Geodesy and Cartography
ISSN 2029-6991 / eISSN 2029-7009
2022 Volume 48 Issue 4: 224–232
https://doi.org/10.3846/gac.2022.15642

UDC 528.8

DEVELOPMENT OF SOFT COMPUTATIONAL SIMULATOR FOR OPTIMIZED DEEP ARTIFICIAL NEURAL NETWORKS FOR GEOMATICS APPLICATIONS: REMOTE SENSING CLASSIFICATION AS AN APPLICATION

Ahmed SERWA*

Faculty of Engineering in El-Mataria, Helwan University, Cairo, Egypt

Received 17 September 2021; accepted 12 November 2022

Abstract. Artificial neural networks (ANN) obtain more importance after the innovation of deep learning (DL) approach. This research is oriented towards development of soft computational simulator for geomatics research using ANN supporting the deep approach. ANN seems to be a black box due to its sensitivity towards initialization, architecture, and behavior. This research gives a spotlight on the dull areas of ANN algorithm by developing a soft computational simulator for it. The applied examples are chosen to cover geomatics data. DANNDO (Deep Artificial Neural Networks Designer and Optimizer) software is developed to achieve the research objective. Multi-layer perceptron (MLP) architecture is applied in this simulator. Geomatics (remote sensing multi- spectral data) is selected to be a testing paradigm to insure the reliability of the developed simulator. The developed simulator proved the high performance of applying both shallow and deep ANN (DANN).

Keywords: artificial neural networks, soft computational simulator, geomatics, remote sensing.

Introduction

Remote sensing automation is the most interesting field of study because of the need for cost reduction (Serwa & Saleh, 2021). Deep Artificial neural networks (DANN) approach is evolved from traditional Artificial Neural Networks (ANN) by taking into consideration deep learning (DL). Some of the behavior of DANN is still not clear specially in remote sensing classification (Serwa, 2017a, 2017b).

A lot of research works handle ANN in geomatics applications in addition, some of this research contains a developed software for the certain research purpose only. While this research applies ANN model using any type of geomatics data. From the previous literature one can see. Ismail et al. (2019) made a study for the use of deep artificial neural network in the optimization of the production of antifungal exochitinase compared with the response surface methodology. Chen and Cheng (2004) made design and realize the software of machining techniques manual but since 2004 it can be considered away from the recent updates. Some examples from previously trials to develop a reliable software package for geomatics applications e.g., (Serwa & El-Semary, 2016). While Mohanty (Mohanty & Majumdar, 1996) tried to develop a software package that applies ANN model for remote sensing classification using C programming language. The same problem can be found, it is temporally faraway from now and a lot of achievements are reached e.g. Serwa (Serwa, 2009, 2017a, 2017b; Serwa & Elbialy, 2020), who enhanced the performance of ANN step by step by monitoring the errors between the neurons of input layer, hidden layers and output layer of multi- layer perceptron (MLP). Sachdeva et al. (2016) developed the package – SFERCB – "Segmentation, feature extraction, reduction, and classification analysis by both SVM and ANN for brain tumors. In Some important research that proved the accuracy enhancement in remote sensing classification can be achieved by reducing the fuzzy in the final results such as Nabil et al. (2020), Serwa and El-Semary (2020), Serwa (2020).

Most recent research concerns a certain application of DANN not a general framework like this research, the reason is that there are some dark areas in DANN must be discovered by a general implementation. To

*Corresponding author. E-mail: *ahmed_serwa@yahoo.com*

find insight into the dark area of DANN one must go through it step by step and do all necessary experiments (Serwa & El-Semary, 2020).

The novelty of this research work is the development of a dependable software package that applies DANN in easy and accurate mode in a generalized methodology and the field of geomatics as an application. This research makes applying DANN very easy and solves all optimization problems.

1. DANN and DL

DANN is numerical approach that simulates the structure of the natural nervous system depending on the signal transmitted through multi hidden layers. Figure 1 indicates the default multi-layer perceptron (MLP) structure with normal learning. One can note that in DANN the number of hidden layers is larger than normal ANN. The weight updates in such case are more complex and the algorithm is very slow but gradually moves toward the optimized solution. Figure 2 shows such case. While ANN is characterized by its shallow architecture as shown in Figure 1. Some literature mentioned that both shallow and deep ANN is equivalent but recent research indicates that in case of DL it is better to get deep structure.

DL is different from DANN in both structure and architecture; it is a series of feature maps connected with both convolution and pooling layers followed by a fully connected ANN as shown in Figure 3. So, one can express DL as an evolution of DANN with some abstraction and expansion. In addition, DL depends on a stable DANN, and this is the reason behind the objective of this research work.

This research work is not concerned with development of DL software package, but it is concerned with DANN one. To achieve DL software package, one must stabilize DANN simulator at first to guarantee the success of DL one.

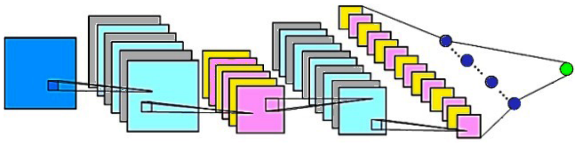


Figure 3. Typical DL architecture

2. DANN methodology

In this research work one uses deep MLP (DMLP) with propagation neural networks (BPNN) algorithm. To develop the software package of DANNDO one must stabilize the equations of BPNN. The network consists of input layer which can be expressed as (p), many hidden layers than can be expressed as (s, j and k) in addition, the final output layer which can be expressed as (l) as shown in Figure 4.

A lot of efforts were exerted, and investigations were made in addition to the real experiments were carried

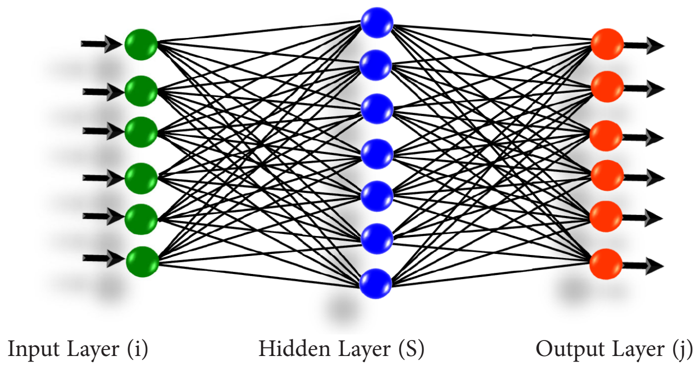


Figure 1. Typical MLP structure with shallow learning

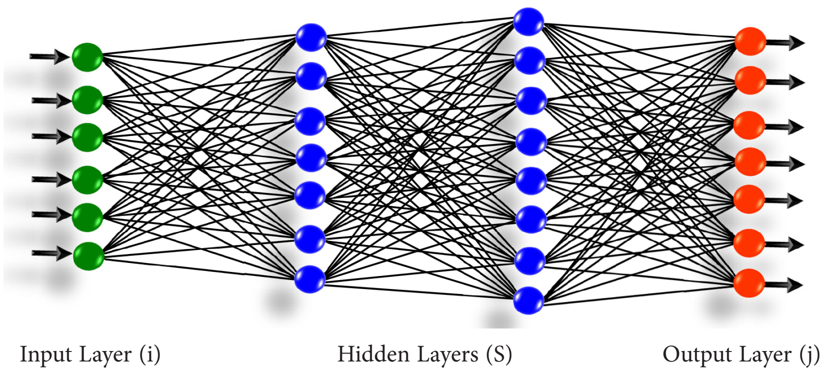


Figure 2. Typical MLP structure with deep learning

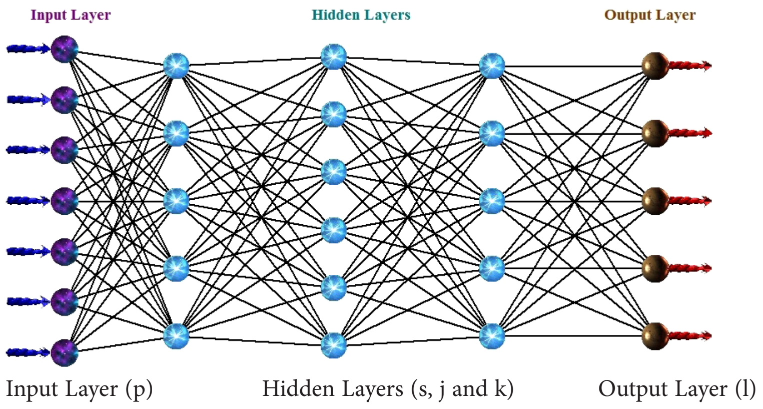


Figure 4. DANN structure

out to reach the optimized method of deep BPNN. Some of the previous equations are under argument due to the wide difference of input and output data. The variety of data in the real applications lead to a lot of assumptions in ANN in general. In this research we take the assumptions that optimize ANN e.g., the input signals to the network are always positive and cannot be negative at all.

A second assumption is the range of the input signal in case of 8-bit image (0–255). The reason for these assumptions is the structure and nature of multi-spectral images.

The input signal for input neurons in layer p is the input vector. One should notice that the input signals are transferred directly to be the output without any reduction as follows:

$$O_p = X_p. \tag{1}$$

One should know that the signal is distributed by certain weights (W_{ps}) to the first hidden layer units. After that the signals are summed as follows:

$$net_s = (\sum O_p W_{ps})\lambda, \tag{2}$$

where: O_p expresses the signal out of p^{th} input unit; W_{ps} is the weight connecting the p^{th} input layer unit with s^{th} hidden layer unit; λ expresses the gain factor that expresses the transferee strength.

After that, the output of the unit O_s in the s^{th} hidden layer can be obtained as follows:

$$O_s = F(net_s), \tag{3}$$

where F is the activation function, and it is taken as follows:

$$O_s = (1 + \exp(-net_s))^{-1}, \tag{4}$$

where: O_s is the output of the neuron in the S^{th} hidden layer which is transferred to the next hidden layer j^{th} as an input as Eq. (2) as follows:

$$net_j = (\sum O_s W_{sj})\lambda. \tag{5}$$

It is easy to get the output values for the following hidden layer which can be calculated as follows:

$$O_j = (1 + \exp(-net_j))^{-1}. \tag{6}$$

The input and the output for the following hidden layer k can be calculated respectively as follows:

$$Net_k = (\sum O_j W_{jk})\lambda; \tag{7}$$

$$O_k = (1 + \exp(-net_k))^{-1}. \tag{8}$$

While the final network output O_l will be as follows:

$$O_l = net_l(\sum O_k W_{kl})\lambda, \tag{9}$$

where: O_l is the network output, k is the ending hidden layer and W_{kl} is the weight matrix between the ending hidden layer and the output layer.

To judge the network performance, the output O_l should be compared with the target output T_l which the training data. The function (E) can be used to do so, and it is given as follows:

$$E = 0.5 \sum_{l=1} (T_l - O_l)^2, \tag{10}$$

where T_l is the target vector, O_l the network output and c is the number of clusters or classes.

The error value E_l in each unit l in the output layer can be obtained as follows:

$$E_1 = O_1(1 - O_1)(T_1 - O_1). \tag{11}$$

The new weights connecting the output units and the ending hidden layer can be calculated as follows:

$$(W_{kl})_{new} = (W_{kl})_{old} + L_R \cdot E_1 \cdot O_k, \tag{12}$$

where L_R is the learning rate which can express the learning percent rate.

E_k is the error value in the neuron k in the ending hidden layer can be computed as:

$$E_k = O_k(1 - O_k) \sum_{k=1}^{K} E_l W_{kl}. \tag{13}$$

In similar way, the new weight connecting the hidden layer k and the previous layer j is calculated as follows:

$$(W_{jk})_{new} = (W_{jk})_{old} + L_R \cdot E_k \cdot O_j. \tag{14}$$

So, the error value in the hidden unit in any hidden layers E_j can be obtained as follows:

$$E_j = O_j(1-O_j)\sum_{j=1}^{J} E_k W_{jk}. \tag{15}$$

To compute the error value in the first hidden layer, Eq. (13) can be used. Then the weight matrix connecting layer j and the input layer P as can be calculated as follows:

$$(W_{pj})_{new} = (W_{pj})_{old} + F_m \cdot E_j \cdot O_j, \tag{16}$$

where F_m expresses the momentum coefficient that judges the transferee between the first hidden layer and the input one.

The process of DANN for remote sensing classification problem can be explained in certain steps as follow:

Step 1: Input MS image.

Step 2: Spatial and spectral Sub-setting.

Step 3: Selecting training and testing data sets.

Step 4: Input the pixel vector as an input for the input layer.

Step 5: Compute the output from the input layer using Eq. *(*1).

Step 6: Compute the input for the first hidden layer using Eq. (2).

Step 7: Compute the output from the first hidden layer using Eq. (4).

Step 8: Compute the input for the next hidden layer using Eq. (5).

Step 9: Compute the output for any hidden layer using Eq. (6).

Step 10: Repeat step 8 and Step 9 for any other hidden layers.

Step11: Compute the input for the neuron in the output layer (the final output) using Eq. (9).

Step 12: Compute the network error using Eq. (10).

Step 13: if the error limit is reached or all pixels is entered go to step 22 otherwise continue.

Step 14: Compute the error in the output neuron using Eq. (11).

Step 15: Update the weights connecting the output layer and the final hidden layer using Eq. (12).

Step 16: Compute the error in the hidden neuron using Eq. (13).

Step 17: Update the weights connecting the last hidden layer and the previous one using Eq. (14).

Step 18: Repeat Step 16 and 17 for any intermediate hidden layers.

Step 19: Compute the error for the starting hidden layer neuron using Eq. (15).

Step 20: Update the weights between the input layer and the first hidden layer using Eq. (16).

Step 21: Go to step 4.

Step 22: Store the final weights.

Step 23: Apply steps 4 to 11 for all image pixels except the training dataset.

Step 24: Display the final classified image with accuracy assessment.

One must carry out both pseudo and flowchart to guarantee full understanding of the algorithm in the research. The process starts with inputting MS image, training, and testing datasets. Then initialize the weights: between input layer and first hidden layer, between hidden layers and finally connecting the ending hidden layer and the output layer. After computing the network output using the initialized weights then compute the error between this output and the training true data. Then the correction must be carried out in the back direction to correct the old weights with new values. The network output can be computed again using the new weights. The process is iterated until reaches the stopping condition when the error is negligible. The final weights are stored to be used for all image data to obtain the final classified image. The implementation of the previous algorithm can be represented using the flowchart methodology as shown in Figure 5.

3. DANNDO development

To achieve the research objective SW development using waterfall model is followed as shown in Figure 6 and can be detailed as follows:

Initiation: A review of current systems was performed to guarantee novelty and feasibility. There is no freeware or open-source software dealing with DANN in remote sensing in the research or educational form.

Feasibility: The feasibility study is carried out on the current commercial systems to see if it feasible to make a new system is feasible or not. After finishing the initiation and feasibility, a plan for stages of the cycle is well determined.

Investigation: A detailed investigation of the users' (geomatics engineers, students, and researchers) needs is carried out to specify the requirements and specifications of the novel system. From this investigation one can identify the inputs to the system, processes inside the system, outputs from the system, and data flows necessary for SW development.

Requirements: The requirements are determined by assuming a real-life problem that can be solved by system. The requirements may include SW, HW and technical support that are involved in the system.

Analysis: The analysis is carried out by simplifying the system into small entities. Then every single entity is studied through its input, processing, and outputs. Then studying the interaction between every entity and the other entities inside the system.

Specifications: The specifications of the outputs can be obtained by the system users in such case remote sensing technology dealers. Output format, accuracy and reports

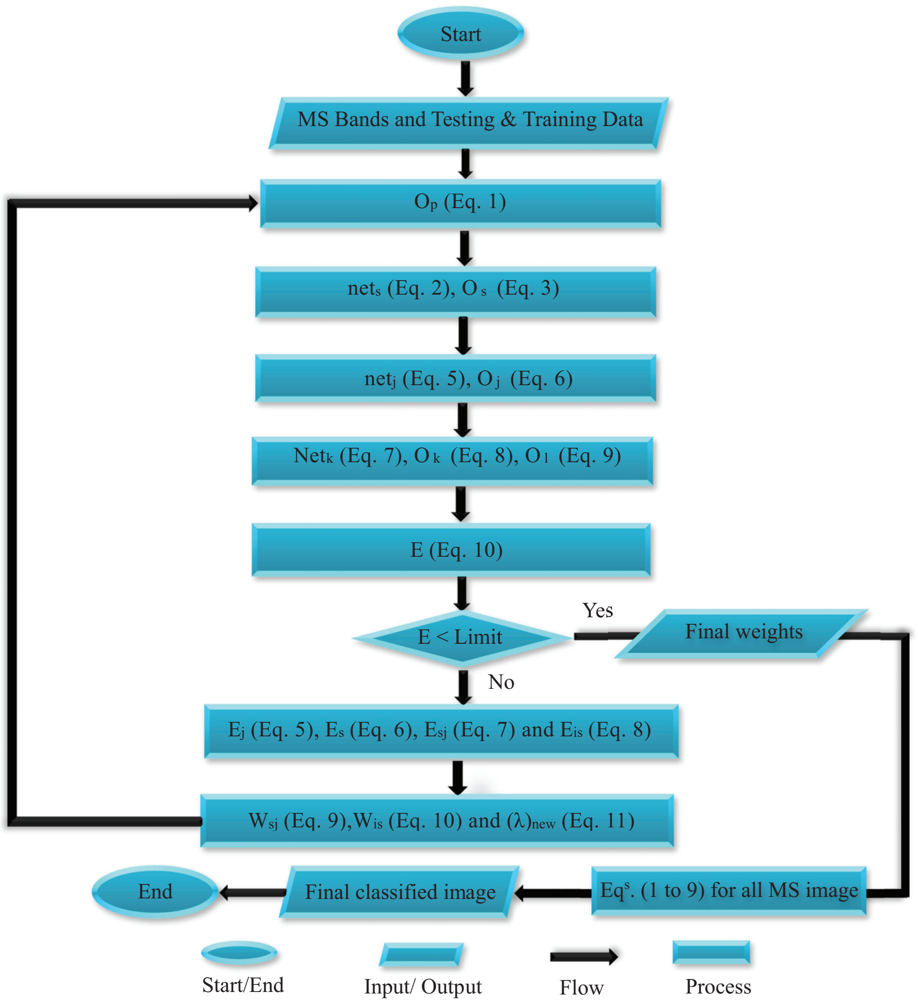

Start
MS Bands and Testing & Training Data
Op (Eq. 1)
nets (Eq. 2), Os (Eq. 3)
netj (Eq. 5), Oj (Eq. 6)
Netk (Eq. 7), Ok (Eq. 8), Ol (Eq. 9)
E (Eq. 10)
E < Limit
Yes
Final weights
No
Ej (Eq. 5), Es (Eq. 6), Esj (Eq. 7) and Eis (Eq. 8)
Wsj (Eq. 9),Wis (Eq. 10) and (λ)new (Eq. 11)
End
Final classified image
Eqs. (1 to 9) for all MS image
Start/End
Input/ Output
Flow
Process

Figure 5. DANN using BPNN

differ from user to another according to the nature of the current application.

After the previous stages the DANNDO was selected for the name of the developed SW.

Design: This stage identifies how the DANNDO system appears and runs in details such as the graphical user interface (GUI) and implementation (coding).

Build: One can use the design stage plan to converts it into computer programming language syntax (code). Codes are written for every part of the system, normally done as modules in the project.

Testing: DANNDO is tested to ensure that it acts well its tasks which were identified during the design stage. Test data are used to insure the reliability of DANNDO.

Implementation: This step ensures that there are very few or no problems for system users. When the system is being installed the users are trained in how to use the new system.

Maintenance: the users of the developed system can report bugs or problems in the system they find. Improvements and modifications can be applied to the system or the cycle and maybe we need to return to the first stage according to the necessity.

3.1. DANNDO modules

DANNDO SW is developed using VB as an interfacing programming language with com components written in C++. It is following rapid application development (RAD) strategy by using the skeleton of a past ENNIGMA. DANNDO has the following forms:

- *Starting form:* Contains author affiliation, institution and contact information for inquires as shown in Figure 7.
- *Identification form:* Identifies the SW name and purpose and indicates the supported data types as shown in Figure 8. It is designed to indicate the general nature of the novel system.
- *Main form:* Contains the commands and the network structure as shown in Figure 9. It is designed to indicate the structure of the applied neural network which include input neurons in the input layer, hidden neurons in the hidden layers and output neurons in the output layer. Each layer type appears in a certain pattern. In addition, both inputs and outputs data have their own pattern.

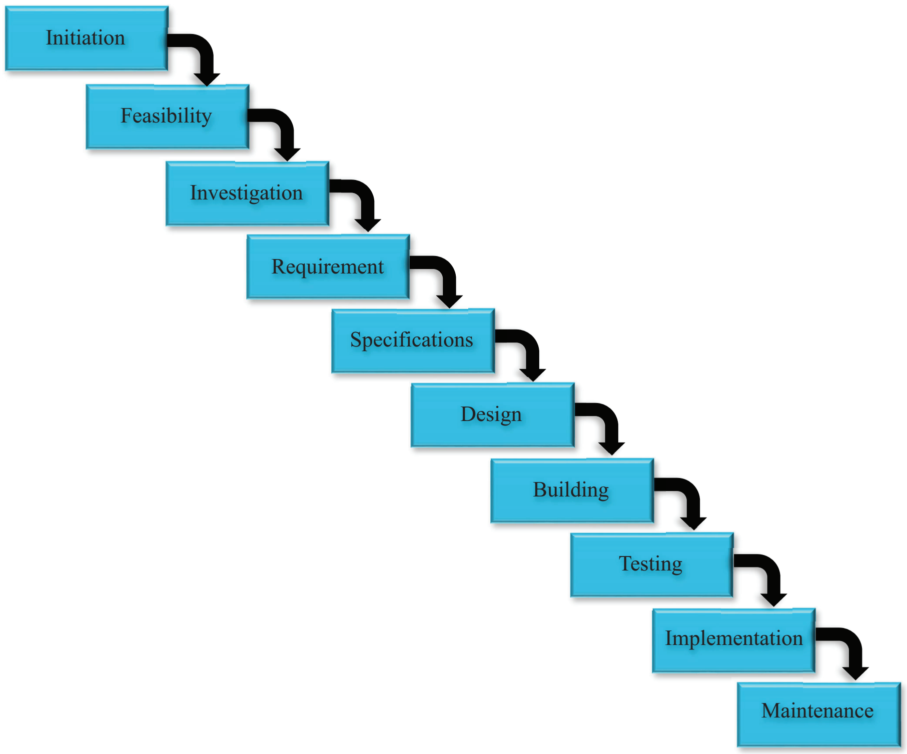


Figure 6. DANNDO development life cycle using waterfall model

AUTOMATICA
Engineering Software
AUTOMATICA
for SW Development
Assoc.Prof.Dr.Eng. Ahmed Serwa
Faculty of Engineering-Mataria- Helwan University
WhatsApp:00201003808123 Email:ahmed_serwa@yahoo.com

Figure 7. Starting form

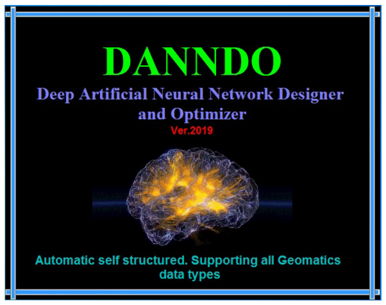
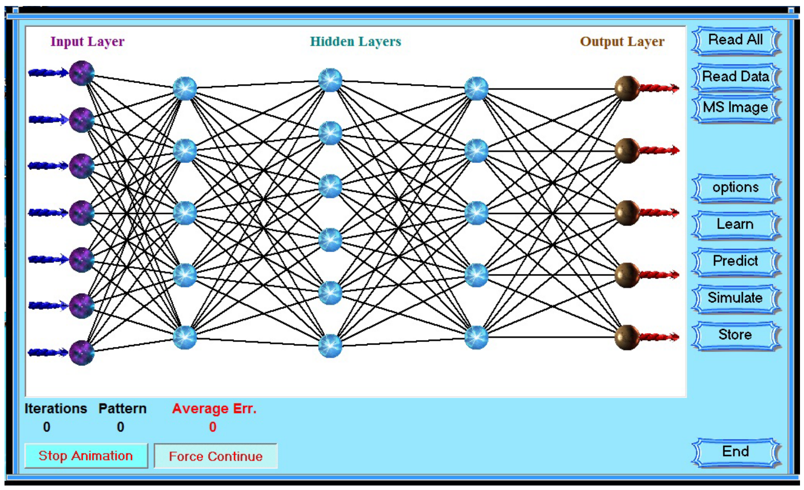

DANNDO
Deep Artificial Neural Network Designer and Optimizer
Ver.2019
Automatic self structured. Supporting all Geomatics data types

Figure 8. Identification form



Iterations 0
Pattern 0
Average Err. 0

Figure 9. Main module with network structure

4. Application

Application overview with flowchart symbology is indicated in Figure 10. The process is started with input MS image to ENVI software to make spatial and spectral subsetting in addition to get reference samples (ROI^s) – for training and testing. ENVI is also used to carry out the basic operations such as radiometric, spectral and geometric corrections according to necessity. DANNDO obtains training data for learning in addition to original image data to construct the final membership file for all MS images.

The input (for training) and output (for classification) of DANNDO is an ASCII file containing both spectral response and class memberships of each pixel in single line as shown in Figure 11. The spectral responses of each pixel in each band (digital numbers) are separated with "," between each. The separator of ";" is used to end both spectral responses and classes memberships. DANNDO is decoding the input training file to form the training vector for each pixel. The decoding includes the input spectral response in each image band in addition to extracting the class memberships which is a binary coded value. In Figure 11 a training area belongs to class 4. The binary code is used because the training is crisp (hard) and the same method is used for the output file. Both input and output file have the same format for simplicity.

4.1. Application data

Application data is a Landsat 8 remote sensing satellite MS image which covers a part of south Port Said city in Egypt and. The used bands are the first seven spectral equal bands (1 to 7) the rest bands cannot be used due the variety in size and nature. Figure 12 indicates the study area on ENVI environment in before spatial and spectral subset. The study area is represented by 600×600 pixels (324 km^2) for the southern part of Port Said city. The satellite image is corrected from all necessary errors such as atmospheric, radiometric, and geometric corrections. The previous corrections done although some precautions were made such as selecting the area with off-cloud and off-relief. The reference is prepared using suite visits with companion to handheld GARMIN GPS map 62s (3 meters accuracy).

Figure 13 shows the reference for both training and testing where the proposed classes were represented, these proposed classes are sandy rocks, silt, sand, water

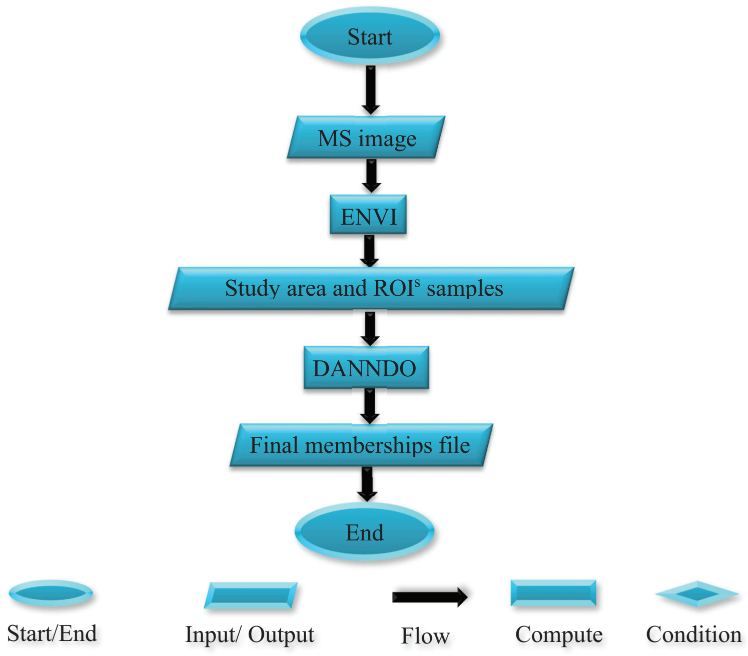
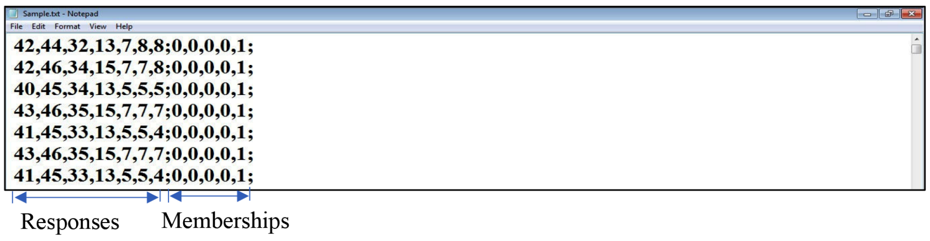
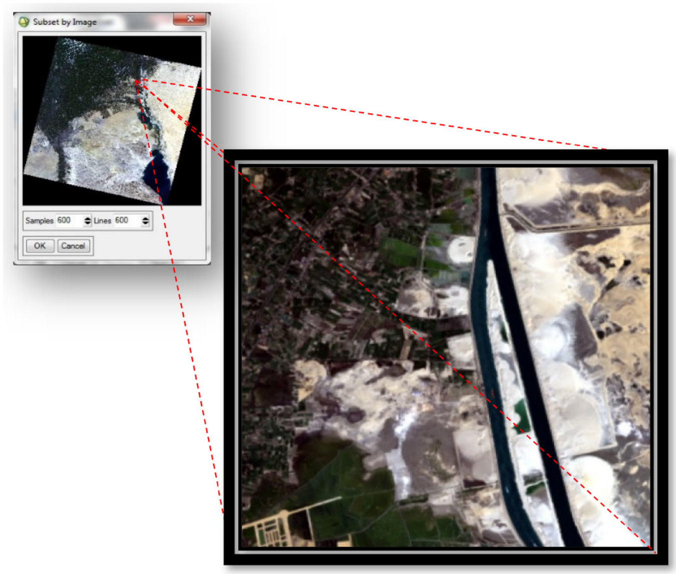


Figure 10. Block diagram of the Application



```
42,44,32,13,7,8,8;0,0,0,0,1;
42,46,34,15,7,7,8;0,0,0,0,1;
40,45,34,13,5,5,5;0,0,0,0,1;
43,46,35,15,7,7,7;0,0,0,0,1;
41,45,33,13,5,5,4;0,0,0,0,1;
43,46,35,15,7,7,7;0,0,0,0,1;
41,45,33,13,5,5,4;0,0,0,0,1;
```

Figure 11. DANNDO input/output file



Samples 600 Lines 600

Figure 12. True color Landsat 8 for the study area

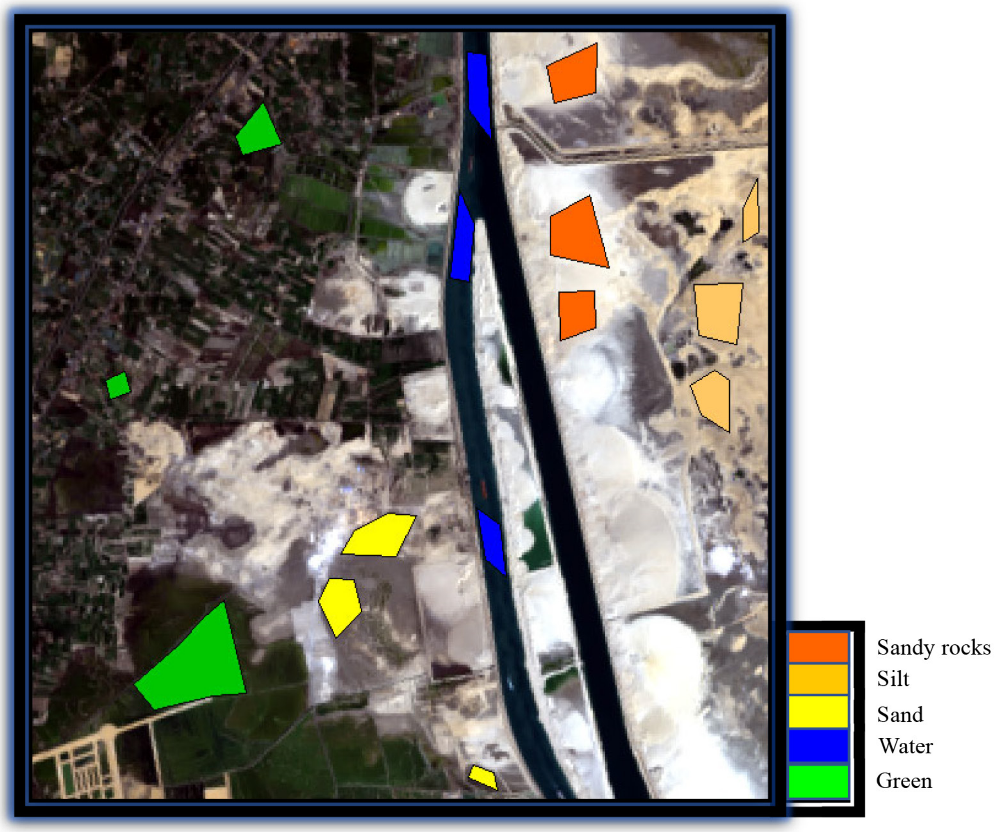
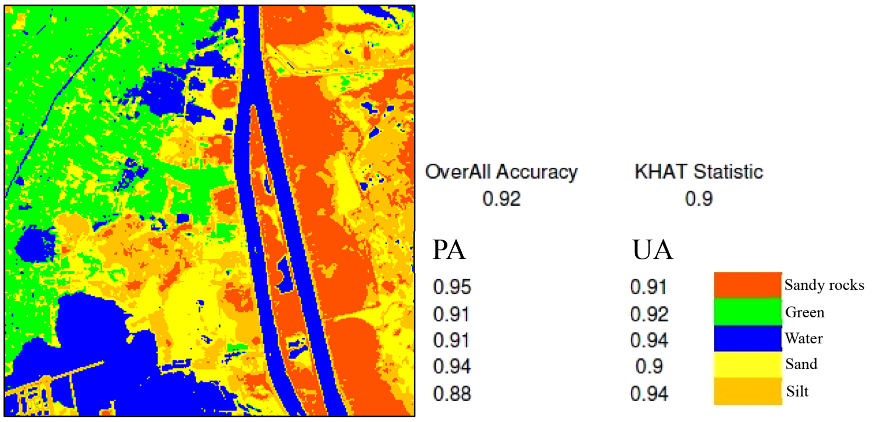


Figure 13. Reference for the study area

and green areas. The selected pixels for both reference and testing are divided by 50% for each in a random way to guarantee the reliability.

5. Results

To obtain the final classification result the binary memberships must be translated to classes colors. Figure 14 shows the final classification result with both user accuracy (UA) and producer accuracy (PU) for each class. Kappa value (KHAT) and the overall accuracy is included in the result. The overall accuracy using DANNDO is compared with the result from ADIPRS software which was developed previously by Serwa (Serwa & El-Semary, 2020) and it is found identical; the explanation for this result is the similarity of coding of BPNN in both software. No



OverAll Accuracy 0.92

KHAT Statistic 0.9

PA	UA		
0.95	0.91	🟧	Sandy rocks
0.91	0.92	🟩	Green
0.91	0.94	🟦	Water
0.94	0.9	🟨	Sand
0.88	0.94	🟧	Silt

Figure 14. Final classification result according to DANNDO

need for any statistical test because the matching case is 100% identical. One must carry out a comparison between any remote sensing commercial software and DANNDO to guarantee the reliability of the last. But in case of the availability of reference for validation there is no need for the comparison between DANNDO and any other alternative software. The reason is that one need to compare the real world with DANNDO results to ensure that it is reliable.

Conclusions

The research objective is achieved by developing and testing the developed research simulator of DANNDO. Although DANNDO uses equations that are common in ANN, it makes a spotlight on DANN by optimizing these equations. DANNDO also simulates the DANN problem visually and computationally so that the user can feel the results in addition to making the decision about convergency or divergency. Use of DANNDO in the research process of remote sensing results in better understanding outcomes. The generality of DANNDO can be noticed in dealing with ASCII format so the user can deal with it easily by converting his own data and use the SW. In the case of remote sensing DANNDO gives an acceptable result so it can be used in remote sensing classification.

References

Chen, Z., & Cheng, Y. (2004). Design and realize the software of machining techniques manual. *Journal of Shaanxi University of Science & Technology*, (2), 45–47.

Ismail, S. A., Serwa, A., Abood, A., Fayed, B., Ismail, S. A., & Hashem, A. M. (2019). A study of the use of deep artificial neural network in the optimization of the production of antifungal exochitinase compared with the response surface methodology. *Jordan Journal of Biological Sciences*, *12*(5), 543–551. http://www.scopus.com/inward/record.url?eid=2-s2.0-85087292210&partnerID=MN8TOARS

Mohanty, K. K., & Majumdar, T. J. (1996). An artificial neural network (ANN) based software package for classification of remotely sensed data. *Computers & Geosciences*, *22*(1), 81–87. https://doi.org/10.1016/0098-3004(95)00059-3

Nabil, A., Serwa, A., & Mostafa, A. E. (2020). Studying the potentiality of using low cost system based on image analysis technique to survey the gravel's size in asphalt mixes. *Engineering Research Journal*, *167*, 257–274. https://doi.org/10.21608/erj.2020.140105

Sachdeva, J., Kumar, V., Gupta, I., Khandelwal, N., & Ahuja, C. K. (2016). A package-SFERCB-"Segmentation, feature extraction, reduction and classification analysis by both SVM and ANN for brain tumors". *Applied Soft Computing*, *47*, 151–167. https://doi.org/10.1016/j.asoc.2016.05.020

Serwa, A. (2009). *Automatic extraction of topographic features from digital images* [PhD thesis]. Faculty of Engineering in Cairo-Azhar University.

Serwa, A. (2017a). Optimizing activation function in deep artificial neural networks approach for landcover fuzzy pixel-based classification. *International Journal of Remote Sensing Applications*, *7*(1). https://doi.org/10.14355/IJRSA.2017.07.001

Serwa, A. (2017b). Studying the effect of activation function on classification accuracy using deep artificial neural networks. *Journal of Remote Sensing & GIS*, *6*, 3. https://doi.org/10.4172/2469-4134.1000203

Serwa, A. (2020). Studying the potentiality of using digital Gaussian Pyramids in multi-spectral satellites images classification. *Journal of the Indian Society of Remote Sensing*, *48*, 1651–1660. https://doi.org/10.1007/s12524-020-01173-w

Serwa, A., & El-Semary, H. H. (2016). Integration of soft computational simulator and strapdown inertial navigation system for aerial surveying project planning. *Spatial Information Research*, *24*, 279–290. https://doi.org/10.1007/s41324-016-0027-9

Serwa, A., & El-Semary, H. H. (2020). Semi-automatic general approach to achieve the practical number of clusters for classification of remote sensing MS satellite images. *Spatial Information Research*, *28*, 203–213. https://doi.org/10.1007/s41324-019-00283-z

Serwa, A., & Elbialy, S. (2020). Enhancement of classification accuracy of multi-spectral satellites' images using Laplacian pyramids. The *Egyptian Journal of Remote Sensing and Space Science*, *24*(2), 283–291. https://doi.org/10.1016/j.ejrs.2020.12.006

Serwa, A., & Saleh, M. (2021). New semi-automatic 3D registration method for terrestrial laser scanning data of bridge structures based on artificial neural networks. *The Egyptian Journal of Remote Sensing and Space Science*, *24*(3), 787–798. https://doi.org/10.1016/j.ejrs.2021.06.003