Sun Magnetograms Retrieval From Vast Collections Through Small Hash Codes *

Rafał Grycuk[1][0000−0002−3097−985X] and Rafał Scherer[1][0000−0001−9592−262X]

Czestochowa University of Technology, al. Armii Krajowej 36, Czestochowa, Poland
{rafal.grycuk,rafal.scherer}@pcz.pl

Abstract. We propose a method for retrieving solar magnetograms based on their content. We leverage data collected by the SDO Helioseismic and Magnetic Imager using the SunPy and PyTorch libraries to create a vector-based mathematical representation of the Sun's magnetic field regions. This approach enables us to compare short vectors instead of comparing full-disk images. To reduce retrieval time, we use a fully-connected autoencoder to compress the 144-element descriptor to a 36-element semantic hash. Our experimental results demonstrate the efficiency of our approach, which achieved the highest precision value compared to other state-of-the-art methods. Our proposed method is not only applicable for solar image retrieval but also for classification tasks.

Keywords: Fast image hash · Solar activity analysis · Solar image description · CBIR of solar images · Magnetogram Image Descriptor · Magnetogram Image Hash.

1 Introduction

Analysing the Sun is crucial for understanding many aspects of our solar system and the universe. By analysing the Sun's internal structure and surface features, we can better understand how the Sun generates and releases energy through nuclear fusion. This information is essential for predicting and understanding the Sun's behaviour, such as the occurrence of solar flares, coronal mass ejections, and other space weather phenomena that can affect Earth and the space environment.

Solar activity can cause disturbances in Earth's magnetic field, leading to geomagnetic storms that can affect power grids, satellites, and communication systems. By monitoring the Sun's activity and predicting space weather, we can take measures to mitigate these effects. By studying the Sun's properties, researchers can better understand the range of conditions that can support life in other star systems.

* The project financed under the program of the Polish Minister of Science and Higher Education under the name "Regional Initiative of Excellence" in the years 2019–2023 project number 020/RID/2018/19, the amount of financing 12,000,000.00 PLN.

ICCS Camera Ready Version 2023
To cite this paper please use the final published version:
DOI: 10.1007/978-3-031-36027-5_19

NASA's Living With a Star (LWS) Program has a mission to explore how solar activity affects Earth, and the Solar Dynamics Observatory (SDO) is a crucial component of this endeavour. The SDO provides extensive data on the solar atmosphere at different wavelengths and with high spatial and temporal resolution, allowing for thorough analysis of the Sun's impact on our planet. One of the key instruments on the SDO is the Helioseismic and Magnetic Imager (HMI), which specializes in examining oscillations and the magnetic field of the solar surface. By generating dopplergrams, continuum filtergrams, and magnetograms (which are maps of the photospheric magnetic field), the HMI enables researchers to obtain valuable insights into the Sun's behaviour.

The magnetograms created by the HMI are particularly important to researchers. However, with the sheer volume of data generated by the SDO spacecraft, it is impossible to manually annotate and search through the entire collection. Although some image retrieval methods exist, they are primarily designed for real-life images and do not suit the needs of solar research. Instead, researchers have turned to semantic hashing to reduce dimensionality by creating short codes that preserve similarity and reflect the content of the input data. This approach was initially introduced by Salakhutdinov and Hinton [17] and has since been used to describe any short codes that reflect content-similarity.

The goal of semantic hashing is to produce compact vectors that accurately reflect the semantic content of objects. This approach enables the retrieval of similar objects through the search for similar hashes, which is a faster and more memory-efficient process than working directly with the objects themselves. Previous works has used multilayer neural networks to generate hashes. Recently, learned semantic hashes have become popular for image retrieval, as demonstrated by research such as that by [18].

Initially, we found out that generating hashes from full-disk solar images would be impractical due to the large size of the image collections in terms of resolution and quantity. As a result, we developed the hand-crafted intermediate descriptors discussed earlier.

A full-disk content-based image retrieval system is described in [1]. The authors checked eighteen image similarity measures with various image features resulting in one hundred and eighty combinations. The experiments shed light on what metrics are suitable for comparing solar images to retrieve or classify various phenomena.

In [3], a general-purpose retrieval engine called Lucene is utilized to retrieve solar images. Each image is treated as a document with 64 elements (representing the rows of each image), and each image-document is unique. Wild-card characters are used in query strings to search for similar solar events. While the Lucene engine is compared to distance-based image retrieval methods in [4], no clear winner emerged. Each tested method has its advantages and disadvantages in terms of accuracy, speed, and applicability. There is a significant trade-off between accuracy and speed, with retrieval times of several minutes required for accurate results.

ICCS Camera Ready Version 2023
To cite this paper please use the final published version:
DOI: 10.1007/978-3-031-36027-5_19

In [10], a sparse model representation of solar images is presented. The method utilized the sparse representation technique proposed in [14] and showed superior performance in both accuracy and speed compared to previous solar image retrieval approaches. The authors of [12] focused on tracking multiple solar events across images with 6-minute cadence by selecting specific solar image parameters. Additionally, sparse codes for AIA images were used in [11], where ten texture-based image parameters were computed for regions determined by a 64×64 grid for nine wavelengths. A dictionary of k elements was learned for each wavelength, and a sparse representation was then computed using the learned dictionary. In [13], a new method for image retrieval using fuzzy sets and boosting is proposed. To overcome the curse of dimensionality that affects solar data, the authors use the Minkowski norm and carefully choose the appropriate value for the p parameter. They also employ a 256-dimensional descriptor, which has been shown to be both efficient and accurate compared to previous approaches.

In this paper, we propose a method for automating solar image retrieval and enabling their fast classification using a solar image hash generated from one-dimensional hand-crafted features by a fully-connected autoencoder. The resulting hash is very compact, consisting of only 36 real-valued elements, but experiments have shown that this is sufficient to accurately describe the images. In the dataset used, the images are annotated only by their timestamp, making it difficult to extract any other meaning or interpret the trained system [15]. The timestamp is treated as a measure of similarity, and after training, the algorithm allows retrieval of images by their visual similarity, regardless of the timestamp proximity. The paper is organized into three sections: the proposed method for generating learned solar hashes is introduced in Section 2; experiments on the SDO solar image collection are described in Section 3; and finally, the paper concludes with Section 4.

2 Solar Magnetic Intensity Hash for Solar Image Retrieval

The Solar Dynamics Observatory's (SDO) instruments are not only the Atmospheric Imaging Assembly (AIA) but also Helioseismic and Magnetic Imager (HMI), which allows for creation of magnetograms of the Sun. In active regions, the magnetic field can be significantly stronger than the average magnetic field of the Sun, with some regions having magnetic fields over 1,000 times stronger. By providing information about the magnetic fields of the entire solar disk, magnetograms are useful in many areas of solar analysis. Taking into consideration the noise present in regular active region images due to bright pixels that represent flares extending beyond the solar disk, the utilization of magnetograms in solar image description or solar image hashing seems like a viable solution to enhance the precision of the hash. By providing information about the magnetic fields of the entire solar disk, magnetograms can effectively reduce the unwanted noise in solar images. As such, using magnetograms to analyze the Sun's activity (as shown in Fig. 1) appears to be a more justifiable approach.

ICCS Camera Ready Version 2023
To cite this paper please use the final published version:
DOI: 10.1007/978-3-031-36027-5_19

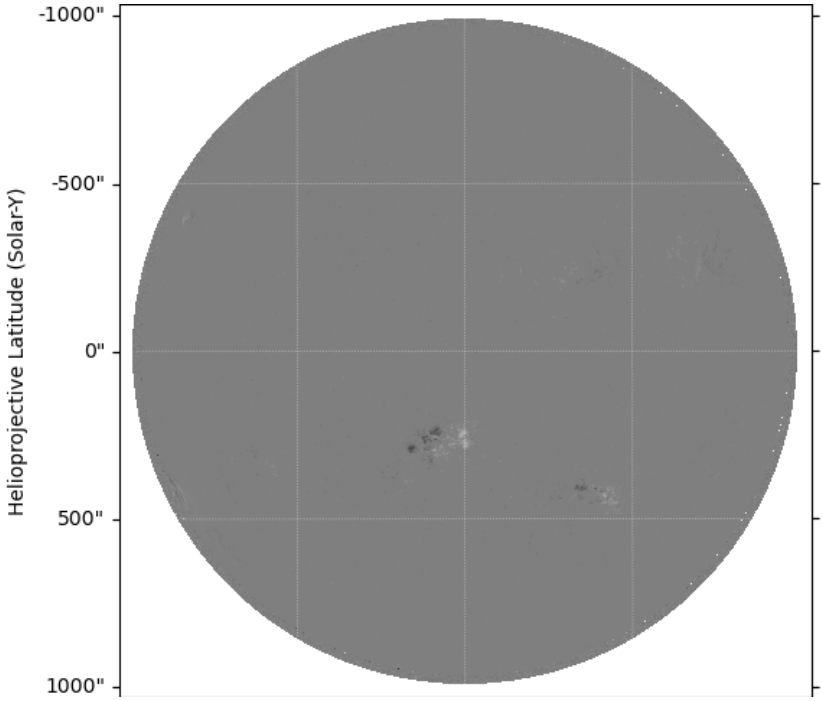

Fig. 1. Example magnetogram image. As can be seen, the image is difficult to analyze without any pre-processing.

Solar images are highly similar to each other, and using general-purpose descriptors for retrieval may not yield accurate results. To address this issue, we propose a novel solar magnetogram hash for solar image retrieval in large datasets. Our experimental setup includes a GeForce RTX 2080 Ti 11GB GDDR6 graphics card, which allows us to utilize 11 GB of memory. Initially, we attempted to design a full-disc autoencoder, but the higher mini-batch values caused out-of-memory exceptions, and the learning time was several days compared to the minutes required in our proposed approach. Therefore, we developed a preprocessing stage to calculate the solar magnetogram descriptor, which was then used to reduce the hand-crafted vectors to x-element real-valued hashes using the autoencoder (see Sec. 2.3). This approach preserves the significant information about active regions while reducing the dimensionality of the data for efficient retrieval. The presented algorithm is composed of four main stages: active region detection, calculating solar image hand-crafted descriptors, encoding to hash, and retrieval.

2.1 Magnetic Region Detection

In this section, we describe the fundamental steps involved in the hashing process. The first step entails adjusting the magnetogram image to more clearly an-

ICCS Camera Ready Version 2023
To cite this paper please use the final published version:
DOI: 10.1007/978-3-031-36027-5_19

notate the magnetic regions, as illustrated in Figure 2. We refer to this process as magnetic region detection, which we conduct by utilizing magnetogram images obtained via the SunPy library [19, 20]. This step enables us to determine the intensity of the magnetic field. Figure 1 and Figure 2 both depict the increase

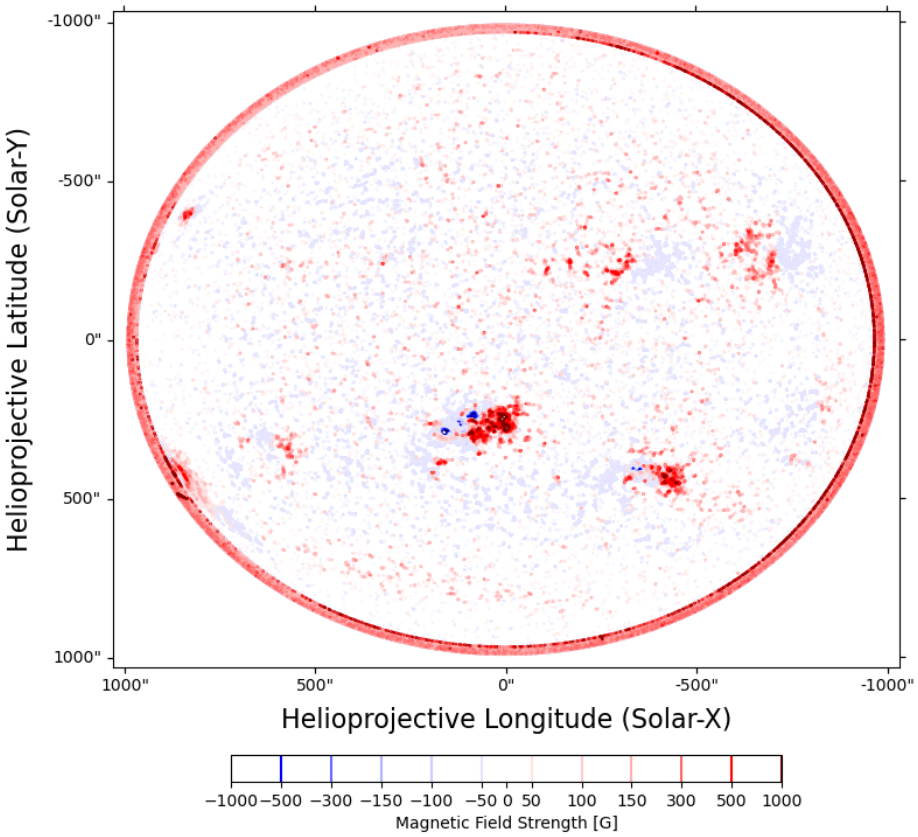

Fig. 2. The magnetic region detection and annotation process. The magnetic regions can be clearly visible. We can observe the polarities (red and blue) and their intensities.

in magnetic field strength around active regions. To define the magnetic field strength, we utilize color intensities, as illustrated in Figure 3. Throughout the solar cycle, the magnetic field undergoes twisting, tangling, and reorganization. It is important to note that magnetic regions (MR) have a strong correlation with Coronal Mass Ejections (CMEs) and solar flares, which makes analyzing them critical for understanding the impact on life on Earth. As illustrated in Figure 3, magnetic region detection (MRD) enables us to determine the north (red) or south (blue) polarities, between which CMEs are most likely to originate. Additionally, tracking and analyzing MRs is valuable for predicting solar flares.

ICCS Camera Ready Version 2023
To cite this paper please use the final published version:
DOI: 10.1007/978-3-031-36027-5_19

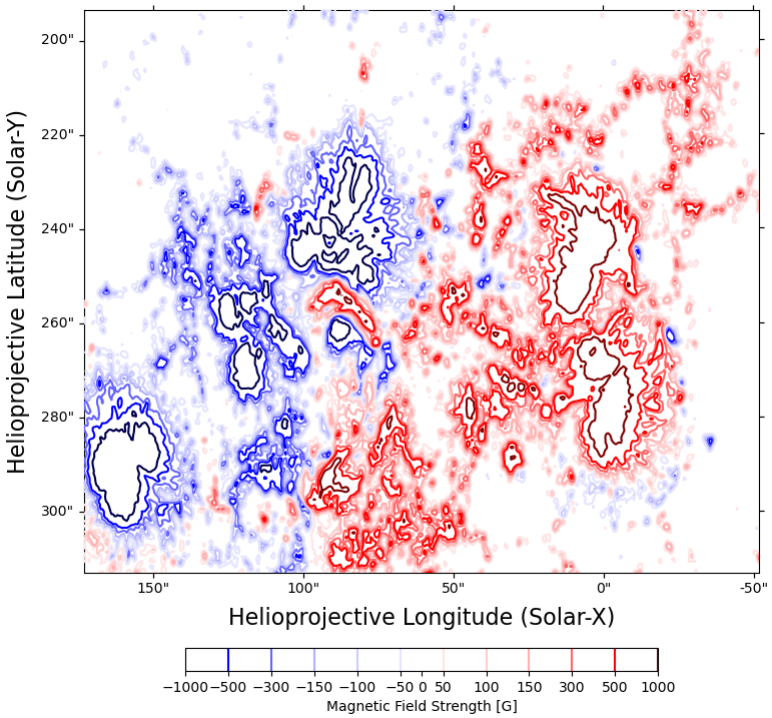

Fig. 3. A magnification of magnetic regions.

By performing magnetic region detection, we can generate a magnetic region intensity image (MRII) that is used in the subsequent steps of the algorithm.

2.2 Calculation of Solar Magnetic Intensity Descriptor

This section describes a method for calculating a magnetic intensity descriptor (MID). After detecting the magnetic fields, as presented in Sec. 2.1, we need to create a mathematical representation of the magnetic field distribution. Comparing high-resolution images is not efficient, therefore we propose a novel approach to represent the Magnetic Region Intensity Image (MRII). We slice the MRII like a pizza, and for each slice, we calculate a magnetic field histogram. The details of the method are as follows. First, we need to set the coordinates of the image center, denoted as *cc*. Fortunately, the radius r is fixed due to the Sun's fixed position on the image. Then, we determine the θ angle empirically and found that 30° provides optimal results. Next, we perform a cropping operation on the obtained slices using the pseudo-code in Alg. 1. We calculate the arc points of the slice (sector) *aps* and *ape* using the following formulas:

$$ape_x = cc_x - 1.5 * r * \sin\theta, \tag{1}$$

ICCS Camera Ready Version 2023
To cite this paper please use the final published version:
DOI: 10.1007/978-3-031-36027-5_19

$$ape_y = cc_y - 1.5 * r * \cos\theta, \tag{2}$$

The trigonometric functions sin and cos are used in these formulas to calculate the row and column coordinates of two points on the arc. A factor of 1.5 is applied to extend the arc beyond the circle's radius slightly. After dividing the MRII into

INPUT: $MRII$ - magnetic region intensity image
r - radius
cc - center coordinates of MRII
θ - angle of the slice
Local Variables:
MC - mask circle matrix
$MMRII$ - mask MRII matrix
ape - coordinates of starting point on the arc
OUTPUT: $CMRII$ - cropped slice of MRII
$MS := CreateBooleanCircleMatrix(cc, r)$
$MMRII := CreatePolygonMatrix([cc_x, aps_x, ape_x, cc_x],$
$[cc_y, aps_y, ape_y, cc_y])$
$CMRII := CombineMasks(MS, MMRII)$

Algorithm 1: Algorithm for cropping the MRII slice.

slices as described in the previous section, the process of cropping is repeated for each subsequent circle segment (slice) until the entire image is covered. This results in a list of MRII slices, each containing the magnetic field intensities for that particular segment. The next step is to create a magnetic field histogram (MFH) for each slice, which allows us to represent the distribution of magnetic fields for each segment of the MRII. The histogram is created with the same scale as the magnetic field intensities, ranging from [-1000;1000] accordingly to the magnetic field range presented in Fig. 3. Finally, all histograms are combined into a single vector called the Magnetic Intensity Descriptor (MID). This vector represents the overall magnetic field distribution for the entire MRII image. The entire process, including cropping and histogram creation, is illustrated in Figure 4 and described in Algorithm 2.

The proposed method enables the generation of a hand-crafted hash for a magnetogram input image, called the magnetic intensity descriptor (MID). By setting θ to 30 deg, the resulting MID vector consists of 144 integer-valued elements, which is significantly smaller than the full-disc image.

2.3 Hash Generation

This section outlines the hash generation process, which takes as input a Solar Magnetic Intensity Descriptor (MID) that is later utilized to generate the corresponding hash. The goal of this step is to obtain a representative hash that describes the solar image and, more specifically, the magnetic regions of the Sun at a given timestamp. This step is crucial because it enables the reduction

ICCS Camera Ready Version 2023
To cite this paper please use the final published version:
DOI: 10.1007/978-3-031-36027-5_19

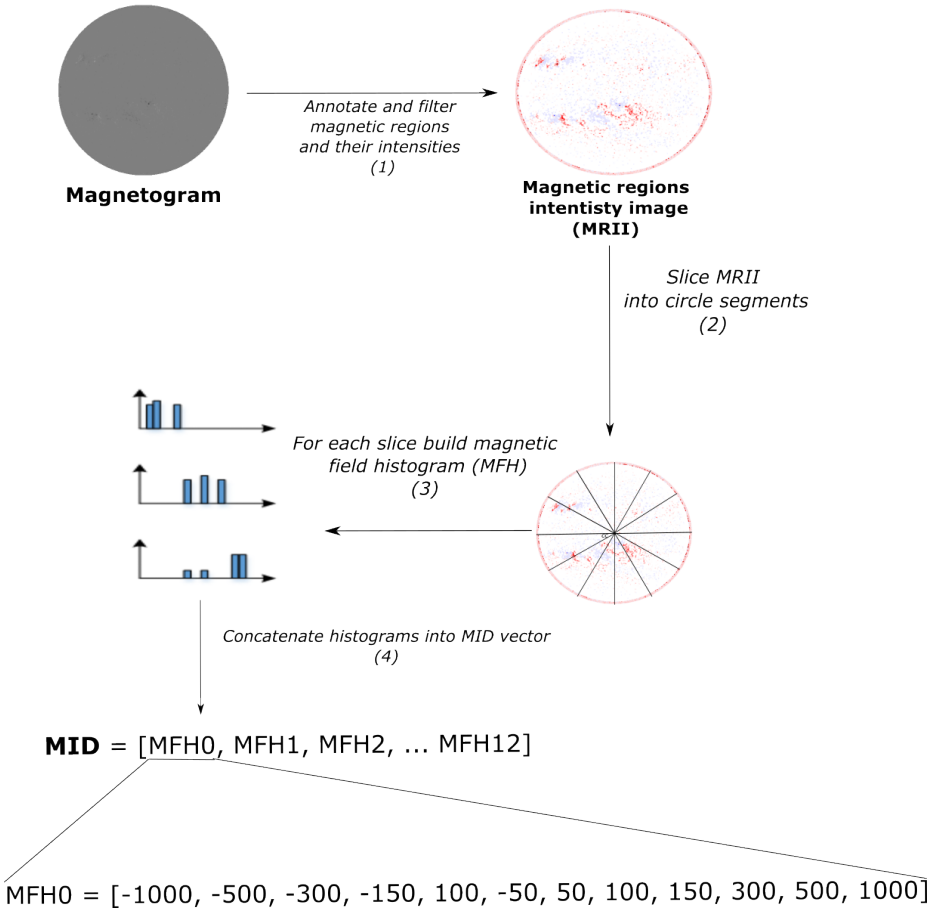

Fig. 4. Steps for calculating the magnetic intensity descriptor (MID).

of data in the retrieval stage (see Section 2.4). To perform this operation, we utilized a fully-connected autoencoder (AE) to encode the previously acquired MID. Autoencoders are utilized in various machine learning tasks, such as image compression, dimensionality reduction, feature extraction, and image reconstruction [5, 7, 16]. Since autoencoders use unsupervised learning, they are ideal for generating semantic hashes. We present the autoencoder model architecture in Table 1. The AE model should be analysed from top to bottom. As shown, the model is relatively simple but enables the reduction of hash length without a significant loss of information about magnetic regions of the magnetogram. We would like to emphasize that only the latent space (encoded) part of the trained AE is used for hash generation, while the decoding part of AE is solely used for training purposes. Through a series of experiments, we determined that 40

ICCS Camera Ready Version 2023
To cite this paper please use the final published version:
DOI: 10.1007/978-3-031-36027-5_19

INPUT: MI - magnetogram image
Local Variables:
$MRII$ - magnetic region intensity image
$SliceList$ - list of slices
$HistogramList$ - list of histograms
OUTPUT: MID - magnetic Intensity descriptor vector
$MRII := MagneticRegionDetection(MI)$
$SliceList := CroppingTheSlices(MRII)$
foreach $MRIISlice \in SliceList$ **do**
  $MFH := BuildSliceIntanceHist(MRIISlice)$
  $HistogramList.Add(MFH)$
end
$MID = ConcatHist(HistogramList)$

Algorithm 2: Algorithm for calculating a magnetic intensity descriptor.

Table 1. Tabular representation of the fully-connected autoencoder model.

Layer (type)	Output	Filters (in, out)	Params
$Input(InputLayer)$	[1, 144]		0
$Linear\_1(Linear)$	[1, 72]	144, 72	10440
$ReLU\_1$	[1, 72]	0	
$Linear\_2(Linear)$	[1, 72]	72, 36	2628
$ReLU\_2$	[1, 36]	0	
$Encoded(latent-space)$	[1, 36]		
$Linear\_4(Linear)$	[1, 36]	36, 72	2664
$ReLU\_4$	[1, 72]	0	
$Linear\_5(Linear)$	[1, 144]	72, 144	10,512
$ReLU\_5$	[1, 144]	0	
$Decoded(Tanh)$	[1, 144]		

epochs are sufficient to achieve a satisfactory level of generalization without the overfitting phenomenon.

Table 1 illustrates the use of a convolutional autoencoder for generating hashes, with the top layer serving as input. A one-dimensional autoencoder was employed because magnetic intensity descriptors are one-dimensional vectors, which reduces computational complexity. This process effectively shortens the hash length while retaining significant information about the active regions of the solar image. The mean squared error function was utilized as the loss function, and we discovered that training the model for 40 epochs was adequate for achieving the necessary level of generalization and preventing network over-fitting. Following the training process, each image descriptor was fed into the latent space (encoded) layers of the autoencoder, yielding a 36-element hash (known as the Solar Magnetic Intensity Hash). This hash can be used for content-based retrieval applications involving solar images. Moreover, the chosen autoencoder architecture was specifically chosen to achieve optimal generalization.

ICCS Camera Ready Version 2023
To cite this paper please use the final published version:
DOI: 10.1007/978-3-031-36027-5_19

2.4 Retrieval

In the final phase of the proposed method, we employ the previously generated hashes for solar image retrieval. Following the preceding steps, we assume that each solar image in our database has an assigned hash. The retrieval process entails executing an image query by comparing the distances between the hash of the query image and the hashes created for all images stored in the dataset. To perform this retrieval, we must have a database of solar images that have undergone hash generation. In the subsequent step, we compute the distance (d) between the query image hash and every hash in the database. The cosine distance measure is utilized for this purpose (see [9] for additional information).

$$\cos(QH_j, IH_j) = \sum_{j=0}^{n} \frac{(QH_j \bullet IH_j)}{\|QH_j\| \, \|IH_j\|}, \tag{3}$$

where $\bullet$ is dot product, QH_j is the query image hash, and IH_j a consecutive image hash. Upon computing the cosine distance, the images in the database are sorted in ascending order by their distance to the query (query hash). The final step of the presented technique enables the retrieval of n images closest to the query, which are then returned to the user. To execute the query, the user must provide the value of the parameter n. Alg. 3 illustrates the complete process as pseudo-code. An alternative method involves retrieving images based on a threshold. To use this approach, a threshold parameter must be supplied instead of n, and images are retrieved if their cosine distance to the query is below the threshold. The proposed method can also accommodate this method; however, the first method is preferable because it is better suited for system users.

INPUT: $ImageHashes$, $QueryImage$, n
OUTPUT: $RetrievedImages$
foreach $ImageHash \in ImageHashes$ **do**
  $QueryImageHash = CalculateHash(QueryImage)$
  $D[i] = Cos(QueryImageHash, ImageHash)$
end
$SortedDistances = SortAscending(D)$
$RetrievedImages = TakeFirst(n)$

Algorithm 3: Image retrieval steps.

3 Experimental Results

This section presents simulation results and a solution for evaluating unlabelled images. Since there was a lack of labelled data, unsupervised learning was employed for encoding descriptors. Consequently, evaluating the proposed method

ICCS Camera Ready Version 2023
To cite this paper please use the final published version:
DOI: 10.1007/978-3-031-36027-5_19

against state-of-the-art approaches was challenging. To overcome this problem, we leveraged the Sun's rotation movement to identify a set of similar images (SI). We hypothesized that consecutive images captured within a small time window would exhibit similar active regions, albeit with slight shifts. The provided solar images at 6-minute intervals, which we could assume were similar due to the nature of the Sun's movement. The only requirement was to adjust the time window difference. Through experimentation, we determined that images captured within a 48-hour window could be regarded as similar. Let us take under consideration an image taken at 2015-10-15, 00:00:00. Based on the above assumptions, we can assume that every image in 24 hours before and in 24 hours after is similar. Only for evaluation purposes, images are identified by the timestamps. The process of determining similar images is presented in Table 2. We conducted a series of experiments to determine image similarity using the

Table 2. Defining image similarity. Based on experiments, we determined that images within a 48-hour window can be treated as similar. This allows to evaluate the method.

Timestamp	SI (similar image)/ NSI (not similar image)
2015-10-13, 23:54:00	NSI
2015-10-14, 00:00:00	SI
2015-10-14, 00:06:00	SI
2015-10-14, 00:12:00	SI
2015-10-14, 00:18:00	SI
2015-10-14, 00:24:00	SI
2015-10-14, 00:30:00	SI
.....	SI
2015-10-15, 00:00:00	QI (query image)
.....	SI
2015-10-15, 23:24:00	SI
2015-10-15, 23:30:00	SI
2015-10-15, 23:36:00	SI
2015-10-15, 23:42:00	SI
2015-10-15, 23:48:00	SI
2015-10-15, 23:54:00	SI
2015-10-16, 00:00:00	NSI

proposed method. A single experiment consisted of the following steps:

1. Executing an image query to retrieve images.
2. Comparing the timestamp of each retrieved image with the query image timestamp.
3. If the timestamp fell within a 48-hour window, the image was deemed similar to the query.

After defining similar images (SI), we can define performance measures *precision* and *recall* [6] [21] based on following sets:

ICCS Camera Ready Version 2023
To cite this paper please use the final published version:
DOI: 10.1007/978-3-031-36027-5_19

- SI - set of similar images,
- RI - set of retrieved images for query,
- $PRI(TP)$ - set of positive retrieved images (true positive),
- $FPRI(FP)$ - false positive retrieved images (false positive),
- $PNRI(FN)$ - positive, not retrieved images,
- $FNRI(TN)$ - false, not retrieved images (TN).

Afterwards, we can define *precision* and *recall* for CBIR systems

$$precision = \frac{|PRI|}{|PRI + FPRI|}, \tag{4}$$

$$recall = \frac{|PRI|}{|PRI + PNRI|}. \tag{5}$$

$$F_1 = 2 * \frac{precision * recall}{precision + recall}. \tag{6}$$

We present the results of our experiments in Table 3, which demonstrate the effectiveness of our method. The experimental outcomes presented in Tab. 3 exhibit promise, as denoted by the mean F_1 score and the high precision values. Our approach yielded an average precision of 0.92581, surpassing the results of Banda [4] (0.848) and Angryk [2] (0.850). Additionally, we outperformed the previous works by Grycuk [8]. Our approach achieves a high value of the precision measure, indicating that most of the images close to the query are correctly retrieved. However, as images move farther from the query, more positive but not retrieved images (PNRI) are retrieved. This phenomenon is due to the Sun's rotation, which results in missing active regions between consecutive images. In the 48-hour cadence, significant active regions may change their position, leading to a significant impact on the hash and an increased distance from the query. We implemented the simulation environment in Python using PyTorch and ran it on hardware consisting of an Intel Core i9-9900k 3.6 GHz processor, 32 GB of RAM, and a GeForce RTX 2080 Ti 11 GB graphics card, all running on Windows Server 2016. Hash creation for 525,600 images took approximately 35 minutes, while the encoding stage took approximately 3 hours. On average, the retrieval time was approximately 350 ms.

4 Conclusions

We introduced a new method for retrieving solar magnetograms that employs a semantic hash. Our technique uses data from the SDO Helioseismic and Magnetic Imager and is implemented using SunPy and PyTorch libraries. We represent the magnetic regions of the Sun as 144-dimensional vectors, allowing for faster comparison and retrieval. To further expedite the process, we employ a fully-connected autoencoder to convert the 144-element magnetic intensity descriptor (MID) to a 36-element solar magnetic intensity hash. Our experiments,

ICCS Camera Ready Version 2023
To cite this paper please use the final published version:
DOI: 10.1007/978-3-031-36027-5_19

Table 3. Experiment results for the proposed algorithm. Due to lack of space, we present only a part of all queries.

Timestamp	RI	SI	PRI(TP)	FPRI(FP)	PNRI(FN)	Precision	Recall	F_1
2018-01-01 00:00:00	238	241	200	38	41	0.84	0.83	0.83
2018-01-04 07:00:00	448	481	401	47	80	0.90	0.83	0.86
2018-01-07 16:00:00	433	481	383	50	98	0.88	0.80	0.84
2018-01-13 07:00:00	403	481	394	9	87	0.98	0.82	0.89
2018-01-20 22:06:00	383	481	377	6	104	0.98	0.78	0.87
2018-01-24 06:12:00	431	481	384	47	97	0.89	0.8	0.84
2018-01-31 11:12:00	436	481	400	36	81	0.92	0.83	0.87
2018-02-05 14:18:00	428	481	391	37	90	0.91	0.81	0.86
2018-02-11 21:24:00	432	481	398	34	83	0.92	0.83	0.87
2018-02-18 10:24:00	417	481	381	36	100	0.91	0.79	0.85
2018-02-21 16:30:00	428	481	395	33	86	0.92	0.82	0.87
2018-02-27 12:36:00	392	481	386	6	95	0.98	0.80	0.88
2018-03-07 19:36:00	434	481	391	43	90	0.90	0.81	0.85
2018-03-08 20:42:00	440	481	394	46	87	0.90	0.82	0.86
2018-03-13 09:42:00	398	481	384	14	97	0.96	0.80	0.87
2018-03-19 11:42:00	448	481	405	43	76	0.90	0.84	0.87
2018-03-24 20:42:00	438	481	401	37	80	0.92	0.83	0.87
2018-03-31 04:42:00	407	481	390	17	91	0.96	0.81	0.88
2018-04-01 14:48:00	392	481	386	6	95	0.98	0.80	0.88
2018-04-09 17:48:00	434	481	398	36	83	0.92	0.83	0.87
2018-04-13 18:48:00	439	481	390	49	91	0.89	0.81	0.85
2018-04-16 20:54:00	435	481	399	36	82	0.92	0.83	0.87
2018-04-19 22:00:00	410	481	394	16	87	0.96	0.82	0.88
2018-04-28 02:00:00	422	481	382	40	99	0.91	0.79	0.85
2018-05-04 07:00:00	432	481	394	38	87	0.91	0.82	0.86
2018-05-09 04:00:00	421	481	395	26	86	0.94	0.82	0.88
2018-05-14 00:06:00	428	481	404	24	77	0.94	0.84	0.89
2018-05-18 16:12:00	452	481	405	47	76	0.90	0.84	0.87
2018-05-22 23:12:00	436	481	398	38	83	0.91	0.83	0.87
2018-05-25 14:18:00	407	481	388	19	93	0.95	0.81	0.87
2018-05-26 21:18:00	410	481	393	17	88	0.96	0.82	0.88
2018-06-01 16:24:00	400	481	391	9	90	0.98	0.81	0.89
2018-06-03 08:24:00	425	481	379	46	102	0.89	0.79	0.84
2018-06-11 10:30:00	425	481	379	46	102	0.89	0.79	0.84
2018-06-13 04:30:00	436	481	396	40	85	0.91	0.82	0.86
2018-06-15 10:30:00	425	481	380	45	101	0.89	0.79	0.84
2018-06-18 18:30:00	426	481	395	31	86	0.93	0.82	0.87
2018-06-23 01:30:00	434	481	400	34	81	0.92	0.83	0.87
2018-06-27 01:36:00	441	481	395	46	86	0.90	0.82	0.86
2018-07-04 09:36:00	437	481	400	37	81	0.92	0.83	0.87
2018-07-08 12:42:00	419	481	406	13	75	0.97	0.84	0.9
Avg.						**0.92581**	**0.81674**	**0.86720**

ICCS Camera Ready Version 2023
To cite this paper please use the final published version:
DOI: 10.1007/978-3-031-36027-5_19

as detailed in Tab. 3, demonstrate the superior precision of our approach compared to other state-of-the-art methods. Additionally, this method can also be used for classification tasks. By utilizing magnetograms instead of images from the Atmospheric Imaging Assembly, which captures the solar atmosphere in one or multiple wavelengths, the proposed method is more robust against noise.

References

1. Banda, J., Angryk, R., Martens, P.: Steps toward a large-scale solar image data analysis to differentiate solar phenomena. Solar Physics **288**(1), 435–462 (2013)
2. Banda, J.M., Angryk, R.A.: Large-scale region-based multimedia retrieval for solar images. In: International Conference on Artificial Intelligence and Soft Computing. pp. 649–661. Springer (2014)
3. Banda, J.M., Angryk, R.A.: Scalable solar image retrieval with lucene. In: 2014 IEEE International Conference on Big Data (Big Data). pp. 11–17. IEEE (2014)
4. Banda, J.M., Angryk, R.A.: Regional content-based image retrieval for solar images: Traditional versus modern methods. Astronomy and computing **13**, 108–116 (2015)
5. Brunner, C., Kő, A., Fodor, S.: An autoencoder-enhanced stacking neural network model for increasing the performance of intrusion detection. Journal of Artificial Intelligence and Soft Computing Research **12**(2), 149–163 (2022). https://doi.org/10.2478/jaiscr-2022-0010
6. Buckland, M., Gey, F.: The relationship between recall and precision. Journal of the American society for information science **45**(1), 12 (1994)
7. Grycuk, R., Galkowski, T., Scherer, R., Rutkowski, L.: A novel method for solar image retrieval based on the parzen kernel estimate of the function derivative and convolutional autoencoder. In: 2022 International Joint Conference on Neural Networks (IJCNN). pp. 1–7. IEEE (2022)
8. Grycuk, R., Scherer, R.: Grid-based concise hash for solar images. In: International Conference on Computational Science. pp. 242–254. Springer (2021)
9. Kavitha, K., Rao, B.T.: Evaluation of distance measures for feature based image registration using alexnet. arXiv preprint arXiv:1907.12921 (2019)
10. Kempoton, D., Schuh, M., Angryk, R.: Towards using sparse coding in appearance models for solar event tracking. In: 2016 19th International Conference on Information Fusion (FUSION). pp. 1252–1259 (2016)
11. Kempton, D.J., Schuh, M.A., Angryk, R.A.: Describing solar images with sparse coding for similarity search. In: 2016 IEEE International Conference on Big Data (Big Data). pp. 3168–3176. IEEE (2016)
12. Kempton, D.J., Schuh, M.A., Angryk, R.A.: Tracking solar phenomena from the sdo. The Astrophysical Journal **869**(1), 54 (2018)
13. Korytkowski, M., Senkerik, R., Scherer, M.M., Angryk, R.A., Kordos, M., Siwocha, A.: Efficient image retrieval by fuzzy rules from boosting and metaheuristic. Journal of Artificial Intelligence and Soft Computing Research **10**(1), 57–69 (2020). https://doi.org/10.2478/jaiscr-2020-0005
14. Mairal, J., Bach, F., Ponce, J., Sapiro, G.: Online learning for matrix factorization and sparse coding. Journal of Machine Learning Research **11**(Jan), 19–60 (2010)
15. Mikołajczyk, A., Grochowski, M., Kwasigroch, A.: Towards explainable classifiers using the counterfactual approach - global explanations for discovering bias in data. Journal of Artificial Intelligence and Soft Computing Research **11**(1), 51–67 (2021). https://doi.org/10.2478/jaiscr-2021-0004

ICCS Camera Ready Version 2023
To cite this paper please use the final published version:
DOI: 10.1007/978-3-031-36027-5_19

16. Najgebauer, P., Scherer, R., Rutkowski, L.: Fully convolutional network for removing dct artefacts from images. In: 2020 International Joint Conference on Neural Networks (IJCNN). pp. 1–8. IEEE (2020)
17. Salakhutdinov, R., Hinton, G.: Semantic hashing. International Journal of Approximate Reasoning **50**(7), 969–978 (2009). https://doi.org/https://doi.org/10.1016/j.ijar.2008.11.006, special Section on Graphical Models and Information Retrieval
18. de Souza, G.B., da Silva Santos, D.F., Pires, R.G., Marananil, A.N., Papa, J.P.: Deep features extraction for robust fingerprint spoofing attack detection. Journal of Artificial Intelligence and Soft Computing Research **9**(1), 41–49 (2019). https://doi.org/10.2478/jaiscr-2018-0023
19. Stuart Mumford, Nabil Freij et al.: Sunpy: A python package for solar physics. Journal of Open Source Software **5**(46), 1832 (2020). https://doi.org/10.21105/joss.01832, https://doi.org/10.21105/joss.01832
20. The SunPy Community et al.: The sunpy project: Open source development and status of the version 1.0 core package. The Astrophysical Journal **890**, 1–12 (2020). https://doi.org/10.3847/1538-4357/ab4f7a, https://iopscience.iop.org/article/10.3847/1538-4357/ab4f7a
21. Ting, K.M.: Precision and recall. In: Encyclopedia of machine learning, pp. 781–781. Springer (2011)

ICCS Camera Ready Version 2023
To cite this paper please use the final published version:
DOI: 10.1007/978-3-031-36027-5_19