed Denial of Service Based on Splunk," Proc. of the IEEE International Conference on Advanced Materials for Science and Engineering (ICAMSE), pp. 397–400, Tainan, Taiwan, 2016.

[16] A. Siddiqa, A. Karim and V. Chang, "Smallclient for Big Data: An Indexing Framework towards Fast Data Retrieval," Cluster Computing, vol. 20, no. 2, pp. 1193–1208, 2017.

[17] A. Voit, A. Stankus, S. Magomedov and I. Ivanova, "Big Data Processing for Full-text Search and Visualization with ElasticSearch," Int. J. of Advanced Comp. Sci. and Appl., vol. 8, no. 12, p. 18, 2017.

[18] N. Shah, D. Willick and V. Mago, "A Framework for Social Media Data Analytics Using ElasticSearch and Kibana," Wireless Networks, vol. 2018, pp. 1–9, DOI: 10.1007/s11276-018-01896-2, 2018.

[19] J. Hansen, K. Porter, A. Shalaginov and K. Franke, "Comparing Open Source Search engine Functionality, Efficiency and Effectiveness with Respect to Digital Forensic Search," Norsk Informasjonssikkerhetskonferanse (NISK), pp. 1-14, 2018.

[20] D. Lande, I. Subach and A. Puchkov, "A System for Analysis of Big Data from Social Media," Information & Security, vol. 47, no. 1, pp. 44–61, 2020.

[21] K. Rodrigues, Y. Luo and D. Yuan, "CLP: Efficient and Scalable Search on Compressed Text Logs," Proc. of the 15th USENIX Symposium on Operating Systems Design and Implement., pp. 183–198, 2021.

[22] R. Ando, Y. Kadobayashi, H. Takakura and H. Itoh, "Understanding Traffic Patterns of Covid-19 Ioc in Huge Academic Backbone Network Sinet," International Journal of Network Security & Its Applications (IJNSA), vol. 13, no. 6, pp. 23-36, 2021.

[23] D. Shahi, Apache Solr: A Practical Approach to Enterprise Search, ISBN: 978-1-4842-1070-3, 2016.

[24] R. Gao, D. Li, W. Li and Y. Dong, "Application of Full Text Search Engine Based on Lucene," Advances in Internet of Things, vol. 2, no. 4, DOI:10.4236/ait.2012.24013, 2012.

[25] A. Białecki, R. Muir, G. Ingersoll and L. Imagination," Apache Lucene 4," Proc. of SIGIR Workshop on Open Source Inf. Retrieval, [Online], Available: http://opensearchlab.otago.ac.nz/paper_10.pdf, 2012.

[26] B. Lublinsky, K. T. Smith and A. Yakubovich, Professional Hadoop Solutions, ISBN: 978-1-118-61193-7, John Wiley & Sons, 2013.

[27] R. Kuć, Apache Solr 4 Cookbook, ISBN-13: 978-1782161325, Packt Publishing, Ltd., 2013.

[28] B. Abu-Salih, P. Wongthongtham, D. Zhu, K. Y. Chan and A. Rudra, Social Big Data Analytics: Practices, Techniques and Applications, ISBN: 978-981-33-6652-7, Springer Nature, 2021.

[29] B. Abu-Salih, P. Wongthongtham, D. Zhu et al., "Introduction to Big Data Technology," Ch. 2 in Book: Social Big Data Analytics: Practices, Techniques and Applications, pp. 15–59, 2021.

[30] C. Gormley and Z. Tong, ElasticSearch: The Definitive Guide - A Distributed Real-time Search and Analytics Engine, ISBN: 9781449358549, O'Reilly Media, Inc., 2015.

[31] S. Bhandarkar and N. BN, "A Full-text-based Search Algorithm *vs* ElasticSearch," Studies in Indian Place Names, UGC Care Journal, vol. 40, no. 74, pp. 2168–2171, 2020.

[32] Y. Gupta and R. K. Gupta, Mastering Elastic Stack, ISBN-13: 978-1786460011, Packt Pub., 2017.

[33] L. Belcastro, F. Marozzo, D. Talia and P. Trunfio, "Big Data Analysis on Clouds," In Book: Handbook of Big Data Technologies, pp. 101–142, DOI:10.1007/978-3-319-49340-4_4, Springer, 2017.

[34] P. Zadrozny and R. Kodali, Big Data Analytics Using Splunk: Deriving Operational Intelligence from Social Media, Machine Data, Existing Data Warehouses and Other Real-time Streaming Sources, ISBN-13: 978-1430257615, Apress, 2013.

[35] B. P. Sigman and E. Delgado, Splunk Essentials, 2nd Ed., ISBN: 9781785882135 1785882139, Packt Publishing, Ltd., 2016.

[36] T. Hryhorova and O. Moskalenko, "Use of Information Technologies to Improve Access to Information in E-learning Systems," Proc. of the 18th International Conference on Data Science and Intelligent Analysis of Information (ICDSIAI 2018), vol. 836, pp. 206–215, Springer, 2018.

[37] A. Aksyonoff, Introduction to Search with Sphinx: From Installation to Relevance Tuning, ISBN: 9780596809553, O'Reilly Media, Inc., 2011.

[38] A. Ali, Sphinx Search Beginner's Guide, ISBN-13: 978-1849512541, Packt Publishing, Ltd., 2011.

[39] R. Maski, "Using Apache Solr for Ecommerce Search Applications," Happiest Minds, IT Services, pp. 1-12, [Online], Available: https://www.happiestminds.com/whitepapers/using-apache-solr-for-ecommerce-search-applications.pdf, 2013.

[40] V.-A. Zamfir, M. Carabas, C. Carabas and N. Tapus, "Systems Monitoring and Big Data Analysis Using the ElasticSearch System," Proc. of the 22nd IEEE International Conference on Control Systems and Computer Science (CSCS), pp. 188–193, Bucharest, Romania, 2019.

[41] J. Hamilton, B. Schofield, M. G. Berges and J.-C. Tournier, "SCADA Statistics Monitoring Using the Elastic Stack (ElasticSearch, Logstash, Kibana)," Proc. of the Int. Conf. on Accelerator and Large Experimental Physics Control Systems (ICALEPCS2017), pp. 451-455, Barcelona, Spain, 2017.

[42] S. P. Chamarthi S. Prasad and S. Magesh, "Application of Splunk towards Log Files Analysis and Monitoring of Mobile Communication Nodes," International Journal of Applied Science and Engineering Research, vol. 3, pp. 478-483, 2014.

[43] D. Mehta, "Splunk Search Processing Language," In Book: Splunk Certified Study Guide, pp. 27–52, Springer, 2021.

[44] A. Chaudhary, K. Akshatha, K. Kodlekere and S. J. Prasad, "Keyword Based Indexing of a Mmultimdia File," IEEE International Symposium on Multimedia (ISM), pp. 573–576, Taichung, Taiwan, 2017.

[45] P. Kumar, P. Kumar, N. Zaidi and V. S. Rathore, "Analysis and Comparative Exploration of ElasticSearch, Mongodb and Hadoop Big Data Processing," in Book: Soft Computing: Theories and Applications, pp. 605–615, Springer, 2018.

[46] P. Zadrozny and R. Kodali, "Visualizing the Results," in Book: Big Data Analytics Using Splunk, pp. 63–96, Springer, 2013.

[47] K. Venkatesh, M. J. S. Ali, N. Nithiyanandam and M. Rajesh, "Challenges and Research Disputes and Tools in Big-data Analytics," Int. J. of Eng. and Advanced Technol., vol. 6, pp. 1949–1952, 2019.

[48] V. Prajapati, Big Data Analytics with R and Hadoop, ISBN 978-1-78216-328-2, Packt Pub., Ltd., 2013.

[49] F. A. Vadhil, M. L. Salihi and M. F. Nanne, "Toward a Secure ELK Stack," International Journal of Computer Science and Information Security (IJCSIS), vol. 17, no. 7, pp. 139-143, 2019.

[50] Splunkbase, "Home | Splunkbase," [Online], Available: http:splunkbase.splunk.com, [Accessed: Dec. 2021].

ملخص البحث:

إنّ النّمو المتصاعد للبيانات المتولّدة من المحكمة المغربية يجعل من الصّعب البحث عن المعرفة القيّمة في مجموعات البيانات المتعدّدة والضّخمة. وإنّ طرق البحث التقليدية غير منسجمة مع سياق البيانات الضّخمة. وفي الحقيقة، فإنّ البحث عن معلوماتٍ معيّنة في البيانات الضّخمة يتطلّب طرقاً متقدمة وأنظمة بحثٍ عالية الفعالية. وللإسهام في استراتيجية التّحول الرّقمي للمحكمة، نهدف الى تطوير حلّ من شأنه أن يعزّز التّطورات التكنولوجية في هذا المجال. ويتمثّل المشروع الذي نحن بصدد القيام به في تطوير طرقٍ وتقنياتٍ جديدة تتعلّق بالذّكاء الاصطناعي من أجل أتمتة محتوى هائلٍ من البيانات التي ينتجها النّظام القضائي في المملكة المغربية، وتصميم نظامٍ قادرٍ على تحليل كمّ هائلٍ من البيانات القضائية. وهدفنا هو كشف ظواهر معينة قائمة وشرحها واستباط معرفةٍ جديدةٍ من المعلومات التي يتمّ تحليلها؛ من أجل تمييز الأشكال وعمل التّوقعات وإجراء التعديلات اللازمة عند الضّرورة. لذا، فإنّ هدفنا من هذه الدراسة -الأولى من نوعها- هو استقصاء تقنيات البحث والفهرسة القائمة في مجال البيانات الضّخمة وفحصها. ويقارن البحث أبرز الحلول المستخدمة لاسترجاع المعلومات بغية اختيار الحلّ الأمثل من بينها، الذي يصلح لأن يكون أساساً لمحرّك البحث الذي ننوي تطويره للبحث عن البيانات الضّخمة القضائية وفهرستها وتحليلها.