Enhancing Solar Flare Image Prediction Using Autoencoders

Alfred Mawuli Dogbatse, Adrian Beard, Masud Rana Rashel and Akm Kamrul Islam

January 16, 2024

Enhancing Solar Flare Image Prediction using Autoencoders

1st Alfred Mawuli Dogbatse
Computational Data Science and Engineering
North Carolina A & T State State University
Greensboro, USA
amdogbatse@aggies.ncat.edu

2nd Adrian Beard
Computational Data Science and Engineering
North Carolina A & T State State University
Greensboro, USA
adbeard1@aggies.ncat.edu

3rdMasud Rana Rashel
Institute of Earth Sciences
University of Évora Pole
Évora, Potugal
mrashel@uevora.pt

4th AKM Kamrul Islam
Computational Data Science and Engineering
North Carolina A & T State State University
Greensboro, USA
akislam@ncat.edu

***Abstract*—The study introduces an innovative framework for enhancing solar flare image prediction through deep learning - autoencoding, addressing limitations of traditional machine learning models. Solar flares impact space weather and Earth's technology, requiring improved prediction models. The research integrates an auto-encoder and advanced techniques like binary thresholding, Hessian matrix eigenvalue calculation, and Canny edge detection for feature extraction and shape analysis. Motivated by societal and economic impacts, it aims to mitigate disruptions caused by solar flares. Recent incidents underscore the urgency for reliable predictive methods. The study combines image processing and machine learning, utilizing an Autoencoder with convolutional and transpose convolutional layers, contributing to feature representation and understanding in solar flare analysis.**

***Index Terms*—solar flare, autoencoding, binary thresholding, hessian matrix, canny edge detection**

I. Introduction

The study aims to develop an advanced framework for predicting solar flare images using deep learning techniques, enhancing our understanding of solar activity and its potential impacts on space weather. Solar flares, sudden and intense bursts of energy on the Sun's surface, have significant implications for space weather and technological systems on Earth. Traditional methods of predicting solar flares rely on manual analysis of observational data, leading to limitations in accuracy and timeliness. As we move towards a technology-dependent society, the need for more precise and efficient prediction models becomes imperative. Literature supports the urgency of improving solar flare prediction methods. For instance, the work of [1] highlights the challenges posed by solar flares to satellite communication systems, underscoring the need for enhanced predictive capabilities. Additionally, the study by [2] emphasizes the potential impact of solar flares on power grids, demonstrating the critical importance of accurate prediction for mitigating these effects. The absence of comprehensive deep learning models for solar flare prediction is evident in the literature. [3] discuss the limitations of current methodologies and advocate for exploring advanced machine learning techniques, while recent works such as [4] acknowledge the potential of deep learning in space weather research but have yet to delve into its application specifically for solar flare prediction. Despite the growing importance of solar flare prediction, a notable research gap exists in the application of deep learning techniques to enhance the accuracy and speed of predictions. Existing models often rely on traditional machine learning algorithms and deep learning architectures. This research seeks to address this gap by introducing a novel approach that harnesses the power of auto-encoder employing advanced techniques like binary thresholding, Hessian matrix eigenvalue calculation, and Canny edge detection to enhances feature extraction and shape analysis for solar flare image predictions. The relevance of applying deep learning to solar flare prediction is supported by recent advancements in related fields. For instance, the success of deep learning in image recognition [5] and time-series analysis [6] suggests its potential applicability to the unique challenges posed by solar flare image data. The motivation behind this research lies in the potential societal and economic impact of improved solar flare predictions. As highlighted by [7], space weather events, including solar flares, can disrupt communication systems, affect satellite operations, and even pose risks to power grids. By developing a more accurate and timely prediction model, our research aims to contribute to the resilience of critical infrastructures and advance our ability to mitigate the effects of solar flares on Earth. The urgency of our research is underscored by recent incidents such as the solar storm of February 2023, which caused disruptions to satellite communication and navigation systems [8]. These events emphasize the critical need for innovative and reliable methods in predicting solar flares, forming the foundation of our study.

II. Related Studies

In ”Forecasting air quality time series using deep learning” the study employs recurrent neural network (RNN) with long short-term memory (LSTM) networks, a type of recurrent neural network (RNN), to analyze temporal dependencies in air quality. LSTMs are well-suited for capturing long-term patterns in time-series data. The research achieves high prediction accuracy by leveraging the ability of RNN and LSTM networks to retain information over extended periods. This results in a more nuanced understanding of the complex temporal dynamics associated with time series data. The conclusion emphasizes the potential of RNN and LSTM-based models in improving the reliability of time series predictions, particularly in capturing intricate temporal relationships [9].

Also, a study conducted by [10] used Long Short-Term Memory (LSTM), AutoRegressive Integrated Moving Average (ARIMA), Seasonal AutoRegressive Integrated Moving Average (SARIMA), and a hybrid model combining LSTM and ARIMA. These methods were employed for time series forecasting of the maximum sunspot number for Solar Cycles 25 and 26. The authors emphasized the optimization of hyperparameters using Bayesian optimization and highlights the outstanding performance of the LSTM-ARIMA hybrid model in achieving the best forecasting results. In their conclusion, hybrid methods, particularly the LSTM-ARIMA model, show promise in enhancing the accuracy of sunspot number forecasting, and LSTM networks exhibit superior learning capabilities compared to ARIMA in the context of time-series data. The forecasted sunspot numbers align closely with NASA’s predictions, suggesting the reliability of the chosen forecasting approach.

In addition, [11] employed fourier transform long short-term memory (FT-LSTM) model - a hybrid deep learning (DL) model - developed for improving the prediction accuracy of monthly discharge time series in the Brahmani river basin at Jenapur station. Additionally, three other popular DL models are mentioned for comparison: LSTM, recurrent neural network, and gated recurrent unit. The study evaluates the performance of these models considering different lag periods (1, 3, 6, and 12) to capture temporal relationships and identify patterns within hydrological data. The results indicate that the FT-LSTM model consistently outperforms the other models across all lag periods in terms of error metrics, demonstrating higher Nash–Sutcliffe efficiency and R2 values. The conclusion emphasizes the effectiveness of the FT-LSTM model in improving the accuracy of monthly runoff forecasts, contributing to the field of hybrid DL models for hydrological forecasting, and offering a promising solution for water resource management and river basin decision-making processes.

Moreover, Long Short-Term Memory (LSTM) model is employed for constructing a water quality parameter prediction model. The methodology involves utilizing the comprehensive pollution index method and Mann-Kendall trend analysis for assessing pollution status and change trends, extracting principal water quality parameters based on pollution share rates, employing the Spearman method for identifying influential factors, and finally, constructing the water quality parameter prediction model using LSTM analysis based on the driving factor analysis outcomes. The LSTM model demonstrated good prediction performance, with the average coefficient of determination (R2) reaching 0.82 for total nitrogen (TN) and 0.86 for dissolved oxygen (DO). Comparative analysis showed that the LSTM model outperforms both the random forest (RF) model in time series prediction and exhibits superior robustness and applicability compared to the AutoRegressive Moving Average with eXogenous inputs model (ARMAX). The findings suggest that the developed LSTM model offers valuable technical assistance for water quality prediction and early warning systems, particularly in economically disadvantaged regions with limited monitoring capabilities, facilitating resource optimization and promoting sustainable development [12].

In a comparison between Support Vector Machine (SVM), representing a traditional machine learning method, and deep learning for image recognition, with a focus on handwritten digital images recognition, deep learning is highlighted for its ability to naturally handle two-dimensional image data, automatically extract features, and its popularity for good learning ability and low generalization error. The results of the comparison demonstrate that the deep learning method is more accurate and more stable in image recognition compared to SVM [13].

The continuous progress of time and technological development has led to the explosive growth of image data on network social media. Images, being a primary mode of communication, are widely utilized due to their rich content and intuitive advantages. Convolutional Neural Network (CNN) emerges as the primary machine learning method in image recognition, involving operations like image eigenvalue extraction and convolution to analyze diverse images. The role of machine learning becomes increasingly significant in the realm of artificial intelligence, with algorithms learning from data to predict outcomes. However, challenges arise in associating low-level image information with high-level semantics. To address this, the paragraph introduces a multi-level information fusion model based on the VGG16 model, enhancing the recognition rate by recovering discarded feature information lost during convolution. The model’s recognition rate is validated using the 0RL Face Database, BioID Face Database, and CASIA Face Image Database, emphasizing the importance of CNN in image recognition improvements [14].

III. Methodologies

The strategy adopted for constructing the models was centered on forecasting images of solar flares. The initial step involved assembling satellite images containing both solar flares and those without them from Kaggle. Afterward, the process begins with reading images from specified paths, converting them to the RGB color space, and applying binary thresholding. Feature extraction is performed using the

Hessian matrix and eigenvalues, contributing to shape analysis and local structure understanding. Binary thresholding is again applied to RGB images, followed by Canny edge detection. The blue channel of the original image is combined with the Canny edge-detected image, and bounding rectangles are drawn around contours. The images are preprocessed, normalized, and organized into lists for training and testing. The study employs an Autoencoder neural network, involving an encoder with convolutional layers and a decoder with transpose convolutional layers. The Autoencoder is trained over 55 epochs, utilizing Mean Squared Error (MSE) as a metric. The trained Autoencoder is then applied to predict the reconstructed images of the first 20 samples in the dataset. This methodology aims to enhance the model's understanding of features in the data, specifically in the context of image reconstruction.

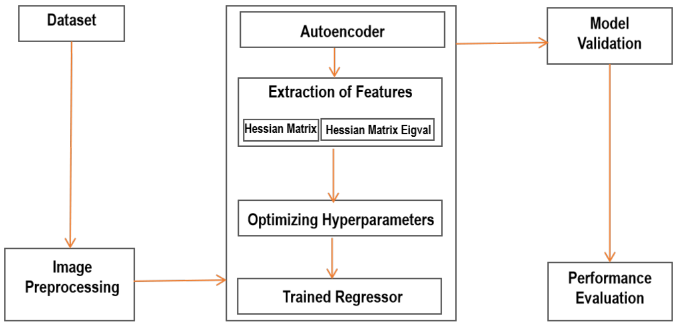

Fig. 1. Model Architecture

A. Dataset

In this study, we employed the Solar Flare Prediction Dataset sourced from Kaggle, which features satellite images depicting both instances of solar flares and images of the sun without flares. The decision to utilize satellite imagery was influenced by its extensive coverage and precise temporal resolution, characteristics inherent to satellites[15]. Forecasting solar flares is a daunting task, often likened to the challenges encountered in weather prediction. In the realm of Machine Learning, this dataset presents a notably intricate challenge due to the intricacy of individual samples, comprising up to 40 images, the relatively limited sample size of 8,000 for training, and the additional complexity posed by it being a regression problem.

B. Data Preprocessing

The initial step involved in preparing the Solar Flare Prediction Dataset for deep learning involved data preprocessing to align it with the requirements of the deep learning algorithm. This was crucial to empower the algorithm in identifying and extracting patterns within the dataset, ultimately enhancing its performance [16][17].

Specifically, the satellite images underwent resizing to dimensions of 180 by 180 pixels with 3 channels (RGB). The pixel values in the satellite dataset, originally ranging from 0 to 255, were then normalized by dividing each value by 255. This normalization process aimed to facilitate the backpropagation process, accelerating training for quicker convergence and improved overall performance [18][19][20].

C. Mathematical and Computational Tools for Feature Extraction and Analysis

The hessian matrix and hessian matrix eigvals in image processing serves several purposes, including feature extraction, shape analysis, and object detection. They are particularly useful when dealing with images containing complex structures, edges, or regions of interest, and they contribute to the extraction of meaningful information for subsequent analysis or decision-making. Also, they are used for the following purposes.

- Blob Detection: The Hessian matrix is effective in detecting blobs or regions with varying intensities in an image. By analyzing the eigenvalues of the Hessian matrix, you can identify regions that correspond to blob-like structures. This is particularly useful in applications such as medical imaging (detecting tumors) or computer vision (detecting objects) [21].
- Corner Detection: Eigenvalues of the Hessian matrix are used to identify corners or junctions in an image. High eigenvalues indicate regions with strong curvatures, typically found at corners. Corner detection is crucial in computer vision tasks like image registration or object recognition [22].
- Shape Analysis: The Hessian matrix provides information about the local curvature and structure of objects in an image. Analyzing the eigenvalues helps in understanding the shape of objects, whether they are edges, corners, or blobs. This is valuable in tasks like shape recognition and classification [23].
- Edge and Ridge Detection: By examining the eigenvalues, the Hessian matrix can be used to detect edges and ridges in an image. Different eigenvalue combinations indicate different types of structures. This is essential in image segmentation and edge detection tasks [24].
- Image Filtering: Hessian matrix-based methods can be employed as filters to enhance certain features in an image. For example, filtering based on eigenvalues might emphasize specific structures or suppress noise [25].
- Scale Selection: The Hessian matrix is used for multi-scale analysis, allowing the detection of structures at different scales. This is crucial for identifying objects or patterns of varying sizes in an image [26].
- Object Recognition: The information derived from the Hessian matrix and eigenvalues can be used as features for object recognition tasks. This is common in computer vision applications where understanding the structure of objects is essential [27].

D. Hessian matrix

In image processing, the Hessian matrix is a mathematical concept used to analyze the second-order spatial derivatives of an image. It provides information about the local structure, curvature, and shape of objects in an image. The Hessian matrix is commonly employed for feature extraction, edge detection, and shape analysis. The Hessian matrix for a grayscale image I(x,y) is a 2x2 matrix of partial derivatives computed with respect to the spatial coordinates x and y. The Hessian matrix H is defined as follows:

$$\text{Hess}, H = \begin{bmatrix} \frac{\partial^2 I}{\partial x^2} & \frac{\partial^2 I}{\partial x \partial y} \\ \frac{\partial^2 I}{\partial x \partial y} & \frac{\partial^2 I}{\partial y^2} \end{bmatrix} \tag{1}$$

Each element of the matrix represents the second-order partial derivative of the intensity values in the image. The elements H_{11} and H_{22} represent the curvature along the x-axis and y-axis, respectively, while H_{12} and H_{21} represent the cross-derivative terms. In practice, the Hessian matrix is often computed using convolution operations with appropriate convolution kernels to approximate the second-order derivatives. This allows for efficient and numerical stable computation of the Hessian matrix for various image processing tasks.

E. Hessian Matrix Eigvals

In image processing, hessian matrix eigvals refers to the function or operation that calculates the eigenvalues of the Hessian matrix. The Hessian matrix is a 2x2 matrix of second-order partial derivatives, often used to analyze the local structure, curvature, and shape of objects in an image. The eigenvalues of the Hessian matrix are crucial in various image processing applications, providing information about the principal curvatures and the nature of he local image structures. The eigenvalues ($\lambda 1$ *and* $\lambda 2$) and corresponding eigenvectors of the Hessian matrix are computed using the hessian matrix eigvals operation. Mathematically, given the Hessian matrix:

$$H = \begin{bmatrix} H_{xx} & H_{xy} \\ H_{xy} & H_{yy} \end{bmatrix} \tag{2}$$

where H_{xx}, H_{xy}, H_{yy} are the second-order partial derivatives of the image intensity values, the eigenvalues ($\lambda 1$ *and* $\lambda 2$) are obtained by solving the characteristic equation:

$$det(H - \lambda I) = 0 \tag{3}$$

where I is the identity matrix. The eigenvalues represent the principal curvatures of the image surface at a particular point. Their magnitudes and signs provide insights into the local structure of the image: If both eigenvalues are positive, the region corresponds to a bright spot or a ridge. If both eigenvalues are negative, the region corresponds to a dark spot or a trough. If one eigenvalue is positive and the other is negative, the region corresponds to a saddle point.

F. Model Training and Evaluation

The images are read from the specified path and are then converted from the BGR color space to the RGB color space using OpenCV. Binary thresholding is then applied to the converted RGB image. Pixels with intensity values

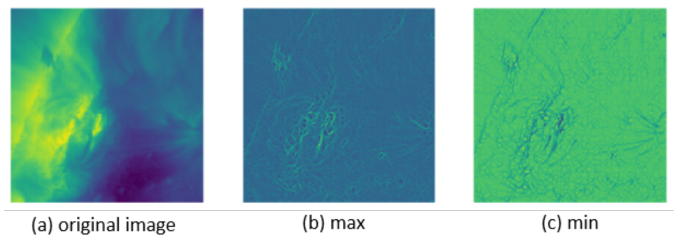

Fig. 2. Image, the max and min eigenvalues of the Hessian matrix

greater than or equal to 200 become 255 (white), and pixels below 200 become 0 (black). The resulting binary image is stored for later use. Again, the original images are read and converted to grayscale using cv2.cvtColor. Next, the Hessian matrix and eigenvalues of the grayscale images were calculated using the hessian matrix and hessian matrix eigvals functions respectively from the scikit-image library. The sigma parameter defines the standard deviation of the Gaussian filter applied before computing the derivatives. The order parameter specifies the derivatives to be computed. These functions were performed for feature extraction, shape analysis, and understanding local structures in the image. The resulting eigenvalues were used for further analysis and visualization. Likewise, the original images are converted from the BGR color space to the RGB color space using cv2.cvtColor and binary thresholding is applied to the RGB image . Pixels with intensity values greater than or equal to 200 become 255 (white), and pixels below 200 become 0 (black). Canny edge detection algorithm was then applied to the binary thresholded image and the parameters 10 and 100 represent the lower and upper thresholds for the edge detection. Furthermore, blue channel (img[:,:,0]) was selected from the original RGB image and performs a weighted combination of the blue channel of the original image and the Canny edge-detected image. The parameters 0.8 and 0.4 represent the weights assigned to the original and Canny images, respectively. Contours were found in the Canny edge-detected image and were iterated through. For each contour it found bounding rectangle was drawn using cv2.boundingRect. Moreover, the code processes the set of images, resizes them, normalizes pixel values, and then collects the preprocessed images along with their start and end annotations into separate lists to be used for training and testing. The MinMaxScaler scales the input data to a specified range, often [0, 1]. Encoder and decoder components of an Autoencoder neural networks were defined using the Sequential API in Keras/TensorFlow. The encoder sequentially adds convolutional layers with batch normalization and ReLU activation functions. The encoder consists of four convolutional layers with increasing numbers of filters (32, 64, 128, 256). Each convolutional layer is followed by batch normalization to stabilize training and a Rectified Linear Unit (ReLU) activation function. Likewise, the decoder sequentially adds transpose convolutional layers (deconvolutional layers) with ReLU activation functions. The decoder consists of four transpose convolutional layers with decreasing numbers of filters (128, 64, 32). The last layer has a number of filters equal to output class and uses ReLU activation.

- Autoencoder Composition and Compilation: The autoencoder combines the previously defined encoder and decoder as sequential layers in a new Autoencoder model. The encoder and decoder are connected sequentially, forming the complete autoencoder architecture and compiles the autoencoder model.
  Compile loss: Specifies the loss function to be minimized during training.
  Compile optimizer: Specifies the optimization algorithm to be used during training.
  Metrics=[”mse”]: Specifies that Mean Squared Error (MSE) to be used as a metric to evaluate the performance of the model during training.
- Autoencoder Training: This stage trained the autoencoder model by fitting it to a dataset of original images and their corresponding masked versions. The training process occurs over 55 epochs, and the Checkpoint model callback is employed to save the model weights when improvements in validation accuracy occur. The trained autoencoder model is stored in a variable, allowing for subsequent analysis and evaluation. This process aims to teach the autoencoder to reconstruct masked images from their original counterparts, facilitating the generation of predictions and potentially enhancing the model's ability to represent and understand features in the data. After training over 55 epochs, the loss and mse reduced from 7.2959 and 134541.8906 to 0.6363 and 0.0088 respectively.
- Prediction: The trained autoencoder was applied to the first 20 images in the dataset and stores the reconstructed images (predictions). The autoencoder attempts to reconstruct each input image, and predictions will contain the model's predictions for these specific samples.
  Put three figures here...

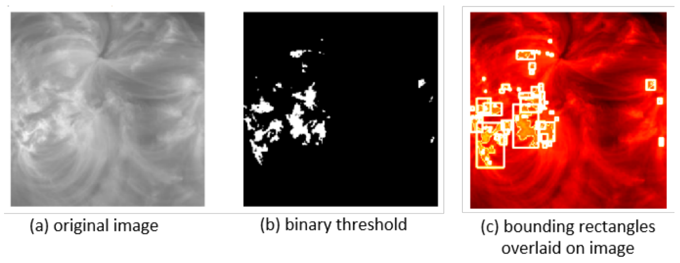

Fig. 3. A side-by-side visualization of an image, its binary threshold and the image with edges and bounding rectangles overlaid

G. Discussion

The catastrophic and devastating impact of solar flares on living organisms lives, and radio communication demands urgent attention. The machine learning process for detecting these solar flares involves various image processing and feature extraction steps. Initially, the images are converted to RGB color space, and binary thresholding is applied. Grayscale conversion, Hessian matrix, and eigenvalue calculations are performed for feature extraction. Binary thresholding, Canny edge detection, and weighted combination with the blue channel are applied for further image analysis. Contours are found, and bounding rectangles are drawn around them. The code then preprocesses images for training an Autoencoder, involving resizing, normalization, and collection of annotations.

The autoencoder architecture is defined with encoder and decoder components using convolutional layers, batch normalization, and ReLU activation. The model is compiled with specified loss and optimization functions, and Mean Squared Error (MSE) is chosen as a metric. Training occurs over 55 epochs, with a checkpoint mechanism for saving weights during improved accuracy. The loss and MSE metrics reduce significantly during training.

Finally, the trained autoencoder is used for prediction on the first 20 images, generating reconstructed images. The overall process aims to enhance the model's ability to reconstruct masked images from their originals, potentially improving feature representation and understanding in the data.

H. Conclusion

The machine learning process for detecting solar flares in images involves multiple steps for image preprocessing, feature extraction, and training an autoencoder model. Images are initially processed by converting them to the RGB color space, applying binary thresholding, and calculating Hessian matrix eigenvalues for feature extraction and shape analysis. Canny edge detection is then applied, and a weighted combination of the original and edge-detected images is performed. Contours and bounding rectangles are found for visual analysis. The dataset is further processed for training, including resizing, normalization, and annotation collection. An autoencoder model is defined, composed, and compiled with specified loss, optimizer, and evaluation metrics. The model is then trained over 55 epochs, with callbacks for monitoring and saving. The process results in reduced loss and mean squared error (mse). The trained autoencoder is applied to predict reconstructed images for the first 20 samples in the dataset. Overall, the workflow encompasses image processing, feature extraction, model training, and prediction.

Acknowledgment

We extend our sincere gratitude to Kaggle for providing the invaluable Solar Flare Image Dataset, a fundamental resource that played a pivotal role in developing our innovative framework for enhancing solar flare image prediction through deep learning. This high-quality data significantly contributed to addressing traditional machine learning model limitations and advancing space weather prediction. Kaggle's collaborative platform for data science enthusiasts and the community's efforts are acknowledged. Our article highlights the synergy between open data initiatives and cutting-edge research, emphasizing Kaggle's role in facilitating access to the Solar Flare

Image Dataset. We thank Kaggle and all contributors for their shared commitment to advancing scientific knowledge and promoting innovation worldwide.

References

[1] [1]Tobiska, W. K. (2008). Mitigating orbit planning, satellite operations, and communication surprises from adverse space weather. In 46th AIAA Aerospace Sciences Meeting and Exhibit (p. 453).

[2] Omatola, K. M., & Okeme, I. C. (2012). Impacts of solar storms on energy and communications technologies. Archives of Applied Science Research, 4(4), 1825-1832.

[3] Bueno, J., Maktoobi, S., Froehly, L., Fischer, I., Jacquot, M., Larger, L., & Brunner, D. (2018). Reinforcement learning in a large-scale photonic recurrent neural network. Optica, 5(6), 756-760.

[4] Kigerl, A., Hamilton, Z., Kowalski, M., & Mei, X. (2022). The great methods bake-off: Comparing performance of machine learning algorithms. Journal of Criminal Justice, 82, 101946.

[5] Pak, M., & Kim, S. (2017, August). A review of deep learning in image recognition. In 2017 4th international conference on computer applications and information processing technology (CAIPT) (pp. 1-3). IEEE.

[6] Torres, J. F., Hadjout, D., Sebaa, A., Martínez-Álvarez, F., & Troncoso, A. (2021). Deep learning for time series forecasting: a survey. Big Data, 9(1), 3-21.

[7] Ferguson, D. C., Worden, S. P., & Hastings, D. E. (2015). The space weather threat to situational awareness, communications, and positioning systems. IEEE Transactions on Plasma Science, 43(9), 3086-3098.

[8] Aa, E., Zhang, S. R., Wang, W., Erickson, P. J., & Coster, A. J. (2023). Multiple Longitude Sector Storm-Enhanced Density (SED) and Long-Lasting Subauroral Polarization Stream (SAPS) During the 26–28 February 2023 Geomagnetic Storm. Journal of Geophysical Research: Space Physics, 128(9), e2023JA031815.

[9] Freeman, B. S., Taylor, G., Gharabaghi, B., & Thé, J. (2018). Forecasting air quality time series using deep learning. Journal of the Air & Waste Management Association, 68(8), 866-886.

[10] Moustafa, S. S., & Khodairy, S. S. (2023). Comparison of different predictive models and their effectiveness in sunspot number prediction. Physica Scripta, 98(4), 045022.

[11] Swagatika, S., Paul, J. C., Sahoo, B. B., Gupta, S. K., & Singh, P. K. (2023). Improving the forecasting accuracy of monthly runoff time series of the Brahmani River in India using a hybrid deep learning model. Journal of Water and Climate Change, jwc2023487.

[12] Gao, Z., Chen, J., Wang, G., Ren, S., Fang, L., Yinglan, A., & Wang, Q. (2023). A novel multivariate time series prediction of crucial water quality parameters with Long Short-Term Memory (LSTM) networks. Journal of Contaminant Hydrology, 259, 104262.

[13] Lai, Y. (2019, October). A comparison of traditional machine learning and deep learning in image recognition. In Journal of Physics: Conference Series (Vol. 1314, No. 1, p. 012148). IOP Publishing.

[14] Lou, G., & Shi, H. (2020). Face image recognition based on convolutional neural network. China communications, 17(2), 117-124.

[15] Kaur, B., & Sharma, P. K. (2019). Implementation of customer segmentation using integrated approach. International Journal of Innovative Technology and Exploring Engineering (IJITEE), 8(6S), 770-772.

[16] Dwivedi, S. K., & Rawat, B. (2015, October). A review paper on data preprocessing: a critical phase in web usage mining process. In 2015 International Conference on Green Computing and Internet of Things (ICGCIoT) (pp. 506-510). IEEE.

[17] Brownlee, J. (2019). Develop deep learning models on theano and TensorFlow using keras. J Chem Inf Model, 53(9), 1689-1699.

[18] Singh, D., & Singh, B. (2020). Investigating the impact of data normalization on classification performance. Applied Soft Computing, 97, 105524

[19] Sola, J., & Sevilla, J. (1997). Importance of input data normalization for the application of neural networks to complex industrial problems. IEEE Transactions on nuclear science, 44(3), 1464-1468.

[20] Raju, V. G., Lakshmi, K. P., Jain, V. M., Kalidindi, A., & Padma, V. (2020, August). Study the influence of normalization/transformation process on the accuracy of supervised classification. In 2020 Third International Conference on Smart Systems and Inventive Technology (ICSSIT) (pp. 729-735). IEEE.

[21] Kong, H., Akakin, H. C., & Sarma, S. E. (2013). A generalized Laplacian of Gaussian filter for blob detection and its applications. IEEE transactions on cybernetics, 43(6), 1719-1733.

[22] Mehrotra, R., Nichani, S., & Ranganathan, N. (1990). Corner detection. Pattern recognition, 23(11), 1223-1233.

[23] Abdel-All, N. H., Soliman, M. A., Hussien, R. A., & El-Nini, W. M. (2013). Extract ridges and ravines using Hessian matrix of 3D image. J. Math. Comput. Sci., 3(5), 1252-1270.

[24] Abdel-All, N. H., Soliman, M. A., Hussien, R. A., & El-Nini, W. M. (2013). Extract ridges and ravines using Hessian matrix of 3D image. J. Math. Comput. Sci., 3(5), 1252-1270.

[25] Guo, S., & Wang, H. (2020). Image domain least-squares migration with a Hessian matrix estimated by non-stationary matching filters. Journal of Geophysics and Engineering, 17(1), 148-159.

[26] Lindeberg, T. (2020). Scale selection. Computer Vision: A Reference Guide, 1-14.

[27] Loncomilla, P., Ruiz-del-Solar, J.,& Martínez, L. (2016). Object recognition using local invariant features for robotic applications: A survey. Pattern Recognition, 60, 499-514.