Scalable Solar Image Retrieval with Lucene

Juan M. Banda
Montana State University
Bozeman, Montana

Rafal A. Angryk
Georgia State University
Atlanta, Georgia

Abstract— **In this work we present an alternative approach for large-scale retrieval of solar images using the highly-scalable retrieval engine Lucene. While Lucene is widely popular among text- based search engines, significant adjustments need to be made to take advantage of its fast indexing mechanism and highly-scalable architecture to enable search on image repositories. In this work we describe a novel way of representing image feature vectors in order to enable Lucene to perform search and retrieval of similar images. We compare our proposed method with other popular alternatives and provide commentary of the performance as well as the benefits and caveats of the proposed method.**

Keywords— ***Image Retrieval; Content-Based Image Retrieval; Big Data Mining; Image Processing; Astroinformatics.***

I. INTRODUCTION

As our Solar Dynamics Observatory (SDO) mission image dataset grows to over 43 million images since the mission launched in 2012, the field of solar physics has entered the era of big data mining in a very dramatic way [1]. The main focus of this work is how to be able to scale our existing content-based image retrieval (CBIR) system [2, 3] from a large-scale system to a big data capable tool that allows researchers to query the entire repository. We have theorized on approaches on how to handle this transition in [1], and with this paper we present preliminary results of a scaled version of the SDO CBIR system. We also tackle the issue of not only querying complete images as a basis of similarity, but rather regions of interest found inside each image. We have made advances in this approach using descriptor signatures [4], but as the data continues to grow, query performance might be impacted, so the need for a highly-scalable mechanism of retrieval is imminent.

Lucene is a retrieval engine traditionally deployed and optimized for text and document retrieval [5]. Based on other research group's efforts [6] of bringing Lucene's capabilities to the content-based image retrieval field, we propose a way to adapt our image parameters to provide region based queries in a big data context. In this work we outline the steps needed to translate our images into image parameters for retrieval and how these are in turn adapted as sets of sequences of strings. This transformation will allow us to represent our images in a way that Lucene can accommodate querying large corpuses in an efficient and responsive way. We also propose a system architecture comprised of two Lucene servers to be able to adapt to higher volumes of data as the dataset keeps increasing.

The overall organization of this paper is as follows: after some background information (section II), we explain how our Lucene index was built (section III) and provide insights into our system's architecture (section IV), query building (section V). Experiments are described and discussed in sections VI and VII. Lastly we provide our conclusions (section VIII) and we finally, in section IX we conclude the paper with an outline of our future work.

II. BACKGROUND INFORMATION

Through the years, a number of content-based image retrieval (CBIR) systems have facilitated general purpose image retrieval tasks. Systems like Photobook [7], Candid [8], Chabot [9], QBIC [10] from IBM. These systems rely on features such as shape, color, or texture to match complete similar images without taking into consideration regions of interest to make their matches. Our previous review work [3, 11, 12] extensively covers the development of CBIR as a field, and how it relates to our application. Newer developments in the CBIR filed have explored fuzzy approaches [13], binning-strategies [14], boosting methods [15], as well as more integrated tools [16].

Work in region-based image retrieval started when researchers noticed that Content-Based Image Retrieval (CBIR) would be improved by matching parts of the image, rather than the complete image [17]. The first retrieval-systems with region-based capabilities where: Blobworld [18], SIMPLIcity [17], Netra [19] and Walrus [20], said systems apart offered traditional full-image retrieval but they relied on some sort of underlying region-of-interest based image representation to retrieve images. With [21] researchers focused on adapting existing algorithms from CBIR into the Region-Based realm without addressing the scalability issue [22, 23, 24] since back then image repositories as large as Flickr, PhotoBucket, and SDO did not exist. Fast-forward into the present, the issue of region-based image retrieval is still not completely solved since there are multiple aspects to consider: efficient image representation of newly created complex scientific (amongst others) image repositories and the scale of the image data being generated. A comprehensive review of region-based image retrieval can be found in [25]

Sophisticated approaches like the one presented in [26] show great promise in color images, but since they rely on dominant color features to work, they do not transfer to our Solar image domain. Approaches using clustering are very popular these days [27, 28], but they have been shown to be optimal for particular datasets with very well defined regions of interest. Other approaches speculate on the types of images they are optimal for [29, 30], but without real experimentation we have yet to determine if they can help with our task at hand. Some region-based retrieval and image representation approaches rely on arbitrarily constructed labels for sections of interest [31, 32, 33, 34] that do not translate to images without clear objects to identify, such as solar image data.

Systems like [35, 36] are beginning to take advantage of Big Data technologies (Hadoop, MapReduce) for large-scale retrieval, but are still not addressing the region-based querying issue. LIRE [6] proposes a Java library based on Lucene and Solr and a set of tools [37] to facilitate the use of Lucene for CBIR purposes. LIRE, for our purposes, is limited in scope in terms of providing only a few image parameters similar to the ones proposed in our work. We are looking to adapt a stock version of Lucene to work for our purpose rather than adapt another CBIR-building framework that uses LIRE. We believe that this will lead to us having less overhead and allow us to focus on just building the necessary functionality. Overall, we want to build on top of Lucene's native text retrieval capabilities and translate our image data into an experimental representation to query our image dataset.

A. *Dataset*

For this work, we utilized solar images from the Atmospheric Imaging Assembly (AIA) module of the SDO mission. We selected images compiled by Schuh et al. [38], spanning a six-month period of data containing around 15,000 images in two wavelengths (193 and 131) featuring 24,000 event instances of four different solar events listed in Table I.

TABLE I. DATASET EVENTS AND LABELS

Label	Event Type	Wavelength
AR	Active Region	193
CH	Coronal Hole	193
FL	Flare	131
SG	Sigmoid	131

This dataset is the same one we used in [4], so we provide a 1 to 1 comparison in terms of retrieval performance. We also made two additional scalability test datasets containing 1.5 million and 4.5 million images, created by artificially combining the original 15,000 images based on their image parameters. Labels are not provided to the artificial images, they are only used to simulate a larger amount of items indexed by Lucene. We decided use the scalability test datasets in order to observe the behavior of the Lucene index with a considerably larger amount of items indexed that the search has to go through in order to provide retrieval results.

We have selected each event (and its wavelength) based on SDO's Feature Finding Team (FFT) modules and the data sources they use to identify particular events as indicated in [39]. We will be analyzing retrieval behavior using both Maximal Bounding Boxes (MBR) and chain codes (boundary outlines) for the labeled regions of two separate events first (the ones that have them). MBR's provide a rectangular area that fully covers a solar event (Figure 1a), while a chain-code provides a pixel outline of the exact solar event area as reported by the FFT modules (Figure 1b).

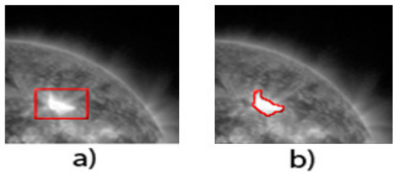


Fig. 1. Solar image with an MBR defined on a), and a chain-code on b)

While FL and SS do not have chain-codes in the original dataset, we have generated them similarly to [4] by using a modified version of the '*fillimage*' property of the `regionprops` function found in Matlab.

B. *Feature Extraction*

As the traditional input of a CBIR system, we extract numerical image parameters to represent our dataset images. Each image is segmented in a 64-by-64 grids. We then calculate ten image parameters from each cell (as shown in figure 2). Previously discussed in [40, 41], many possible parameters were tested on the solar images based on factors such as computational expense and classification accuracy, but it was determined that only the ten parameters selected were sufficient for the task. These image parameters are: entropy, fractal dimension, the mean intensity, the third and fourth moments, relative smoothness, the standard deviation of the intensity, Tamura contrast, Tamura directionality and uniformity. More details as well as their formulas are beyond the scope of this work and can be found in [3].

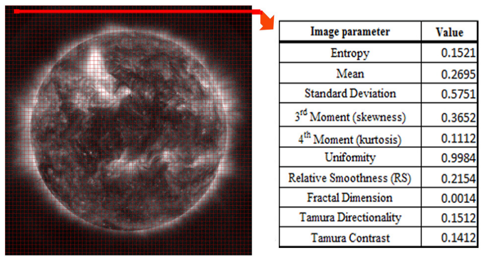


Image parameter	Value
Entropy	0.1521
Mean	0.2695
Standard Deviation	0.5751
3rd Moment (skewness)	0.3652
4th Moment (kurtosis)	0.1112
Uniformity	0.9984
Relative Smoothness (RS)	0.2154
Fractal Dimension	0.0014
Tamura Directionality	0.1512
Tamura Contrast	0.1412

Fig. 2. Example of feature extraction on SDO image cells, from [4]

C. *L2 Retrieval System*

We will compared our proposed system with a Euclidean distance-based system we introduced in [4] for region-based retrieval of solar images. The system retrieval accuracy is: AR: 72%, CH: 85%, FL: 65% and SG: 63%, with an average of 71%. As we mentioned before we are testing under the same software/hardware conditions.

D. *Hardware Utilized*

All of our experiments have been tested on separate virtual machines (VM) running Ubuntu Linux 12.04 with 30 GB's or RAM and using 6 cores of an AMD FX-8150 Black Edition processor. For the L2 system we are using PostgreSQL 9.2 and for the Lucene system we use Apache Lucene version 4.0. We provide the same hardware conditions for each system to run to be able to deliver 1-1 comparisons in terms of performance.

III. BUILDING THE LUCENE INDEX

In this work we attempt a new way of converting our image parameter data into text strings that could be efficiently indexed and searched by Lucene. This section describes how we transform the image parameters into **documents** (**image-row-document**) that are in turn put together as a **document of documents** (**image-document**) that can be search through using Lucene. Please note that our approach while similar in essence to the Visual Bag of Words (VBoW) model, it is different from it in several ways: 1) Each **image-document** represents a 64 element row of each image, not a certain visual descriptor as in the VBoW model. 2) In this paper every **image-document** is set to be unique, another difference between the VBoW. 3) Lastly, Lucene treats the items that indexes as documents (which are split into terms), making it natural for us to divide each image by the number of rows into **image-row-documents** and then calling the complete set an **image- document**. The granularity of this approach is what makes our queries work properly, as we will provide more details about this in the Query building section. The process is described in Algorithm 1.

Algorithm 1 – Create image document:

```
Input: Set of 64x64x10 image parameters extracted from
imagefile (as described in Fig. 2).

1) For every n^th row in image (where n = 1 to 64)
     a. For every j^th feature (j = 1 to 10)
          i. For every i_j^th image parameter in image
             row (where i = 1 to 64)
               1. Concatenate each image
                  parameter as a string with a
                  blank space character between
                  each one.
     b. Once the full row is processed. Generate a
        unique image-row-document id using image number
        concatenated with row number.
2) Once a full image is processed. Create an image
   document id, using the image number. This image
   document will now contain an ordered list of the image-
   row-document ids inside.
```

NOTE: In algorithm 1, all image parameters are concatenated at once. However, additional granularity can be achieved by treating them separately. Something we want to analyze in the future and you can read more about in the Future Work section.

Notice we have two separate sets of documents to index independently: **image-documents** and **image-row-documents**. While this could be handled by two instances of Lucene running on the same server, we have opted for a two server architecture allowing us to upgrade and replace which ever component of the system needs scaling as the number of documents increases. For this paper we used both instances in the same Virtual Machine (VM), so we can provide a 1-1 comparison with the L2-based system. We have not tested this in a two-server scenario, but since we the worst-case scenario in this work (both Lucene instances in one server), the two-server approach should be considerably faster.

IV. EXPERIMENTAL ARCHITECTURE:

As explained in the previous section, we used two Lucene servers each containing an individual document index. In our proposed architecture we used the SDO Retrieval Front-End server (Fig. 3) to translate the user query into **image-row-documents** using the Image Database. Once they are created, we used them to query the Lucene instance containing the image-row-document index. With the list of similar image-row-documents returned, we will now search for matching **image-documents** using the image document index for matching sections. Once we have narrowed down the list to a set of **image-documents**, the image database provides the image names and file location of the query results to the user through the SDO Retrieval Front-End.

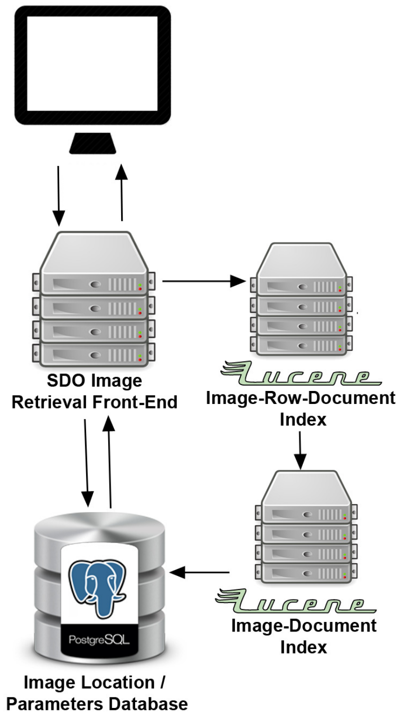


Fig. 3. Proposed system architecture, from user query all the way through results presentation

V. QUERY BUILDING AND CRITERIA

In order to be able to query our image-row-documents and image documents index, in subsection *A* we propose a way of building queries and in subsection *B* we explain the special parameters used for the results filtering.

A. Query construction

In order to mimic the user selecting a particular region of an image to query the repository with, we have opted to use the event labels provided in [38]. We present two different algorithms to construct the queries being sent to **image-row-document** index.

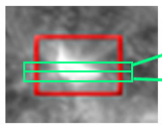
Algorithm 2 – Construct image-row-document query for MBR:

```
Input: Event or user provided label coordinates $(x_1,y_1)$ for
the top left and $(x_2,y_2)$ for the bottom right (Fig. 4).

1) For every n^th row in image (where n = $y_1$ to $y_2$)
     a. For every i^th image parameter in MBR row (where
        i = $x_1$ to $x_2$)
          i. Concatenate each image parameter as a
             string with a blank space character
             between each one.
2) Each set of concatenated strings will be used
   separately as a query.
```

Algorithm 3 – Construct image-row-document query for chain codes:

```
Input: Event or user provided label coordinates as a set of
points ($x_a$, $y_a$),($x_b$, $y_b$) representing each row of the chain-
code area (starting on row a to b).

1) For every n^th set of points (where n = a to b)
        a. Concatenate each image parameter as a string
           with a blank space character between each one.
2) Each set of concatenated strings will be used
   separately as a query.
```

Once we have the sets of returned similar **image-row-documents** we use this set to query the **image-document** index and return similar images.



0.8181 0.3524 0.6151 0.9811 0.6314 .....

0.7413 0.2316 0.7129 0.3321 0.9781 .....

Fig. 4. Generation of **image-row-document** query for MBR sample.

B. *Filtering parameters*

In order to provide the user with images that contain similar events/regions to the ones queried, we made some adjustments to the querying methodology. Each **image-row-document** query is adjusted with wildcards to not only provide an exact match for the query, but also mimic a range query. Lucene supports multiple wildcards per string, allowing us to tweak all image parameters in the same query. Figure 5 shows an original **image-row-document** query (a) and a modified wildcard **image-row-document** query (b). In our previous analysis [4] we rarely found two identical query results (distance between the query to image database equal to zero), so keeping a strict search will not search (non-wildcard) will not be too useful.

...... 0.1587 0.5846 0.8953 0.1239 0.1485 0.2356 0.8745 ....

a)

...... 0.15?7 0.58?6 0.89?3 0.12?9 0.14?5 0.23?6 0.87?5 ....

b)

Fig. 5. Image-row-document query a) original and b) wilcards inserted.

The decimal position where we place the wildcards can be adjusted to limit or increase the number of matches, Fig 5. Show it placed on the third decimal. Our initial experiments will address the tuning of said parameter, as shown in figure 7 that shows the 3rd decimal being the best performing.

In order to determine a match for each image document returned, we analyze the percentage of returned image-row-documents to match by 70, 80, 90% thresholds. Based on experimentation we found that a threshold of 80% achieved the best retrieval results (Fig. 5).

VI. EXPERIMENTS – FILTERING AND ADJUSTED BASELINE

In order to set our filtering parameters, we use as a base-line result the system built for [4], we will call these results: retrieval L2, since they are Euclidean distance based. We also started using no wildcards and a threshold of 100% matches per document. With these very restrictive parameters, the results are not surprising as we can see in figure 6. For all experiments we present the average results of 100 random queries.

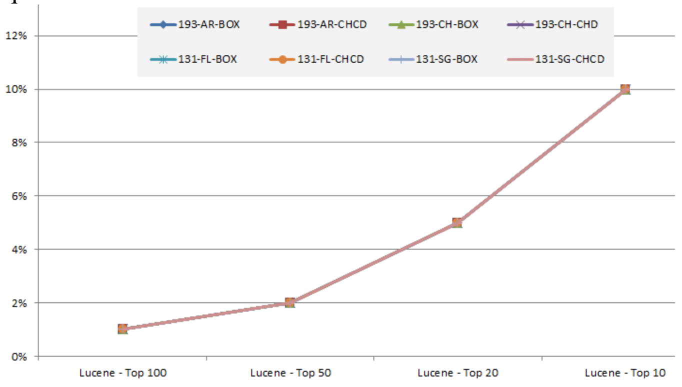
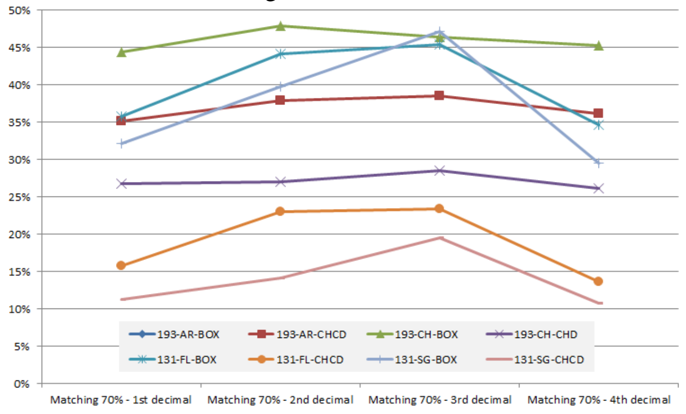
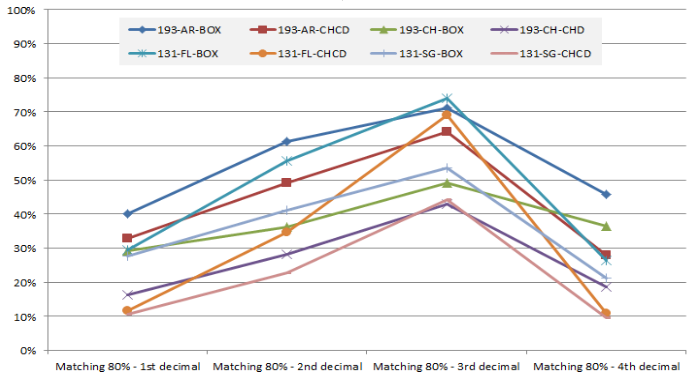


Fig. 6. Strict filtering Lucene retrieval precision results.

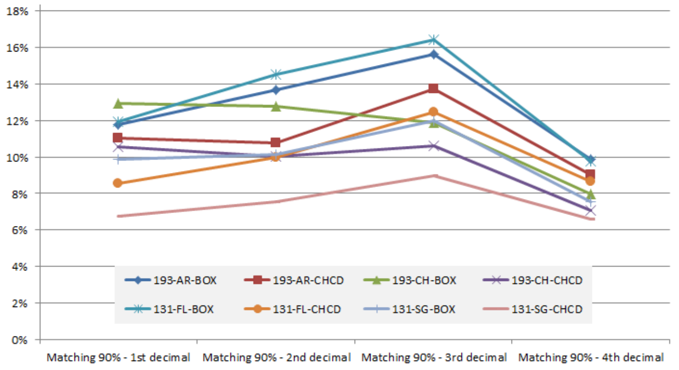
As we suspected, the restricted filtering settings produce only one match, the query, for all scenarios of using the top 100, 50, 20, and 10 results. This shows that not using wildcards and a 100% matching criteria will not produce any useful results due to the granularity level of our image parameter floating point values. In order to find a solid matching criterion, we will introduce the use of wildcards and different matching percentages only for the top 50 scenario that we want to use in our CBIR system, since we want to return initially the top 50 results to the user. For consistency, unless explicitly stated otherwise, we will use top 50 results for Lucene and L2 through this work.



a)



b)



c)

Fig. 7. Lucene retrievalprecision results at: a) 70%, b) 80%, and c) 90%. BOX results use the MBR labels and CHCD results use the chain codes.

Figure 7 clearly shows that the matching documents parameter has to be set to 80%, since that is the only one that produces results with retrieval precision higher than 70%. The other two matching percentage options tested never reach over 25% retrieval precision, a considerably low number. This figure also allows us to see which decimal place to use for the wildcard placement. In 7b), we can see that the best results were achieved by using the wildcard in the 3rd decimal. This result was also very consistent over a) and c).

After these experiments we can justify using a matching criterion of 80% and the third decimal to place the wildcards in the queries. All future experiments, unless explicitly stated, will be using these settings.

VII. Experiments – Comparison against traditional retrieval

Another observation we made after analyzing figure 7 is that in a) and c) we have very evident performance gaps between the usage of MBR's (labeled **BOX**) and chain codes labeled (**CHCD**). We theorized this would be an issue due to the nature of these labels having smaller irregular **image-row-document** segments that might be matched to more **documents**. In Fig. 8 we analyze their performance in terms of query processing time by averaging the performance time with all BOX and CHCD labels and comparing it against the L2-system best results processing best time.

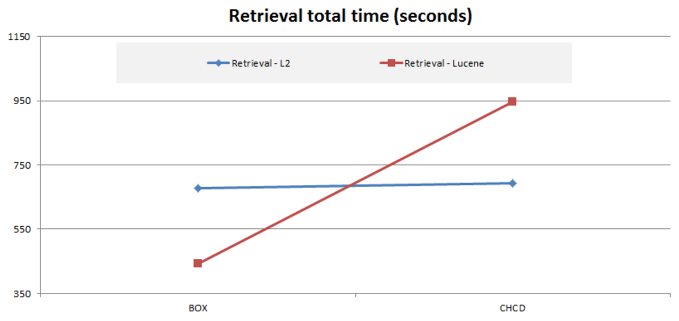


Fig. 8. L2 and Lucene retrieval times comparison between MBR's and chain codes for 1,000 random queries.

Figure 8 shows that there are considerable performance issues when using chain codes bringing the performance to similar levels as a traditional distance-based systems (L2), something that will have an impact when we scale the system to millions of images as we show at the end of this section. We will now drop the chain codes from our analysis and add the other two events (AR and CH) that only have MBR's to our analysis.

In Fig 9, we present retrieval results disregarding the wavelengths and only focusing on effectiveness on retrieving particular events.

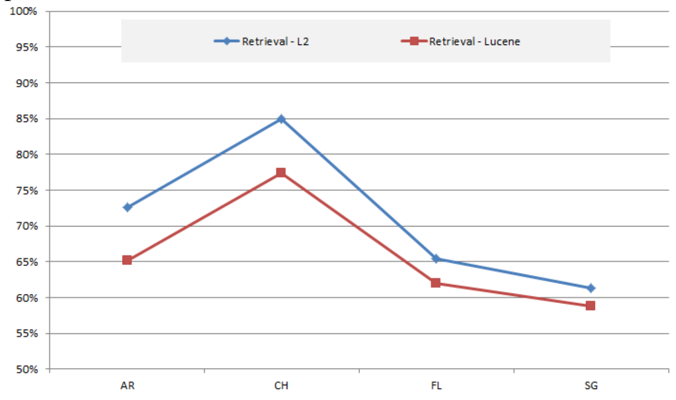


Fig. 9. Baseline L2 and Lucene retrieval precision per event for finiding the top 50 similar results.

While a 2% to 5% difference in performance might not be that considerable, we are looking at Lucene as a scalable solution rather than a 1 to 1 match of retrieval performance. Many parameters and adjustments can be made to Lucene to match and exceed the performance of our L2-based system. In figure 10 we show that only modifying the top *N* results parameter we can achieve some performance gains, getting us closer to the L2 system performance.

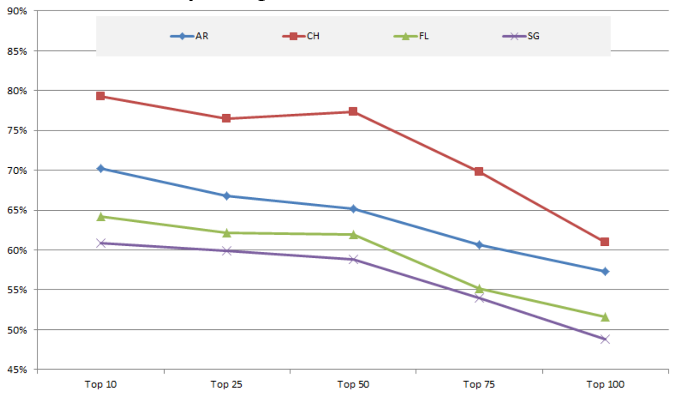


Fig. 10. Lucene retrieval precision comparison for multiple top *N*'s.

As we restrict the number of top *N* documents to 10 we observe that we are able to recover the few percentage points that we lost when changing between the L2 and Lucence systems at top 50 results. This most restrictive condition actually tells us that the first 'similar' documents retrieved are actually more relevant to the query image and class. Consequently, the more results we look at (top 100) we see a considerable drop in performance (almost 20%) since those documents don't seem to be as related to our query.

Overall, we have shown that we can get 'as good' performance from the Lucene retrieval system when compared to our previous L2-based system. We have extensively tested multiple parameters to fine-tune the Lucene system to perform acceptably. However, the advantages of this approach greatly lie on the scalability tests. As we add more documents to our system, the L2 system that depends on on-the-fly distance calculations will suffer. In figure 11 we compare average retrieval times (not performance) of 100 random queries in both the Lucene and L2 systems through 3 different datasets: the original one with 15K images, and the two artificial + original ones that go up to 1.5 and 4.5 million records.

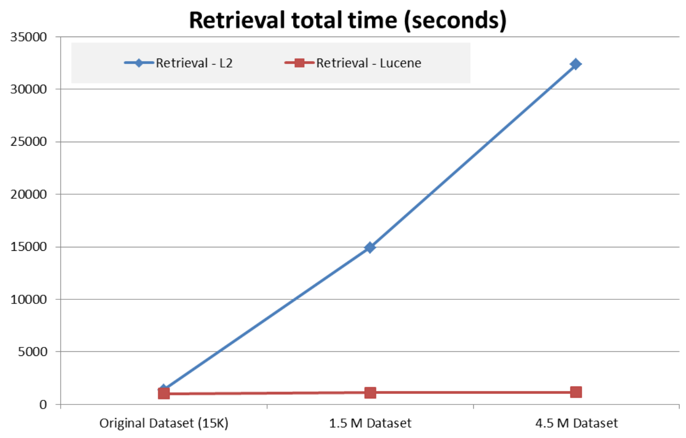


Fig. 11. Retrieval scalability test between L2 and Lucene system.

The scalability test shows immediately why the proposed Lucene approach is better fitted for larger image repositories. We see that for the original dataset the performance difference between Lucene and L2 is not that considerable, but the gap immediately widens as we greatly increase the number of images by 100% and 300% respectively.

VIII. Conclusions

In this work we have presented very strong arguments to demonstrate how the proposed Lucene system is considerably better than our previous L2 system, for Big Data-scale retrieval (Fig. 11). While the retrieval performance oscillates around the same percentages (Fig. 9, 10) between the systems, the retrieval time for the considerably larger datasets (Fig. 11) is the most conclusive analysis to support our claims.

We have demonstrated (again, as in [4]) that the chain codes and their more detailed representation of events/queries, do not really provide any considerable retrieval performance improvements (Fig. 7) and actually degrade performance (Fig. 8). This is quite interesting since once would assume that a more detailed version of the query event would serve better for search, but the results here and on [4] continue to show otherwise.

Our proposed method of building the Lucene indexes (section III) combined with the experimental system architecture (section IV) and the query building and criteria (section V) all integrate together very well in creating a system that is able to handle 4.5 million records indexed and performs the same as one that was built for 15K records, thanks to the advantages of Lucene and its retrieval capabilities. As the results here are encouraging, there are plenty of improvements and things to add to this system to better fine tune it and be able to release it on a production environment.

IX. Future Work

Having presented very promising results for our Lucene based system, there are many modifications that we want to test in order to improve performance and have a more robust system before releasing a public version. Our initial system does not take advantage of Lucene's numeric field data type. This data type has been constantly improved over time to provide range queries of its on, by creating mappings of the numeric values to strings. We are curious to see how the performance of such fields compares and/or improves our existing approach. This data type also comes with several configurable parameters for range query precision that we are very interested in exploring. While we might add some overhead to our system in terms of converting our queries on the fly, we are interested to see how Lucene would be able to handle this natively.

One interesting idea that came about when comparing our approach to the visual bag of words (WBoW) approach is that we might be able to 'cluster' similar words and avoid having unique image-row documents for all images and rater have only one that is used interchangeably through the repository. While this might reduce our index complexity and, we believe finding said 'duplicate' image-row documents beforehand would be computationally very expensive, and this is not a good solution for when new documents are added. Still, we are thinking of ways to implement this and observe what type of performance changes we will see.

Lastly, Lucene in the last few years has been adding Machine Learning tools in their core, such as classifiers, so we are also interested in looking at them for retrieval purposes. Mahout is another project that takes advantage of Lucene to provide Data Mining and Pattern Recognition facilities, and we will be surely exploring in the future. As the people of LIRE [6] before us, we have shown that Lucene is a viable alternative for Solar Image retrieval and we would love to see other contributions built on top of our proposed architecture.

Acknowledgments

This work was supported by two National Aeronautics and Space Administration (NASA) grant awards, 1) No.NNX09AB03G and 2) No. NNX11AM13A..

References

[1] Banda, J., Schuh, M., Angryk, R., Pillai, K.G., McInerney, P.: Big data new frontiers: Mining, search and management of massive repositories of solar image data and solar events. In New Trends in Databases and Information Systems Advances in Intelligent Systems and Computing 241.

[2] http://cbsir.cs.montana.edu/sdocbir/php/index.php

[3] J.M. Banda, R. A. Angryk, P. C. H. Martens. "imageFARMER – Introducing a framework for the creation of large-scale content-based image retrieval systems". International Journal of Computer Applications 79(13):8-13, October 2013. Published by Foundation of Computer Science, New York, USA.

[4] J.M. Banda & R. A. Angryk. “Large-Scale Region-Based Multimedia Retrieval for Solar Images”. In Lecture Notes in Computer Science: Artificial Intelligence and Soft Computing, Volume 8467, 2014, pp. 649-661, Springer International Publishing, ISBN: 978-3-319-07172-5. DOI: 10.1007/978-3-319-07173-2_55

[5] http://lucene.apache.org/

[6] Lux Mathias, Savvas A. Chatzichristofis. Lire: Lucene Image Retrieval – An Extensible Java CBIR Library. In proceedings of the 16th ACM International Conference on Multimedia, pp. 1085-1088, Vancouver, Canada, 2008

[7] Pentland, A., Picard, R., Sclaro, S.: Photobook: Content-based manipulation of image databases. International Journal of Computer Vision 18(3) (1996) 233-254

[8] Ogle, V., Stonebraker, M.: Chabot: Retrieval from a relational database of images.Computer (1995) 40-48.

[9] Kelly, P., Cannon, T., Hush, D.: Query by image example: the comparison algorithm for navigating digital image databases approach. Storage and Retrieval for Image and Video Databases (1995) 238-248.

[10] Flickner, M., Sawhney, H.t.: Query by image and video content: The qbic system. Computer 28(9) (1995) 23-32.

[11] J. M. Banda, R. Angryk, P. C. H. Martens. “On dimensionality reduction for indexing and retrieval of large-scale solar image data”. Solar Physics: Image processing in the petabyte era. Springer 2013. Online: DOI:10.1007/s11207-012-0027-4, in print: Volume 283, Issue 1, pp. 113-141.

[12] J.M. Banda, R. Angryk, P. Martens. “On the surprisingly accurate transfer of image parameters between medical and solar images”, ICIP-IEEE '11, Brussels, Belgium, September 2011, pp. 3730-3733.

[13] J. A. Piedra-Fernandez, Gloria Ortega, James Z. Wang and M. Canton-Garbin, "Fuzzy Content-Based Image Retrieval for Oceanic Remote Sensing," IEEE Transactions on Geoscience and Remote Sensing, vol. 52, no. 9, pp. 5422-5431, 2014.

[14] Kamel A, Mahdi YB, Hussain KF (2013) Multi-bin search: improved large-scale content-based image retrieval. In: 20th International Conference on Image Processing (ICIP), IEEE, pp 2597–2601. doi:10.1109/ICIP.2013.6738535

[15] Kumar, M. Sasi, and Y. S. Kumaraswamy. "A Boosting Frame Work for Improved Content Based Image Retrieval." Indian Journal of Science and Technology 6.4 (2013): 4312-4316.

[16] Hare, J.S., Samangooei, S., Dupplaw, D.P.: OpenIMAJ and ImageTerrier: Java libraries and tools for scalable multimedia analysis and indexing of images. In: Proceedings of the 19th ACM MM, MM 2011, pp. 691–694. ACM, New York (2011).

[17] James Z. Wang, Jia Li and Gio Wiederhold, ``SIMPLIcity: Semantics-Sensitive Integrated Matching for Picture Libraries,'' IEEE Transactions on Pattern Analysis and Machine Intelligence, vol. 23, no. 9, pp. 947-963, 2001.

[18] Carson, C., Thomas, M., et al.: Blobworld: A system for region-based image indexing and retrieval. Visual Information and Information Systems (1999) 509-517.

[19] Wei-Ying Ma, B.S. Manjunath. “NeTra: A toolbox for navigating large image databases”. Multimedia Systems, May 1999, Volume 7, Issue 3, pp 184-198.

[20] Apostol Natsev, Rajeev Rastogi, and Kyuseok Shim. 1999. WALRUS: a similarity retrieval algorithm for image databases. In Proceedings of the 1999 ACM SIGMOD international conference on Management of data (SIGMOD '99). ACM, New York, NY, USA, 395-406. DOI=10.1145/304182.304217

[21] Jing, Feng, Li, Mingjing, Zhang, Lei, Zhang, Hong-Jiang, Zhang, Bo. “Learning in Region-Based Image Retrieval”. Image and Video Retrieval - Lecture Notes in Computer Science, Volume 2728, pp 205-215. Springer Berlin Heidelberg, 2003.

[22] Jing, Feng, Li, Mingjing, Zhang, Lei, Zhang, Hong-Jiang, Zhang, Bo. “An efficient and effective region-based image retrieval framework.” In IEEE Trans Image Process. 2004 May;13(5):699-709.

[23] Jing, Feng; Mingjing Li; Hong-Jiang Zhang; Bo Zhang, "Relevance feedback in region-based image retrieval," Circuits and Systems for Video Technology, IEEE Transactions on , vol.14, no.5, pp.672-681, May 2004.

[24] Jia Li, James Z. Wang, and Gio Wiederhold. 2000. IRM: integrated region matching for image retrieval. In Proceedings of the eighth ACM international conference on Multimedia (MULTIMEDIA '00). ACM, New York, NY, USA, 147-156. DOI=10.1145/354384.354452

[25] Huang, Wei; Gao, Yan; Chan, Kap Luk "A Review of Region-Based Image Retrieval", Journal of Signal Processing Systems, Volume.59, Issue.2, pp.143, 2010, ISSN: 19398018.

[26] Nishant Shrivastava, Vipin Tyagi, “Content based image retrieval based on relative locations of multiple regions of interest using selective regions matching”, Information Sciences, Volume 259, 20 February 2014, Pages 212-224, ISSN 0020-0255.

[27] Zakariya, S.M.; Ali, R.; Ahmad, N. "Combining visual features of an image at different precision value of unsupervised content based image retrieval", Computational Intelligence and Computing Research (ICCIC), 2010 IEEE International Conference on, On page(s): 1 – 4.

[28] Amory, A.A.; Sammouda, R.; Mathkour, H.; Jomaa, R.M. "A content based image retrieval using K-means algorithm", Digital Information Management (ICDIM), 2012 Seventh International Conference on, On page(s): 221 – 225.

[29] Rangkuti, A.H.; Hakiem, N.; Bahaweres, R.B.; Harjoko, A.; Putro, A.E. "Analysis of image similarity with CBIR concept using wavelet transform and threshold algorithm", Computers & Informatics (ISCI), 2013 IEEE Symposium on, On page(s): 122 – 127.

[30] Elumalaivasan, P.; Suthir, S.; Ravikumar, S.; Pandiyaraju, V.; Munirathinam, T. "CBIR: Retrieval of similar images using median vector algorithm", Green Computing, Communication and Conservation of Energy (ICGCE), 2013 International Conference on, On page(s): 1 – 5

[31] Amores, J.; Sebe, N.; Radeva, P. "Context-Based Object-Class Recognition and Retrieval by Generalized Correlograms", Pattern Analysis and Machine Intelligence, IEEE Transactions on, On page(s): 1818 - 1833 Volume: 29, Issue: 10, Oct. 2007

[32] Hui Wei; Xiao-Mei Wang; Loi Lei Lai "Compact Image Representation Model Based on Both nCRF and Reverse Control Mechanisms", Neural Networks and Learning Systems, IEEE Transactions on, On page(s): 150 - 162 Volume: 23, Issue: 1, Jan. 2012

[33] Chang-Yong Ri; Min Yao "Semantic Image Segmentation Based on Spatial Context Relations", Information Science and Engineering (ISISE), 2012 International Symposium on, On page(s): 104 – 108

[34] Helala, Mohamed A.; Selim, Mazen M.; Zayed, Hala H. “A Content Based Image Retrieval Approach Based On Principal Regions Detection”. International Journal of Computer Science Issues (IJCSI);Jul2012, Vol. 9 Issue 4, pp204.

[35] Venkatraman, S.; Kulkarni, S. "MapReduce neural network framework for efficient content based image retrieval from large datasets in the cloud", Hybrid Intelligent Systems (HIS), 2012 12th International Conference on, On page(s): 63 – 68.

[36] Gu, C., Gao, Y.: A Content-Based Image Retrieval System Based on Hadoop and Lucene. In: Cloud and Green Computing (CGC), November 1-3, pp. 684–687 (2012).

[37] Mathias Lux, Glenn Macstravic. “The LIRE Request Handler: A Solr Plug-In for Large Scale Content Based Image Retrieval”. MultiMedia Modeling - Lecture Notes in Computer Science Volume 8326, 2014, pp 374-377.

[38] Schuh, M., Angryk, R., Pillai, K., Banda, J., Martens, P.: A large-scale solar image dataset with labeled event regions. 20th IEEE Int. Conf. on Image Processing (ICIP) (2013) 4349-4353.

[39] Martens, P., Attrill, G., Davey, A., Engell, et. al : Computer vision for the solar dynamics observatory (sdo). Solar Physics 275 (2012) 79-113.

[40] Banda, J., Angryk, R.: An experimental evaluation of popular image parameters for monochromatic solar image categorization. 23rd Inter. FLAIRS Conf. (2010) 380-385.

[41] Banda, J., Angryk, R.: On the effectiveness of fuzzy clustering as a data discretization technique for large-scale classification of solar images. Proceedings of the 18th IEEE International Conference on Fuzzy System (2009) 2019-2024.