

# International Journal of Computing and Artificial Intelligence



E-ISSN: 2707-658X  
P-ISSN: 2707-6571  
IJCAI 2022; 3(1): 26-40  
Received: 12-11-2021  
Accepted: 14-12-2021

**Sylvester Akpah**  
University of Mines and  
Technology, Tarkwa South  
Ghana

**Yao Yevenyo Ziggah**  
University of Mines and  
Technology, Tarkwa, Ghana

**Daniel Mireku-Gyimah**  
University of Mines and  
Technology, Tarkwa, Ghana

**Corresponding Author:**  
**Sylvester Akpah**  
University of Mines and  
Technology, Tarkwa South  
Ghana

## Assessing the suitability of convolutional auto encoder as an unsupervised tool for image classification in artisanal small-scale gold mining environment

**Sylvester Akpah, Yao Yevenyo Ziggah and Daniel Mireku-Gyimah**

DOI: <https://doi.org/10.33545/27076571.2022.v3.i1a.43>

### Abstract

Efforts to control or stop illegal Artisanal Small-Scale Gold Mining (ASGM) in Ghana, which is causing significant environmental degradation, have faced numerous challenges because these illegal activities are carried out in remote areas, inaccessible by the current practice of using 4WD vehicles or trekking by foot. This paper sought to assess the suitability of using an Unmanned Aerial Vehicle (UAV) to capture the locations and features of all ASGM sites and use a Convolutional Autoencoder (CAE) to classify the defined sites into legal and illegal ASGM sites. The classification process used by the CAE involved three main stages, namely encoding, latent space learning, and decoding. The encoder accepts the UAV captured images as input, processes the input images to extract salient features and the decoder decodes the salient features to reconstruct the input image and define a site as an ASGM site. To classify a defined ASGM site as legal or illegal, a python program was integrated into the CAE which makes use of known point coordinates of all legal ASGM sites. A site is flagged as illegal if its point coordinates do not match those in the legal ASGM sites database, otherwise, it is a legal site. The performance of the CAE was measured using the following performance metrics: accuracy, precision, recall, and FI-score. The results of the CAE proved superior giving a classification accuracy of 97.52% when compared with the results obtained from other classification algorithms, namely Random Forest (RF) and Support Vector Machine (SVM) with 93.23% and 95.66% respectively. In this paper, it has been demonstrated that UAVs can be used to capture the locations and features of all ASGM sites, which otherwise would have been inaccessible by the use of 4WD vehicles or trekking, and classify the captured location into legal and illegal ASGM sites using a CAE, to facilitate the control and prevention of illegal ASGM in Ghana.

**Keywords:** artisanal small-scale gold mining, convolutional autoencoder, image classification, unmanned aerial vehicle, convolutional neural network

### 1. Introduction

Artisanal and Small-Scale Gold Mining (ASGM) in Ghana is an old vocation that has intensified in the last decade. This is largely attributed to the relatively high demand for gold, employment opportunities, and the economic benefits derived from the ASGM sector (Aryee *et al.*, 2003; Hilson and Potter, 2003; Owusu-Nimo *et al.*, 2018) <sup>[19, 5, 29]</sup>. ASGM operations have evolved from the use of rudimentary tools such as pickaxe, hammer, shovel, and chisel (Thomas *et al.*, 2002; Hilson, 2009; Bansah *et al.*, 2016a) <sup>[33, 18]</sup> to the use of heavy mining equipment such as bulldozers, excavators, crushers, and processing plants (Bansah *et al.*, 2016b; Bansah *et al.*, 2018) <sup>[7, 6]</sup>. Largely characterised by the lack of long-term planning, use of inappropriate mining methods, and total disregard for the environment, ASGM has led to significant environmental degradation (Kusimi *et al.*, 2014) <sup>[23]</sup>.

Of particular interest is illegal ASGM popularly known in the local parlance as “*Galamsey*”, which has become a nuisance as its activities can be directly linked to very destructive impacts on rivers and other water bodies (Mantey *et al.*, 2017; Owusu-Nimo *et al.*, 2018; Bansah, 2019) <sup>[26, 29]</sup>, as well as agricultural lands and forest reserves (Ontoyin and Agyemang, 2014; Owusu-Nimo *et al.*, 2018) <sup>[29]</sup>. It is now the policy of the Government of Ghana to stop illegal ASGM. Unfortunately, these activities are carried out in remote and inaccessible areas making it difficult to locate and stop them. This is because the current methods employed to gain access to these illegal sites include the use of 4WD vehicles and/or trekking which have been ineffective. Identifying the locations of illegal ASGM sites quickly would be the first logical step of overcoming the problem of inaccessibility and thus be able to control and prevent the practice.

This paper seeks to assess the suitability of deep feature learning techniques of AutoEncoders (AEs) to analyse UAV captured images of ASGM sites and subsequently classify them into legal and illegal ASGM sites. The present study thus leverages the capability of a Convolutional AutoEncoder (CAE) as a powerful feature extraction tool to learn highly descriptive feature representations of objects of interest contained in the UAV images. The main contribution of this approach is to bring to bear, and advance research in, CAEs as a modern image classification approach in heterogeneous environments such as ASGM sites.

### 1.1 Feature Extraction using AutoEncoders

AEs have gained a lot of popularity in recent times with published research works in computer vision and object detection. Yousefi-Azer *et al.* (2017) [38] proposed an unsupervised deep feature learning technique based on AEs for malware classification and network-based anomaly detection. Experimental results of their model architecture showed improvements in malware classification and detection of network-based anomaly intrusions of 97.38% compared to other traditional techniques. A feature extraction approach using stacked AEs in hyperspectral imaging to detect phenome state was proposed by Zabalza *et al.* (2016) [39]. A significant reduction in complexity and an improved efficacy rate of data abstraction and classification accuracy proved that their approach achieved a 96.87% classification accuracy. Zhang *et al.* (2018) [42] proposed a deep sparse AE for extracting deep feature representations for fault diagnosis. After carefully studying the input vectors to learn highly descriptive features of the images, the results showed that their approach achieved an accuracy rate of 99.3% of locomotive adhesion diagnosis state. In Gao *et al.* (2019) [15], a contractive stacked AE was proposed for data-based fault diagnosis. The stacked AE experimented on the Case Western Reserve University dataset produced a classification accuracy of 97.82%. Similarly, Zhang *et al.* (2019) [41] used stacked sparse AE to monitor in real-time high power disk laser welding statuses. The results proved that their approach achieved a higher classification accuracy of 96.68% and stronger robustness in monitoring welding status as compared to Support Vector Machine (SVM) and Random Forest. In the work of Supratak *et al.* (2014) [32], the authors tested the suitability of an unsupervised deep feature learning algorithm based on stacked AEs to automatically learn feature representations from Electroencephalography (EEG) data to design patient-focused seizure detectors. The experiments carried out by using a labeled dataset from the CHB-MIT database proved that all of the test seizures detected a mean latency rate of 3.36 seconds and a low false detection rate. Li *et al.* (2016) [25] compared Stacked AE to Fisher's ratio and Principal Component Analysis (PCA) to

extract deep feature representations to predict the existence of telecommunication churns. The experimental results proved that the average runtime of stacked AE was about 2 hrs while the average runtime of Fisher's ratio analysis was well over 7 hrs. Wetzel (2017) [35] employed conventional AE and variational AE to learn deep features of latent parameters. Results from this study showed a significant improvement of 95.68% classification accuracy over other image classifiers. Latif *et al.* (2017) [24] presented the use of variational AE to extract deep feature representations from speech signals and classify human emotions from the learned representations. The results showed that their technique yielded a classification accuracy of 64.86%. Although AEs have been applied in numerous domains, their application as a classification tool in ASGM remains underexplored. Hence, assessing the applicability and performance of AE as a classifier for legal and illegal ASGM site detection has some scientific value worth investigating.

## 2. Resources and Methods Used

### 2.1 Study Area

Tarkwa-Nsuaem Municipality, located in the Western Region of Ghana (Fig. 1) was selected as the study area for this research. The Municipality represents one of the largest active gold mining enclaves in the country, thereby presenting a vast concentration of ASGM activities. In addition to legal ASGM which is sanctioned by the Small-Scale Gold Mining Law PNDCL 218 of 1989 and the Minerals and Mining Act (2006), the Municipality is replete with illegal ASGM activities which have resulted in significant environmental degradation. The locations of most of these illegal ASGM sites are inaccessible thus making the study area suitable for this research.

### 2.2 Resources Used

A DJI Phantom 4 UAV equipped with a digital camera of resolution 4000 × 2250 pixel and a Field of View (FOV) of 70° was used to capture the aerial images of the ASGM sites at different altitudes and orientations. The UAV has a maximum flight time of approximately 30 minutes. A Dell G5 15 laptop with Intel Core i7 (3.1 GHz base frequency, 6 cores) was used to train the CAE model. The system is a CUDA-capable device and runs on Windows 10 Professional operating system. It features an NVIDIA GeForce RTX 1060 (10GB GDDR5 Dedicated) and 64GB RAM with a 128-bit interface. Python programming language was used and anaconda jupyter notebook was the model development environment. For the experiments, Keras (Chollet, 2015) [11] deep learning library running on TensorFlow v1.12.0 (Abadi *et al.*, 2016) [1] backend was also used.

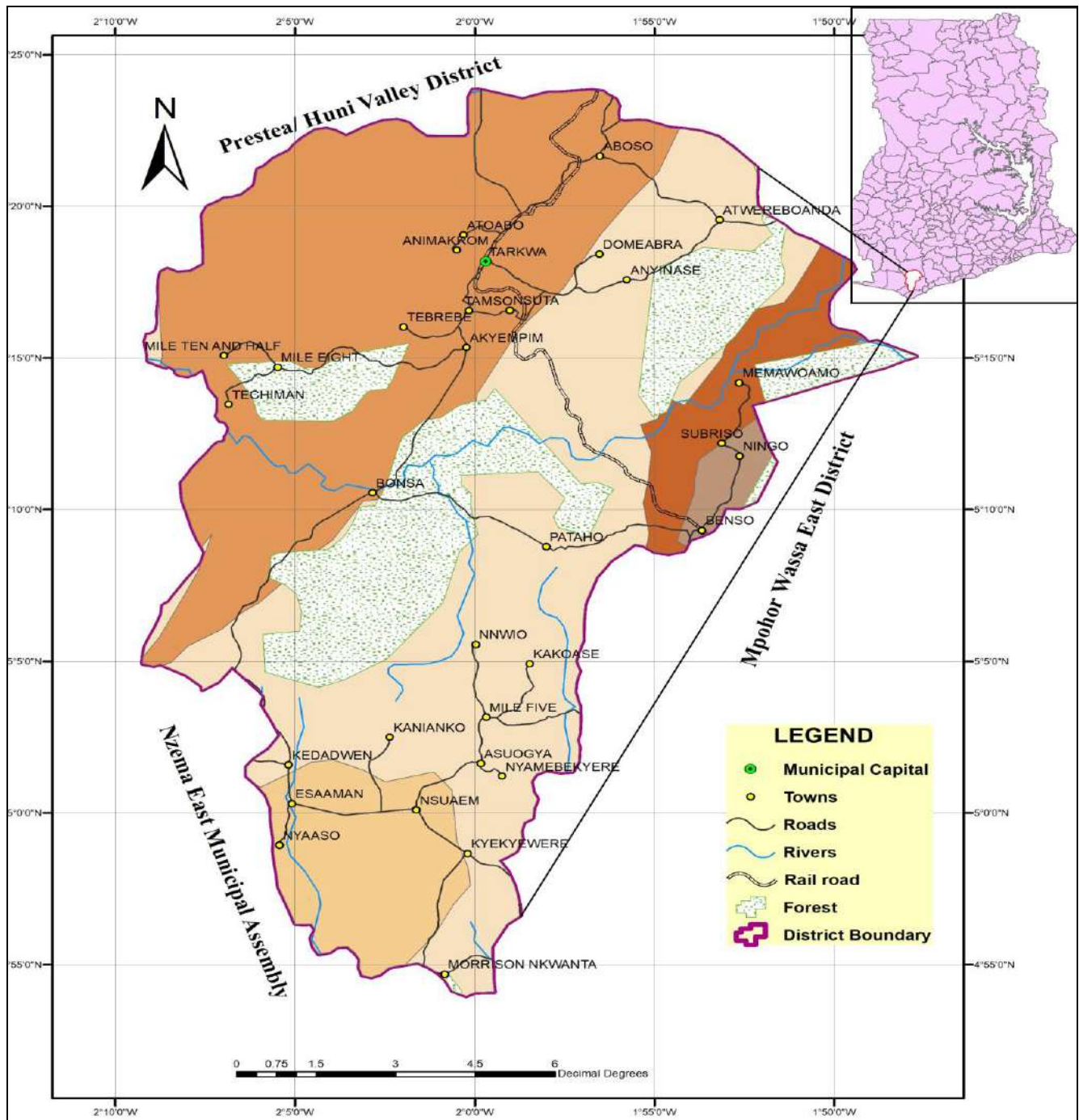


Fig 1: Map of Study Area

**2.3 Methods Used**

The study was implemented according to the following steps. Aerial data of the various ASGM sites was captured using a UAV as the first step. Data analysis which included image pre-processing and data augmentation followed next. Training of the CAE model was the last stage. Details of each of the implementation phases are discussed in the following subsections.

**2.3.1 Flowchart of the ASGM CAE**

The flowchart for the implemented CAE is shown in Fig. 2. The dataset applied in the first step involved the collected aerial images of the various ASGM sites which were

captured with UAV. The images were pre-processed using the Contrast-Limited Adaptive Histogram Equalization (CLAHE) (Zuiderveld, 1994; Jintasuttisak and Intajag, 2014) [45] to make them ready for the ASGM CAE model ahead of training. The data set was split into two sets: training and validation. The training set was used to train the model with the validation set used to evaluate the trained model’s generalisation using its classification accuracy. The model was optimised after several iterations to boost its performance. Accurate classification is achieved when the images being validated against the trained model belong to the right class.



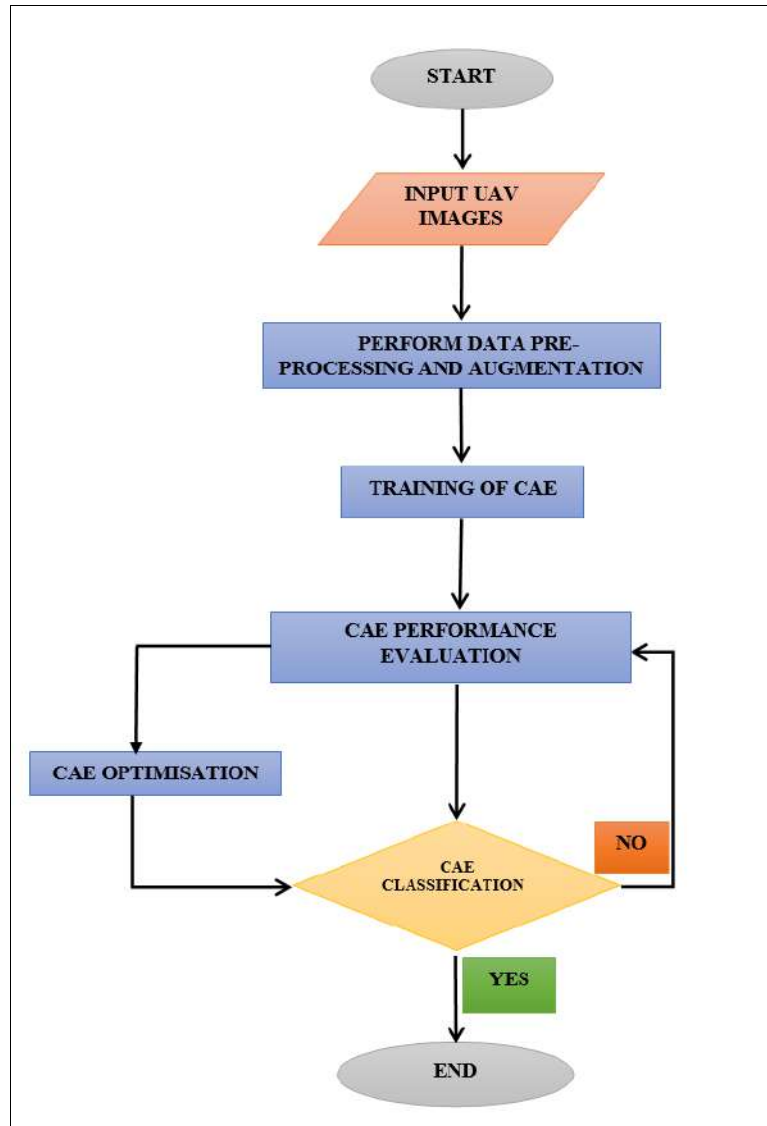


Fig 2: Flowchart of the CAE

**2.3.2 Data Acquisition**

The DJI Phantom 4 UAV equipped with a Global Navigation System (GNS) was used for the data collection. As shown in Fig. 3, the still photography mode of the UAV camera was used even though there was a live video stream option. Four automatic flight plans were performed using the UAV at two different altitudes (i.e., 100, and 150 meters). The images were captured between 12<sup>th</sup> and 16<sup>th</sup> October 2020 between the hours of 11:00 am to 2:00 pm each day when the weather was fine with minimal cloud cover and the wind speed was around 2 m/s. Table 1 shows the various flight plans undertaken by the UAV.



Fig 3: The DJI Phantom 4 UAV used for Data Collection

Table 1: Flight Details for the Two Flight Altitudes

SN	Parameters	100m	150m
1	Speed (m/s)	2	2
2	Shooting Angle	67°	67°
3	Front Lap	75%	75%
4	Side Lap	75%	75%
5	Resolution (cm/px)	1.78	2.03
6	No. of Flights	4	4
7	Total Flight Duration	1hr 45 mins	1hr 35 mins
8	Number of Images Collected	1 926	1 874

**2.3.3 Data Pre-processing**

Image preprocessing was the initial phase of the image classification process. Image preprocessing reduces irrelevant information contained in the images as well as improving the proportion of effective information. It further enhances the contrast of images. For this study, some of the captured UAV aerial images were exposed to excessive shadow at the edges due to cloud cover. To ensure that the above problem does not affect the training results of the CAE, the study adopted the Contrast-Limited Adaptive Histogram Equalization (CLAHE) (Zuiderveld, 1994) [45] as its image pre-processing technique to improve image contrasts. This is carried out by computing several histograms that make up the various sections of the image

and reallocates the luminance values across the entire image. The adopted CLAHE was adopted because it presents two main sub-sections namely: Block Size (BS) and Clip Limit (CL). In this case, the BS and CL were used to control the quality of the images produced during the optimization process. When the illumination rate of an image was too high, it meant that the CL values had increased because input images generally have low-intensity values and a larger CL flattened the histogram values. The CLAHE pre-processing technique is presented according to the following phases:

**Phase 1:** The original input image was divided into several non-overlapping background region proposals. The total number of image tiles presented was equal to  $M \times N$  and  $8 \times 8$  which was a good value to preserve the most representative image features contained in the images.

**Phase 2:** The histogram of the background region proposals contained in the input image was calculated according to the total number of gray levels in the image.

**Phase 3:** The contrast histogram of the background region proposal presented by the CL value was computed in eqn. (1) as:

$$M_{avg} = \left( \frac{MrX \times MrY}{M_{gray}} \right) \quad (1)$$

where  $M_{avg}$  is the average number of pixel values,  $M_{gray}$  is the total number of gray levels contained in the background region proposal, MrX and MrY represent the total number of pixels contained in the X and Y dimensions found in the background region proposal.

The CL was computed in eqn. (2) as:

$$M_{CL} = M_{clip} \times M_{avg} \quad (2)$$

Where  $M_{CL}$  is the original CL,  $M_{clip}$  represent the regularized CL within the range of (0, 1)? In the event where the number of presented pixels was far greater than  $M_{CL}$ , the pixels were clipped. The number of clipped pixels was defined as  $M_{\Sigma clip}$ , with the average number of remaining pixels awaiting distribution to each gray level expressed in Eqn. (3) as:

$$M_{avggray} = M_{\Sigma clip} \times M_{gray} \quad (3)$$

To have a fair understanding of the histogram clipping rule, the following statements were made in Eqns. (4, 5 and 6) as:

If  $H_{region}(i) > M_{CL}$  then

$$H_{region\_clip}(i) = M_{CL} \quad (4)$$

Else if  $H_{region}(i) + N_{avggray} = M_{CL}$  then

$$H_{region\_clip}(i) = M_{CL} \quad (5)$$

$$\text{Else } H_{region\_clip}(i) = H_{region}(i) + M_{CL} \quad (6)$$

Where  $H_{region}(i)$ ,  $H_{region\_clip}(i)$  represents the original and

clipped histograms located in each region proposal found at the  $i$ -th gray level.

**Phase 4:** Reorganize the remaining pixel values until all of them have been distributed. The redistribution of the pixel values is expressed in Eqn. (7) as:

$$\text{Step} = M_{gray} / M_{remain} \quad (7)$$

Where  $M_{remain}$  represent the total number of clipped pixels. The program began by searching through the entire image array to identify the minimum to maximum gray level values. The program then redistributed each pixel value to the gray level when the obtained number of pixels in the gray level was less than  $M_{CL}$ . In the case where the remaining pixel values were not entirely distributed before the search ends, the program computes a new step as shown in Eqn. (7) and then commences a new search by looping through the entire dataset until all pixels were distributed.

**Phase 5:** Rayleigh transform was employed to improve the intensity values of each image background region proposal. By this, the clipped histogram was transformed to a cumulative probability represented as  $P_{input}(i)$ , which was then used to create a transfer function. The Rayleigh transform function is represented in Eqn. (8) as:

$$y(i) = y_{min} + \sqrt{2\alpha^2} \ln \left( \frac{1}{1 - P_{input}(i)} \right) \quad (8)$$

Where  $y_{min}$  is classified to be the lower bound value of each pixel and  $\alpha$  is a scaling parameter of the Rayleigh distribution. The output probability density value for each intensity of an input image was represented in Eqn. (9) as:

$$p(y(i)) = \left( \frac{y(i) - y_{min}}{\alpha^2} \right) \cdot \exp \left( -\frac{(y(i) - y_{min})^2}{2\alpha^2} \right) \text{ for } y(i) \geq y_{min} \quad (9)$$

The result showed that an improved  $\alpha$  value produced a significant contrast enhancement of the images.

**Phase 6:** The output of the Rayleigh function specified in Eqn. (9) was re-scaled using a linear contrast stretch which was expressed in Eqn. (10) as:

$$y(i) = \frac{N(i) - N_{min}}{N_{max} - N_{min}} \quad (10)$$

Where  $N(i)$  denotes the input data value obtained from the Rayleigh function and  $N_{max} - N_{min}$  represents the minimum and maximum values presented by the Rayleigh function.

During the pre-processing phase, the images were resized and converted to grayscale with pixel values ranging from 0 to 255 with a dimension of 28 x 28.

### 2.3.4 Data Augmentation

Data augmentation is an important feature in machine learning as it adds more variation to the data in the database and enhances model performance. Mikołajczyk and Grochowski (2018) [28] described data augmentation as the process of increasing the total number of images in the dataset by applying several augmentation techniques to the original images. In this paper, four augmentation techniques namely; 90° rotation, shear, saturation, and blur were

randomly applied to the original images in the dataset. During the data augmentation phase, duplicates of the original images were created. After which a total of 15 200 images were identified in the dataset.

### 2.3.5 Dataset Construction

The dataset contained 15 200 unannotated aerial images which were divided into six classes namely; excavators, sluice board, trucks, persons, tailings dump, and crusher. The main task here was to reconstruct and classify aerial images captured at ASGM sites. Specifically, the task was to train the CAE that uses unsupervised learning on unannotated images. The images were then resized to the pixel size of  $28 \times 28$  before been fed into the neural network to commence training.

### 2.3.6 Data Splitting

The dataset ( $D$ ) was divided into two parts: a training set to train and find optimal model parameters and a validation set, to validate and measure the generalisation capability of the model (Fig. 4). The percentage ratio data split of 80/20 was chosen because improper splitting can cause high variation and bias in the model results. According to Inyaem (2018) [21], the 80/20 dataset split produces the best possible outcome on any data set. The training set contained 12 160 (80%) data samples and the validation set had 3 040 (20%) data samples. Thus, the performance of the proposed model will be assessed on 20% of data that was not used during the training phase, inversely, this data has never been exposed to the model. This ensures an equitable review of the model. The object class distribution of the images in the dataset is

shown in Fig. 5. Two variable labels were then created namely,  $x\_train$  which held the training input dataset was set to 0 and  $y\_train$  which held the validation dataset was set to 1. The pixel values of the training dataset were then rescaled in the range 0 to 1 inclusive for purposes of this study. This rescaling was necessary because it created consistency in data variability and improved the convergence speed of the algorithm. With RGB, 256 colors were contained in each channel, hence the input and output were bounded for each pixel equivalent to a minimum and maximum values of 0 and 1.

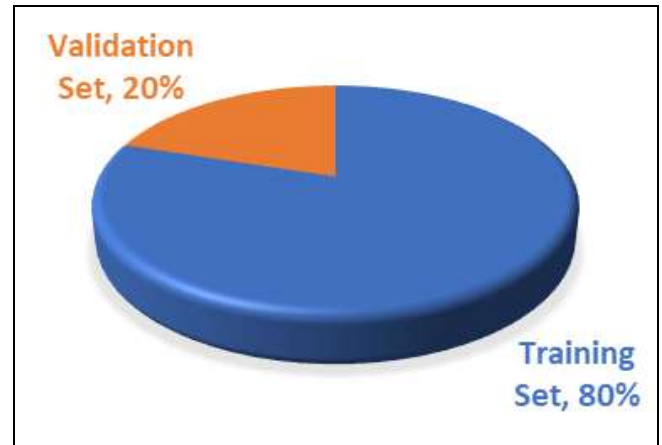


Fig 4: Training and Validation Sets

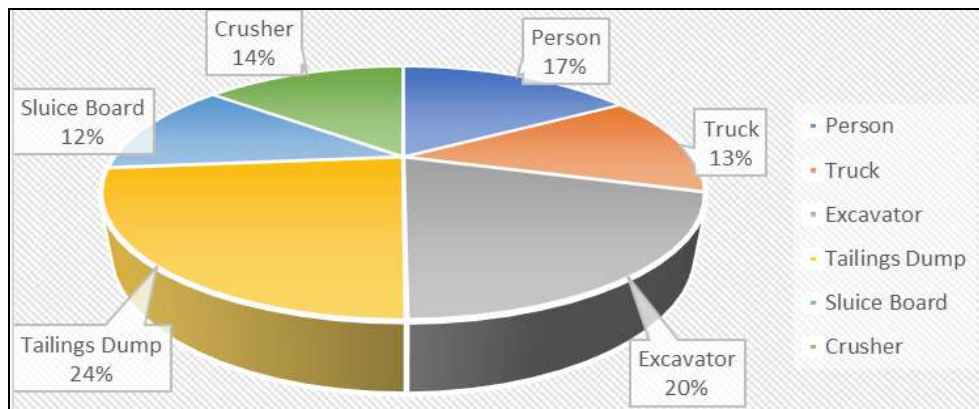


Fig 5: Object Class Distribution in the Dataset

## 2.4 Proposed CAE Structure

Autoencoder is an unsupervised machine learning technique that considers an input image and tries to reconstruct it back to the original input image. The CAE is a type of CNN that tries to reconstruct an input image patch at the output. The design of a CAE is made up of a balanced framework that contains two main phases, namely the encoding phase and the decoding phase. The encoding phase denotes half of the network, which includes convolution and max-pooling layers. On the other hand, the decoding phase denotes the other half of the network which is concerned with reconstructing the input image patches from the small-sized data representations which is made up of deconvolution and upscaling layers (Dolgikh, 2018) [14]. To achieve efficient feature learning, this paper seeks to leverage for the first time the capabilities of a CAE to classify legal and illegal ASGM sites from UAV captured images. The main goal for adopting CAE is its efficiency in quickly learning the deep

feature representations of an input image by following a particular distribution pipeline. This approach proved the effectiveness of the CAE, especially when hidden feature representations follow a defined Gaussian distribution channel. The strength of the CAE has further been echoed in the literature that during the training phase, the CAE system has no control over the learning of feature representations located in latent space. This means that each distribution channel captures a particular representative feature from the input image. Since the main goal here is to classify deep feature representations located in the hidden layers of the model, each class in the dataset was forced to pursue a defined Gaussian distribution. Moreover, to ensure that the model classifies accurately, the distributions are separated from one another. The upper shape of a CAE as shown in Fig. 6 depicts a typical structure of the proposed CAE which regenerates the input image.



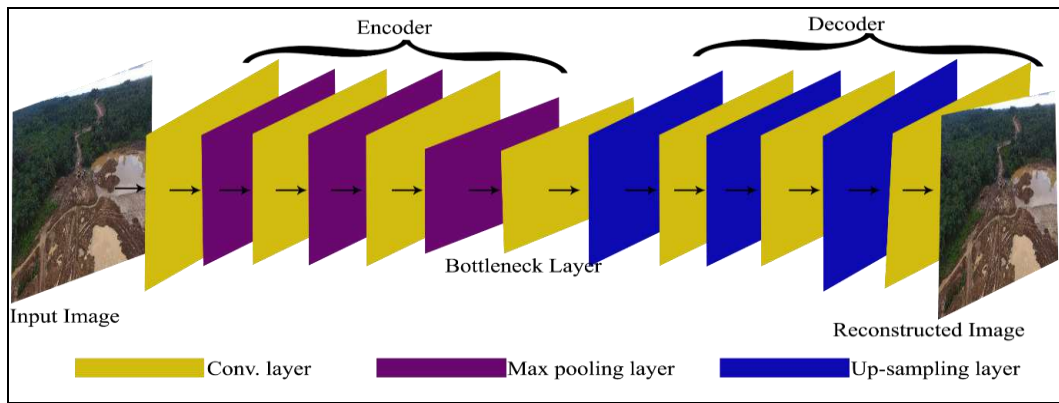


Fig 6: Structure of the Proposed CAE

### 2.4.1 CAE Architecture

The proposed CAE is aimed at predicting high-resolution images from low-resolution images provided as input. It was implemented by applying the following neural network layers as shown in Table 2 and 3 respectively. Two convolutional two-dimensional (Conv 2D) layers with output shape of (28, 28, 1) with a rectified linear units (ReLU) activation function (Clevert *et al.*, 2015) [12] was added as shown in Eqn. (11). The two layers were distinctively stored in outer layer 1 to extract highly descriptive image features from the input image before focusing on the smaller feature maps.

$$y = \max(0, x) \quad (11)$$

A Max-pooling layer, with an output shape of (14, 14, 64) was connected to the previous Conv 2D layers which reduced the dimensionality of the input images into smaller feature maps. A Dropout layer, with the shape of (14, 14, 64), was employed to regularise the neural network layers which were being executed in parallel. 2D Conv layers were introduced and applied one after the other with the same output shape of (14, 14, 128). The layers were then calculated using the activation function expressed in equation (4.36), with their results stored in outer layer 2 to undergo further processing. Another Max-pooling layer with the output shape of (14, 14, 128) was then connected to the previous 2D Conv layers. 2D Conv layers with the output shape of (64, 64, 256) was introduced and applied after the earlier max-pooling layer.

An UpSampling layer with the output shape of (14, 14, 128) was then introduced and applied to the neural network to expand the small image map into larger maps. Two 2D Conv layers were applied one after the other with the same output shape of (14, 14, 64) using the activation function of ReLU as expressed in equation (4.36). An addition procedure was carried out on the earlier 2D Conv layer with the outer layer 2 and was stored to undergo further processing. An UpSampling layer with the output shape of (14, 14, 64) was again introduced and applied to expand the small image feature maps into larger maps. An additional two 2D Conv with the same output shape of (14, 14, 32) using the same activation function ReLU expressed in equation (4.36) were again applied one after the other. An addition operation was performed on the earlier Conv layers with the results stored in the output shape (28, 28, 16) to undergo further processing. The final layer of the CAE was applied to the 2D Conv layer with the output shape of (28,

28, 1) to obtain the reconstructed image. The network architecture is shown in Fig.4.9

Table 2: Network Parameters of the Encoder

Layer Type	Filters	Stride	Output Shape
Input			28, 28, 1
2D Convolutional	64	3 x 3/1	28, 28, 64
2D Convolutional		3 x 3/1	28, 28, 64
Max-pooling		2 x 2/2	14, 14, 64
2D Convolutional	128	3 x 3/1	14, 14, 128
2D Convolutional		3 x 3/1	14, 14, 128
Max-pooling		1 x 1/1	14, 14, 128
2D Convolutional	64	3 x 3/1	7, 7, 64

Table 3: Network Parameters of the Decoder

Layer Type	Filters	Stride	Output Shape
Up-sampling		2 x 2/2	14, 14, 128
Convolutional (2D)	64	3 x 3/1	14, 14, 64
Convolutional (2D)		2 x 2/2	14, 14, 64
Up-sampling		1 x 1/1	14, 14, 64
Convolutional (2D)	32	1 x 1/1	14, 14, 32
Convolutional (2D)			14, 14, 32
Add	16	2 x 2/2	28, 28, 16
OutPut Layer		3 x 3/1	28, 28, 1

### 2.4.2 CAE Training

A two-stage approach was adopted to train the CAE namely, reconstruction and regularisation. In the reconstruction stage, the encoder and the decoder were trained to reduce the reconstruction loss in the neural network. The regularisation stage has two subsections: (i) the discriminator being the first subsection was trained to separate the results from the encoder and generate test samples from the known Gaussian distributions and (ii) the weights of the discriminator were fixed, and the encoder was trained to outwit the discriminator. Taking into consideration that the CAE can overfit the data, the dropout regularisation was employed in the dense layers of the neural network to monitor the generalisation performance on the validation set. The CAE was trained with stochastic gradient descent at momentum of 0.7 for 1000 epochs, and a learning rate of 0.001. The learning rate was chosen based on the learning rate range test.

### 2.4.3 Pseudocode of CAE

The proposed CAE pseudocode provided an overview of information flow to analyse the UAV images to ensure an effective classification.

<b>Pseudocode 1</b> <i>Process Flow for Image Classification</i>	
	Start
1.	Import libraries to be used using the import
2.	keywords;
3.	Assign variables with the 'path' variable appended
4.	with "legal_train and illegal_train"; This contains the
5.	path to the legal and illegal sites train datasets;
6.	Input: Aerial_images;
7.	Resize each image to a 28 x 28 dimension;
8.	Convert the resized images into a "grayscale";
9.	CAE.Init();
10.	CAE.Train (Aerial_image_patch);
11.	W ← CAE.Hidden_layer_weights();
12.	CAE.Init_convolution_kernel(W);
13.	maps ← CAE.Convolution(Aerial_image);
14.	Filters_maps ← CAE.Filter(maps);
15.	CAE.MaxPooling(Filters_maps);
16.	Image_feature ← CAE.Output();
	Print the classification report of the testing_data using
	keras "classification_report" function;
	End

#### 2.4.4 One-Hot-Encoding

To train the encoder head, the images in the dataset had to be segmented. To achieve that, a few dense or fully connected layers were added to the encoder part and the labels were converted into one-hot encoding vectors. According to Brett (2013) [9], Hackeling (2017) [17], and Russac *et al.* (2018) [37] in one-hot encoding, categorical data is always converted into a vector of numbers. The reason has been that machine learning algorithms experience challenges working with categorical data directly. To achieve that, one boolean column for each category or class was generated. For this study, the one-hot encoding was assigned a row vector for each image with a dimension of  $1 \times 2$ . The vector consisted of all zeros except for the class that it represented, for which it was assigned 1.

#### 2.4.5 Optimiser

The CAE model was compiled using RMSProp optimiser (Hinton, 2012) [20]. RMSProp optimiser utilizes the magnitude of recent gradients to normalize the gradients. A moving average was always kept over the root mean squared (hence RMS) gradients, by which the current gradient was divided. Fine-tuning the hyperparameters at certain stages as part of the model training was necessary to improve the generalization capability of the CAE. To achieve this, a batch size of 64 was used in the training since the specification of the computer used in the training could only handle up to a batch size of 64.

#### 2.4.6 Reconstruction Loss Function

To achieve an efficient image reconstruction, the loss function plays a crucial role. The reconstruction loss  $l_r$  is

described as computing the difference between the input and reconstructed image using the mean squared error (MSE). The loss was computed after every batch between the batch of predicted output and the ground truth using the MSE, pixel by pixel. The MSE was expressed in Eqn. (11) as:

$$l_{MSE} = \sum_{m=1}^X \sum_{n=1}^Y (a_{mn} - \hat{a}_{mn})^2 \quad (11)$$

Where  $X$  represents all the images contained in a particular batch,  $Y$  represents the total number of pixels contained in

each input image,  $a_{mn}$  denote the  $n$ -th pixel value of the  $m$ -th original input image, and  $\hat{a}_{mn}$  denote the value of  $n$ -th pixel value of the  $m$ -th reconstructed image. According to Zhao *et al.* (2016) [40], adopting the MSE loss function results in the generation of hazy reconstructed images with substantial loss of information from the original input images. To avert this problem, a CAE was introduced to learn highly descriptive UAV captured image features by optimising the Structural Similarity (SSIM) index in the loss

function:  $l_{SSIM}(a_{mn}, \hat{a}_{mn}) = 1 - SSIM(a_{mn}, \hat{a}_{mn})$ . The optimisation of the SSIM loss function encouraged the CAE to adequately preserve the highly descriptive feature information in the image patches. The optimized SSIM loss function is expressed in Eqn. (12) as:

$$SSIM(a, \hat{a}) = [l(a, \hat{a})^\alpha \cdot c(a, \hat{a})^\beta \cdot s(a, \hat{a})^\gamma] \quad (12)$$

Where  $l(\cdot)$ ,  $c(\cdot)$ , and  $s(\cdot)$  in Eqn.(12) represent the brightness level, contrast, and structural comparison functions, whereas  $\alpha$ ,  $\beta$ , and  $\gamma$  denote the component weights.

#### 2.4.7 Freezing CAE Layers

In the work of Goutam *et al.* (2020) [16], the authors affirmed that freezing some layers of a CAE during training prevents its weights from being modified as well as minimises the computational time for training the neural network. This technique is often used in transfer learning, where the base model (trained on some other dataset) is frozen. This completely avoids the backward pass to the layer which results in a significant speed boost. In this paper, the first 19 layers of the model were set to false (frozen), which decreased the training time of the model as compared to fully training all the layers of the model as shown in Table 4.



**Table 4:** Compilation of Weights of Encoder and Fully Connected Parts of CAE

Autoencoder.get_weights () [0] [1]				
Array ([[	-0.06961541,	-0.13086976,	0.06880782,	-0.17125794,
	0.11113851,	0.06572483,	0.02621359,	-0.05917768,
	-0.11628202,	0.09554031,	-0.07953136,	0.027147768,
	0.04537337,	0.14425784,	-0.10137215,	-0.02907431,
	-0.0734989,	0.13339205,	0.03168559,	-0.07791986,
	0.01853126,	-0.08251979,	0.15800025,	-0.09370226,
	-0.06152673,	-0.03043551,	-0.13401306,	-0.07720321,
	0.33415118,	0.0556346,	0.0748269,	0.07720321]],
[[	0.3960594,	0.10101996,	0.14296336,	-0.17684872,
	-0.02469493,	0.09734134,	-0.14296336,	0.16751678,
	-0.11488952,	0.131156705,	0.00483114,	-0.03776508,
	-0.22710389,	-0.1151738,	0.20672224,	-0.08622494,
	-0.0962623,	0.111811011,	0.16107674,	-0.15987864,
	0.0660238,	-0.05852052,	0.08559294,	-0.16740653,
	-0.09856632,	0.01963332,	-0.16328155,	-0.08238394,
	0.01267421,	-0.12417085,	0.0748269,	0.19871375]],
[[	-0.04455828,	-0.17048655,	-0.20608611,	-0.09370429,
	-0.09502622,	0.08972999,	0.06537726,	-0.03650601,
	0.15058747,	0.22072856,	0.18455137,	0.0362407,
	-0.04140273,	0.00323136,	-0.01676794,	0.11073503,
	-0.08076645,	0.12683366,	0.25174496,	0.09826355,
	0.060144,	-0.13479899,	-0.02915362,	0.06155493,
	-0.15445675,	-0.13378724,	-0.16701058,	-0.1683176,
	-0.04121149,	0.00905693,	-0.10834365,	0.04149643]],
dtype = float32)				
Full_model.get_weights () [0] [1]				
Array ([[	-0.06961541,	-0.13086976,	0.06880782,	-0.17125794,
	0.11113851,	0.06572483,	0.02621359,	-0.05917768,
	-0.11628202,	0.09554031,	-0.07953136,	0.02714772,
	0.04537337,	0.14425784,	-0.10137215,	-0.029074991,
	-0.0734989,	0.13339205,	0.03168559,	-0.07791986,
	0.01853126,	-0.08251979,	0.15800025,	-0.09370226,
	-0.06152673,	-0.03043551,	-0.13401306,	-0.13140804,
	0.33415118,	0.0556346,	0.17088927,	0.07720321]],
[[	0.3960594,	0.10101996,	0.14296336,	-0.17684872,
	-0.02469493,	0.09734134,	-0.18198167,	0.16751678,
	-0.11488952,	0.131156705,	0.00483114,	-0.03776508,
	-0.22710389,	-0.1151738,	0.20672224,	-0.08622494,
	-0.962623,	0.111811011,	0.16107674,	-0.15987864,
	0.0660238,	-0.05852052,	0.08559294,	-0.16740653,
	-0.09856632,	0.01963332,	-0.16328155,	-0.08238394,
	0.01267421,	-0.12417085,	0.0748269,	0.19871375]],
[[	-0.04455828,	-0.17048655,	-0.20608611,	-0.09370429,
	-0.09502622,	0.08972999,	0.06537726,	-0.03650601,
	0.15058747,	0.22072856,	0.18435137,	0.0362407,
	-0.04140273,	0.00323136,	-0.01676794,	0.11073503,
	-0.08076645,	0.12683366,	0.25174496,	0.09826355,
	0.060144,	-0.13479899,	-0.02915362,	0.06155493,
	-0.15445675,	-0.13378724,	-0.16701058,	-0.1683176,
	-0.04121149,	0.00905693,	-0.10834365,	0.04149643]],
dtype = float32)				

**3. Model Evaluation Metrics**

To evaluate the effectiveness of the CAE model, a comparative experiment was conducted between the CAE, Support Vector Machine (SVM) (Vapnik, 1998; Cortes and Vapnik, 1995)<sup>[34, 13]</sup> and Random Forest (Breiman, 2004)<sup>[8]</sup> respectively. In this paper, accuracy, precision, recall, and F1-score are the performance metrics used to assess the classification accuracy between the detection results and the ground truth test dataset. The metrics are computed in Eqns. (13) to (16), with a detailed description of the metrics found in the literature (Zhou *et al.*, 2019; Zhang *et al.*, 2018; Janssens *et al.*, 2016; Chang *et al.*, 2008)<sup>[44, 41, 42, 22, 10]</sup>.

Accuracy is the most intuitive multiclass performance measure used for validating the performance of classifiers (Alsalem *et al.*, 2018)<sup>[4]</sup>. It is the ratio of correctly predicted observations to the total observations. Accuracy was expressed in Eqn. (13) as:

$$Accuracy = \frac{TP + TN}{TP + FP + FN + FP} \tag{13}$$

Precision measures the exactness of a classifier to correctly predict positive observations against the total predicted positive observations (Alsalem *et al.*, 2018)<sup>[4]</sup>. It defines the

percentage of True Positive (TP) predictions among all other detections made by the system. The precision metric is computed in Eqn. (14) as:

$$Precision = \frac{TP}{TP + FP} \quad (14)$$

Recall measures the ratio of correctly predicted positive observations to all observations in the actual class (Alsalem *et al.*, 2018) [4]. The recall is expressed as a fraction of positive instances that are correctly classified as computed in Eqn. (15) as:

$$Recall = \frac{TP}{TP + FN} \quad (15)$$

From Eqn. (15), when the total number of detected objects increased, the probability of correctly detected objects (TP) as well as the probability of falsely detected objects (FP) also increased, thus, the precision value is decreased. Inversely, increasing the number of detected objects decreases the probability of wrongly missed objects (FN), hence, the recall value also increases. This implies that the precision and recall are inversely proportional.

The F1-score metric assesses the overall performance of a classifier as the harmonic mean of precision and sensitivity (Agaian *et al.*, 2014) [2]. The metric is given by Eqn. (16).

$$F1\ Score = 2 \times \frac{Precision \times Recall}{Precision + Recall} \quad (16)$$

From Eqn. (16), it can be deduced that as both FP and FN approached zero, F1- score also approached unity.

From the equations, True Positive (TP) represents the number of correctly detected objects by the neural network model, True Negative (TN) is the number of correctly missed objects. False Positive (FP) is the number of wrongly detected objects and False Negative (FN) is the number of wrongly missed objects.

## 4. Results and Discussion

### 4.1 Comparison of CAE, SVM and RF

The dataset used in this study is made up of aerial images captured at different ASGM using a UAV. The dataset contained a total of 15 200 unannotated images divided into 12 160 and 3 040 training and testing images respectively. The CAE was trained to accept UAV captured images as input, transform the original input image data in the latent space representation and reconstruct the original image by obtaining an optimal approximation of the underlying data representation by minimizing the reconstruction error. Table 4 shows the performance of the CAE model compared to SVM and RF when using a fixed input image size of (28 × 28). To measure how the proposed CAE model performed over the number of epochs, a function was plotted to show the loss changes over time for the training and validation sets as shown in Fig. 7. It was deduced that the lower the loss, the less the difference between the original images and the reconstructed images, thereby improving the performance of the CAE model.

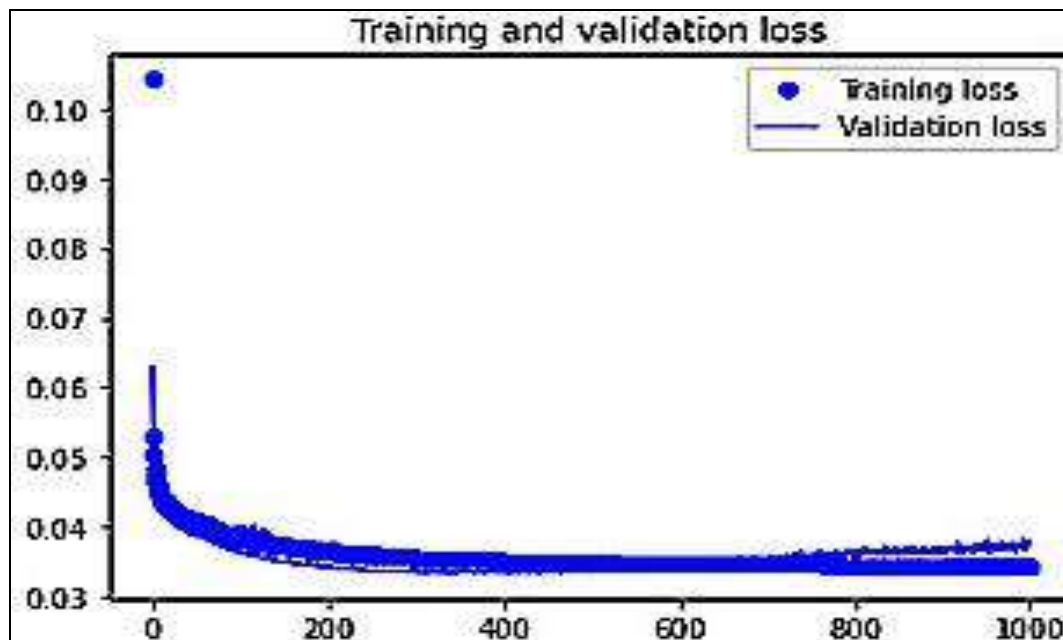


Fig 7: Training and Validation Loss Plots

It can be inferred in Fig. 7 that the loss rate was reduced to 0.055 for the training data and around 0.065 for the validation set. From the graph, it is clear that the proposed CAE was not overfitting, as the validation loss (that is, the loss calculated on unseen data samples) minimises during training. It can further be noticed that even though the validation loss was reducing, the reduction was much slower per the 1000 epochs, thereby with more iterations, the

proposed CAE could perform better. Finally, the validation set was used to reconstruct some of the images through the proposed CAE model and compared them to the original data inputs to access the effectiveness of the CAE.

In Table 5, it can be observed that the CAE achieved the best classification accuracy of 97.52% which was closely followed by SVM and RF which produced 95.66% and 93.23% respectively. The interpretation here is that the CAE

had the least misclassification rate of 2.48% while the SVM and RF had 4.34% and 6.77% respectively. The results further proved the dominance of the CAE as an efficient feature extractor, mainly because it is contrasted with a CNN, which is considered as a benchmark in supervised image classification, where having a deeper architectural network supports the performance of the network. It was also deduced that due to the nature of the CAE network, it was able to learn more complex features compared to SVM and RF. To further prove the dominance of the CAE, a morphological approach was adopted to close the small blob regions around the ground truth mask before training the image classifiers. A careful assessment was then carried out to identify the cutoff areas around the objects of interest so that the detailed object features such as texture, edges and contours are not lost. The autoencoder in the CAE helped to initialise the classifier module with the proper weights during its unsupervised training process.

The performance of the classification model SVM can be said to be comparable to that of CAE by achieving the highest precision of 97.66%. However, the major shortfall of SVM and RF is that they both have more complex network structures that require larger training dataset and a large number of trainable parameters. By contrast, the CAE produced comparable performance in a simplified manner with significantly fewer network parameters than the SVM and RF. Thereby, making the CAE more robust for image classification in heterogeneous environments such as an ASGM site. This can be confirmed by the results in Table 4. It is worth noting that all three models were trained with the reconstruction loss function expressed in Eqn. (11). CAE presented the best performance, which is mainly due to the balanced nature of the loss function. Such a loss function enabled an enhanced feature distribution learning.

Concerning the spent per training epoch, the CAE stands out without a doubt, as it required less than half the time compared to SVM and RF and even less than nine times when parameter fine-tuning was applied. This showed that the CAE was more cost-effective compared to SVM and RF not only in terms of model performance but also in processing time. During the training phase, it was noticed that the choice of an ideal filter size greatly supported the convolution process of the CAE model. In the experiments carried out, a  $3 \times 3$  filter size was used to extract the image features. By this, the CAE was able to extract more feature information and disregard unwanted feature information. The overall performance of the proposed CAE compared to SVM and RF proved its dominance as the CAE was able to learn the complex feature representations of the objects of interest (i.e., excavator, truck, person, tailings dump, crusher, and sluice board) contained in the UAV images.

**Table 5:** Classification Accuracies of the CAE, SVM and RF

Method	Input Image	Accuracy (%)	Precision (%)	Recall (%)	F1-Score (%)
Random Forest	$28 \times 28$	93.23	94.93	91.46	91.66
SVM	$28 \times 28$	95.66	97.66	94.00	89.00
CAE	$28 \times 28$	97.52	96.72	95.46	97.52

### Classification of ASGM Sites into Legal and Illegal Sites

Classifying an ASGM site into legal and illegal ASGM sites is an important part of this study as it will go a long way to push the frontiers of using UAVs to quickly capture the

locations and features of these sites and classify them using the CAE. In applying the CAE for image classification, the positional coordinates of the objects and features of interest identified in the UAV captured images were used. To achieve this, Mongo DB, an open-source document database management system was used to design two databases. The first database was used to hold all point coordinates of legal mine concessions within the study area and the second database held all detectable landforms, features and equipment that are used to define a site as an ASGM site. The point coordinates of all officially registered mine concessions were obtained from the Minerals Commission office in Tarkwa. The Minerals Commission Ghana is the official government agency responsible for supervising mining activities in Ghana, from the authorisation of mining to supervision of the enterprise and giving out licences. After the UAV captured images were pre-processed and resized into defined pixel size of  $28 \times 28 \times 1$ , the images trained on the CAE to begin the reconstruction process. With the constructed ASGM dataset consisting of images categorised into six classes namely; excavators, tailings dump, crusher, people, sluice board and trucks, the CAE was able to learn highly descriptive features of interest contained in the images. Figure 8 shows the process flow of image reconstruction and classification into legal or illegal ASGM sites. Figure 8(A) shows a sample UAV captured image that was fed into the encoder part of the CAE to begin the encoding process by learning salient features of the UAV image. In order not to lose huge amount of information when slicing and stacking up the data, the CAE was able to keep the spatial information of the input image, and gently extracted the needed information using the Convolution layer.

Deducing from Fig. 6, the results demonstrated that a flat 2D image was extracted to a thick square (Conv1), then continues to become a long cubic (Conv2) and another longer cubic (Conv3) and so on throughout the encoding phase. The entire process was designed to retain the spatial relationships that existed in the data. In the middle is the latent space with hidden layers which is composed of 10 neurons as shown in Fig. 8(B) tries to learn the objects of interest contained in the UAV image. Figure 8(C) shows the an image which has gone through the decoding process where all the cubics were flattened and reconstructed back to a 2D flat image. At this stage, it can be deduced that the CAE was able to successfully learn the very important features contained in the images and reconstruct the image which looked the same as the input image.

The objects and features learned by the CAE were then crosschecked with the detectable landforms, features and equipment database in order to define the site as an ASGM site or not. Should the learned objects match the objects in the database, that particular site is defined as an ASGM site. Once a site was defined as an ASGM site by the decoder, the output based on the point coordinates of the learned objects identified was passed to a python program which compared the point coordinates of the learned objects to the coordinates contained in the database of all legal ASGM sites. A site is flagged as illegal if its coordinates did not correspond with the coordinates in the database. This is shown in Fig. 8(D). Figure 8(E) depicts a defined site which has been classified as a legal ASGM site. This was achieved after the point coordinates of the objects of interest matched point coordinates within the legal ASGM database.





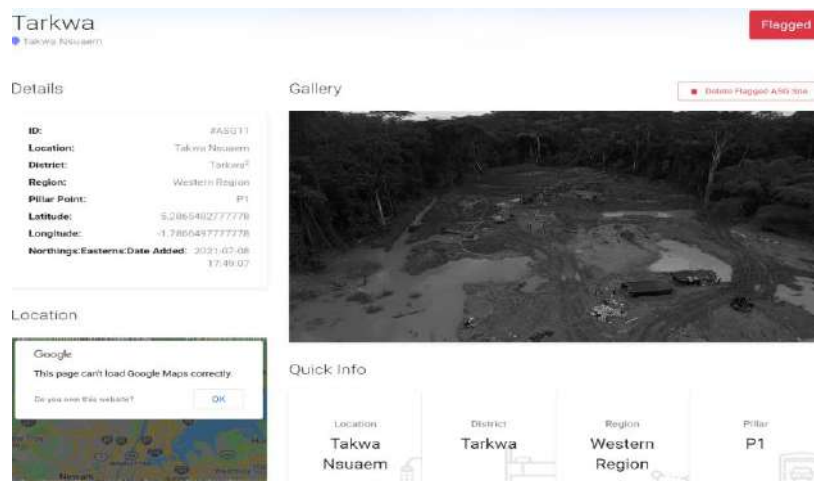
(A) UAV Captured Image as Input for Encoder



(B) The Latent Space Learning Process



(C) Reconstructed Image as Output from Decoder



Tarkwa  
Takwa Nsuaem

Flagged

Details

ID: #ASG11  
Location: Takwa Nsuaem  
District: Tarkwa<sup>2</sup>  
Region: Western Region  
Pillar Point: P1  
Latitude: 5.2865402777778  
Longitude: -1.7866497777778  
Northings: Easterns: Data Added: 2021-07-08 17:49:07

Gallery

Details Flagged ASG Site

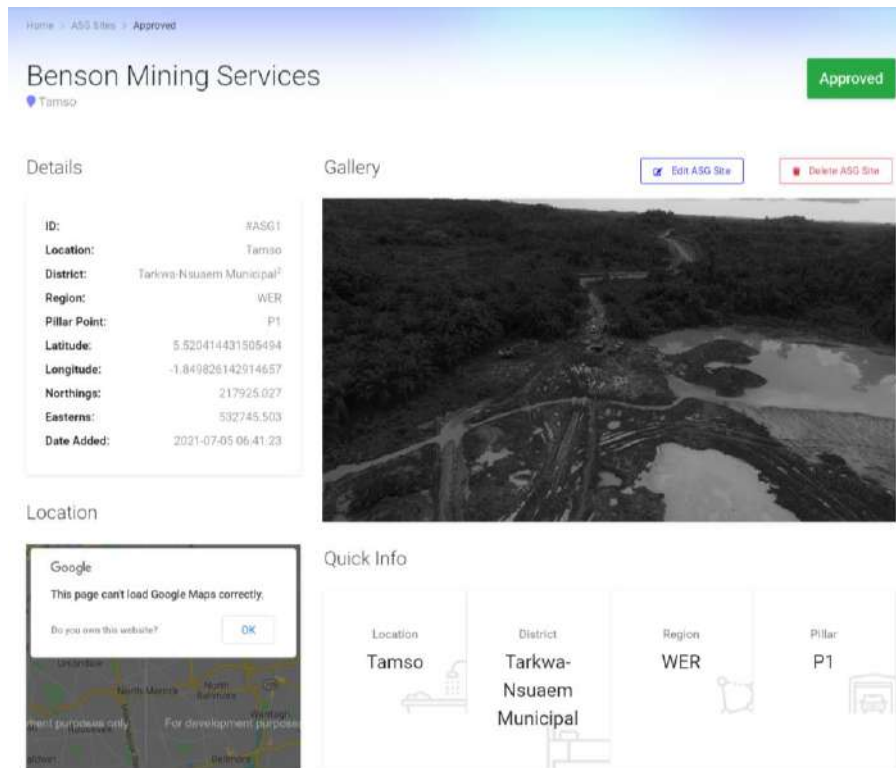
Location

Google  
This page can't load Google Maps correctly.  
Do you see this message?

Quick Info

Location	District	Region	Pillar
Takwa Nsuaem	Tarkwa	Western Region	P1

(D) ASGM Site Flagged as an Illegal Site



(E) ASGM Site Flagged as a Legal Site

Fig 8: The Image Classification Process of the CAE

## 5. Conclusion

This paper has demonstrated the effectiveness of the CAE as an efficient image classifier for the classification of UAV captured images into legal and illegal ASGM sites thus to facilitate the control and stoppage of illegal ASGM in the Ghana. The paper has further proved that UAVs can be used to capture the locations and features of all ASGM sites, which otherwise would have been inaccessible by the use of 4WD vehicles or trekking. The CAE was trained on 12 160 images and tested on the remaining 3 040 images in the dataset. The CAE operated in three main stages to define and classify a site as legal or illegal. The encoder accepted the UAV captured images as input, the latent space processed the input images by learning important image features and the decoder decoded the learned features to reconstruct the original input images thus to define a site as an ASGM site. A python program which makes use of known point coordinates of all legal ASGM sites was used to classify a defined site as legal or illegal ASGM sites. The classification performance of the CAE, SVM and RF was measured using the following evaluation metrics namely; accuracy, precision, recall, and F1-score. Experimental results after analysing the models showed that the CAE performed better by presenting a classification accuracy of 97.52% as against 95.66% and 93.23% for SVM and RF respectively. It can therefore be concluded that, UAVs and CAE is a suitable approach for image capturing and classification in ASGM environments.

## 6. References

1. Abadi M, Tensor Flow AABP. Large-Scale Machine Learning on Heterogeneous Distributed Systems. Proceedings of the 12<sup>th</sup> USENIX Symposium on Operating Systems Design and Implementation (OSDI'16), Savannah, GA, USA, 2016, 265-283.
2. Aghaian S, Madhukar M, Chronopoulos AT. "Automated Screening System for Acute Myelogenous Leukemia Detection in Blood Microscopic Images", IEEE Systems Journal. 2014;8:995-1004.
3. Akara S, Ling L, Yike G. "Feature extraction with stacked autoEncoders for epileptic seizure detection", Proceedings of the 2014 36th Annual International Conference of the IEEE Engineering in Medicine and Biology Society, Chicago, USA, 2014, 4184-4187.
4. Alsalem MA, Zaidan AA, BB Zaidan AA, Hashim M, Albahri OS, Albahri AS, *et al.* "Systematic Review of an Automated Multiclass Detection and Classification System for Acute Leukaemia in Terms of Evaluation and Benchmarking, Open Challenges, Issues and Methodological Aspects", Journal of Medical Systems. 2018;42:204.
5. Aryee BNA, Ntibery BK, Atorkui E. "Trends in the Small-Scale Mining of Precious Minerals in Ghana: A Perspective on its Environmental Impact", Journal of Cleaner Production. 2003;11:131-140.
6. Bansah K, Dumakor-Dupey N, Stemm E, Galecki G. "Mutualism, Commensalism or Parasitism? Perspectives on Tailings Trade between Large-Scale and Artisanal and Small-Scale Gold Mining in Ghana", Resources Policy. 2018;57:246-254.
7. Bansah KJ, Yalley AB, Dumakor-Dupey N. "The Hazardous Nature of Small-Scale Underground Mining in Ghana", Journal of Sustainable Mining. 2016b;15:8-25.
8. Breiman L. "Consistency for a Simple Model of Random Forests", Technical Report 670, UC Berkeley, 2004. <http://www.stat.berkeley.edu/~breiman>, Accessed: March 12, 2021.
9. Brett L. Machine Learning with R, Packt Publishing Limited, Birmingham, 2013, 458.

10. Chang YW, Wang YC, Liu T, Wang ZJ. "Fault Diagnosis of a Mine Hoist using PCA and SVM Techniques" *Journal of China University of Mining and Technology*. 2008;18:327-331.
11. Chollet F. Keras, 2015. <https://github.com/fchollet/keras>, Accessed: January 7, 2021.
12. Clevert DA, Unterthiner T, Hochreiter S. "Fast and Accurate Deep Network Learning by Exponential Linear Units (elus)". *Proceedings of the 4th International Conference on Learning Representations, ICLR 2016, San Juan, Puerto Rico, 2015*, 1-15.
13. Cortes C, Vapnik V. "Support Vector Networks", *Proceedings of the 1995 Conference in Machine Learning, Tahoe City, California, USA*. 1995;20:273-297.
14. Dolgikh S. "Spontaneous Concept Learning with Deep Autoencoder", *International Journal of Computational Intelligence System*. 2018;12:1-12.
15. Gao Y, Gao L, Li X, Zheng Y. "A Zero-Shot Learning Method for Fault Diagnosis Under Unknown Working Loads", *Journal of Intelligent Manufacturing*. 2019;31:899-909.
16. Goutam K, Balasubramanian S, Gera D, Sarma RR. "Layer Out: Freezing Layers in Deep Neural Networks", *Journal of Nature*. 2020;295:1-9.
17. Hackeling G. *Mastering Machine Learning with Scikit-Learn*, Packt Publishing Limited. Birmingham, 2017, 238.
18. Hilson G. "Small-Scale Mining, Poverty and Economic Development in Sub-Saharan Africa: An Overview", *Resources Policy*. 2009;34:1-5.
19. Hilson G, Potter C. Why is Illegal Gold Mining Activity So Ubiquitous in Rural Ghana?, *African Development Bank, Blackwell Publishing Ltd, Malden, USA*, 2003, 237-270.
20. Hinton G. "Neural Network for Machine Learning", *Online Lecture Notes*, 2012. <http://www.cs.toronto.edu/~hinton/coursera/lecture6/lec6.pdf>, Accessed: February 11, 2021.
21. Inyaem U. "Construction Model using Machine Learning Techniques for the Prediction of Rice Produce for Farmers", *Proceedings of the 2018 IEEE 3rd International Conference on Image, Vision and Computing (ICIVC), Chongqing, China, 2018*, 870-874.
22. Janssens O, Slavkovikj V, Vervisch B, Stockman K, Loccufier M, Verstockt S. "Convolutional Neural Network Based Fault Detection for Rotating Machinery", *Journal of Sound and Vibration*. 2016;377:331-345.
23. Kusimi JM, Amisigo BA, Banoeng-Yakubo BK. "Sediment Yield of a Forest River Basin in Ghana", *Catena*, 2014;123:225-235.
24. Latif S, Rana R, Qadir J, Epps J. "Variational Auto Encoders for Learning Latent Representations of Speech Emotion: A Preliminary Study", *Proceedings in 2018 Inter speech, Speech Research for Emerging Markets in Multilingual Societies, Hyderabad, India, 2017*, 3107-3111.
25. Li R, Wang P, Chen Z. "A Feature Extraction Method Based on Stacked Auto-Encoder for Telecom Churn Prediction" *Proceedings of the AsiaSim 2016, SCS Autumn Sim 2016: Theory, Methodology, Tools and Applications for Modeling and Simulation of Complex Systems, Beijing, China, 2016*, 568-576.
26. Mantey J, Owusu-Nimo F, Nyarko K, Aubynn A. "Operational Dynamics of "Galamsey" within Eleven Selected Districts of Western Region of Ghana", *Journal of Mining and Environment*. 2017;8(1):11-34.
27. Masci J, Meier U, Ciresan D, Schmidhuber J. "Stacked convolutional Auto-encoders for Hierarchical Feature Extraction", *Proceedings of the 21<sup>st</sup> International Conference on Artificial neural networks and Machine Learning, Espoo, Finland. 2011*;1:52-59.
28. Mikołajczyk A, Grochowski M. "Data Augmentation for Improving Deep Learning in Image Classification Problem", *Proceedings of the 2018 International Interdisciplinary PhD Workshop (IIPhDW), Krakow, Poland, 2018*, 1-7.
29. Owusu-Nimo F, Mantey J, Nyarko KB, Appiah-Effah E, Aubynn A. "Spatial Distribution Patterns of Illegal Artisanal Small-Scale Gold Mining (Galamsey) operations in Ghana: A Focus on the Western Region", *Heliyon*, 2018;4:534.
30. Pareto V. *Cours d'économie Politique*, Librairie Droz, Genève, Switzerland, 1964, 424.
31. Russac Y, Caelen O, He L. "Embeddings of Categorical Variables for Sequential Data in Fraud Context", *Proceedings of the 2018 International Conference on Advanced Machine Learning Technologies and Applications (AMTLA2018), Cairo, Egypt. 2018*;723:542-552.
32. Supratak A, Li L, Guo Y. "Feature Extraction with Stacked AutoEncoders for Epileptic Seizure Detection", *Proceedings of the 36th Annual International Conference of the IEEE Engineering in Medicine and Biology Society, Chicago, USA, 2014*, 4184-4187.
33. Thomas H, Felix H, Micheal P. "Global Report on Artisanal & Small-Scale Mining", *MMSD (Mining, Minerals and Sustainable Development)*. 2002;70:36-37.
34. Vapnik VN. "An Overview of Statistical Learning Theory", *Journal of IEEE Transactions on Neural Networks*. 1999;10:988-999.
35. Wetzel SJ. "Unsupervised Learning of Phase Transitions: From Principal Component Analysis to Variational Auto Encoders" *Journal of Physical Review E*. 2017;96:1-8.
36. Xavier G, Yoshua B. "Understanding the difficulty of training deep feedforward neural networks", *Proceedings of the Thirteenth International Conference on Artificial Intelligence and Statistics, 2010, Chia Laguna Resort, Sardinia, Italy, 2010*, 249-256.
37. Xu J, Sun X, Zhang Z, Zhao G, Lin J. "Understanding and Improving Layer Normalization", *Proceedings of the 33rd Conference on Neural Information Processing Systems (NeurIPS 2019), Vancouver, Canada, 2019*, 1-19.
38. Yousefi-Azer M, Varadharajan V, Hamey L, Tupakula U. "AutoEncoder-based Feature Learning for Cyber Security Application", *Proceedings of the 2017 International Joint Conference on Neural Networks (IJCNN), Alaska, Anchorage, 2017*, 3854-3861.
39. Zabalza J, Ren J, Zheng J, Zhao H, Qing C, Yang Z, *et al.* "Novel Segmented Stacked Auto Encoder for Effective Dimensionality Reduction and Feature



- Extraction in Hyperspectral Imaging”, *Journal of Neurocomputing*. 2016;185:1-10.
40. Zhao H, Gallo O, Frosio I, Kautz J. “Loss Functions for Image Restoration with Neural Networks”, *IEEE Transaction in Computer Imaging*. 2016;3:47-57.
  41. Zhang Y, You D, Gao X, Wang C, Li Y, Gao PP. “Real-Time Monitoring of High-Power Disk Laser Welding Statuses based on Deep Learning Framework”, *Journal of Intelligent Manufacturing*. 2019;31:799-814
  42. Zhang C, Cheng X, Liu J, He J, Liu G. “Deep Sparse AutoEncoder for Feature Extraction and Diagnosis of Locomotive Adhesion Status”, *Journal of Control Science and Engineering*, 2018, 1-9.
  43. Zhang Z, Jiang T, Li S, Yang Y. “Automated Feature Learning for Nonlinear Process Monitoring — An Approach using Stacked Denoising Autoencoder and K-Nearest Neighbor Rule”, *Journal of Process Control*. 2018;64:49-61.
  44. Zhou P, Zhou G, Zhu Z, He Z, Ding X, Tang C. “A Review of Non-Destructive Damage Detection Methods for Steel Wire Ropes”, *Journal of Applied Science*. 2019;9:1-16.
  45. Zuiderveld K. “Viii.5.-Contrast Limited Adaptive Histogram Equalization”, In *Graphics Gems*; Heckbert, P.S., Ed.; Academic Press: Cambridge, MA, USA, 1994, 474-48.