Design and Implementation of an Automatic Deep Stacked Sparsely Connected Auto-encoder (ADSSCA) Neural Network Architecture for Lithological Mapping under thick Vegetation using Remote Sensing

Charlie Gael Atangana Otele (nonoinfo86@gmail.com)
URIFIA
Mathias Akong Onabid
URIFIA
Patrick Stephane Assembe
University of Yaoundé-I

Research Article

Keywords: ADSSCA, Lithology Mapping, Neural Network, Autoencoder, Hidden Layers, Landsat-8 images

Posted Date: February 7th, 2023

DOI: https://doi.org/10.21203/rs.3.rs-2537926/v1

Additional Declarations: No competing interests reported.

Design and Implementation of an Automatic Deep Stacked Sparsely Connected Auto-encoder (ADSSCA) Neural Network Architecture for Lithological Mapping under thick Vegetation using Remote Sensing.

Charlie Gael Atangana Otele[a,b,*], Mathias Akong Onabid[a,c] and Patrick Stephane Assembe[d]

[a] *URIFIA, Department of Mathematics and Computer Science, Faculty of Science, P.O. Box 67 Dschang, University of Dschang, West Region – Cameroon*

[b] *Department of Computer Science, Higher Teacher Training College, The University of Bamenda,P.O. Box 39-Bambili, North West Region - Cameroon,*

[c] *Department of Computer Science, Higher Technical Teacher Training College, The University of Bamenda, P.O. Box 39-Bambili, North West Region - Cameroon,*

[d] *Postgraduate School of Science, Technology and Geosciences, Geophysics and Geo-Exploration Unit, University of Yaoundé-I, Yaoundé, Cameroon*

[*] Corresponding author:*nonoinfo86@gmail.com*

ABSTRACT

This article, proposes the design and implementation of a hybrid Deep Artificial Neural Network (DANN) to be used in mapping lithology using non-preprocessed Landsat images collected from tropical heterogeneous environments. In it, a sophisticated stacking of the hidden layers is performed through the introduction of an autoencoder topology where a wise variation of the dropout is defined at the encoder block to face the nonlinearity imposed by the limiting factors of such environments. The decoder however is left over without any dropout in order to ensure the reconstruction of the compressed data collected from the encoder. There is a relationships between the number of neurons

per hidden layer of the encoder block, the number of hidden layers of the encoder, the dropout percentage suitable to better model the dataset. The resulting architecture which we call Automatic Deep Stacked Sparsely Connected Autoencoder (ADSSCA) is an optimized hybrid neural network architecture based on well formulated rules providing in advance, the definition of, the network topology, the total number of neurons and the number of hidden layers to be used in an extremely noisy environments. The implementation of the ADSSCA on raw landsat-8 images from an area of southern Cameroon produced an overall accuracy of 92.76%. In addition, five lithological classes where identified with similar individual accuracies.

Keywords: ADSSCA, Lithology Mapping, Neural Network, Autoencoder, Hidden Layers, Landsat-8 images

1. Introduction

The network architecture and the learning algorithm are two key problems in designing a neural network system [1]. The application of DANN to remote sensing enables one to cover vast exploratory areas with little cost when compared to other exploration methods such as geological, geochemical and geophysical methods [2].

Till date, the quest for a methodology that addresses the design of a suitable DANN architecture that accurately provides the exact numbers of layers and their interconnectivity, computing units per layer, the choice of activation functions and intermediate computing units is an ongoing research area. Thus, the main concern of this research is to propose a procedure for building a robust deep learning model for multispectral lithological mapping that handles these aspects.

Of recent, deep learning has become an alternative for solving more complex, fuzzy problems including signal compression, language translation and image processing and classification. With regards to image classification, many studies focus on the application of DANN for the classification of natural images with few investigating DANN applied to multispectral image classification. Even these few are carried out on commercial Geographical Information System (GIS) platforms such as ENVI and ARGIS which do not offer the flexibility for code customization. The design of deep learning models rooted in artificial neural network induction technique explores the idea of building a multilayer perceptron with adequate layers of representations [3]. These layers pass through a training phase using data from there to get the ability of extracting more informative features suitable for the classification task. However, the complexity of the model increases with the addition of more layers which may result to model overfitting. A possible way out to this problem would be to use the dropout percentage wisely and reduce the amount of layers with a well formalized heuristic approach to yield a less complex model.

The aim of this work is the design and implementation of a robust and less complex deep learning model based on a wise utilization of the dropout. This model could be suitable for remotely sensed image classification with open source tools. Specifically, a technique for dimensioning a less complex ANN suitable for tropical forest zones without any preprocessing model. In addition, the relationships between the dropout and some hyper parameters are established to build an optimal and intelligent sparse network topology where the number of neurons per layer and the number of layers per topology are known in advance for a given problem. Furthermore, an interval for selecting the

initial number of neurons of the first hidden layer is proposed given that the ANN model configuration is still done using trial an error method. This model should be able to map the lithology of heterogeneous environments such as the Southern Regions of Cameroon where the vegetation hides the few available outcrops without any preprocessing of data.

2. Related works

Some of the neural network architectures developed recently to solve the problem of remote sensing image classification include the Simple Linear Iterative Clustering-Convolutional Neural Network (SLIC-CNN), the Deep Artificial Neural Network(DANN), the Stacked Denoising Autoencoder and the multilayer perceptron Neural Network(MLPNN). [4], who worked on Intelligent High-Resolution Geological Mapping using Unmanned Aerial Vehicle (UAV) experimented the SLIC-CNN with the objective of identifying and confirming the lithological distributions and establish the rock boundaries, obtained accuracies of 54.26% for granite dykes and 90.4% for orthogenesis. However, the SLIC-CNN is limited with image conditions and cannot therefore accurately distinguish all the dominant lithologies found. Furthermore, the collection of rock information is not possible using UAV and cameras since these minerals are covered by sand, soil or forest making it difficult to be used in heterogeneous environments. [5] worked on the detection of hydrothermal alteration zones in a tropical region using satellite remote sensing data and found that the application of remote sensing data for geological mapping in tropical environments is limited by three main obstacles including the dense vegetation cover, the persistent cloud cover and finally the limited bed rock exposures. The preprocessing techniques such as the atmospheric correction, the spectral band selection through the removal of

overlapping and inactive bands and the radiometric corrections were applied to face the above mentioned obstacles. The mineralogy of soils and rocks was detected using the Directed Principal Component (DPC) analysis of four band ratio images using vegetation index, clay mineral index, ferric iron oxide index and ferrous iron oxide index [5]. However, this model is very complex due to the preprocessing step which in addition cannot be done without loss of data. [6] used Principal Component Analysis (PCA), Independent Component Analysis (ICA) and specialized spectral band ratios to extract spectral information related to vegetation, iron oxide/hydroxide minerals, carbonate group minerals and silicification using Landsat-8 data at regional scale. Satellite-based mineral prospective maps at regional and district scales were generated for the study area by implementing the fuzzy logic model to the most informative thematic layers derived from PCA, ICA and band ratio technique.

[7], designed a DANN model for quick and accurate rock discrimination with Smartphone in which pictures of rocks used were collected through the phone's camera and the lithology of some rocks were quickly and accurately identified in a very short time. Despite the overall accuracy of 95.30% obtained, the classifier performed poorly in the recognition and classification for granites. This probably, could have been because of the high variability of their mineral content. In addition, their application could not locate images with higher similarity. [8], worked on application of lithological mapping using hyper-spectral image. They used DANN techniques to map the lithology using six diverse datasets and obtained an overall accuracy greater than 90%. Yet, some classes showed fairly unstable and low classification results. [9], proposed a DANN model based on the convolutional neural network (CNN) for lithological classification on multi-sensor

data from Mount Isa region, Australia. This lithological mapping was implemented on ArcGIS both with sentinel-2B and ASTER images. The results showed that the concurrent use of sentinel-2B and ASTER data is more accurate (75%) comparatively to the respective use of ASTER (70%) and Sentinel-2B (73%) data. [10], used a Stacked Denoising Autoencoder for Gaofen-1 satellite image classification and obtained an overall accuracy of 95.7%. However, the ability of the model to accurately discriminate between classes of similar signatures is relatively reduced, an indication that the model can fail to identify some relevant lithological classes with very low spectral distances. In addition, the number of neurons and layers of the model were randomly and manually configured through trial and error method. [11], worked on DANN with images in order to investigate the confusion and bias obtained by the classifiers on class property violations. It was found that many of the reported erroneous cases in popular DANN image classification systems occurred because the trained models confuse one class with another or show biases toward some classes over others. Then, they obtained an overall accuracy of 72% for confusion errors and 66.8% for biased error on several popular natural images. These results show that well known deep learning classifiers can generate considerable mistakes in their results thereby affecting their robustness. In [12], a Multilayer Perceptron Neural Network (MLPNN) applied to Landsat-8 (L8) images was developed to address the problem of lithological mapping using multispectral images in forest zones. Two NN models were proposed: the first one integrating a pre-processing system and the second one without any pre-processing and is therefore less complex than the first. Since these two models almost perform equally, it is thus possible to design a classification system using non calibrated images. Limitations of the two models include

the low rate of the overall accuracy (53.01% and 49.98%) and the considerable distance between some class accuracies (e.g. 96.69% for granites1 and only 15.18% for gneisses) due to a large amount of misclassification of pixels belonging to some lithological classes. Further, this MLPNN was not automated and lacked a suitable methodology that considers the specificities of the study area. Besides, the trial and error-based design of these two classification systems, their dimensioning was time consuming and error prone because of the partial exploration of the search space of potential architectures and configurations. These issues motivated the need to formulate a well-established step by step technique to design an architecture that could address the above limitations. They also necessitated the review of some deep learning architecture designed to address the lithological mapping this last decade. Despite improved performances, these algorithms lacked well established methodology that addressed in details the model configuration regarding the number of layers to be used, the number of units per layer and the connectivity between neurons of the various layers. In addition, the existing models integrated a preprocessing system for data calibration and dimension reduction, making them more complex in terms of running time and system memory allocation. Furthermore, most of the research on the application of remote sensing focus on spectral response of the targets and therefore the preprocessing step is imposed as condition to geological mapping.

These above mentioned aspects motivated the proposal of the design of an Automatic Deep Stacked Sparsely Connected Autoencoder (ADSSCA) neural network architecture using well formulated rules to automatically generate a robust deep learning model without any pre-processing system.

3. Methodology of the ADSSCA

3.1 Anatomy of an Automatic Deep Stacked Sparsely Connected Autoencoder (ADSSCA) Classification Model

An autoencoder is a Feed-Forward Artificial Neural Network (FFANN), where the final layer is having the same number of computing units as the input layer [13].The FFANN has at least an intermediate layer. The purpose of the autoencoder is to produce another representation of the inputs. The existing different types of autoencoders include stacked autoencoders, sparse autoencoders, denoising autoencoders and deep autoencoders. The novelty in this research is the proposal of deep learning architecture called ADSSCA which combines both the topology of a stacked autoencoder and the architecture of a deep neural network. In it, the deep learning system integrates in its hidden block a stacked autoencoder topology where a strong but decreasing dropout penalizes the encoding part producing a sparsely connected configuration while it is removed in the decoding part. The Fig. 1 below illustrates the anatomy of the ADSSCA learning model. The ADSSCA learning model is made up of three main blocks including the data preparation model, the encoder block and the decoder block. The system selects five multispectral bands out of the eleven bands of the L8 as described in [12]. These bands are then introduced into the ADSSCA model for training and error correction. The stopping criteria of the training process are met at the lower loss score obtained or after a given amount of iterations. The model and the related hyper parameters are saved and used for classification.

The design of this ADSSCA model is motivated by four main reasons: First, the reduction of the model complexity by bypassing the preprocessing step almost prior to

the existing classification system. In fact, the ANN models are data dependent and very sensitive to noises and small changes on data leading to overfitting. Therefore, it is very difficult and if not impossible to let the model converge in a multiclass classification context when data are note previously preprocessed [12]. Secondly the automatic design of the neural architecture and model configuration also called the ADSSCA design philosophy is coming to reduce overfitting, the black box nature of the ANN models along with their subjectivity in model configuration given that the dimensioning of the existing ANN is still done based on trial and error method. The third consideration concerns the reduction of the model variance using fixed size network architecture. The solution to this problem is to use the dropout percentage wisely. In fact, most of the machine learning models always improve the model accuracy by reducing drastically the model variance if the diversity constrain is satisfied. The final decision of the ensemble is obtained thanks to a consensus algorithm such as the majority voting or the weighted average. However, this approach is computationally intensive and time consuming with the level of complexity exponentially increasing as the number of classifiers is evolving. The dropout is the solution to those problems by permitting during the learning phase to sample a fixed model into different network architectures with a satisfactory architectural diversity constrain but, only one network is used at the test phase for the prediction [14]. The fourth consideration concerns the optimization of the search space of potential network topologies given that the search space of possible architectures is endless.

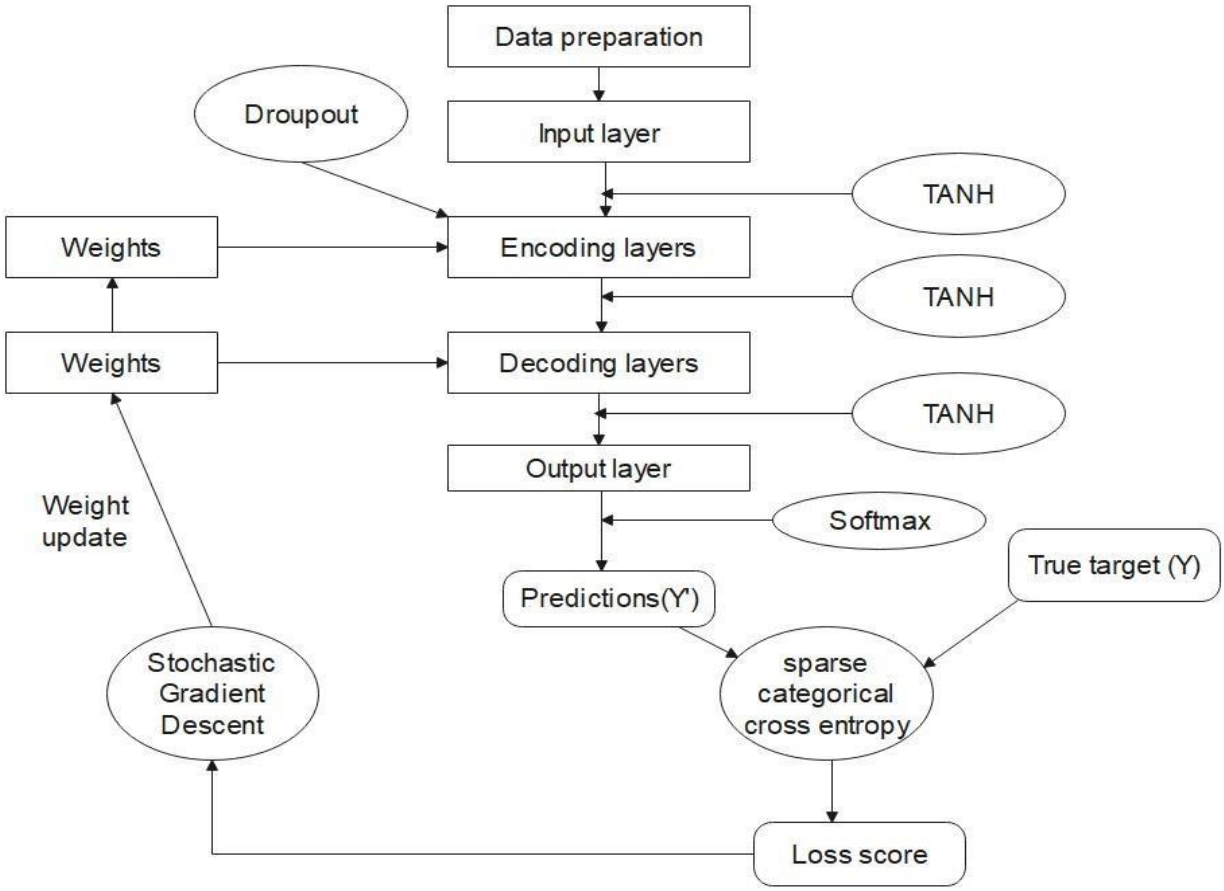

Fig.1 Functioning of the ADSSCA learning and error correction mechanism

3.1.1Data Preparation Model

The data preparation model describes the strategy used in this research to organise training and testing set in a way the ADSSCA can easily handle them including reading, writing and data labelling. We propose the following algorithm taking into consideration the extraction of ROI (Region of Interest) and the design of the training dataset.

Algorithm ADSSCADataPreparation{

-Read the image (5 bands)

-For each band do

**extraction of n ROI per label*

**merge the n ROI belonging to the same class in one matrix per band*

*-Design a N*M*P matrix where N is the number of patterns, M is the number of features and P is the number of bands.*

-Create a table containing for each class all the pixel values located in each band

-Let T=[NumberOfPaterns, NumberOfFeatures]

-For i from 1 to N do

**For j from 1 to M do*

Let t=[NumberOfFeatures]

For k from 0 to p do

Add($X^{i,j,0}$*, t);*

Add(t, T);

}

The data preparation algorithm described above is a generic model to turn remote sensing data into a suitable data format that can be directly used as input to the various machine learning algorithms. Its integration in the ADSSCA system is to facilitate the transfer of the raw remote sensing inputs intothe learning and testing modules. Given that most of the open source systems are design to receive natural images which differ from remote sensing images thanks to their multidimensionality nature, the data preparation module is an alternative to facilitate the use of those existing machine learning systems for remote sensing.

3.1.2 The ADSSCA Encoder Block

Figure 2 shows two main blocks; the input layer where a mapping between input features from the data preparation and input neurons is established and the hidden encoding layer representing the first part of the ADSSCA hidden layers. Each hidden layer in the

encoder block contains a dropout coefficient different from zero which determines the percentage of links disconnected between k^{th} layer and $(k+1)^{th}$ layer. This situation is translated in the following network topology by dropped connexions where some neurons in the k^{th}layer are not linked to all the neurons of the $(k+1)^{th}$ layer.

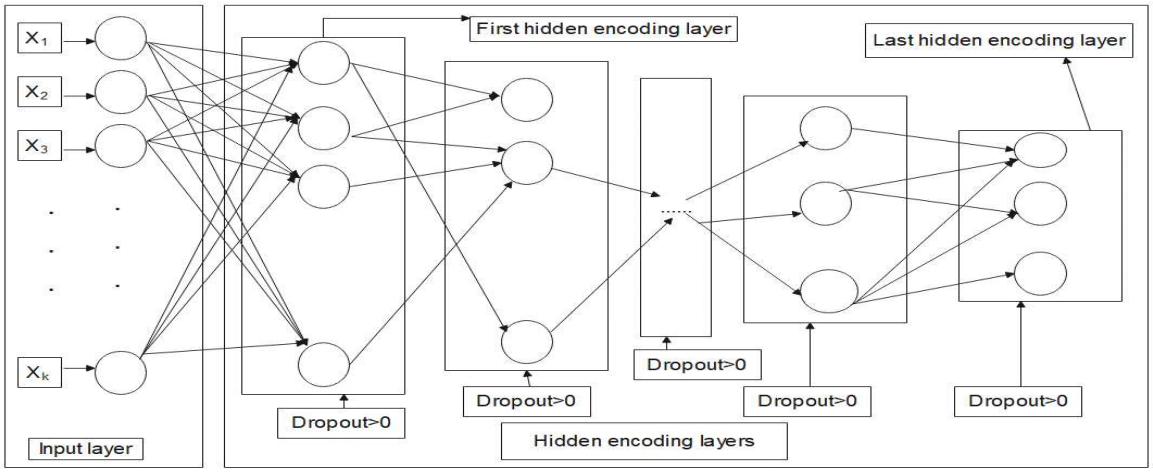

Fig. 2 Architecture of the ADSSCA encoder

3.1.3 The ADSSCA Decoder Block

The figure 3 represents the last block of the ADSSCA including the decoder and the output layer. The decoder has the same network capacity like the encoder however, the decoder is a fully connected artificial neural network. Combining both the decoder and the encoder as illustrated in figure2 and figure3 is an attempt to establish a specific composition of many functions able to capture the data context. The last encoding layer and the first decoding layer have the same number of neurons but, for practical reasons, the dropout was removed in the first decoding layer as it is also the case in all the other decoding layers of the ADSSCA producing a fully connected decoder.

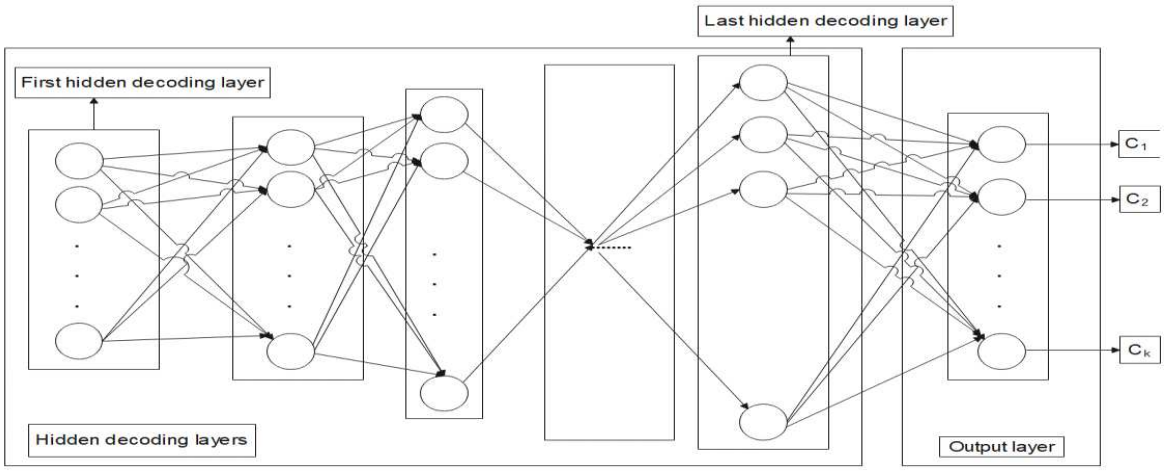

Fig.3 Architecture of the ADSSCA decoder

3.2 Architecture Design

The architecture represents the overall structure of the network including its number of units and the description of their inter-connection [15]. This work uses a chained-based architecture with some added considerations such as the disruption of some connections units, the integration of the encoding and decoding parts. The inactivity of some computing units within the model contributes in the reduction of active neurons and therefore encourages the flow of the gradient from the output to layers nearer the source as described in [15]. The encoding part combined with the dropped connections models the noisy environment while the decoding part adds additional features to refine the final representation of data. The study area in this research is heterogeneous and requires special functions to properly capture this heterogeneity. The existence of this specialized model is ensured by the universal approximation theorem of [16] stating that a feed forward artificial network containing at least one intermediate layer with any activation function is able to yield an approximation of any Borel measurable function from one finite-dimensional space to another with a not null error, with the condition of using enough intermediate computing units. The Borel measurability concept emphasizes on

the approximation by any ANN of any continuous function on a closed and bounded subset of R^n said to be Borel measurable. However, this approximation theorem says nothing with precision about the neural network topology and architecture, the required quantity of layers, the quantity of computing units per layer and the design of a specialized neural architecture.

3.2.1 Layer Configurations and Topology Design

The core of a neural network is a layer, a processing module that can filter data [3]. Specifically, layers extract representations out of the data fed into them. Classical neural network architecture is made up of three layers: an input layer with the computing units equals to the number of input features, an output layer with the number of computing units equals to the number of classes and finally a hidden layer with a random number of neurons. This type of network architecture can produce promising results in Lithological Mapping (LM) provided that it is combined with an adequate preprocessing model for data calibration and feature extractions. An alternative solution to this problem is to add more hidden layers until the model produces desired results. Fig. 4 illustrates a MLPNN with more than one hidden layer for multiclass classification. It is a DANN learning model including more than one intermediate layer for the extraction of more informative features. However, increasing the hidden layers even with a relevant preprocessing model using L8 multispectral images produces poor results during the classification of a satellite image with a MLPNN integrated in a geographical information system platform such as ENVI as seen in [12]. Furthermore, they stated that, the overfitting problem is most often present and even when the model presents good performances in training data; the generalization of the model in unknown data produces poor results.

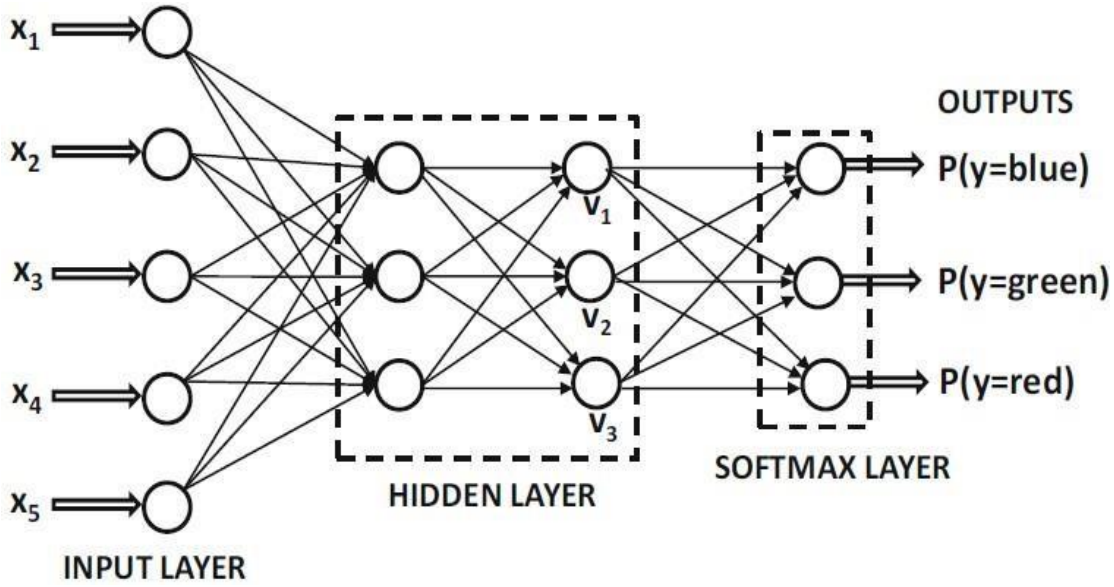

Fig. 4 An example of Multilayer feed forward neural network [17]

To solve the above mentioned weaknesses, we propose some additional rules to design an ADSSCA for remote sensing image classification:

Rule 1: Replace the hidden layers by an autoencoder topology where the number of encoding layers and the number of neurons per layer is to be determined.

Rule 2: Conceive encoding layers with strong but decreasing dropouts penalizing connection units, to be removed in all the decoding layers.

Rule 3: Establish the following relationship between the number of connections units per layer, the number of layer and the dropout percentage of each layer for the automatic configuration of the hidden part of a deep learning:

$$X_k = X_0 - 100 * K * b_0 \quad (1)$$

where k represents the encoding layer position, $X_k$the number of connection units in the k-encoding layer, X_0 is the initial number of neurons and b_0 is the initial dropout percentage coefficient.

One problem with Rule 3 is the selection of initial optimal parameters minimizing the complexity in terms of system memory usage and training time of the model while maximizing the global accuracy. This problem was solved by the introduction of a real constant called ADSSCA step. This is done by setting

$$b_{k+1} = b_0 - q(k+1) \quad (2)$$

where $0 \leq b_0 < 1$ and the ADSSCA step q is a real number defined as $0 \leq q \leq b_0$The following function OptADSSCA defines a data structure containing the overall accuracy of the ADSSCA, the number of neurones of the k-layer and the related dropout coefficient. The algorithm returns the best ADSSCA capacity by applying the three rules defined above.

SOptADSSCA(){
-Input layer: Choose the number of neurones equals the number of input variables;
-Output layer: choose the number of neurones equals the number of classes;
-Let N be the maximum number of iterations of the global artificial neural network optimisation function (OptADSSCA());
- Let S be a data structure containing the classification accuracy (p), the number of neurons in the K-layer of the hidden part of the ADSSCA and the corresponding dropout percentage;
-For i=0 to N Do
**Let X_0 be the initial number of neurons;*
**Let K be the layer position of the encoder part;*
**Let b_0 the dropout coefficient be a real number from [0..1];*
**generate the number of neurons at the k-hidden layer of the ADSSCA following*
$X_k=X_0-100*k*b_0$;

**perform the training and testing of ADSSCA (D);*

If (S.p) <D.p))

Assign parameters of D to S;

Return (S);

else

Return ("the threshold is not yet obtained");

}

S.p represents the minimal allowed threshold after training and testing which is given by the user. S.p must be greater than zero.

3.2.2 Choice of the Total Number of Layers

The total number of layers of a given ANN model is one of the most useful hyper parameter still obtained up to date using trial and error configuration mechanism. The knowledge of this parameter including a well established technique to search for it could drastically reduce the configuration time while improving the model performance and model robustness as well. In fact, one layer can be considered as a particular function enabling the neural model to focus on a specific aspect of the problem. The composition of such function creates a family composition of functions required to tackle complex situations. The calculation of this number to fit our model is given as follows:

Let NL_K be the overall number of layers in the ADSSCA model. Then we can write this formula.

$$NL_K = NL_{K^I} + NL_{K^{II}} + 2 \quad (3)$$

Where;

NL_{K^I} is the number of layers of the encoding part and

$NL_{K^{II}}$ is the number of layers of the decoding block.

Given that the ADSSCA encoder is equal to the decoder in terms of the number of layers, we have finally

$$NL_K = 2NL_{K'} + 2 \quad (4)$$

The above formula can be interpreted as the overall contribution of the various ADSSCA layers in the classification task. Given that the ADSSCA architecture is made up of two main parts including the encoder and the decoder with equal number of layers and processing units, the contribution of the two parts is equivalent to 100%. Therefore, the

contribution of the encoder is assumed to be 50% as well as that of the decoder. The total dropout values that can be used to design the encoder are given in the following equation:

$$b_k = \sum_{i=0}^{n} b_i \tag{5}$$

where b_i represents the dropout percentage obtained in layer *i*. With $0<b_i<1$.

The equation (3) is a suitable alternative to compute the number of layers found in the encoding block as defined in the following relation:

$$NL_k = \sum_{i=0}^{n} \left(sgn\left(b_i\right)\right) \tag{6}$$

Where,

$$sgn(x) = \frac{x}{|x|} = \begin{cases} -1 \; if \; x < 0 \\ 0 \; if \; x = 0 \\ 1 \; if \; x > 0 \end{cases} \tag{7}$$

Therefore, the total number of layers is obtained using (6) in (4) as defined in equation (8):

$$NL_k = 2\sum_{i=0}^{n} \left(sgn\left(b_i\right)\right) + 2 \tag{8}$$

The two layers added at the end of equation (8) represent both the input and the output layer of the global ADSSCA model.

3.2.3 Choice of the Number of Neurons and the Dropout Percentage Coefficient per Layer

The resolution of equation (1) in Rule 3 requires the initialization of three hyper parameters: the number of computing units in any encoding layer, the related dropout percentage and the ADSSCA step. Two proposed possibilities are the manual initialization and the automatic initialization, respectively. The manual initialization involved the random selection of both the number of computing units in the first encoding layer corresponding to the highest number of units per layer and the associated dropout percentage coefficient responsible for the sparsely connected property of the

encoding part. A good ADSSCA step value is the one that will generate a less complex neural network model while contributing in the improvement of the overall precision of the system. Therefore, the choice of a good ADSSCA step value is capital. In the automatic initialization however, the two hyper parameters can be generated by formulating the problem as an optimization one where equation (1) is an assumed objective function with imposed constraints. The solutions of equation (1) are summarized in Table 2 found at the implementation section, (4.3).

One problem of machine learning algorithms is that it has to be problem specific and data dependent. The architecture design and configuration therefore are to be problem specific and data dependent. One advantage of the built model is the design of a generic methodology ready to perform in almost all the figures. This is ensured by randomly initializing the number of neurons in the first encoding layer. Freedom is then given to the user at the beginning of each configuration to explore the neural search space architectures. This will enable one to find the suitable configurations corresponding to a specific problem and data into consideration.

The best initial number of neurones should be the one contributing to model accuracy improvement while minimising the model complexity. That value cannot be unique for all the problem situations due to the fact that the neural network response varies according to the training data corresponding to a given problem. However, we can propose a certain boundary that will enable us to reduce the search space of initial values. The equation (1) defines the value of X_K representing the number of neurons of layer k. Therefore, the total number of neurons of the ADSSCA can be found using the following equation:

$$N_K = N_{K^I} + N_{K^{II}} + N_{K^{III}} \quad (9)$$

where:

- N_K represents the total number of neurons of the model

- N_{K^I} represents the number of neurones of the encoder and the decoder blocks
- $N_{K^{II}}$ represents the number of neurons at the input layer corresponding to the number of input features
- $N_{K^{III}}$ represents the number of neurons at the output layers corresponding to the number of targets or classes

$$N_{K^I} = \sum_{k=2}^{n-1} X_k \tag{10}$$

Using equation (1) in equation (10) we have:

$$N_{K^I} = \sum_{k=2}^{n-1} \left(X_0 - 100 * k * b_0\right) \tag{11}$$

where:

- X_0 represents the initial number of neurons we are looking for
- b_0 represents the initial dropout value
- k represents the layer position within the ADSSCA hidden block

$$N_k = \sum_{k=2}^{n-1} \left(X_0 - 100 * k * b_0\right) + N_{K^{II}} + N_{K^{III}} \tag{12}$$

Equation (11) enables one to obtain a certain boundary to be used when selecting the initial number of neurones in model configuration:

$$\sum_{k=2}^{n-1} \left(X_0 - 100 * k * b_0\right) = \sum_{k=2}^{n-1} \left(X_0\right) - \sum_{k=2}^{n-1} \left(100 * k * b_0\right)$$

$$= (n-2)X_0 - 100b_0\left(\sum_{k=1}^{n} \left(k\right) + (n+1)\right)$$

$$= (n-2)X_0 - 100b_0\left(\frac{n(n+1)}{2} + (n+1)\right) \tag{13}$$

N_{K^I} is positive, this includes that $X_0 \gg 50(n+1)(b_0)$ (14)

Equation (14) provides the lower bound from which one can select the initial value of the neurons to configure the ADSSCA model. The problem with this technique remains the lack of the upper bound to select the neurons initial value.

3.2.4 Choice of the Transfer Functions

The mathematical expression used to find out the output of an ANN represents the transfer function. Its fundamental aim is to change an input signal of a computing unit in an ANN into an output signal. Each computing unit or neuron in an ANN is incorporating a transfer function which determines the activation or not of the computing unit, subject to whether the input signal of each computing unit is relevant for the model's output. The activation function location within an ANN is illustrated in Fig. 5.

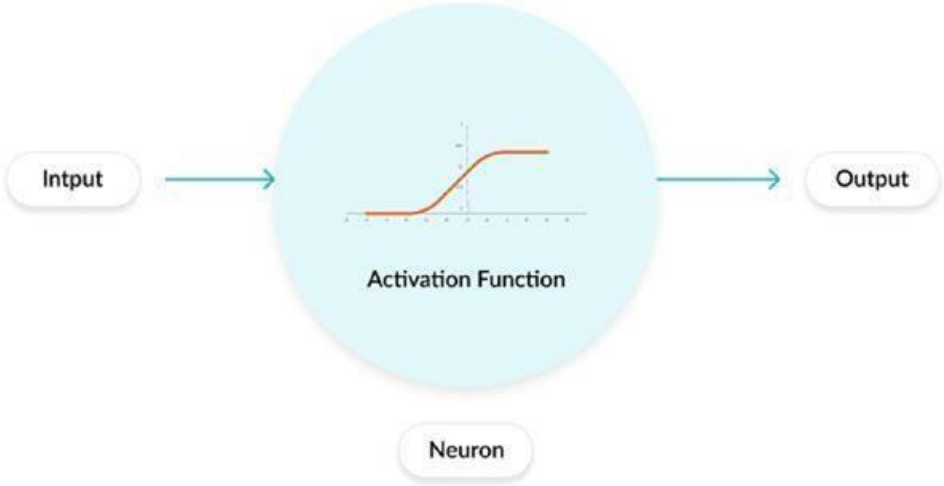

Fig. 5 The placement of the activation function [18]

The transfer function is fundamental in ANN architecture because it enables the ANN to converge by increasing the convergence speed. In a multinomial classification study such as this, where there is the need to perform back propagation and to tackle the complexity of changing model parameters in data from extremely vegetated areas, the nonlinear tangent hyperbolic and Softmax activator were used as activation functions. They fulfilled the three requirements for multinomial classification problems as in lithological mapping from remote areas and they allow the use of the gradient descent as learning algorithm. The mathematical formula of the Hyperbolic Tangent (*Tanh*) and Softmax are respectively given by equation (15) and equation (17):

$$\tanh(x) = \frac{2}{2+ exp^{(-2x)}} = 2 * sigmoid\ (2x) - 1 \quad (15)$$

where *sigmoid* function is defined in equation (16) as:

$$sigmoid(x) = \frac{1}{1+ exp^{(-x)}} \quad (16)$$

$$softmax(x) = \frac{exp^{x}}{\sum_{j=0}^{k} exp^{(x)j}} \quad (17)$$

The nonlinear transfer functions fit the ADSSCA because the concerned area to map is a natural environment and therefore is nonlinear by nature. Then, using a nonlinear transfer function enables the ADSSCA model to face the non-linearity inherent to complex data such as multispectral images of heterogeneous environments. The Hyperbolic Tangent is used in all the layers of the model except in the final layer where Softmax is used (see Table 1). There was a possibility to mix *tanh* and *logistic (sigmoid)* in both the input and hidden layers but practical reasons imposed only *tanh* as a unique combination that could yield better accuracy as shown in Table-1. Additionally, *tanh* speeds up the training and the convergence of the learning algorithm, and, stabilizes the overall accuracy despite the random initialization of the weights.

3.2.5 Selecting the Loss Function

To control the output of an ANN, the loss function is useful to measure its misfit from what is expected. Indeed, computing a distance score between the network predictions and the true targets captures the good performance of the network on this specific example. The inconsistencies between the predicted value and the actual value of the model are represented by the loss function [19]. This function is very useful in that it

summarizes all the aspects of a classifier in a single statistical value so that any increase of that statistical value is accountable for a suitable model

[20]. We can notice several functions for ANN weight error estimations [21][22][23][24][25][26] but the sparse categorical cross entropy [18] has been preferred because of the non-binary categorical outcome expected. The mathematical formulation is given by equation (18):

$$Loss = \sum_{j=0}^{m} \sum_{i=0}^{n} Y_{ij} * \log(\hat{Y}_{ji}), \quad (18)$$

where y denotes the desired output and $\hat{y}$ is the probability for each output category.

3.2.6 Choice of the Learning Strategy

After having chosen a loss function, there is need for it to be reduced since its minimization has direct positive impact on model accuracy improvement. This can be done through the learning strategy using the gradient descent algorithm used for finding suitable NN weights contributing in the minimization of the loss function, [18]. The algorithm technique is advantageous in the situation where it is difficult or even impossible to analytically compute the NN weights as it is the case in this research. For this reason, it is imperative that these weight values should be searched by an optimizer. As learning strategy, we combine stochastic Gradient Descent and mini-batch to obtain mini-batch Stochastic Gradient descent [17] which uses a batch B=$\{j_1 \ldots j_m\}$ of training points for the update. The weights are then given by:

$$\bar{W} = \bar{W} - \alpha \sum_{i \in B} \frac{\delta L_i}{\delta \bar{W}} \quad (19)$$

Where $W = (w_i - w_d)$ represents the ANN weights and $L_i$the loss contributed by the i^{th} training point.

L is the loss function of the entire neural network defined by

$$L = \sum_{i=1}^{n} L_i \quad (20)$$

The reason for choosing mini-batch stochastic gradient descent is to put together both the stochastic property and the efficiency of batch gradient optimizer. In fact, the Mini-batch gradient optimizer combines the stochastic property and the batch gradient efficiency by splitting the dataset dedicated for training in small sample sizes call batches used for error calculation and coefficient updates [18]. This has as consequence to increase the model update frequency compared to that of the batch gradient descent, and therefore allows for a more robust convergence, while preventing us from falling under the situations of local minima. This study used a batch size of 150 pixels to produce suitable results.

3.2.7 Dropout Percentages

Dropout is a regularization technique that randomly turns off a percentage of neurons at each layer, at each training step. This improves the network robustness since it does not rely on any particular set of inputs to make predictions. The knowledge is distributed amongst the whole network.

A strong dropout is imposed in the encoding part to face the heterogeneity of the study area that is caused by the dense tropical forest and the thick alteration cover which both hide the signal from the few available outcropping rocks. Also, the strong evapo-transpiration of the forest densifies the air water content (cloud) which increases noise.

The hypothesis is that the heterogeneity of the study area imposes a strong variation of the signal. For that reason, the strength of the signal from a given rock is very high where a rock exposure is available or not far from the surface while, it is very low in some areas where a deep forest layer is covering the outcrops or where the rocks are found far away from the surface due to height alteration and weathering. Therefore, a suitable model of deep learning should better capture the dominant signals of the study area taking into consideration the heterogeneity of the area with sparse deep learning topology. This enables the emphasize on the learning of redundant patterns representing the dominant signal of the area so as to generate a heterogeneous representation of data very close to that of the study area. Within, the dropout randomly penalizes some neurons to fit the context of the natural environment. Since the study area does not follow any statistical distribution, the randomness property of the dropout is therefore an opportunity to design a model capturing the maximum variability found in the natural environment. In addition, no extra programming is added to increase the complexity of the model.

The dropout percentage within a layer does not follow any statistical distribution. However, it takes into account the non-linearity of the natural environment by acting within each encoding layer as a regulariser. This is carried out by contributing in setting the number of neurones to be turned off but also in generating using equation (1) the number of neurones per layer. Furthermore, the equation (2) introduces the dropout contribution in the selection of an optimal ADSSCA step coefficient in terms of network capacity and global model accuracy.

4. **Implementation of the ADSSCA on Landsat-8 images.**

4.1 **The Study Area**

Cameroon is located in the Central African forest zone between longitude 12.354722(120 00' E) and latitude 7.369722 (60 00' N), but the area of interest is between Yaoundé (Latitude=030 52' N & Longitude=110 32' E), Sangmelima (Latitude=030 00' N & Longitude=120 15' E). The latitude of this country suggests its position in the northern hemisphere. The rocks of the study area are enriched with variety of mineral resources: these include the gneiss found around Yaoundé, the schists found along the Nyong river of Mbalmayo, the granites found toward the Sangmelima and Ebolowa zones and mica-schists found around Metet according to the existing geological map as shown in Fig. 6b. The study area is located within the densely vegetated Southern Cameroon, a region dominated by a thick lateritic soil making it difficult to find outcrops. Generally, north to south, the lithology is regrouped into two domains, the Central Africa Fold Belt and the Ntem complex Fig. 6a. However, the impact of human activities due to the various constructions and agriculture are progressively creating heterogeneous environment where multiple transitions between forest and buildings are most often found.

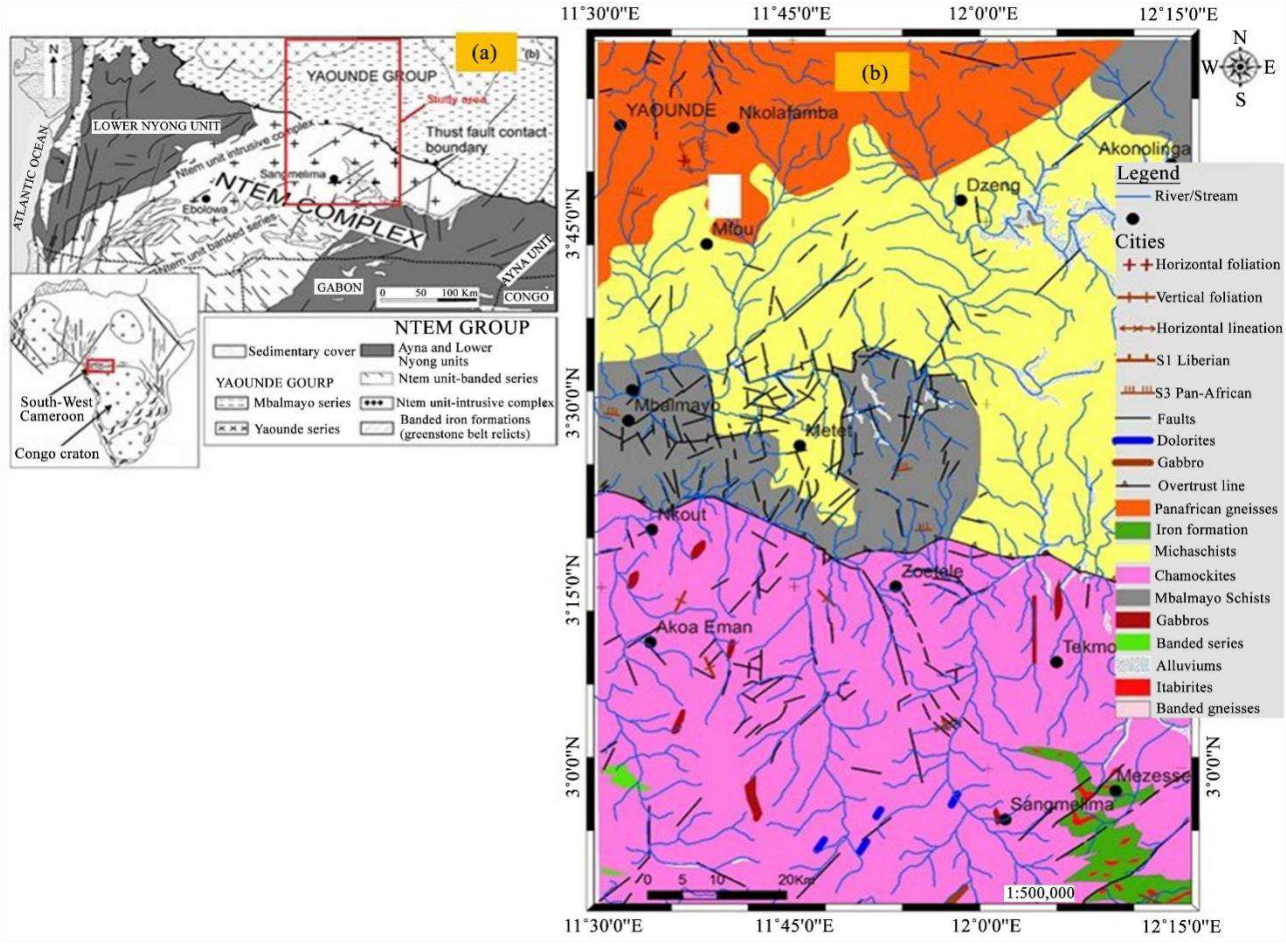

Fig. 6 Geological map of the study area [27]

4.2 Database Description

The imagery used in this work is the result of electromagnetic measurements provided by two sensors including Operational Land Imager (OLI) and Thermal Infrared (TIR) on-board of the L8 satellite. Taken at 705km of altitude, the 12th of January was suitable to collect these electromagnetic measurements since that date on the year of 2015 corresponds to the dry season in the Cameroon territory. The period of 9:27 am representing the time of image collection was appropriate to reduce the amount of noise including cloud. The scene (reference code LC81850582015012LGN01) from(http://earthexplorer.usgs.gov) corresponding to a 185 km by 181 km area is

captured at three spatial resolutions. This scene was subset from of 7731x7591x7 to 3500x3500x5 using a rasterio library [28] making an area of 185 km by 50 km. L8 bands characteristics are summarized in Table 1 below.

Table 1 Characteristic of L8 Bands [29].

L8 Spectral windows, Grand surface values (m) and Bands (μm)			
Band (B) numbers	Ground surface	Band names	Spectral windows
B1	30 m	Costal/ Aerosol	[0.435:0.451]
B2	30 m	Blue	[0.452:0.512]
B3	30 m	Green	[0.533:0.590]
B4	30 m	Red	[0.636:0.673]
B5	30 m	NIR	[0.851:0.879]
B6	30 m	SWIR-1	[1.566:1.651]
B7	30 m	SWIR-2	[2.107:2.294]
B8	15 m	Pan	[0.503:0.676]
B9	30 m	Cirrus	[1.363:1.384]
B10	100 m	TIR-1	[10.60:11.19]
B11	100 m	TIR-2	[11.50:12.51]

Landsat-8 multispectral imagery offering wavelength bands ranging from 0.435µm to 12.51µm may be sensitive to the alteration minerals in heavy vegetation cover [6].The vegetation has high absorption in the interval of 0.45 µm to 0.68 µm and high reflectance in the near-infrared (0.85 µm to 0.87µm). In addition, most minerals exhibit high reflectance in the short-infrared domain (2.10 µm to 2.29µm)

as well as high emissivity in the thermal infrared domain (10.6 µm to 12.51 µm). This is particularly the case of mineral assemblage of metamorphic rocks which dominate the study area, [30], [27]. Therefore, we selected for this study band-2 to 4 and also band-7 and 11. Fig. 7a displays the original image while Fig. 7b displays the study area in true colors extracted from the original image.

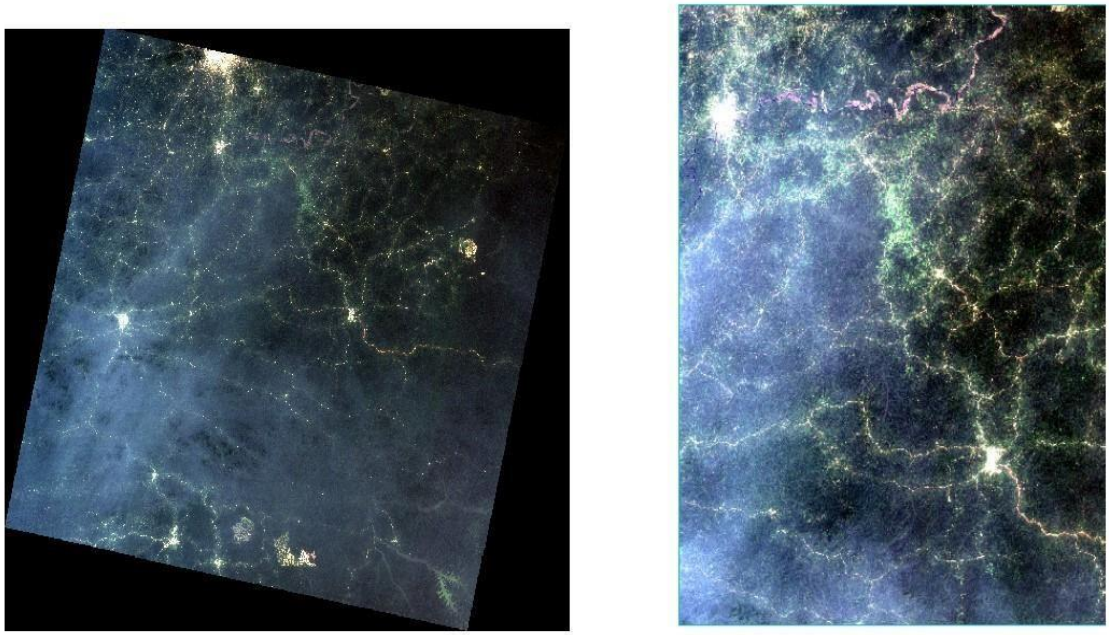

Fig. 7 *a) Raw image representing the study area b) extracted image from the study area*

4.3 ADSSCA Implementation

Considering the three rules formulated in section 3.2.1, this study developed a deep learning architecture based on one input layer with five neurons to receive the five L8 bands and five neurons in the final layer corresponding to the five dominant rock types assumed in the study area. The design of the middle part of the deep learning model follows Rule-1. A ten-layered feed forward NN is used as the hidden layers of our model. The reason is that a fully trained autoencoder hidden part behaves similarly as the Principal Components Analysis (PCA) algorithm and therefore can produce suitable descriptors to be used as input in an artificial NN in order to produce very impressive performances compared to if the normal layer disposition where used [13]. Table 2 summarizes the total configuration of the ADSSCA implemented with 300 neurons and 0.5 dropout percentages as initial parameters.

Table 2 *Configuration of the various layers of the ADSSCA*

Layer type	Number of neurons per layer	Dropout percentage	Activation functions
Input layer	5	0	Tanh
First encoding layer	300	0.5	Tanh
Second encoding layer	250	0.4	Tanh
Third encoding layer	200	0.3	Tanh
Fourth encoding layer	150	0.2	Tanh
Fifth encoding layer	100	0.1	Tanh
First decoding layer	100	0	Tanh
Second decoding layer	150	0	Tanh
Third decoding layer	200	0	Tanh
Fourth decoding layer	250	0	Tanh

Fifth decoding layer	300	0	Tanh
Output layer	5	0	Softmax

4.4 Training of the ADSSCA

A good training phase must end at the number of iterations representing the meeting point between the training curve and the test curve which in this case was not expected to be far from 600 iterations. To keep a good compromise between variance and error which would therefore avoid model overfitting the process was limited around the intersection between the two curves.

The model loss in Fig. 8 indicates a value of 0.17 corresponding to the base line error of 7.24% committed by the model for both training and test phases after 600 iterations.

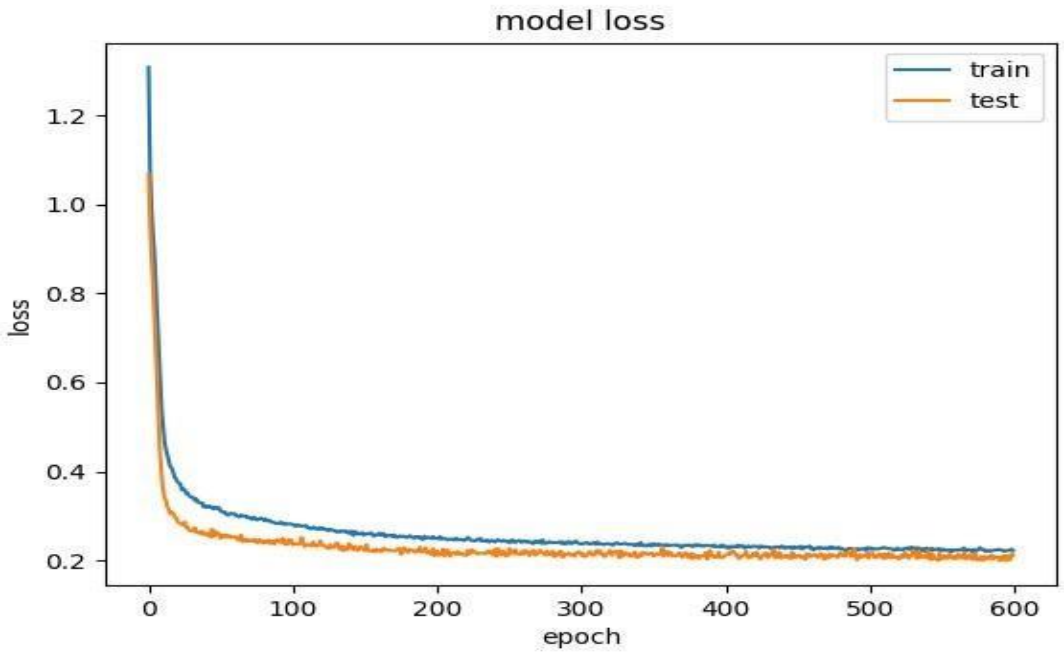

Fig. 8 Learning curves of the loss for both the training and test limited at 600 iterations.

This model loss is showing the diagram depicting the loss curve which illustrates sparse categorical cross entropy loss on the training dataset on 600 iterations. These 600iterations correspond to the point where the lowest less score is obtained during the experimental training phase. That low loss score is a proof that the dimensioning of the ADSSCA model is suitable with respect to the specific problem it is trying to solve; the accurate discrimination of lithology, consequence of an improvement in the reduction of lithological class confusion. There is a direct relationship between the dimensioning of an ANN and its ability to learn [31]. The dimensioning of an ANN also called the neural network capacity aimed at configuring the number of connections units and the number of layers [31].

The model loss diagram illustrates that the highest error committed by the system in the training process was 1.3 and was reduced up to 0.17 representing the lowest loss score obtained by the ADSSCA model after 600 iterations. This loss score corresponding to the base line error of 7.24% represents the overall distance between the expected error or error of an ideal system which is zero and that of the ADSSCA system.

4.5 Classification using the ADSSCA Model Previously Trained

Beside the proposed learning curves of the loss illustrated in Fig. 8, we proposed the curves for classification accuracy shown in Fig. 9 as statistical indicators assessing the performances of the ADSSCA classification with respect to the various classes observed in this research during both the training and the test phases. The accuracy represents a relevant indicator to assess the performance of the classifier. In terms of accuracy, the ADSSCA performed an overall accuracy of 92.76% obtained after 600 iterations. Indeed, both the training and test curves are almost about to meet as seen in Fig.9.

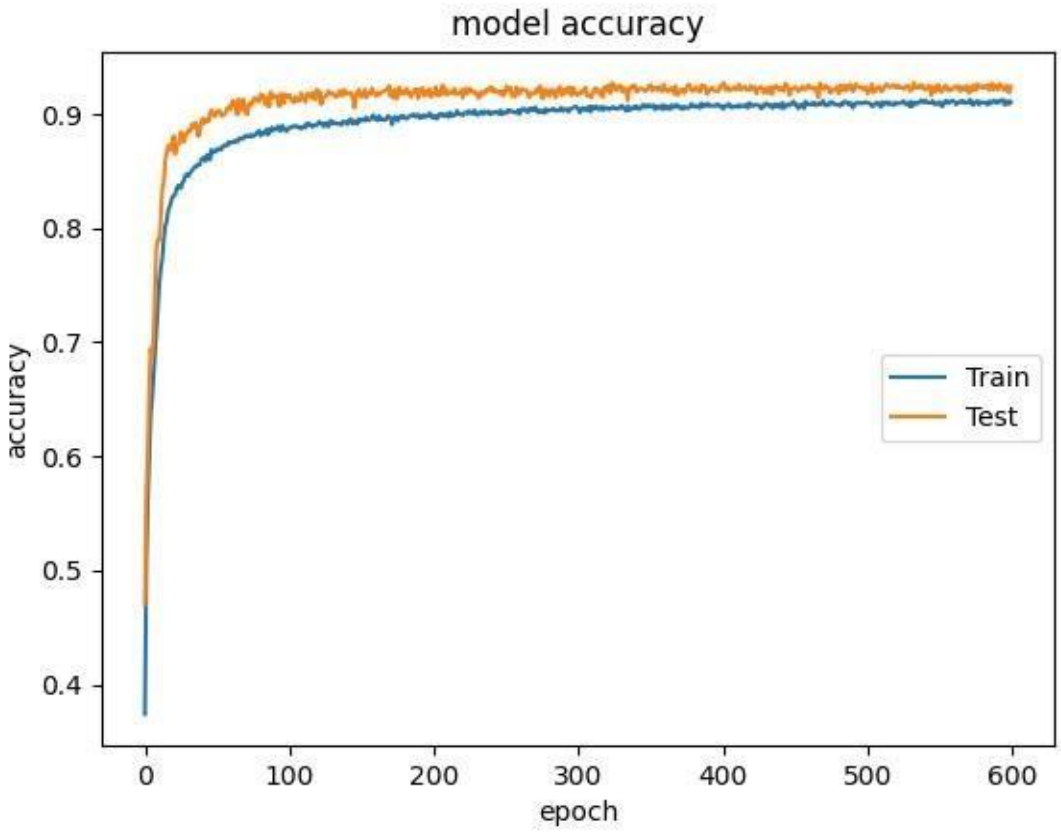

Fig. 9 Model accuracy of both training and test limited at 600iterations

4.6 The Resulting lithological Map.

The application of the ADSSCA classifier to L8 data generated a lithological map describing the spatial distribution of dominant rock types of the study area as shown in Fig. 10.

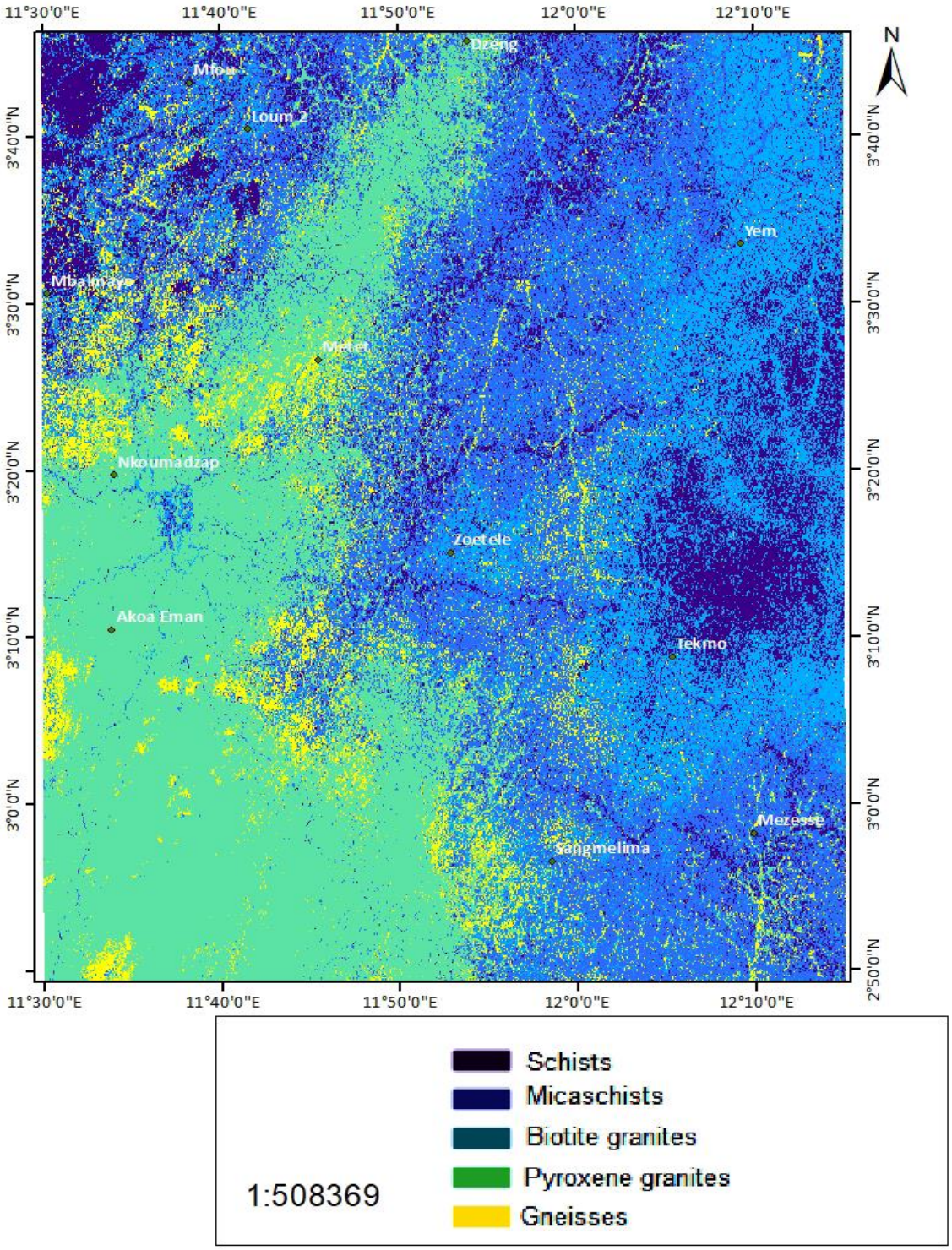

Fig. 10 Output of the lithological map obtained after 600 iterations

A summary report of the performances of the ADSSCA classification system developed in this research is presented in Table 3 for each lithological class. This table shows the five dominant lithologies at the first column including: Schists, Micaschists, Biotite Granites, pyroxene granites and Gneisses. The second column shows the precision for each rock type. The accuracy of a classification model represents not only a statistical value, but also the ability of a classification system to accurately identify the appropriate class a given sample belongs to. The mathematical formula is given by TP/ (TP+FP). In this formula, TP corresponds in number to the correct samples of a given class while FP corresponds to the samples counted as not correct for their respective classes. The third column presents the Recall values of each rock type. The recall represents a property that has a classification model to identify correct samples of the suitable class. Its formula is given by

TP/ (TP+FN). TP corresponds in number to the correct samples of a given class while FN corresponds to the samples not correct for the given class. Column 4 illustrates the values of F-beta score with beta=1.The F-beta score represents a weighted harmonic average of the accuracy and the recall. That score is always comprised between 0.0 and 1.0, where the value 1.0 corresponds to the situation of equal importance between the recall and the accuracy. Column 5 shows the Support values. These values include samples counted per class representing the Ground truth of each

lithological class.

Table 3 ADSSCA Classification performances

Thematic name	Accuracy	Recall	F1-score	Support
Gneisses	0.88	0.86	0.87	1019
Schists	0.97	0.97	0.97	914
Micaschists	0.95	0.95	0.96	1015
Biotite Granites	0.91	0.94	0.93	1015
Pyroxene Granites	0.91	0.92	0.91	977

The classification report of the above table shows that Schists and micaschists have the highest accuracy values indicating 97% and 95% respectively and equally the highest recall and F1-score values for an overall accuracy of 92.76%.

5 Discussions

The results of the study are presented in the form of lithological map, the training and testing curves and the performance table of the ADSSCA classification system. The produced map shows the spatial distribution of five lithological classes with the accuracies ranging from 88% (Gneisses) to 97% (Schists). Compared to what was obtained with the two MLPNN in [12], these results are far better in terms of individual accuracy and even in the overall accuracy as illustrated in Table 4.

Table 4 Comparison of accuracy results between the three classifiers (MLPNN (ENVI), MLPNN (KERAS) and ADSSCA)

Classifiers	Gneisses	Schists	Micaschists	Biotite Granites	Pyroxene Granites	Overall accuracy
MLPNN(ENVI)	15%	33%	79%	96%	34%	51%
MLPNN(KERAS)	28%	33%	74%	50%	30%	43%
ADSSCA	88%	97%	95%	91%	91%	92%

This improvement of classification results is due to the architectural innovations of the ADSSCA, but also to the software environment which involves Tensor flow and Keras which have well optimized libraries specialized in deep learning induction.

The training and testing curves illustrate the performances of our classification system during the training and testing periods. The model accuracy in Fig. 6 shows a global accuracy of 92.76%which is constant for more than 200 iterations. This uniformity of training and testing curves proves the stability of our classification results. The model loss as seen in Fig. 5 illustrates an overall loss score of 0.17 corresponding to the base line error of 7.24% generated by the model in a heterogeneous environment. This error is also constant after about 200iterations. This proves that the model is robust and can still perform well in highly noisy environment with inaccurate data. However, this error still needs to be reduced. Nevertheless, the two plots including the loss and accuracy curves during the training phase suggest that the ADSSCA model has a good fit on the problem.

The classification report in Table 3 presents a significant difference in terms of class accuracies between the higher class accuracy (97% for Schists) and the lower class accuracy (88% Gneisses). This possibly results on one hand from the imbalanced distribution of training data per lithological class, or from the classifier's inability to identify robust discriminative descriptors for those classes (Gneisses). Furthermore, when performing the trial and error-based classification methods, the results obtained are not stable, the global accuracy falls between 64% and 68% and the gap between the lithological classes is large. Our methodology significantly improves classification results and meets state of art standards.

The design of the ADSSCA has been well spelled out; however, the gap in terms of class accuracy remains perfectible. Hypothetically, introducing convolutional layers in the ADSSCA architecture would improve the classification accuracy. Furthermore, the integration of an ensemble classifiers or a data mining multi agent model in the fully connected section of a Deep Convolutional Neural Network (DCNN) is a good option to tackle the problem of lithological class confusion.

6 Conclusions

This research demonstrates the design of an automatic classification system applied to remote sensing, the ADSSCA, based on well formulated rules, for lithological mapping in forest and heterogeneous environments. The implementation of this ADSSCA system in the densely vegetated region of Southern Cameroon resulted to an overall accuracy of 92.76% with a less complex model including a prior data preparation model. Five lithological classes including the schists, the micachists, the biotite granites, the pyroxene ganites and the gneisses, representing the dominant classes of the study area were

discriminated. The originality of the ADSSCA is based on the absence of the preprocessing model, the open source nature of all the tools used offering the flexibility for code customization and architecture design, the minimization of a black box nature of ANN hidden layers through a well formulated methodology with precise number of layers to be used, the number of neurons per layers and the connectivity percentage automatically established between neurons of the various hidden layers. In an attempt to explain how the black box proceed to map the study area, it has been demonstrated that there exists a strong relationship between the dropout percentage, the number of layers and the number of neurons per layer for an optimal ANN architecture. The ADSSCA step coefficient was proposed to limit the number of layers required for the encoder block with respect to the dropout. This contributed to stabilize the classification results obtained no matter the random initialization of weight coefficients prior to each training phase. The heuristic formalized in this work could be used to design for the first time, robust models to effectively handle strongly noisy data in the context of low and unbalanced training samples though the overall accuracy still needs to be improved. Further investigations could be performed by improving the architectural structure of the ADSSCA through the integration of convolutional layers in the system.

Funding

This research is not supported by any organization. All the founds are provided by the authors of this work. No extra source of founding.

Conflicts of interest

There is not any conflict of interest as all the materials used in this work are open source and have been referenced including the data used.

Ethical approval

This work is an original work from common efforts of the authors. However, all the new ideas and materials used are all referenced.

Consent to participate

Charlie Geal Atangana Otele is carrying out the research under the complete supervision of Mathias Akong Onabid. Patrick Stephane Assembe provided access to data and Laboratory facilities. All in all the authors contributed equally in writing the manuscript and are working as a research team.

Consent for publication

All the data used for this work are publicly available through (reference code LC81850582015012LGN01) from (http://earthexplorer.usgs.gov) and the programming code is available under the link https://bitbucket.org/charlinois/adssca/src/ANN/

Author contributions

Charlie Geal Atangana Otele is carrying out the research under the complete supervision of Mathias Akong Onabid. Patrick Stephane Assembe provided access to data and Laboratory facilities. All in all the authors contributed equally in writing the manuscript and are working as a research team.

Computer Code Availability

- Name of code: ADSSCA
- Developers and contact address
    - Charlie Gael Atangana Otele, *URIFIA, Department of Mathematics and Computer Science, Faculty of Science, P.O. Box 67 Dschang, University of Dschang, West Region – Cameroon. Department of Computer Science, Higher Teacher Training College, The University of Bamenda,P.O. Box 39-Bambili, North West Region – Cameroon. nonoinfo86@gmail.com*

- *Year first available: 2022*

- *Hardware required: PC with 8GB of RAM*
- *Software required: free python 3.8 interpreter and pycharm to run the open source code; Tensorflow, keras, sklearn, rasterio, numpy, matplotlib all of them open source. Ubuntu20.04, 64-bit as operating system open source.*
- *Programming languge: Python*
- *Program size:15.15KB without including the images*

Uses of the various libraries and packages

- **Tensorflow**

In this research, tensorflow was used to build the deep neural network topology and train the model as described in the manuscript. The Keras Application Programming Interface compatible with Tensorflow enabled the design of the encoding part and decoding part of the ADSSCA model, by providing the flexibility to connect the various network layers in the way they could provide the desired model heterogeneity suitable for the multispectral data covering a forest zone. In addition the sparcity introduced in the encoding layers and left in the decoding layers was made possible thanks to a combination of the Tensorflow and Keras libraries. In fact, the two libraries combined enabled the use of the dropout hyper parameter responsible to turn out randomly a number of interconnecting neurons within a neural network.
https://pypi.org/project/tensorflow/

- **Sklearn**

This library was used to perform some few data pre-processing including the scaling of data. In addition a proper distribution of training and testing sets was also ensured by splitting our data set based on the recommended proportions. Furthermore, the selection of the best model based on model metric evaluations were also made possible thanks to this library.
https://pypi.org/project/sklearn/

- **Rasterio**

This library was used to open and read the landsat multispectral image one band after the other. The extraction of Region of Interest was done using well defined target windows considered as control point collected from the Cameroon geological map. These target windows contribute for signal calibration.
https://rasterio.readthedocs.io/en/latest/installation.html

- **Numpy**

Numpy is the fundamental package for scientific computing in Python. It was used to model our data set due to the multidimensionality of the multispectral images. As described in the algorithm on data preparation found in the text, each landsat spectral band was stored in a two dimensional array. Numpy was used to stack them band by band in a multidimensional array in such a way that a feature vector having as values corresponding pixel values for each band could be extracted; one per band with the dimension equals to the number of bands included in the analysis.

https://numpy.org/install/

- **Matplotlib**

This library was used to plot the training and testing curves accountable for the performances of the ADSSCA classification system. The display of the classification result map with legends representing the output of the classier was done using this library. *https://matplotlib.org/stable/users/installing/index.html*

Development tools include Python version 3.8 *(https://www.python.org/downloads/)* and Pycharm (*https://www.jetbrains.com/pycharm/download/#section=linux)*. The ADSSCA model is entirely developed in python using the Pycharm integrated development environment to facilitate programming.

Link to the code

https://bitbucket.org/charlinois/adssca/src/ANN/

References

[1] Venkatesan, R., Li, B., (2018) Convolutional Neural Network in Visual Computing: A Concise Guide. Arizona State University, Phoenix, USA, 163, 1-163,. ISBN: 9781315154282.

[2] Maliheh Abbasadeh, Ardeshir Hezarkhani, Enhancement of hydrothermal alteration zones using the spectral feature fitting method in Rabor area, Kerman, Iran, Arab J Geosci (2013) 6:1957-1964, DOI 10.1007/s12517-011-0495-0.

[3] Chollet, F., (2018) Deep Learning with Python. Manning, 386,1-386,. ISBN:9781617294433.

[4] Xuejia, S., Linfu, X., Xiangjin, R., Xiaoshun, L., Jiwen, L., Zeyu, L., (2020). Intelligent High- Resolution Geological Mapping Based on SLIC-CNN, ISPRS Int. J. Geo-Inf. 23,1-23, Doi:10.3390/ijgi9020099.

[5] Pour, Amin Beiranvand, Hashim, Mazlan, van Genderen, John, Detection of hydrothermal alteration zones in a tropical region using satellite remote sensing data: Bau

goldfield, Sarawak, Malaysia, Ore Geology Reviews (2013), doi:10.1016/j.oregeorev.2013.03.010.

[6] J. Didero TakodjouWambo, A. Beiranvand Pour, S. Ganno, P.D. Asimow, B. Zoheir, R. dos Reis Salles, J. Paul Nzenti, B. Pradhan, A.M. Muslim, Identifying high potential zones of gold mineralization in a sub-tropical region using Landsat-8 and ASTER remote sensing data: a case study of the Ngoura-Colomines goldfield, Eastern Cameroon, Ore Geology Reviews (2020), doi: https://doi.org/10.1016/ j.oregeorev.2020.103530.

[7] Guangpeng F., Feixiang C., Danyu C., Yan Li., Yanqi D., (2020). A Deep Learning Model for Quick and Accurate Rock Recognition with Smart phones, Deep Learning in Mobile Information Systems ID 74625224, https://doi.org/10.1155/2020/7462524.

[8] Bei, Y., Shufang, T., Qiuming, C., Yunzhao, G., (2020). Application of Lithological Mapping based on Advanced Hyperspectral Imager (AHSI) Imagery Onboard Gaofen-5 (GF-5) Satellite, Remote Sens. 12, 3990, 19, 1-19, doi: 10.3390/rs12233990.

[9] Brandmeier, M. and Chen, Y., (2019). Lithological Classification Using Multisensor Data And Convolutional Neural Networks, Int. Arch. Photogram. Remote Sens. Spatial Inf. Sci., XLII-2/W16, 55-59; DOI 10.5194/isprsarchives-XLII-2-W16-55-.

[10] Peng, L., Wenzhong, S., Xiaokang, Z., (2018). Remote Sensing Image Classification Based on Stacked DenoisingAutoencoder, Remote Sens, 10,16, 1-12 doi:10.3390/rs10010016. https://doi.org/10.3390/rs10010016.

[11] Yunchi T., Ziyuan Z., Vincente O., Gail K., Baishakhi R., (2020). Testing DNN Image Classifiers for Confusion and Bias Errors, arXiv:1905.07831v3, ACM ISBN 918-1-4503-7121-6, 13, 1-13 https://doi.org/101145/3377811.3380400.

[12] Atangana, C. G. O., Onabid, M. A., Assembe, P. S., Nkenlifack, M., (2021). Updated Lithological Map in the forest zone of the center south and East Regions of Cameroon using Multilayer Perceptron Neural Network and Landsat images. Journal of Geoscience and Environment Protection, 9, 120-134. https://doi.org/10.4236/gep.2021.96007.

[13] Skansi, S., (2018). Introduction to Deep Learning: From Logical Calculus to Artificial Intelligence. Springer International Publishing AG, part of Springer Nature, 189, 1-189,. ISBN 978-3-31973004-2 DOI: 10.1007/978-3-319-73004-2.

[14] Srivastava, N., Hinton, G., Krizhevsky A., Sutskever, I., Salakhutdinov R., (2014); Dropout: A Simple Way to Prevent Neural Networks from Overfitting. Journal of Machine Learning Research 15 (2014) 1929-1958.

[15] Goodfellow, L., Bengio, Y., Courville, A., (2016). Deep Learning, The MIT Press Cambridge, Massachusetts London. England 197 1-599.

[16] Hornik k., Stinchombe M., White H. (1989). Multilayer Feed Forward Networks are Universal Approximators, Neural Networks, Vol. 2, 359-366.

[17] Aggarwal, Charu, C., (2018). Neural Networks and Deep Learning, IBM T. J. Watson Research Center, International Business Machines, 1-493, 121-125. ISBN 978-3-319-94462-3 https://doi.org/10.1007/978-3-319-94463-0.

[18] El-Amir, H., Hamdy M., (2020). Deep Learning Pipeline: Building a Deep Learning Model with Tensor flow, Apress, 556, 297-334, ISBN-13 (electronic): 978-1-4842-5349-6;https://doi.org/10.1007/978-14842-5349-6.

[19] Jia Song, Shaohua Gao, Yunqiang Zhu and Chenyan Ma., (2019). A Survey of Remote Sensing Image Classification Based on CNNs, Big Eart Data, 3:3, 232-254, DOI: 10.1080/20964471.2019.1657720.

[20] Russel, D., Robert, J.,(1999). Neural Smithing: Supervised Learning in Feed Forward Artificial Neural Networks, 155,.ISBN-13: 978-0262527019.

[21] Crammer, K., Singer, Y., (2001). On the Algorithmic Implementation of Multiclass Kernel-based Vector Machines, Journal of Machine Learning Research vol. 2. School of Computer Science and Engineering Hebrew University, Jerusalem 91904, Israel, 28, 1-28.

[22] Tang, Y., (2013). Deep Learning Using Linear Support Vector Machines, 6, 1-6,arXiv preprint arXiv:1306.0239.

[23] Chopra, S., Hadsell, R., and LeCun Y., (2005). Learning a Similarity Metric Discriminatively, with Application to Face Verification. Computer Vision and Pattern Recognition (CVPR). IEEE Computer Society Conference on, volume 1, 546 539–546, DOI: 10.1109/cvpr.2005.202.

[24] Janocha, K., Czarnecki, W. M., (2017). On Loss Functions for Deep Neural Networks in Classification. Schedae Informaticae, Vol. 25, 49-59.arXivpreprint arXiv:1702. 05659,. DOI:10.4467/20838476si.16.004.6185.

[25] Zhao, H., Gallo, O., Frosio, I., and Kautz, J., (2015). Loss Functions for Neural Networks for Image Processing. Arxiv Preprint Arxiv:1511.08861.

[26] Salman, K., Hossein, R., Syed, A., S., Mohammed, B., (2018). A Guide to Convolutional Neural Networks for Computer Vision., 187, 65-68.A Publication in the Morgan and Claypool Publishers Series ISBN: 9781681730226.

[27] Assembe, S.P., Ndougsa-Mbarga, T., Enyegue A Nyam, F.M., Ngoumou, P.C., Meying, A., Gouet, D.H., ZangaAmougou, A., Ngoh, J.D., (2020). + − Evidence of Porphyry Deposits in the Ntem Complex: A Case Study from Structural and Hydrothermal Alteration Zones Mapping through Landsat-8 OLI, Aeromagnetic and Geological Data Integration in the Yaounde-Sangmelima Region (Southern Cameroon). Advances in Remote Sensing 9, 53–84. https://doi.org/10.4236/ars.2020.92004.

[28] Marek-Spartz, M., (2015). Comparing Map Algebra Implementations for Python:

Rasterio and ArcPy Department of Resource Analysis, Saint Mary's University of Minnesota, Minneapolis, MN 55404.14, 1-14.

[29] Ihlen, V., (2019). Landsat 8 Data Users Handbook, U.S. Geological Survey, Version 5.0. 114, 1-114.

[30] Sheldrake, T., Higgins, O., (2021). Classification, Segmentation and Correlation of Zoned Mineral Computers and Geosciences 156 (1-12), 12. https://doi.org/10.1016/j.cageo.2021.104876.

[31] Brownlee, J., (2016). Deep Learning With Python, Machine Learning Mastery, Melbourne Australia, V1.7. 1-245, 111-112.