**A parallel CBIR system using Alluxio and Apache Spark**

Tejas Vaichole- 19BCE1295, Sanika Kulkarni – 19BCE1328, Sandhya S – 19BCE1614

# **Abstract.**

The task of Content Based Image Retrieval (CBIR) is becoming increasingly complex due to the large number of images available on the internet. This task involves retrieval of similar images based on an input image given by the user. To enable faster computation of similar images, the proposed work uses Apache Spark and Alluxio, previously known as Tachyon. Spark is an open-source software used for processing Big Data. It provides parallelism that reduces computational time. Alluxio on the other hand is a virtual distributed storage system. Although models using Spark for CBIR have been proposed earlier, the proposed model aims at reducing the retrieval time of images by optimizing this task by modifying the feature extraction mechanism. Histogram of oriented gradients (HOG) feature descriptor has been used to find the similarity between images. The K Nearest Neighbours (KNN) algorithm has been used and optimized to compute the top K similar images to query images.

# **1. Introduction**

With the spread of the Internet, the amount of data produced by humans has increased significantly. The number of images used has increased due to the availability and use of cell phones, cameras, smartphones and other devices [1]. Content-Based Image Search (CBIR) is one of the most popular ways to obtain similar images on demand. Image searches are not based on keywords or annotations, but rather on features extracted directly from image data. [2].

CBIR can be divided into two main stages. The first step is the indexing step, which retrieves the image elements of the collection and stores them in the database. This is a standalone step that only happens once. Another stage is the image retrieval stage, where the user provides an image of the request and the feature is retrieved. Similar images are retrieved by calculating the distance between the retrieved query image function and the function stored in the database.

Various work has been carried out in this field starting from very simple CBIR to complicated ones. But as more complicated the system gets; more time is consumed in retrieving the images. This can be solved by using the already developed parallel and distributed technology to reduce the time consumed. In this work, one of the most popular approaches is used i.e. using Apache spark.

# **2. Literature Survey**

In this section, the various works related to the CBIR system have been discussed. The first proposed approaches are centralized CBIR systems, such as QBIC [3], Virage [4], MARS [5], TinEye[6], etc. However, with the increasing generation of image data, the conventional CBIR systems consume a lot of time. This drawback has motivated the researchers to increase the performance of CBIR systems using parallel interactive retrieval systems based on semantic and visual descriptors [1], image retrieval optimization using meta-heuristic algorithms [1], and parallel retrieval systems for large-scale images using cloud computing [1].

Hadoop which is a popular big data framework has been used in many works like [7-14]. In [7], they used and improvised the K-Means clustering algorithm by optimizing the selection of the initial cluster center and iteration procedure. Premchaiswadi et al. used Hadoop MapReduce and overcame the drawback of parallel computing technique using cluster platform architecture [8]. They also introduced a new feature extraction algorithm i.e. ACCC(Auto Color Correlogram and Correlation) which reduced the processing time. Costantini et al. used Hadoop for the indexing phase and spark in the retrieval phase. This allowed the system to be completely scalable and to reduce the computational cost. [10] also uses Hadoop and MapReduce for high performance targeting big data. They also proposed a novel bag of visual words (BOVW) technique based on a proposed chain-clustering binary search-tree (CC-BST) algorithm to build the visual statements for representing the image. Mohammed et al. have proposed a parallel and distributed computation using the HIPI framework (Hadoop Image Processing Interface) and HDFS (Hadoop Distributed File System) as a storage system, and exploited the high power of GPUs (Graphic Processing Units) [11]. Similarly, [12] uses CBIR on Hadoop MapReduce framework and Local Tetra Patterns (LTrPs) as their feature descriptor. Apart from using Hadoop [14] has used Lucene to better their results.

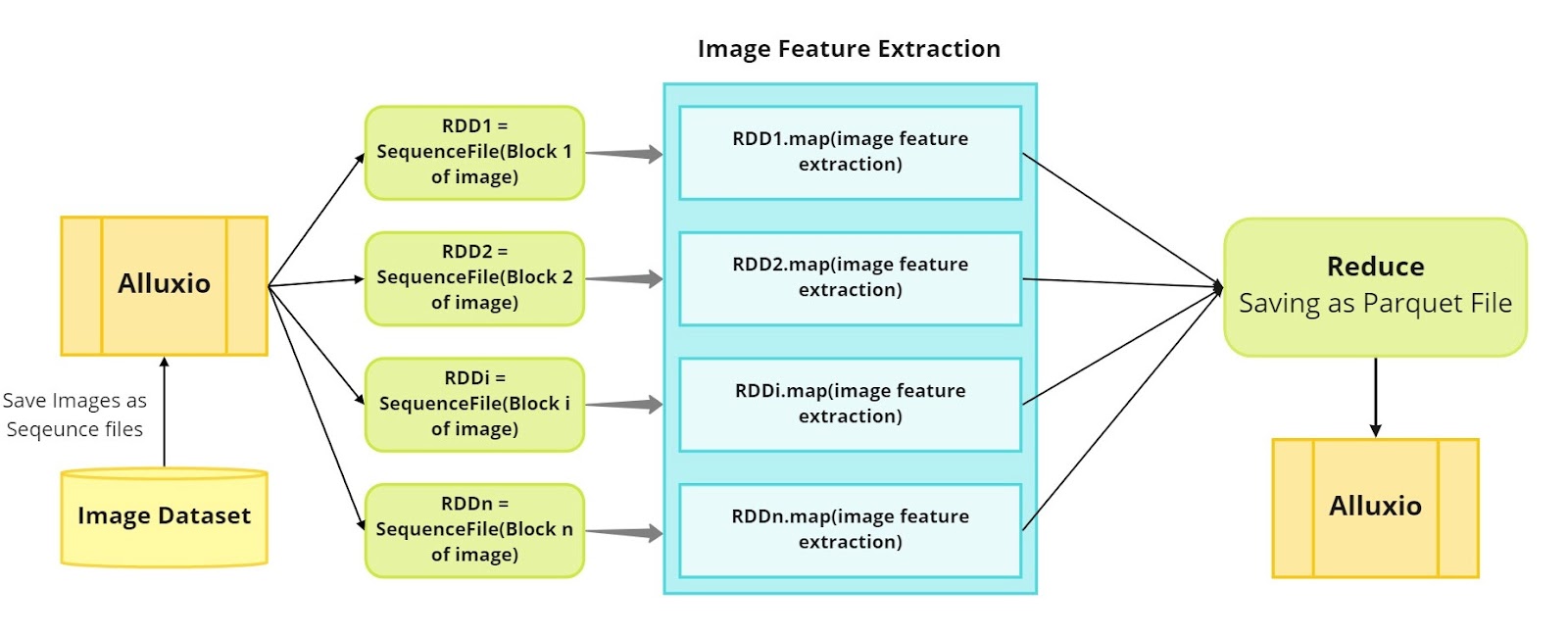
One of the recent advancements in the field of parallel CBIR involves the use of Apache Spark as seen in [1][15-16]. [1] uses Apache Spark and Tachyon for data storage. They have used a CS-LBP feature descriptor and parallel KNN algorithm to find the distance between images. Duan et al. also used Apache Spark to reduce the computational cost of frequent read and write on HDFS [15]. They have used Avro to combine the files created after feature extraction. Similar to [1], [16] uses Apache Spark and KNN algorithm for the task of CBIR. They have used MapReduce model framework to index the large-scale images and Spark has been used as a proportionate method of retrieving the index, which runs on the higher layer of MapReduce and Hadoop distributed file system (HDFS) environment. HDFS provides an in-memory data storage and fast retrieval mechanism using the indexing process.

**3. Proposed Work**

This research work proposes a Content-Based Image Retrieval system using Apache Spark and Alluxio. The detailed explanation about Apache Spark and Alluxio is given in the further sub-sections.

***Apache Spark***

Apache Spark is an opensource distributed processing system used for big data. It is based on the Hadoop MapReduce model used for large-scale data processing. Alternative approaches, such as Hadoop's MapReduce, are optimized to work in memory while writing data to and from a computer's hard drive. So spark is faster than other computational models. Extend MapReduce problems to more types of computations, including interactive queries and streaming. [17]. Various CBIR studies have shown that Spark helps increase search times when compared to other systems. For this reason, Apache Spark is preferred for computational purposes.



***Figure 1. Indexing step***

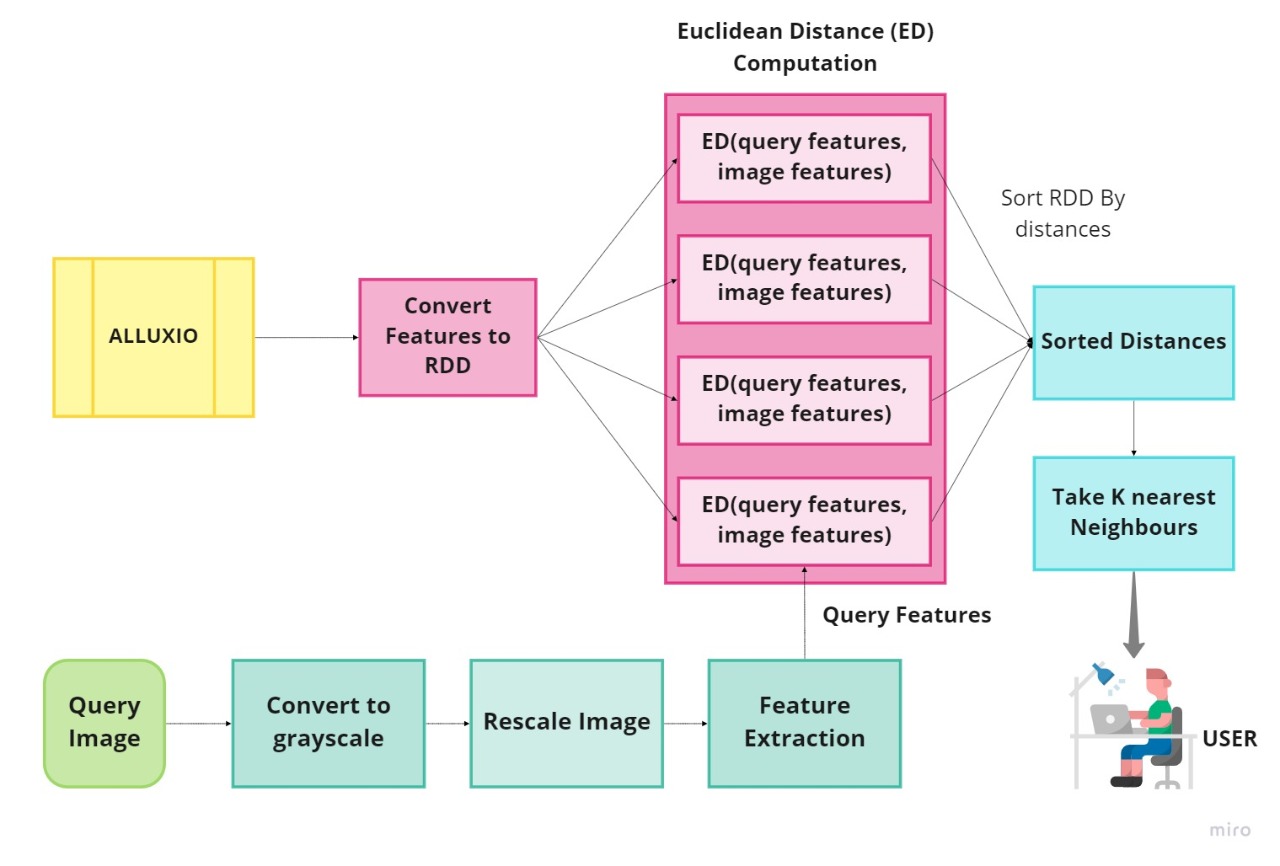
***Alluxio***

Alluxio, formerly known as Tachyon, is a virtual distributed storage system. Bridging the gap between data-driven applications and storage systems [18]. It can also be used as a distributed shared caching service, allowing computing applications that interact with Alluxio to transparently cache frequently accessed data, especially from remote locations, providing in-memory I/O throughput. In addition, Alluxio's tiered storage that can use both memory and disk (SSD/HDD) makes scaling data-driven applications cost-effective.

This study proposes a CBIR architecture utilizing spark and alluxio for processing and storage. The Alluxio is used to store images and features extracted from them for faster access. Spark, on the other hand, is used to extract features from an image and obtain similar images based on the needs of the image. The architecture of our system can be divided into two main phases: indexing and search phase. These steps are detailed below.

***Indexing Step***

Indexing is an offline step which is not performed during the retrieval of similar images. In this step, the features are extracted from the database images and stored in the storage system. To extract the features the images are first stored in the alluxio in the sequence file format. SequenceFile is a flat file consisting of binary key/value pairs. It is extensively used in MapReduce as input/output formats [20]. For each image the sequence file contains the following information <fileName: String, image: Bytes >. Then stored sequence files are read from alluxio in spark. The sequence files are decoded to obtain images. These images are converted to grayscale and then resized for feature extraction. Histogram of Oriented Gradients (HOG) is used to extract the features [19]. HOG is used in computer vision for object detection purposes. The technique counts occurrences of gradient orientation in the localized portion of an image. This method is quite similar to Edge Orientation Histograms and Scale Invariant Feature Transformation (SIFT). The HOG



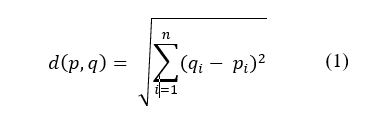
**Figure 2. Searching step**

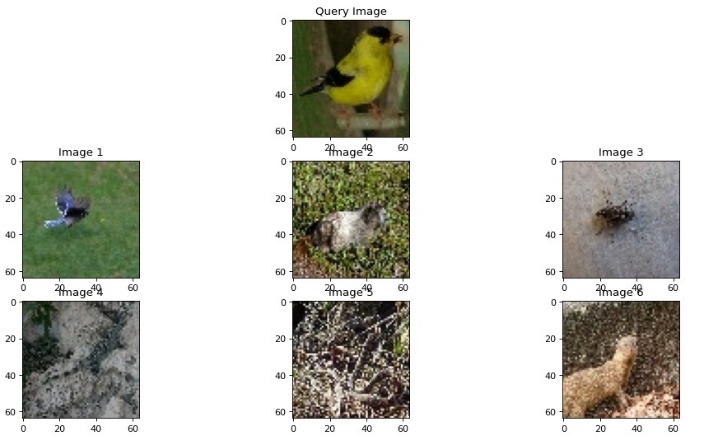
descriptor focuses on the structure or the shape of an object. As spark uses distributed computing based on MapReduce the features are computed parallely. Hence the feature extraction takes very less time. The extracted features are converted to spark dataframe and stored in alluxio in parquet format. These features then can be read and used for the image retrieval steps. The overall indexing step is shown in Figure 1.

***Searching Step***

This is an online step where similar images are extracted based on the extracted features. Query image is first taken as input from the user. From the query image the HOG features are extracted using the same method used for extracting features from database images.

Once the query image feature is extracted, retrieval of similar images is carried out, using stored database features. A parallel K-Nearest Neighbor (KNN) algorithm is used in spark. In KNN algorithm the Euclidean distance between the query feature and extracted feature of the images from the database is calculated. The Euclidean distance formula between two features p and q of length n is given in equation (1) below.







**Output of query images**

**4. Experimental Setup**

Experiments conducted as part of the research work were performed on the local system, i.e., on one node only. All experiments were performed on a VBOX virtual machine (VM) on a Windows host system. The virtual machine used Ubuntu 20.04 LTS operating system with 4 GB of RAM and 4 cores. Here are the specs of the machine the VM is running on: 8 GB RAM, 9th gen i5 processor and Windows 10 operating system also provided distributed and parallelized computations using Alluxio 2.7.0, Spark 3.1.1, Java 11 and Python 3.8. Python with pyspark was used for coding. Opencv and scikit image were used to compute the feature from the image.

# **5. Summary**

The proposed models were compared with various research works. Our model outperforms many works. The comparison of the performance is given below in Table 5. And visualised in Figure 8. The below results have been compared with a system which uses a single node and parallel KNN similar to what was used in the method.

**6. Results**

Results of the proposed system were evaluated on two datasets of different sizes. The datasets used were TinyImagenet [21] and ImageClef [22] dataset. TinyImageNet dataset is the small version of the ImageNet dataset [23]. Dataset is available on Kaggle. It had around 1 million images out of which we used only 20,000 images. Images were 64x64. The results obtained are tabulated below in Table 1.

**Table 1. Search time on TinyImageNet**

|  |  |
| --- | --- |
| **Number of Images** | **Average Search Time (s)** |
| 200 | 4.09 |
| 500 | 4.44 |
| 1000 | 4.56 |
| 2000 | 5.22 |
| 5000 | 8.40 |

**7. Conclusion**

The model proposed in this article uses Apache Spark distributed processing to parallelize computation. KNNs using spark-based dataframes have significantly improved results due to their faster processing power than RDDs. The HOG function provides a good measure for finding similarities between images. Alluxio which is used for storage provides faster read and write access at the speed of memory. Therefore, the proposed model improves the search time or search time for similar images compared to the existing model.

# **8. References**

[1] S. Mezzoudj, A. Behloul, R. Seghir, and Y. Saadna, “A parallel content-based image retrieval system using spark and tachyon frameworks,” J. King Saud Univ. - Comput. Inf. Sci., vol. 33, no. 2, pp. 141–149, 2021.

[2] G. Schaefer, “Effective and efficient browsing of large image databases,” in Handbook of Research on Digital Libraries, IGI Global, 2009, pp. 142–148.

[3] Flickner, M., Sawhney, H., Niblack, W., Ashley, J., Huang, Q., Dom, B., Gorkani, M., Hafner, J., Lee, D., Petkovic, D., Steele, D., & Yanker, P. (1995). Query by image and video content: the QBIC system. *Computer*, *28*(9), 23–32.

[4] Bach, J. R., Fuller, C., Gupta, A., Hampapur, A., Horowitz, B., Humphrey, R., Jain, R. C., & Shu, C.-F. (1996). Virage image search engine: an open framework for image management. In I. K. Sethi & R. C. Jain (Eds.), *Storage and Retrieval for Still Image and Video Databases IV*. SPIE.

[5] Rui, Y., Huang, T. S., & Mehrotra, S. (1997). Content-based image retrieval with relevance feedback in MARS. *Proceedings of the 1997 International Conference on Image Processing. Part 2 (of 3)*.

[6] (N.d.). Researchgate.Net. Retrieved December 5, 2021, from https://www.researchgate.net/figure/Screenshot-of-TinEye-reverse-image-search-result\_fig2\_305683970

[7] Jianfang Cao,Min Wang, Hao Shi,Guohua Hu, and Yun Tian1; Mathematical Problems in Engineering Volume 2016

[8] Premchaiswadi, W., Tungkatsathan, A., Intarasema, S., & Premchaiswadi, N. (2013). Improving performance of content-based image retrieval schemes using Hadoop MapReduce. 2013 International Conference on High Performance Computing & Simulation (HPCS).

[9] Costantini, L., & Nicolussi, R. (2015). Performances evaluation of a novel Hadoop and Spark based system of image retrieval for huge collections. Advances in Multimedia, 2015, 1–7.

[10] Sakr, N. A., ELdesouky, A. I., & Arafat, H. (2016). An efficient fast-response content-based image retrieval framework for big data. *Computers & Electrical Engineering: An International Journal*, *54*, 522–538.

[11](N.d.-c). Researchgate.Net. Retrieved December 5, 2021, from https://www.researchgate.net/publication/324844953\_A\_New\_Parallel\_and\_Distributed\_Approach\_for\_Large\_Scale\_Images\_Retrieval

[12] U.S.N. Raju, George, S., Praneeth, V. S., Deo, R., & Jain, P. (2015). Content based image retrieval on Hadoop framework. *2015 IEEE International Congress on Big Data*, 661–664.

[13] Zhang, J., Liu, X., Luo, J., & Lang, B. (2010). DIRS: Distributed image retrieval system based on MapReduce. *5th International Conference on Pervasive Computing and Applications*, 93–98.

[14] Gu, C., & Gao, Y. (2012). A content-based image retrieval system based on Hadoop and Lucene. *2012 Second International Conference on Cloud and Green Computing*, 684–687.

[15] Duan, Y., Jiang, L., Lin, X., & Chen, X. (2016). An improved content based image retrieval system on Apache spark. Proceedings of the 2016 International Conference on Mechatronics, Control and Automation Engineering

[16] Hussain, D. M., & Surendran, D. (2021). The efficient fast-response content-based image retrieval using spark and MapReduce model framework. *Journal of Ambient Intelligence and Humanized Computing*, *12*(3), 4049–4056.

[17]*Apache Spark - Introduction*. (n.d.). Tutorialspoint.Com. Retrieved December 5, 2021, from https://www.tutorialspoint.com/apache\_spark/apache\_spark\_introduction.htm

[18] *Alluxio - data orchestration for the cloud*. (2019, March 12). Alluxio.Io. https://www.alluxio.io/

[19]Dalal, N., & Triggs, B. (2005). Histograms of oriented gradients for human detection. *2005 IEEE Computer Society Conference on Computer Vision and Pattern Recognition (CVPR’05)*, *1*, 886–893 vol. 1.

[20]*SequenceFile*. (n.d.). Apache.Org. Retrieved December 5, 2021, from<https://cwiki.apache.org/confluence/display/HADOOP2/SequenceFile>

[21] L. Han, “Tiny ImageNet Challenge.” .

[22] “IAPR TC-12 Benchmark,” Imageclef.org. [Online]. Available: http://www.imageclef.org/photodata. [Accessed: 05-Dec-2021].

[23] J. Deng, W. Dong, R. Socher, L.-J. Li, K. Li, and L. Fei-Fei, “ImageNet: A large-scale hierarchical image database,” in 2009 IEEE Conference on Computer Vision and Pattern Recognition, 2009, pp. 248–255.