Proceedings of ISERME 2025
19-20 September 2025
https://doi.org/10.31705/ISERME.2025.16

A Web-Based System for Rock Classification Leveraging RGB and Hyperspectral Imaging

***Kaito Takizawa[1], Natsuo Okada[2], Okhala Muacanhia[1], Narihiro Owada[3], George Paul Mathews[4], Yoko Ohtomo[2], and Youhei Kawamura[2]**

[1]Division of Sustainable Resources Engineering, Graduate School of Engineering, Hokkaido University, Kita-13, Nishi-8, Sapporo 060-8628, Japan

[2]Division of Sustainable Resources Engineering, Faculty of Engineering, Hokkaido University, Hokkaido University, Kita-13, Nishi-8, Sapporo 060-8628, Japan

[3]Graduate School of International Resource Sciences, Department of Earth Resource Engineering and Environmental Science, Akita University, Akita 010-0852, Japan

[4]School of Mining and Geosciences, Nazarbayev University, Astana, 010000, Kazakhstan

[*]Corresponding author – takizawa.kaito.b6@elms.hokudai.ac.jp

Abstract
This study introduces a novel scientific approach that integrates hyperspectral imaging and artificial intelligence to enhance rock type classification. A core contribution of this work is the development of an original segmentation algorithm capable of identifying subtle mineralogical variations in core samples. This algorithm enables the automated classification of diverse rock types with high accuracy and interpretability. The original algorithms were implemented into a user-friendly application that streamlines the image analysis process, thereby reducing dependency on expert geological interpretation. This enables rapid and reliable evaluation, even by non-specialist users. To validate the application's performance, a case study was conducted. Comparing the segmentation-based rock type classification with conventional visual inspection and Python-based scripts, confirming comparable accuracy. The findings demonstrate that the proposed system offers both scientific novelty and practical value, contributing to the advancement of non-contact, efficient, and accurate geotechnical analysis in both research and field environments.

Keywords: Hyperspectral imaging, Mineralogy, Mineral processing, Spectroscopy,

1 Introduction

Hyperspectral data, which provides unique spectral reflectance signatures for minerals and rocks, is a powerful tool for a wide range of applications, including mineral identification, environmental monitoring, agriculture, and medical imaging [1], [2]. Machine learning methods that use this data have shown high accuracy in classifying minerals [3]. However, the high dimensionality and complexity of hyperspectral data pose significant challenges for effective analysis [4].

Previous studies have used Python scripts to analyze hyperspectral data for applications in mining and civil engineering [5]. While effective, these scripts require programming skills and command-line operations, making them inaccessible to non-specialists. This underscores the need for more intuitive and user-friendly solutions [3].

Traditional machine learning techniques for hyperspectral image classification also have limitations, such as challenges with feature selection, a tendency to ignore spatial information, and poor scalability with large

datasets [6], [7]. Deep learning models have been developed to overcome these issues. They automatically learn important features, improve accuracy and efficiency, and reduce the risk of overfitting [8]. Deep learning also addresses scalability challenges through parallel processing and specialized hardware [9]. For example, Convolutional Neural Networks (CNNs) can recognize spatial patterns, a key advantage over traditional machine learning [10].

Recent studies show that using AI with hyperspectral data has greatly improved geological tasks, like mineral identification [11]. Conventional methods for mineral or rock identification are often time-consuming, costly, and don't always guarantee accuracy [12]. The advancement of tools using machine learning and deep learning can address these limitations. For example, a web application for rock type identification achieved 89.2% accuracy on thin section images [13]. Another application, APiS [14], allows users to create and apply classification models. However, these applications often rely on pre-trained models, which limits their flexibility and may not suit specific user needs. Additionally, some, like those requiring a MATLAB license, have high barriers to entry.

This study introduces a new application called IROMIE (a blend of the Japanese words for "color" and "appearance"), which features an intuitive graphical user interface (GUI) designed to enhance the accessibility and usability of mineral processing. IROMIE provides the same analytical capabilities as existing Python-based scripts while being significantly easier to use. A key feature is the ability for users to build and train models using their own datasets. This enables flexible and customized rock classification without relying on predefined models. The entire workflow—from model creation to prediction—can be completed without any programming expertise. Furthermore, since the application is built on an open-source Python platform, it is completely free to use.

In this paper, we will first provide a detailed description of the methodology, including the development environment and a flowchart for building the CNN model. We will then describe the algorithms used in the application and compare its performance to a Python script using the same dataset.

2 Application Development and System Design

2.1 Flowchart of the application

In the provided Figure , the rock classification application developed in this study operates according to a specific processing workflow. The primary task of this application is rock type segmentation, which involves classifying regions within a rock image. This can be performed on both RGB and hyperspectral images.

The segmentation task consists of three main steps: preprocessing, inference (or training), and post-processing. The application can operate in two distinct modes: training a new model with labeled data or performing inference using a pre-trained model. In both modes, the preprocessing stage converts the input image into a format suitable for CNN-based analysis. Subsequently, a VGG16-based network is used to estimate the rock type of each pixel. The resulting segmentation mask is then refined in the post-processing stage to remove noise and merge adjacent regions, which produces the final classification output. By implementing this task as an independent module within the application, users can seamlessly perform analyses as needed.

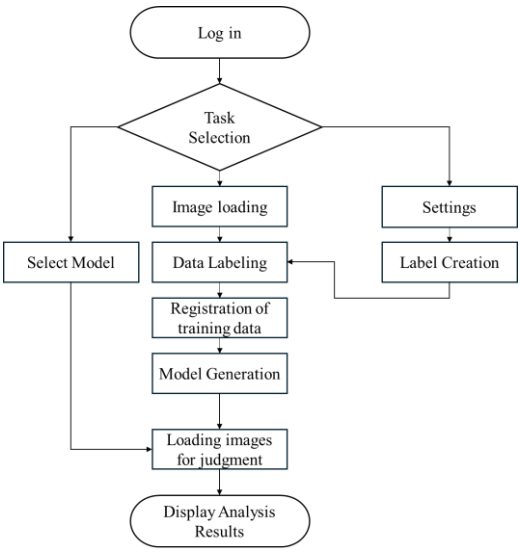

Figure 1: Flowchart of the application.

2.2 System Architecture and Technologies

The development environment for the application developed in this research is shown in Table . This web application was primarily built using TypeScript (v5.2.2) and Python (v3.9.18). The back-end server was developed with Node.js (v20.10.0), and PostgreSQL (v16.1) was used for the database. The

development environment ran on CentOS Stream release 9.

TypeScript was chosen for its static typing, which improves code readability and makes the system easier to maintain. Python's extensive libraries and flexibility were leveraged for data processing and scripting. Node.js was ideal for handling server-side tasks due to its asynchronous processing and scalability. Finally, PostgreSQL provided reliable and efficient database management. This combination of technologies ensures high performance and scalability for the entire system.

Table 1: Development environment

Component / Area	Technology	Version
Frontend	TypeScript	5.2.2
Backend	Python	3.9.18
Server Environment	Node.js	20.10.0
Database	PostgreSQL	16.1
Development Environment	OS	CentOS Stream 9

2.3 User Authentication and Information Protection

A user authentication feature has been implemented in this application to guarantee data security and the protection of personal information. Users gain access by first entering their email address and then supplying an authentication code sent to that email. This mechanism is critical for ensuring that shooting data, training data, and discrimination models are managed individually for each user, which effectively restricts access to data from separate sites. This design plays a vital role in preventing data leakage and safeguarding user privacy.

Additionally, the application provides an organization's administrator with the ability to meticulously configure individual user privileges. Figure illustrates the interface for modifying user permissions. The administrator possesses comprehensive authority to execute all functions, including uploading training images, generating discrimination models, and applying judgments to unknown images. Conversely, subordinate users are given specific authorizations, such as the permission to create training images. This layered privilege management system was developed to mitigate the risk of accidental deletion or overwriting of critical data, thus increasing data security.

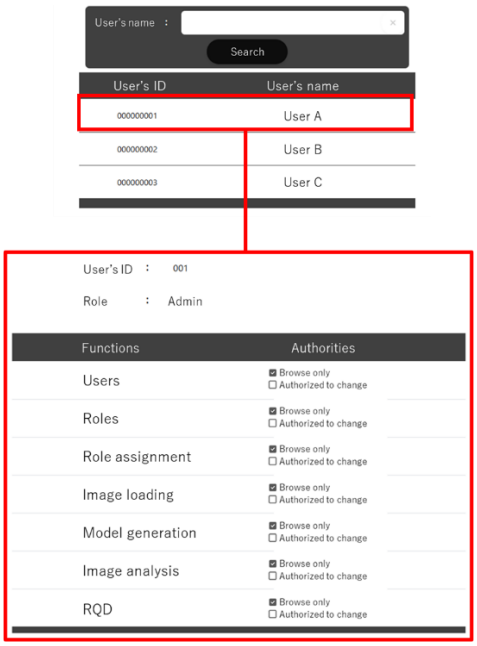

Figure 2: Interfaces of user modification

2.4 Segmentation task on the App

The application offers a streamlined and intuitive workflow that guides the entire process, from the preparation of training data to the generation and application of a classification model (Figure). The process begins with the preparation of the training dataset. An image file for analysis is uploaded from a local computer, with supported formats including .dat, .png, and .jpg. Subsequently, labels are assigned to specific regions of the image by intuitively dragging the mouse over the desired areas. This method significantly enhances the efficiency of the labelling process. Furthermore, the components used for analysis can be configured with multiple names and colours as shown in the figure, allowing them to be customized according to the specific objective of the analysis (Figure). Once the training data is prepared, the next step, discriminant model generation, can be initiated by clicking the "Generate Model" button. The application's status updates to indicate that the model is being built and a confirmation message is displayed upon completion. This process can be repeated at any time to reflect new data or label modifications. Finally, the generated model can be used for the classification of unknown images. A new image is uploaded, and the model is applied, which assigns a label to each pixel. The results are visually displayed using a color-

coded map, with the flexibility to freely set the name and colour for each label. This visual representation facilitates an easy understanding of the composition and spatial distribution of components within the image.

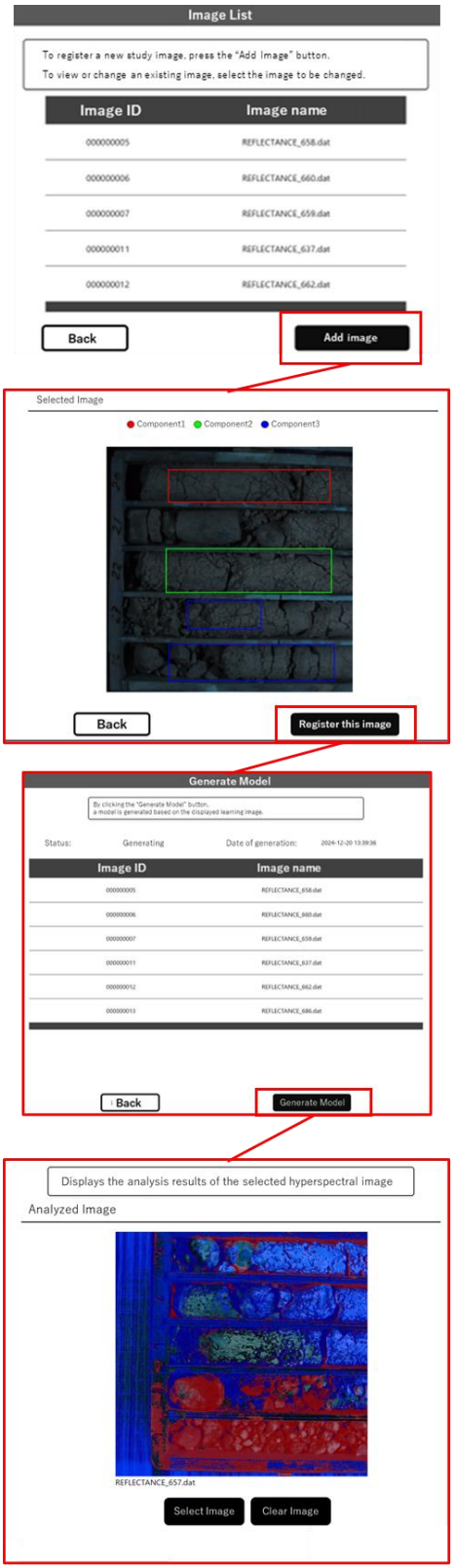

Figure 3: Workflow of segmentation

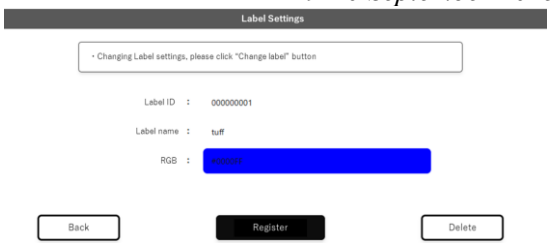

Figure 4: Label registration

3 Methodology

3.1 Hyperspectral data

Hyperspectral (HS) data is structured as a three-dimensional "data cube," with two spatial axes and a third axis for spectral information. After a process of averaging and normalization, this cube is converted into a one-dimensional array. This new format allows for the creation of spectral plots that show the brightness values across different wavelengths, as illustrated in Figure . In these graphs, the vertical axis indicates relative intensity, and the horizontal axis represents the wavelength in nanometres. HS data offers a fine spectral resolution, typically just a few nanometres, and can cover more than 100 bands. This level of detail surpasses that of conventional RGB data, providing clear, distinct spectral curves and peaks that are characteristic of specific materials.

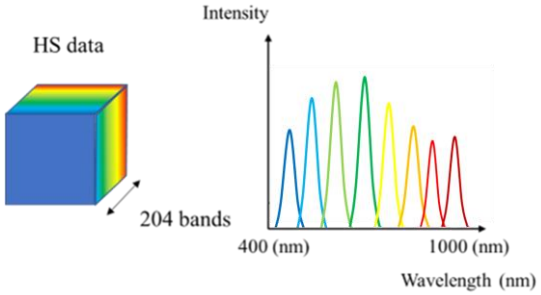

Figure 5: Structure of hyperspectral data

3.2 Algorithm of CNN

Convolutional Neural Networks (CNNs) are a class of deep learning algorithms known for their strong performance in image recognition tasks [25]. They function by extracting features from input data through a core operation called convolution. This process, which is essentially the inner product of the input data and a kernel (or filter), produces an output defined by the equation:

$$a_{ij} = \sum_{s=0}^{m-1} \sum_{t=o}^{n-1} w_{st} x_{(i+s)(j+t)} + b \qquad (1)$$

Here, x represents the image, k is the kernel's size, w represents the weights, and b is the bias.

This convolution is repeated across several layers, each typically comprising a convolutional operation, a non-linear activation function (like ReLU), and a pooling step. The ReLU layer introduces non-linearity, which is crucial for learning complex patterns, while the pooling layer (often using max-pooling) reduces the feature map's dimensions and provides translational invariance. As a CNN's layers deepen, it learns to recognize increasingly complex and abstract features. The final output is derived from the aggregated information across all layers.

The model is trained by adjusting its weights to minimize prediction errors through an iterative backpropagation process. This continues until the weights converge, signaling that the model is ready.

For reliable performance, a dataset is commonly divided into three subsets: a training set to update the model's parameters, a validation set to tune hyperparameters and prevent overfitting, and a test set to provide an unbiased evaluation of the final model's ability to generalize to new data.

3.3 Architecture of 1D-CNN

In this study, a Convolutional Neural Network (CNN) was developed using the Keras Sequential API to classify one-dimensional sequences. The model's architecture, including the design choices for each layer, is detailed in **Error! Reference source not found.**.

The model processes one-dimensional input sequences with a shape of (204, 1), representing 204 timesteps and a single feature. The first layer is a 1D convolutional layer with 64 filters, a kernel size of 3, and "same" padding to maintain the input's spatial dimensions. The ReLU activation function is applied to introduce non-linearity, enabling the layer to extract low-level features.

This is followed by a 1D max-pooling layer with a pool size of 2, which reduces the spatial dimensions. This step helps to decrease computational complexity and achieve translational invariance. The second convolutional layer has 128 filters with a kernel size of 3 and a ReLU activation function, designed to extract more abstract features. A second max-pooling layer with a pool size of 2 further reduces dimensions while retaining essential information.

To capture higher-level patterns, a third convolutional layer with 256 filters is used, also with a kernel size of 3 and ReLU activation. A dropout layer with a rate of 0.5 is placed after this layer to prevent overfitting by randomly deactivating half of the neurons during training.

The output from these feature extraction layers is flattened into a 1D vector. This is fed into a fully connected dense layer of 256 units with ReLU activation, which learns complex combinations of the high-level features. To further combat overfitting, a second dropout layer with a rate of 0.25 is included.

The final dense layer, with 4 units, uses the SoftMax activation function to output a probability distribution across the four classes, enabling the final classification.

The model was compiled with the categorical cross-entropy loss function, a standard choice for multi-class problems. The Adam optimizer, with its default learning rate of 0.001, was used for efficient training. Accuracy was selected as the primary metric to evaluate performance during both training and validation.

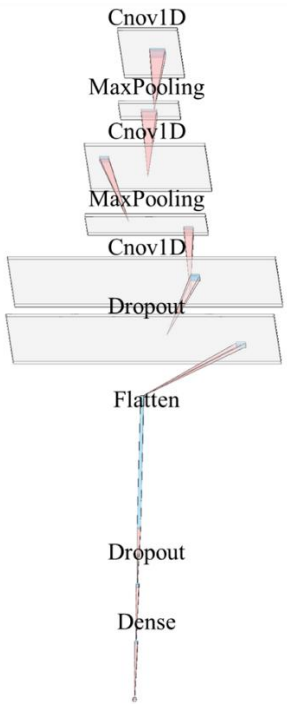

Figure 6: CNN architecture

4 Dataset preparation for segmentation

The classification task focused on tuff, mudstone, and other rock types sourced from advanced borehole cores in tunneling. The labeled dataset required for this machine learning task was created using the developed application's learning module. Users can

efficiently build their training data by selecting a label name and specifying a corresponding Region of Interest (ROI). For this experiment, three rock types, designated as Rocks A, B, and C, were chosen as the classification classes, and a total of 351,006 pixels of hyperspectral data were collected for training. This dataset was then divided to evaluate the model: 80% was used for training, while 10% each was allocated for validation and testing.

5 Results of segmentation

This experiment involved training machine learning models to classify three types of rocks. The classification dataset consisted of hyperspectral images from actual borehole core samples, as described in Section 3.1. The classification model was built based on the spectral information from these images.

The trained model was then applied to hyperspectral images that were not part of the training set to determine the rock type. The results of this analysis are shown in Figure 7(a).

To evaluate the application's accuracy, its results were compared with rock type classifications determined by visual inspection and a machine learning model built with a Python script, as shown in Figure 7(b). The ground truth data (visual labels) used for comparison are presented in Figure 7(c), and the application's results were found to be in high agreement. Furthermore, no significant performance differences were found between the model built on the application and a conventional Python-based model. These findings suggest that the application-based learning and classification process performs as well as traditional programming-based methods, confirming its high practical value in the field.

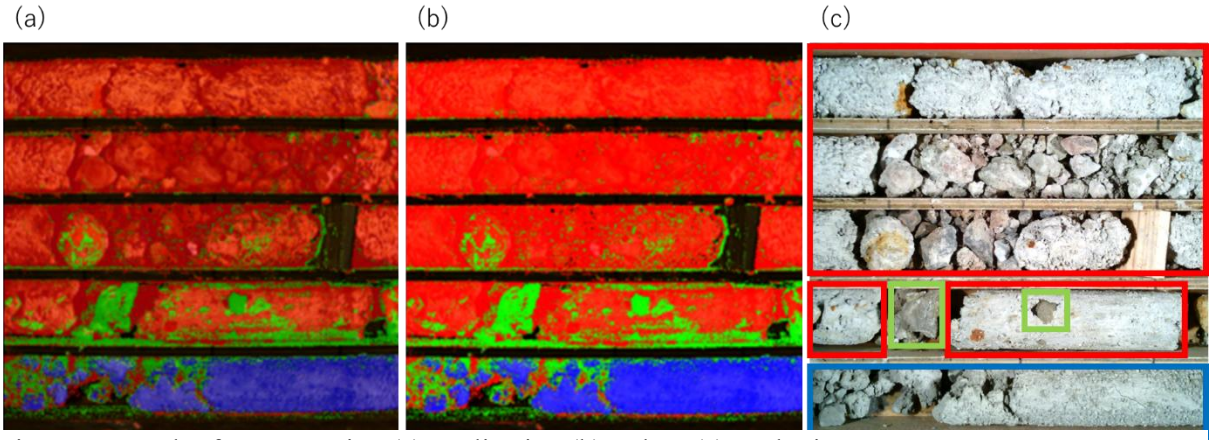

Figure 7: Result of segmentation (a)Application (b)Python (c)Geologist

6 Limitations

Despite the segmentation model's effectiveness, it has several limitations. First, even within the same rock type, subtle spectral differences can arise from variations in geological facies across different sites. Because of this, the current application requires the creation of site-specific models rather than a single, universal model that can be used everywhere.

Additionally, since the application processes images via the web, it must transmit and render large hyperspectral image files over an internet connection. The significant data volume of these images makes this process much slower than local processing with Python scripts on a personal computer. These factors pose considerable challenges to the application's scalability and real-time performance in field environments.

7 Conclusion

In this study, a novel, web-based application was developed to automate lithological classification based on hyperspectral imaging and RGB imagery of borehole core samples. The application incorporates custom-built algorithms for hyperspectral segmentation, trained with convolutional neural networks (CNNs).

The segmentation module demonstrated high concordance with expert classification and maintained robust performance even when applied to hyperspectral data from sites not included in the training set. This underscores its generalizability and practical applicability across a range of geological settings.

The integration of sophisticated algorithms within a user-friendly interface has been shown to substantially reduce operational workload and cognitive demands, thereby expanding the application's usability to encompass non-

specialist users. The system is a scientifically grounded and field-ready tool that enhances the accuracy and efficiency of geological evaluations in diverse applied contexts, including mining, tunnelling, and civil engineering.

Acknowledgements

The authors wish to thank S. Utsuki from UGS-Utsuki Geo Solution for his assistance with the project's progress. We also express our gratitude to Y. Shinohara of data-harness for his help in implementing the Python script as a web application. Finally, we thank N. Otsuka, a technical staff member at the Faculty of Engineering, Hokkaido University, for her support with the programming aspects of this work.

References

[1] D. Kumar, S. P. S, and R. Jha, "Identifying Rocks and Mineral Resources Using Hyper Spectral Analysis," in 2024 2nd International Conference on Artificial Intelligence and Machine Learning Applications Theme: Healthcare and Internet of Things (AIMLA), 2024, pp. 1–6.

[2] Y. Zhong and L. Zhang, "An Adaptive Artificial Immune Network for Supervised Classification of Multi-/Hyperspectral Remote Sensing Imagery," IEEE Transactions on Geoscience and Remote Sensing, vol. 50, no. 3, pp. 894–909, 2012.

[3] J. M. Bioucas-Dias, A. Plaza, G. Camps-Valls, P. Scheunders, N. Nasrabadi, and J. Chanussot, "Hyperspectral Remote Sensing Data Analysis and Future Challenges," IEEE Geosci Remote Sens Mag, vol. 1, no. 2, pp. 6–36, 2013.

[4] J. R. Laura, L. R. Gaddis, R. B. Anderson, and I. P. Aneece, "Chapter 4 - Introduction to the Python Hyperspectral Analysis Tool (PyHAT)," in Machine Learning for Planetary Science, J. Helbert, M. D'Amore, M. Aye, and H. Kerner, Eds., Elsevier, 2022, pp. 55–90.

[5] M. F. Goodchild, "Spatial Thinking and the GIS User Interface," Procedia Soc Behav Sci, vol. 21, pp. 3–9, 2011.

[6] S. Li, W. Song, L. Fang, Y. Chen, P. Ghamisi, and J. A. Benediktsson, "Deep Learning for Hyperspectral Image Classification: An Overview," IEEE Transactions on Geoscience and Remote Sensing, vol. 57, no. 9, pp. 6690–6709, 2019.

[7] B. Nikparvar and J. C. Thill, "Machine Learning of Spatial Data," ISPRS International Journal of Geo-Information 2021, Vol. 10, Page 600, vol. 10, no. 9, p. 600, Sep. 2021.

[8] J. Figueroa Barraza, E. López Droguett, and M. R. Martins, "Towards Interpretable Deep Learning: A Feature Selection Framework for Prognostics and Health Management Using Deep Neural Networks," Sensors (Basel), vol. 21, no. 17, p. 5888, Sep. 2021.

[9] L. Alzubaidi et al., "Review of deep learning: concepts, CNN architectures, challenges, applications, future directions," Journal of Big Data 2021 8:1, vol. 8, no. 1, pp. 1–74, Mar. 2021.

[10] H. Son, S. Kim, H. Yeon, Y. Kim, Y. Jang, and S. E. Kim, "Visual Analysis of Spatiotemporal Data Predictions with Deep Learning Models," Applied Sciences 2021, Vol. 11, Page 5853, vol. 11, no. 13, p. 5853, Jun. 2021.

[11] T. Long, Z. Zhou, G. Hancke, Y. Bai, and Q. Gao, "A Review of Artificial Intelligence Technologies in Mineral Identification: Classification and Visualization," Journal of Sensor and Actuator Networks 2022, Vol. 11, Page 50, vol. 11, no. 3, p. 50, Aug. 2022.

[12] S. Zhang, Y. Yang, F. Sun, and B. Fang, "Application of Image Sensing System in Mineral/Rock Identification: Sensing Mode and Information Process," Advanced Intelligent Systems, vol. 5, no. 11, Nov. 2023.

[13] S. Paucar, C. Mejía-Escobar, and V. Collaguazo, "DESCRIPCIÓN AUTOMÁTICA DE SECCIONES DELGADAS DE ROCAS: UNA APLICACIÓN WEB".

[14] N. Okada et al., "APiS (AI powered intelligence spectrum analyser): hyperspectral imaging machine learning analyser app for mineral processing," Int J Min Reclam Environ, vol. 0, no. 0, pp. 1–27, 2024

.