AN INVESTIGATION ON SELF-NORMALIZED DEEP NEURAL NETWORKS FOR HYPERSPECTRAL IMAGE CLASSIFICATION

M.E. Paoletti[1] *Student Member, IEEE,* J.M. Haut[1] *Student Member, IEEE,*
J. Plaza[1] *Senior Member, IEEE,* A. Plaza[1] *Fellow, IEEE*

***Abstract*— Computational advances have allowed for the development of deep learning (DL) applied to remote sensing data and, particularly, to hyperspectral image (HSI) classification. Deeper architectures are able to establish a better separation of the characteristics of the data, allowing for a better and accurate performance. However, it is known that employing very deep architectures with many abstraction levels can result in a loss of information due to the fact that deep networks often normalize each data individually, without considering the set of adjacent data. To address this issue, this paper implements a self-normalizing neural network (SNN) in order to extract high-level abstract representations without losing information due to the data initialization. The selected activation function (scaled exponential linear units or SELU) normalizes the data considering their neighborhood's information and a special dropout technique (α-dropout), obtaining good classification performance while maintaining the data characteristics across the successive layers. Obtained results show that the proposal improves the performance with few training samples.**

I. INTRODUCTION

Remote sensing (RS) allows us to obtain information about Earth's surface through airborne and spaceborne sensors that operate from the visible to the middle infrared wavelength range [1], obtaining images with different spatial and spectral resolutions [2]. In particular hyperspectral images (HSI) can be considered cubes of images of $n1 \times n2$ pixels with thousands of narrow spectral bands d, containing a very detailed information about the properties of the objects appearing in the image since this objects have different spectroscopic features [3], creating distinctive spectral signatures [4].

Nowadays, improvements in spatial resolution and growing revisit frequencies have exponentially increased data availability. For instance, the ESA Sentinel-1 generates about 1.5GB of data per day; the NASA EOSDIS project produces about 16 TB of data per day, and the NASA Jet Propulsion Laboratorys Airbone Visible/Infrared Imaging Spectrometer (AVIRIS) has a data collection rate of 2.5 MB/s (nearly 9 GB/h)[5]. This amount of data presents new processing and storing challenges due to its high volume, variety and the generation velocity [6]. Moreover, the great dimensionality of HSI data creates the need to use non-conventional analysis techniques, originally designed for multispectral data.

In this sense, deep neural networks (DNNs) represent a powerful tool for HSI analysis [1], being able to learn more complex models and extract more abstract features from the data. However, DNNs such as deep multilayer perceptrons (MLPs) or deep convolutional neural networks (CNNs), need large amounts of training data due to severe overfitting problems. In particular, the accuracy obtained by feed-forward neural networks (FNNs) and MLPs is highly affected by the depth of the network, reaching only good results in applications with shallow topologies and being less competitive than other types of nets such CNNs [7], [8]. This learning problem is caused by the imbalance between the large number of parameters that must be trained and the few training samples available in advance. This is also due to the loss of information in each layer of the deep model due to a poor propagation of activations and gradients [9] produced by the internal covariate shift phenomenon [10]. Focusing on this issue, it can be observed in DNNs that the original data distribution changes during the training as the parameters of the network's layers change, slowing down the NN's training. This problem is more critical in HSI, where the presence of noise, data redundancies and inter/intra-class variability makes learning even more difficult.

Employing careful data standardization accompanied by stochastic regularization and a proper activation function are basic tools to improve the convergence of the net, preventing the vanishing gradient problem [7]. In this work, we use a self-normalized neural network (SNN) to perform HSI classification. SNNs are based on scaled exponential linear units (SELUs), an activation function that incorporates self-normalizing properties to the nodes in each layer of the net, allowing robust and stable learning process for deep architectures while driving neuron activations to zero mean and unit variance. In order to prove the usefulness of this kind of neural networks in HSI processing, this work proposes a MLP-based model with self-normalizing properties. The parallel structure of the method, composed by a large amount of independent units, allows us to develop a parallel model through graphics processing units (GPUs). The parallel implementation in GPU presents lower latency, size and power consumption that implementations in CPU, while allowing massive, fast and scalable data processing.

The remainder of the paper is organized as follows. Section II introduces the mathematical concepts behind the SNN, and the characteristics of the proposed model for HSI classification. Section III presents the experiments carried out over a well-known HSI dataset and the obtained results. Finally, section IV summarizes the work and indicates future research lines and improvements.

[1]J.M. Haut, M.E. Paoletti, J. Plaza and A. Plaza are with the Hyperspectral Computing Laboratory, Department of Technology of Computers and Communications, Escuela Politécnica, University of Extremadura, PC-10003 Cáceres, Spain.(e-mail:juanmariohaut@unex.es;mpaolett@unex.es;jplaza@unex.es; aplaza@unex.es).

II. METHODOLOGY

FFN/MLP models are traditionally unable to improve their performance with deep architectures and very complex data, such as HSI scenes, due to data inherent perturbations and the internal covariate shift phenomenon, which produces a poor propagation of activations and gradients with loss of information. In order to avoid this problem, SNNs introduce three improvements related with the input data, the neurons activation outputs, and the internal regularization of the data.

Let us suppose that we have a deep MLP with two layers, where $l-1$ is the lower layer with n neurons and l is the higher layer with m neurons. The output of $L-1$ is denoted by $\mathbf{x}^{(l-1)} \in \mathbb{R}^{n\times d}$, a matrix of $n \times d$, where d is the number of inputs of the neural network. On the other hand, the input of layer l is another matrix defined as $\mathbf{z}^{(l)} \in \mathbb{R}^{m\times d} = \mathbf{W}^{(l-1,l)}\mathbf{x}^{(l-1)}$, where $\mathbf{W}^{(l-1,l)} \in \mathbb{R}^{m,n}$ is the weight matrix that connects the n nodes in layer $l-1$ with the m nodes in layer l, and its output is $\mathbf{x}^{(l)} \in \mathbb{R}^{m\times d} = f\left(\mathbf{z}^{(l)}\right)$ where $f(\cdot)$ is the activation function. The SNN uses two metrics: the mean and the variance. All activations in layer $l-1$, i.e. $\{x_i^{(l-1)}\}_{i=1}^n$, and layer l, i.e. $\{x_i^{(l)}\}_{i=1}^m$, have mean $\mu^{(l-1)} = \mathrm{E}(x_i^{(l-1)})$ and $\mu^{(l)} = \mathrm{E}(x_i^{(l)})$ respectively, where $\mathrm{E}(\cdot)$ indicates expectation. Also all $\{x_i^{(l-1)}\}_{i=1}^n$ and $\{x_i^{(l)}\}_{i=1}^m$ have variance $\nu^{(l-1)} = \mathrm{Var}(x_i^{(l-1)})$ and $\nu^{(l)} = \mathrm{Var}(x_i^{(l)})$ respectively, where $\mathrm{Var}(\cdot)$ indicates the variance of a random variable. On the other hand, each activation function in the higher layer $\{x_i^{(l)} = f\left(z_i^l\right)\}_{i=1}^m$ has as input $z_i^{(l)} = \mathbf{w}_i^{(l-1,l)T}\mathbf{x}^{(\mathbf{l-1})}$, where $\mathbf{w}_i^{(l-1,l)}$ is the weight vector extracted from $\mathbf{W}^{(l-1,l)}$ that connect the i-th neuron in layer l with the n neurons in layer $l-1$. Two new metrics can be defined: the mean of the weight vector $\omega^{(l)} = \sum_{j=1}^n w_{i,j}^{(l-1,l)}$ and the second moment $\tau^{(l)} = \sum_{j=1}^n (w_{i,j}^{(l-1,l)})^2$. The main idea behind SNNs is that there is a function $g : \Omega \to \Omega$ that maps means and variances from one layer to the next one:

$$\begin{pmatrix} \mu^{(l-1)} \\ \nu^{(l-1)} \end{pmatrix} \mapsto \begin{pmatrix} \mu^{(l)} \\ \nu^{(l)} \end{pmatrix} : \begin{pmatrix} \mu^{(l)} \\ \nu^{(l)} \end{pmatrix} = g \begin{pmatrix} \mu^{(l-1)} \\ \nu^{(l-1)} \end{pmatrix} \tag{1}$$

with a stable and attracting fixed point in Ω, depending on $(\omega^{(l)}, \tau^{(l)})$. To implement self-normalization, the SNNs adjust the properties of g by the activation function, performing the scaled exponential linear units (SELU), given by Eq. (2) [7], which introduces self-normalizing properties like variance stabilization, setting neurons activations to zero mean and unit variance, i.e. $\mu^{(l-1)} = \mu^{(l)} = 0$ and $\nu^{(l-1)} = \nu^{(l)} = 1$, and allowing SNNs to be robust to data perturbations. This activation function ensures the following properties:

1) Negative and positive values to control the mean.
2) Saturation regions to dampen the variance if it is too large in the lower layer.
3) A slope bigger than one to increase the variance if it is too small in the lower layer.
4) A continuous curve to ensure a fixed point depending on $\omega^{(l)}$ and $\tau^{(l)}$.

$$\mathrm{selu}(x) = \lambda \begin{cases} x & x < 0 \\ \alpha e^x - \alpha & x \geq 0 \end{cases} \tag{2}$$

Also, SNN maintains the normalization of layer activations when it propagates them through the network, getting closer to the stable and fixed point $(0; 1)$ and selecting $\omega^{(l)} = \sum_{j=1}^n w_{i,j}^{(l-1,l)} = 0$ and $\tau^{(l)} = \sum_{j=1}^n (w_{i,j}^{(l-1,l)})^2 = 1$ for normalized weights initialization, so $\mathrm{E}(\mathbf{w}_i) = 0$ and $\mathrm{Var}(\mathbf{w}_i) = 1/n$. From $z_i^{(i)} = \mathbf{w}_i^{(l-1,l)T}\mathbf{x}^{(l-1)}$ we can extract two relations or moments: $\mathrm{E}(z_i^{(i)}) = \mu^{(l-1)}w^{(l)}$ and $\mathrm{Var}(z_i^{(i)}) = \nu^{(l-1)}\tau^{(l)}$. Also, $z_i^{(i)}$ approaches a normal distribution with density $p_N(z_i^{(i)}; \mu^{(l-1)}w^{(l)}, \sqrt{\nu^{(l-1)}\tau^{(l)}})$. With this relations, mapping function g in Eq. (1) can be redefined as:

$$\begin{pmatrix} \mu^{(l-1)} \\ \nu^{(l-1)} \end{pmatrix} \mapsto \begin{pmatrix} \mu^{(l)} \\ \nu^{(l)} \end{pmatrix} :$$

$$\mu^{(l)}\left(\mu^{(l-1)}, \omega^{(l)}, \nu^{(l-1)}, \tau^{(l)}\right) = \int_{-\infty}^{\infty} \mathrm{selu}(z_i^{(l)}) p_N(z_i^{(l)}; \mu^{(l-1)}\omega^{(l)}, \sqrt{\nu^{(l-1)}\tau^{(l)}})\,\mathrm{d}z$$

$$\nu^{(l)}\left(\mu^{(l-1)}, \omega^{(l)}, \nu^{(l-1)}, \tau^{(l)}\right) = \int_{-\infty}^{\infty} \mathrm{selu}(z_i^{(l)})^2 p_N(z_i^{(l)}; \mu^{(l-1)}\omega^{(l)}, \sqrt{\nu^{(l-1)}\tau^{(l)}})\,\mathrm{d}z - (\mu^{(l)})^2. \tag{3}$$

On the other hand, SNN implements α-dropout method as a new regularization technique in order to keep the mean and variance as inputs to original values, ensuring the self-normalizing property even after regularization. During training, α-dropout randomly sets inputs to the infimum of the SELU activation function. Moreover, in a random way, this technique sets input data to α' since the low variance region value of SELU is defined as $\lim_{x\to-\infty} selu(x) = -\lambda\alpha = \alpha'$, preventing overfitting and information loss.

III. EXPERIMENTS

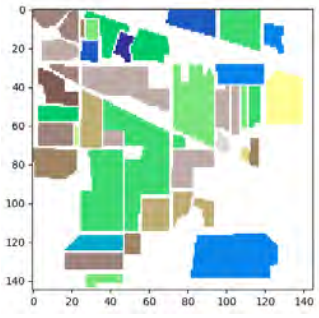
In this section we describe the experiments performed in order to demonstrate the effectiveness of the proposed SNN-based method to analyze HSI imagery through deep neural networks, allowing a robust and faster learning without high variance and reaching high levels of abstract representations. As a result, it is expected to obtain better accuracy than other related methods.

A. Experimental Configuration

Our experiments have been executed on a hardware environment composed by a 6th Generation Intel® Core™i7-6700K processor with 8M of Cache and up to 4.20GHz (4 cores/8 way multitask processing), 40GB of DDR4 RAM with a serial speed of 2400MHz, a GPU NVIDIA GeForce GTX 1080 with 8GB GDDR5X of video memory and 10Gbps of memory frequency, a Toshiba DT01ACA HDD with 7200RPM and 2TB of capacity, and an ASUS Z170 pro-gaming motherboard. On the other hand, the software environment is composed by Ubuntu 16.04.4 x64 as operating system, CUDA 8 and Python.

B. Dataset description

Experiments have been carried out over Indian Pines scene, collected by the Airborne Visible/Infrared Imaging Spectrometer (AVIRIS) sensor [11], with a size of 145 lines by 145 samples, was acquired over a mixed agricultural/forest area, early in the growing season. The original scene comprises 224 spectral bands in the wavelength range from 400 to 2500nm, nominal spectral resolution of 10 nm, moderate spatial resolution of 20 meters by pixel, and 16-bit radiometric resolution. After an initial screening, 22 spectral bands were removed from the data set due to noise. Fig. 1 shows the ground-truth map available for the scene with 16 ground-truth classes. About half of the pixels in the image (10366 of 21025) contain ground-truth information, which comes in the form of a single label assignment (Table 1).



Color	Land cover type	Samples
	Background	10659
	Alfalfa	54
	Corn-notill	1434
	Corn-min	834
	Corn	234
	Grass/Pasture	497
	Grass/Trees	747
	Grass/pasture-mowed	26
	Hay-windrowed	489
	Oats	20
	Soybeans-notill	968
	Soybeans-min	2468
	Soybean-clean	614
	Wheat	212
	Woods	1294
	Bldg-Grass-Tree-Drives	380
	Stone-steel towers	95
	Total samples	21025

Fig. 1: Ground-truth of the Indian Pines $145 \times 145 \times 202$ HSI scene.

C. Performance Evaluation

In order to test the performance of the method over HSI data, several deep MLP models have been executed. Table I shows the configuration of the deep MLP networks used, composed by one input layer (with 200 nodes), four hidden layers and one output layer (with 16 nodes). Our experiment compares the SNN-based MLP with SELU and a classical MLP with ReLU as activation function. In both networks, the Adam optimizer[12] is employed with the same learning rate and model topology. Also, the single-layer feedforward network developed by Ghamisi *et al.* in [13] has been used for comparative purposes. Table II shows the obtained results as the mean of 5 executions, from the carried out comparative using 5%, 10%, 15%, 20% and 25% of randomly selected samples per class as training data. Execution times are very similar between each proposed method and are not the main point of this work. As we can observe, the self-normalized MLP reaches the best overall accuracy (OA) in all cases, in particular with few training data (1.43 percentage points over traditional and 1.16 over [14]) due to its great potential to maintain normalized data through a very deep network, allowing the extraction of better features in all layers, as fig. 3 indicates, where orange line shows the evolution of the self-normalized MLP accuracy in successive epochs, being 500 epochs the maximum considered. We can observe that the proposed method is able to achieve better accuracies in less epochs than the traditional MLP. Finally, Fig. 2 shows the obtained classification maps by traditional MLP (center) and self-normalized MLP (right), both trained with 10% of random selected samples per class. Also, Table III shows the corresponding reached accuracy for each Indian Pines class. With this result, the proposed method is shown to behave in a similar way as the original MLP with ReLU algorithm, but increasing its results in similar epochs.

NETWORKS CONFIGURATION	
Hidden neurons	$144 - 144 - 144 - 64$
Dropout SELU	$0.15 - 0.15 - 0.15 - 0.05$
Dropout RELU	$0.30 - 0.30 - 0.30 - 0.10$
Optimizer	Adam
Learning Rate	0.0008

TABLE I: Self-normalized MLP and traditional MLP networks architectures and optimizer parameters.

Training	Traditional	Proposed	Ghamisi et al. [14]
5%	76.86 (0.77)	**78.29 (1.36)**	77.13 (1.04)
10%	84.41 (0.65)	**85.66 (0.91)**	83.10 (0.62)
15%	88.10 (0.17)	**89.63 (0.24)**	85.28 (0.56)
20%	90.42 (0.59)	**90.97 (0.47)**	87.17 (0.48)
25%	92.15 (0.31)	**92.16 (0.45)**	87.97 (0.50)

TABLE II: Obtained overall accuracies (OAs) by the traditional MLP, self-normalizing MLP and the SLFNN proposed in [14]. Each experiment has been executed 5 times in order to test the robustness and stability of each method via standard deviation (in parenthesis).

Class name	FNN-ReLU	Proposed
Alfalfa	42.59 (22.25)	**46.67 (23.41)**
Corn-notill	80.71 (2.37)	**82.04 (3.80)**
Corn-min	66.04 (2.50)	**80.07 (3.29)**
Corn	70.34 (5.96)	**77.35 (6.15)**
Grass/Pasture	88.65 (3.31)	**92.72 (2.09)**
Grass/Trees	94.62 (1.35)	**95.37 (1.01)**
Grass/pasture-mowed	64.62 (18.11)	**83.85 (20.12)**
Hay-windrowed	**99.22 (0.15)**	98.77 (0.41)
Oats	**49.00 (9.17)**	43.00 (13.27)
Soybeans-notill	83.41 (2.72)	**85.43 (2.27)**
Soybeans-min	**86.41 (2.22)**	83.43 (3.36)
Soybean-clean	75.41 (6.52)	**75.57 (4.58)**
Wheat	98.49 (1.01)	**99.25 (0.38)**
Woods	96.85 (1.37)	**97.45 (0.40)**
Bldg-Grass-Tree-Drives	**64.26 (3.63)**	61.53 (0.69)
Stone-steel towers	90.53 (3.40)	**92.00 (3.80)**
Overall accuracy	84.41 (0.65)	**85.66 (0.91)**
Average accuracy	78.20 (2.47)	**80.91 (2.65)**
Kappa	82.18 (0.76)	**83.68 (1.02)**

TABLE III: Classification accuracies obtained by different deep neural networks tested using the Indian Pines dataset.

IV. CONCLUSIONS AND FUTURE WORK

In this work we have explored the application of a self-normalized neural networks approach in order to perform

Fig. 2: Indian Pines classification maps with 10% of training data: original ground truth (left), MLP with ReLU with 84.41% of OA (center) and self-normalized MLP with 85.66% of OA (right).

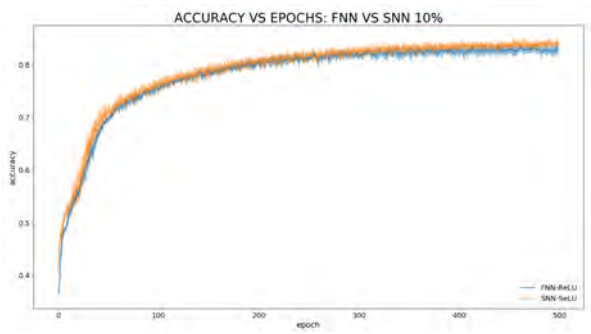


Fig. 3: Accuracy vs epoch from traditional MLP (blue) and self-normalized MLP (orange) using 10% of training data from Indian Pines.

HSI remote sensing data classification. The obtained results over the well-known Indian Pines dataset reveal that self-normalized techniques are suitable not only in networks with a very deep number of layers, reaching better classification accuracies than traditional deep network architectures, but also for very noisy/complex input data. As future works we will explore the suitability of this technique with different kinds of deep neural networks, such as convolutional neural networks, and in combination with the different available methods for data normalization.

V. Acknowledgement

This work has been supported by Ministerio de Educación (Resolución de 26 de diciembre de 2014 y de 19 de noviembre de 2015, de la Secretaría de Estado de Educación, Formación Profesional y Universidades, por la que se convocan ayudas para la formación de profesorado universitario, de los subprogramas de Formación y de Movilidad incluidos en el Programa Estatal de Promoción del Talento y su Empleabilidad, en el marco del Plan Estatal de Investigación Científica y Técnica y de Innovación 2013-2016. This work has also been supported by Junta de Extremadura (decreto 297/2014, ayudas para la realización de actividades de investigación y desarrollo tecnológico, de divulgación y de transferencia de conocimiento por los Grupos de Investigación de Extremadura, Ref. GR15005).

References

[1] M.E. Paoletti, J.M. Haut, J. Plaza, and A. Plaza. A new deep convolutional neural network for fast hyperspectral image classification. {*ISPRS*} *Journal of Photogrammetry and Remote Sensing*, 2017.

[2] Antonio Plaza, Jon Atli Benediktsson, Joseph W. Boardman, Jason Brazile, Lorenzo Bruzzone, Gustavo Camps-Valls, Jocelyn Chanussot, Mathieu Fauvel, Paolo Gamba, Anthony Gualtieri, Mattia Marconcini, James C. Tilton, and Giovanna Trianni. Recent advances in techniques for hyperspectral image processing. *Remote Sensing of Environment*, 2009.

[3] C.I.Chang. *Hyperspectral imaging: techniques for spectral detection and classification*. Plenum publisher, New York, 2003.

[4] Mathieu Fauvel, Yuliya Tarabalka, Jn Atli Benediktsson, Jocelyn Chanussot, and James C. Tilton. Advances in spectral-spatial classification of hyperspectral images. *Proceedings of the IEEE*, 2013.

[5] J.M. Haut, M.E. Paoletti, J. Plaza, and A. Plaza. Cloud implementation of the K-means algorithm for hyperspectral image analysis. *Journal of Supercomputing*, 73(1), 2017.

[6] Victor Andres Ayma Quirita, Gilson Alexandre Ostwald Pedro Da Costa, Patrick Nigri Happ, Raul Queiroz Feitosa, Rodrigo Da Silva Ferreira, Dario Augusto Borges Oliveira, and Antonio Plaza. A New Cloud Computing Architecture for the Classification of Remote Sensing Data. *IEEE Journal of Selected Topics in Applied Earth Observations and Remote Sensing*, 2017.

[7] Gnter Klambauer, Thomas Unterthiner, Andreas Mayr, and Sepp Hochreiter. Self-Normalizing Neural Networks. In *Advances in Neural Information Processing Systems 30: Annual Conference on Neural Information Processing Systems*, pages 972–981, Long Beach, CA, 2017.

[8] Z. Zhong, J. Li, Z. Luo, and M. Chapman. Spectral-spatial residual network for hyperspectral image classification: A 3-d deep learning framework. *IEEE Transactions on Geoscience and Remote Sensing*, 56(2):847–858, Feb 2018.

[9] Rupesh Kumar Srivastava, Klaus Greff, and Jrgen Schmidhuber. Training Very Deep Networks. *CoRR*, abs/1507.06228, 2015.

[10] Sergey Ioffe and Christian Szegedy. Batch Normalization: Accelerating Deep Network Training by Reducing Internal Covariate Shift. In *Proceedings of the 32nd International Conference on Machine Learning (ICML)*, pages 448–456, Lille, France, 2015.

[11] Robert O. Green, Michael L. Eastwood, Charles M. Sarture, Thomas G. Chrien, Mikael Aronsson, Bruce J. Chippendale, Jessica A. Faust, Betina E. Pavri, Christopher J. Chovit, Manuel Solis, Martin R. Olah, and Orlesa Williams. Imaging spectroscopy and the Airborne Visible/Infrared Imaging Spectrometer (AVIRIS). *Remote Sens. Environ.*, 65(3):227–248, Sep. 1998.

[12] Diederik Kingma and Jimmy Ba. Adam: A method for stochastic optimization. *arXiv preprint arXiv:1412.6980*, 2014.

[13] Pedram Ghamisi, Naoto Yokoya, Jun Li, Wenzhi Liao, Sicong Liu, Javier Plaza, Behnood Rasti, and Antonio Plaza. Advances in Hyperspectral Image and Signal Processing.

[14] Pedram Ghamisi, Javier Plaza, Yushi Chen, Jun Li, and Antonio Plaza. Advanced supervised spectral classifiers for hyperspectral images: A review. *IEEE Geoscience and Remote Sensing Magazine (GRSM)*, 2017.