Sun Magnetogram Hash for Fast Solar Image Retrieval

Rafał Grycuk
Częstochowa University of Technology
Częstochowa, Poland *rafal.grycuk@pcz.pl*

Rafał Scherer
Częstochowa University of Technology
Częstochowa, Poland *rafal.scherer@pcz.pl*

Abstract

We propose a novel method for fast retrieving of full-disk solar magnetograms obtained by the Solar Dynamics Observatory (SDO) spacecraft. Due to the high resolution at which these images are produced, an effective search mechanism is essential for managing this large dataset. Based on the magnetogram images and later on Magnetic Region Intensity Image, we generate magnetic field distribution vector. We also use a small fully-connected autoencoder to encode these features and create a concise Magnetic Field Layer-Circle Segment Solar Hash. This method significantly reduces the data required to describe solar images, allowing for nearly real-time retrieval of images similar to the query image. Since HMI (Helioseismic and Magnetic Imager) images are not labeled, we define similarity based on images produced within a short timeframe. The efficiency and accuracy of our method have been validated through experimental results.

Keywords: fast image hash, solar activity analysis, solar image description,

1. Introduction

Studying the Sun is vital for gaining insights into the broader mechanics of our solar system and the universe. By exploring the Sun's internal structure and surface phenomena, we can deepen our understanding of how the Sun produces and distributes energy through nuclear fusion. This knowledge is crucial for predicting and interpreting the Sun's activities, such as solar flares, coronal mass ejections, and other space weather events that can impact Earth and its surrounding space environment.NASA's Living With a Star (LWS) Program is dedicated to studying the Sun's impact on Earth, and the Solar Dynamics Observatory (SDO) plays a pivotal role in this initiative. SDO provides comprehensive data on the solar atmosphere across multiple wavelengths, capturing details at fine spatial and temporal scales. A key instrument on SDO, the Helioseismic and Magnetic Imager (HMI), is instrumental in examining the solar surface's oscillations and magnetic fields. The HMI produces several types of solar data, including dopplergrams, continuum filtergrams, and magnetograms, which are detailed maps of the photospheric magnetic field. The concept of semantic hashing was first introduced by Salakhutdinov [7] and has since been applied to generate concise codes that effectively indicate content similarity in various types of data.

Semantic hashing aims to generate compact vectors that accurately represent the semantic content of objects. This method allows for the efficient retrieval of similar objects by searching for comparable hashes, a process that is faster and uses less memory than interacting directly with the objects. Prior studies have utilized multilayer neural networks to create these hashes. More recently, the adoption of learned semantic hashes has shown promise in image retrieval applications, as demonstrated in research such as that by [8].

Initially, it was determined that using hashes generated from full-disk solar images would be impractical due to the large size and volume of the image collections. Consequently, we

developed the hand-crafted intermediate descriptors mentioned previously.

A full-disk content-based image retrieval system is detailed in [1], where the authors evaluated eighteen image similarity measures across various image features, resulting in one hundred and eighty different combinations. Their experiments highlighted which metrics are most effective for comparing solar images to retrieve or classify different phenomena.

In [4], a general-purpose retrieval engine named Lucene is employed to fetch solar images. Each image is treated as a document comprising 64 elements (corresponding to the image's rows), making each image-document distinctive. Wild-card characters are incorporated in query strings to locate similar solar events. Although the Lucene engine is benchmarked against distance-based image retrieval methods in [3], no definitive superior method was identified. Each method presents its own set of pros and cons regarding accuracy, speed, and practicality. Notably, there is a significant trade-off between accuracy and retrieval speed, with several minutes needed to achieve precise results.

In this paper, we introduce a novel solar semantic hash for solar image retrieval. We use hand-crafted features (magnetic field layer circle segment histogram) and reduce it by applying a fully-connected autoencoder. This hash is remarkably compact, consisting of just 30 elements, yet our experiments confirm its adequacy for accurately capturing the essence of the images. The images in our dataset are labeled solely by their timestamps. We treat the timestamp as a similarity metric, and post-training, our algorithm facilitates the retrieval of images based on visual similarity, independent of timestamp proximity.

2. Proposed Method for Solar Magnetogram Hashing

The first step is to enhance the magnetogram image by clearly marking the magnetic regions, as depicted in Fig. 1 (left). This enhancement is known as magnetic region detection, conducted using magnetogram images sourced from the SunPy library [10, 9]. This step is crucial for determining the strength of the magnetic field. As illustrated in Fig. 1, the strength of the magnetic field intensifies around active regions. We quantify the strength of the magnetic field by using colour intensities, which are clearly displayed in Fig. 1 (right). Throughout the solar cycle, the

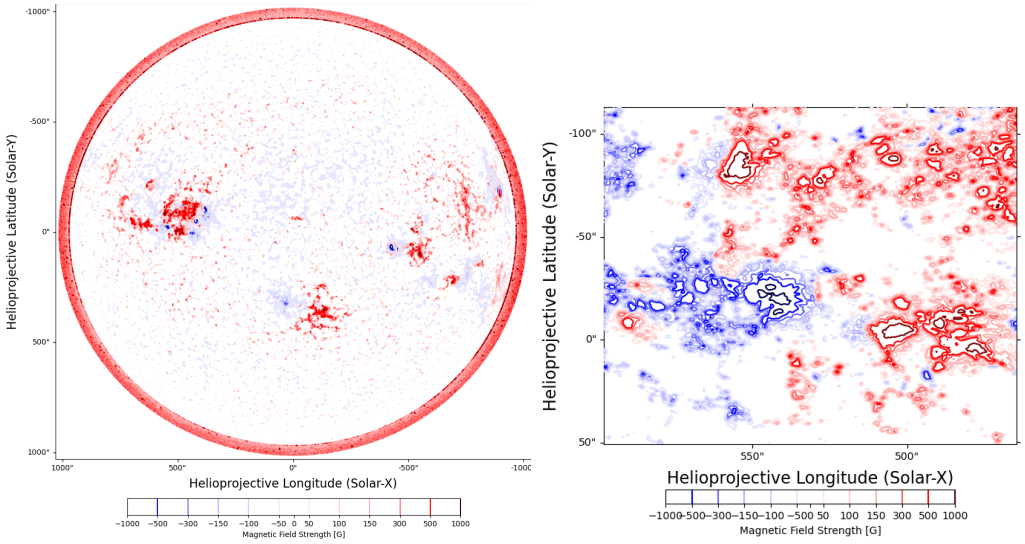

Fig. 1. Magnetic region detection and annotation process. Magnetic regions can be clearly visible. We can observe the polarities (red and blue) and their intensities (left). A magnification of magnetic regions (right).

magnetic field undergoes complex changes, including twisting, tangling, and reorganizing. It's

important to recognize that magnetic regions (MRs) are closely associated with Coronal Mass Ejections (CMEs) and solar flares, which significantly impacts Earth. As demonstrated in Fig. 1, magnetic region detection (MRD) helps identify the north (red) and south (blue) polarities of these regions. CMEs are most likely to occur between these polarities. Moreover, monitoring and analyzing MRs is crucial for predicting solar flares. The MRD process produces a magnetic region intensity image (MRI), which is vital for the subsequent steps of the analysis algorithm.

Direct comparison of high-resolution images is inefficient; thus, we segment MRI into sectors, and we crop every sector into circle segments. We obtain the list of magnetic field layer circle segments and calculate a histogram for each segment. The algorithm is detailed as follows: Initially, we establish the coordinates of the image center, denoted as cc. Fortunately, the radius r remains fixed, thanks to the Sun's consistent positioning within the image. Next, we determine the slicing angle θ. This value was chosen based on empirical findings from our research, which indicated that 30° yields the most optimal results. In the next step, we divide the radius r into several layers for our descriptor, utilizing the radius segment parameter rs. The value of rs is adjustable, which significantly influences the descriptor's effectiveness. In Fig. 2, we demonstrate this process by dividing the radius r into four layers to clearly illustrate the process. However, after thorough research, we found that dividing the radius into 10 segments yields the most optimal results. Subsequently, we perform a cropping operation on the resulting sectors as detailed in Alg. 1. To determine the arc points of each slice (sector), aps and ape, we compute $ape_x = cc_x - 1.5 * rs * \sin\theta$, $ape_y = cc_y - 1.5 * rs * \cos\theta$. The arc points of the sector are determined using trigonometric functions sin and cos. These formulas are utilized to compute the row and column coordinates of two points along the arc. To ensure the arc extends slightly beyond the circle's radius, a factor of 1.5 is applied to the calculated coordinates. The arc points derived from these calculations are then used to crop the relevant slice from the MRI. The variable cc represents the center of the circle, with cc_x and cc_y indicating the x and y coordinates of the center, respectively.

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

Algorithm 1: Algorithm for cropping the MRI slice.

The process of cropping the magnetic region intensity image (MRI) into slices is replicated for each subsequent layer circle segment (slice). This results in a list of MRI slices, each representing different magnetic field intensities. In the next step, a magnetic field layer circle segment histogram (MFLCSH) is created for every slice. This allows us to visualize the distribution of magnetic field strengths across each segment of the MRI. The histogram maintains a consistent scale for magnetic field intensities, ranging from $[-1000; 1000]$. In the final step, all individual histograms (MFLCSH) are aggregated into a single vector — the Magnetic Field Distribution Vector (MFDV). The entire methodology, including the segmentation and histogram generation, is depicted in Fig. 2 and outlined in Alg. 1.

The calculation process for the Magnetic Field Distribution Vector (MFDV) plays a crucial

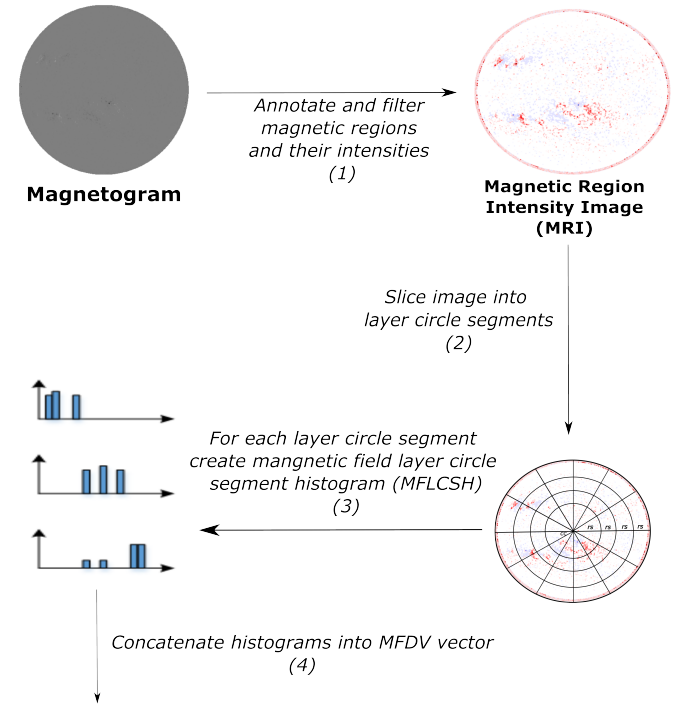

Fig. 2. Algorithm steps for calculating the Magnetic Field Distribution Vector.

role in substantially reducing data volume during the encoding stage. The main goal of this process is to create an intermediate, hand-crafted mathematical representation of magnetogram images, which is utilized in subsequent steps. Through this approach, we achieve MFDV of 120-length (12 sectors and 10 layers), representing a significant reduction compared to the full-disc magnetogram image or even MRI.

Now we use a fully-connected autoencoder (AE) to encode the MFDV. The structure of the autoencoder model is detailed in Table 1, which should be examined from top to bottom. The design of the model is simple yet efficient, effectively shortening the hash length while preserving crucial information about the magnetic regions in the magnetogram. Importantly, only the encoded segment of the autoencoder's latent space is utilized for hash creation; the decoding segment is used exclusively during the training phase. After extensive testing, we determined that training for 40 epochs strikes the right balance between generalization and the avoidance of overfitting. Table 1 illustrates the use of a convolutional autoencoder for generating hashes, where the initial layer serves as the input. A one-dimensional autoencoder was chosen because MFDV's are one-dimensional vectors, which simplifies the computational demands. This approach allows us to efficiently reduce the length of the hash while maintaining crucial information about the solar image's active regions. The mean squared error function was selected as the loss function, and training the model for 40 epochs proved sufficient to attain the desired level of generalization without leading to overfitting. Following the training period, each image descriptor was funneled through the autoencoder's encoding layers, generating a 30-element hash termed the Magnetic Field Layer-Circle Segment Solar Hash (MFLCSSH), suitable for content-based retrieval tasks involving solar images. The autoencoder architecture was carefully chosen to ensure optimal generalization.

In the final stage of our method, we utilize the hashes generated earlier for retrieving solar images. After the initial phases, we consider that each solar image in our database has an associated hash. The retrieval process starts by submitting an image query and comparing the hash

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

Layer (type)	Output	Filters (in, out)	Params
$Input(InputLayer)$	[1, 120]		0
$Linear\_1(Linear)$	[1, 60]	120, 60	7,260
$ReLU\_1$	[1, 60]	0	
$Linear\_2(Linear)$	[1, 60]	60, 30	1,830
$ReLU\_2$	[1, 30]	0	
$Encoded(latent-space)$	[1, 30]		
$Linear\_4(Linear)$	[1, 30]	30, 60	1,860
$ReLU\_4$	[1, 60]	0	
$Linear\_5(Linear)$	[1, 120]	60, 120	7,320
$ReLU\_5$	[1, 120]	0	
$Decoded(Tanh)$	[1, 120]		

of the query image against the hashes of all images in the dataset. This comparison involves calculating the distance (d) between the hash of the query image and each hash in the database. For this purpose, we use the cosine distance measure, which is effective for determining similarity between vectors (refer to [6] for more details). To facilitate this retrieval, it is essential to maintain a database of solar images that have been processed to generate hashes.

$$\cos(QH_j, IH_j) = \sum_{j=0}^{n} \frac{(QH_j \bullet IH_j)}{\|QH_j\| \, \|IH_j\|}, \tag{1}$$

where $\bullet$ is a dot product, QH_j is the query image hash, and IH_j a consecutive image hash. After calculating the cosine distance, the images in the database are sorted in ascending order based on their proximity to the query hash. In the concluding step of our proposed method, the top n images that are nearest to the query are selected and presented to the user. The user must specify the value of n to execute the query effectively. This process is outlined in pseudo-code in Alg. 3. Alternatively, image retrieval can also be performed using a threshold for the cosine distance. In this method, rather than specifying n, the user sets a threshold value, and only those images whose cosine distance to the query falls below this threshold are retrieved. This threshold-based approach is supported by our technique, allowing for flexible retrieval options. However, the method of retrieving the top n closest images is generally more user-friendly and is the recommended approach. The details of the retrieval process are documented in Alg. 2.

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 2: Image retrieval steps.

3. Experimental Results

In this section, we describe the details of the simulation results and our approach for assessing unlabeled images using unsupervised learning techniques to encode descriptors, necessitated by

the absence of labeled data. The lack of labeled data posed challenges in comparing our method against state-of-the-art techniques. To address this, we leveraged the rotational movement of the Sun as a natural experiment to identify a group of similar images (SI). We hypothesized that images captured in close temporal succession would exhibit similar solar active regions, with only minor positional shifts. The dataset consisted of solar images taken at intervals of 6 minutes. Given the Sun's rotational dynamics, images captured within such short intervals are likely to bear resemblance to each other. We adjusted the temporal window for similarity to refine our analysis. Through testing, we determined that images taken within a 48-hour period could be deemed similar for our purposes. Consider an example where an image is captured on 2014-06-15 at 00:00:00. Based on our previous assumptions, any image taken 24 hours before or after this timestamp would be considered similar. We rely solely on timestamps to identify these images for evaluation. A series of experiments was conducted to evaluate image similarity using the proposed method. The process for each experiment consisted of Executing an Image Query: Initiate a query to retrieve a set of images from the database, Comparing Timestamps: Analyze the timestamps of the retrieved images in comparison to the timestamp of the query image, and Identifying Similar Images: Determine which images have timestamps within a 48-hour window of the query image's timestamp, classifying these as similar. Once the set of similar images (SI) is defined, we can establish performance measures such as precision and recall, referencing studies like those by [5, 11]. These measures are based on the following sets: 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). For Content-Based Image Retrieval (CBIR) systems, evaluating performance using precision, recall, and the F_1 score is essential. The results of the performed experiments are outlined in Table 2. Moreover, they seem to be encouraging, specially the average of F_1 score and robust precision values. Our technique obtained an average precision of 0.919, which outperforms the 0.848 and 0.850 precision levels reported by Banda et al. [3] and Angryk et al. [2], respectively. The majority of magnetogram images similar to the query were accurately retrieved; however, magnetograms further from the query were often misclassified as positive but were not retrieved ($PNRI$). Notably, this error was significantly lower than in prior studies. The high $PNRI$ rates are likely due to the Sun's rotational movement, which can cause magnetic regions to shift or disappear within a 48-hour window. This dynamic significantly influences the semantic hash and impacts the query outcomes, resulting in reduced recall values. Our experiments verified these observations. The simulation environment utilized Python with SunPy and PyTorch libraries, running on a system equipped with an Intel Core I9-9900k 3.6 GHz processor, 32 GB RAM, a GeForce RTX 2080 Ti 11 GB graphics card, and Windows Server 2016. Creating hashes for 525,600 images (one year) took roughly 5 hours and 25 minutes, and the learning phase required about 21 hours. The average retrieval time was 450 ms.

4. Conclusions

Our approach introduced a new semantic hash for retrieving solar magnetograms that are similar to a given query image. We utilized data from the Solar Dynamics Observatory Helioseismic and Magnetic Imager, processed with SunPy and PyTorch libraries, to convert the magnetic regions on the Sun into a vector representation. This method compares vectors of 30-length rather than full-disk images, significantly reducing computational demands. Additionally, the incorporation of a fully-connected autoencoder enhances the speed of the process. Evidence of the effectiveness of our method is provided in Table 1, where it achieved the highest precision compared to other advanced methods. Besides solar image retrieval, this technique can be also effective for classification tasks, utilizing magnetograms over Atmospheric Imaging Assembly images (AIA) of the solar atmosphere across various wavelengths, offering greater resistance to

Table 2. 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)	Prec.	Recall	F_1
2014-01-01 00:00:00	233	241	193	40	48	0.83	0.80	0.81
2014-01-04 20:06:00	439	481	398	41	83	0.91	0.83	0.87
2014-01-07 00:12:00	428	481	396	32	85	0.93	0.82	0.87
2014-01-09 02:18:00	413	481	385	28	96	0.93	0.80	0.86
2014-01-16 22:18:00	446	481	397	49	84	0.89	0.83	0.86
2014-01-23 02:18:00	443	481	400	43	81	0.90	0.83	0.86
2014-01-27 23:24:00	405	481	376	29	105	0.93	0.78	0.85
2014-01-31 05:24:00	396	481	385	11	96	0.97	0.80	0.88
2014-02-08 15:30:00	447	481	406	41	75	0.91	0.84	0.87
2014-02-14 13:30:00	408	481	399	9	82	0.98	0.83	0.90
2014-02-22 08:30:00	398	481	387	11	94	0.97	0.80	0.88
2014-02-27 23:30:00	429	481	388	41	93	0.90	0.81	0.85
2014-03-02 16:36:00	413	481	395	18	86	0.96	0.82	0.88
2014-03-06 02:42:00	400	481	382	18	99	0.95	0.79	0.86
2014-03-09 14:42:00	433	481	396	37	85	0.91	0.82	0.86
2014-03-13 09:42:00	440	481	400	40	81	0.91	0.83	0.87
2014-03-21 02:42:00	448	481	404	44	77	0.90	0.84	0.87
2014-03-27 05:48:00	442	481	394	48	87	0.89	0.82	0.85
2014-03-30 06:48:00	441	481	391	50	90	0.89	0.81	0.85
2014-04-03 00:54:00	445	481	408	37	73	0.92	0.85	0.88
2014-04-04 04:00:00	403	481	388	15	93	0.96	0.81	0.88
2014-04-11 20:00:00	439	481	396	43	85	0.90	0.82	0.86
2014-04-18 09:06:00	399	481	394	5	87	0.99	0.82	0.90
2014-04-23 12:12:00	430	481	388	42	93	0.90	0.81	0.85
2014-04-28 23:18:00	407	481	383	24	98	0.94	0.80	0.86
2014-05-01 07:18:00	439	481	397	42	84	0.90	0.83	0.86
2014-05-05 06:18:00	443	481	398	45	83	0.90	0.83	0.86
2014-05-12 04:18:00	428	481	400	28	81	0.93	0.83	0.88
2014-05-18 21:24:00	399	481	388	11	93	0.97	0.81	0.88
2014-05-21 06:24:00	437	481	401	36	80	0.92	0.83	0.87
2014-05-27 05:24:00	446	481	396	50	85	0.89	0.82	0.85
2014-05-28 13:30:00	409	481	397	12	84	0.97	0.83	0.89
2014-06-03 01:30:00	419	481	371	48	110	0.89	0.77	0.83
2014-06-05 14:30:00	415	481	391	24	90	0.94	0.81	0.87
2014-06-12 21:36:00	430	481	382	48	99	0.89	0.79	0.84
2014-06-14 12:36:00	402	481	388	14	93	0.97	0.81	0.88
2014-06-19 16:42:00	435	481	389	46	92	0.89	0.81	0.85
2014-06-23 20:48:00	424	481	404	20	77	0.95	0.84	0.89
2014-07-02 07:48:00	400	481	385	15	96	0.96	0.80	0.87
Avg.						**0.924**	**0.816**	**0.865**

noise and making it a dependable option for practical applications.

References

[1] Banda, J., Angryk, R., and Martens, P.: Steps toward a large-scale solar image data analysis to differentiate solar phenomena. In: *Solar Physics* 288.1 (2013), pp. 435–462.

[2] Banda, J. M. and Angryk, R. A.: Large-scale region-based multimedia retrieval for solar images. In: *International Conference on Artificial Intelligence and Soft Computing*. Springer. 2014, pp. 649–661.

[3] Banda, J. M. and Angryk, R. A.: Regional content-based image retrieval for solar images: Traditional versus modern methods. In: *Astronomy and computing* 13 (2015), pp. 108–116.

[4] Banda, J. M. and Angryk, R. A.: Scalable solar image retrieval with lucene. In: *2014 IEEE International Conference on Big Data (Big Data)*. IEEE. 2014, pp. 11–17.

[5] Buckland, M. and Gey, F.: The relationship between recall and precision. In: *Journal of the American society for information science* 45.1 (1994), p. 12.

[6] Kavitha, K. and Rao, B. T.: Evaluation of Distance Measures for Feature based Image Registration using AlexNet. In: *arXiv preprint arXiv:1907.12921* (2019).

[7] Salakhutdinov, R. and Hinton, G.: Semantic hashing. In: *International Journal of Approximate Reasoning* 50.7 (2009), pp. 969–978.

[8] Souza, G. B. de, Silva Santos, D. F. da, Pires, R. G., Marananil, A. N., and Papa, J. P.: Deep Features Extraction for Robust Fingerprint Spoofing Attack Detection. In: *Journal of Artificial Intelligence and Soft Computing Research* 9.1 (2019), pp. 41–49.

[9] Stuart Mumford, Nabil Freij et al.: SunPy: A Python package for Solar Physics. In: *Journal of Open Source Software* 5.46 (2020), p. 1832. URL: https://doi.org/10.21105/joss.01832.

[10] The SunPy Community et al.: The SunPy Project: Open Source Development and Status of the Version 1.0 Core Package. In: *The Astrophysical Journal* 890 (1 2020), pp. 1–12. URL: https://iopscience.iop.org/article/10.3847/1538-4357/ab4f7a.

[11] Ting, K. M.: "Precision and recall". In: *Encyclopedia of machine learning*. Springer, 2011, pp. 781–781.