CLUSTERING SATELLITE IMAGE TIME SERIES DATA BASED ON GROWING SELF-ORGANIZING MAPS

Rodrigo de Sales da Silva Adeu

Master Dissertation submited to the Graduate Program in Applied Computation, supervised by Dra. Karine Reis Ferreira and Dr. Pedro Ribeiro de Andrade Neto, approved in July, 31 of 2020.

URL of the original document:
<http://urlib.net/8JMKD3MGP3W34R/4325CC8>

INPE

São José dos Campos

2020

CLUSTERING SATELLITE IMAGE TIME SERIES DATA BASED ON GROWING SELF-ORGANIZING MAPS

Rodrigo de Sales da Silva Adeu

Master Dissertation submited to the Graduate Program in Applied Computation, supervised by Dra. Karine Reis Ferreira and Dr. Pedro Ribeiro de Andrade Neto, approved in July, 31 of 2020.

Aluno (a): *Rodrigo de Sales da Silva Adeu*

"CLUSTERING SATELLITE IMAGE TIME SERIES DATA BASED ON GROWING SELF-ORGANIZING MAPS"

Aprovado (a) pela Banca Examinadora em cumprimento ao requisito exigido para obtenção do Título de **Mestre** em

Computação Aplicada

Dr. **Marcos Gonçalves Quiles**

Presidente / UNIFESP / São José dos Campos - SP

(X) Participação por Video - Conferência

(X) Aprovado ( ) Reprovado

Dra. **Karine Reis Ferreira**

Orientador(a) / INPE / São José dos Campos - SP

(X) Participação por Video - Conferência

(X) Aprovado ( ) Reprovado

Dr. **Pedro Ribeiro de Andrade Neto**

Orientador(a) / INPE / São José dos Campos - SP

(X) Participação por Video - Conferência

(X) Aprovado ( ) Reprovado

Dra. **Michelle Cristina Araujo Picoli**

Membro da Banca / INPE / São José dos Campos - SP

(X) Participação por Video - Conferência

(X) Aprovado ( ) Reprovado

Dra. **Luciana Alvim Santos Romani**

Convidado(a) / EMBRAPA / Campinas - SP

(X) Participação por Video - Conferência

(X) Aprovado ( ) Reprovado

Este trabalho foi aprovado por:

( ) maioria simples

(X) unanimidade

São José dos Campos, 31 de julho de 2020

"The beautiful thing about learning is that nobody can take it away from you".

B.B. KING

*To my parents **Juscelino** and **Izabel**, to my brother **Daniel** and to my wife **Lucimara***

ACKNOWLEDGEMENTS

ABSTRACT

Mapping Earth land use and land cover is crucial to understand agricultural dynamics. Recently, analysis of time series extracted from Earth observation satellite images has been widely used to produce land use and land cover information. In time series analysis, clustering is a common technique performed to discover patterns on data sets. In this work, we evaluate the Growing Self-Organizing Maps (GSOMs) algorithm for clustering satellite image time series and compare it with Self-Organizing Maps (SOMs) algorithm. This paper presents two case studies using satellite image time series associated to samples of land use and land cover classes, highlighting the advantage of providing a neutral factor (called spread factor) as a parameter for GSOM, instead of the SOM grid size. We first compare GSOM with traditional SOM, analyzing the resultant network topology, the algorithm running time, the cluster accuracy and the neighborhood maintenance. In the second case study, we changed the dataset, increasing the number of samples and repeating the analysis. We finish concluding that it is possible to cluster satellite image time series with GSOM, avoiding the SOM grid size additional parameter. Besides that, GSOM keeps most of SOM properties and can be considered as a suitable alternative to SOM.

Palavras-chave: Growing Self-Organized Map. Land Use and Land Cover. Machine Learning. Clustering. Unsupervised Learning.

CLUSTERING DE SÉRIES TEMPORAIS DE IMAGENS DE SATÉLITE UTILIZANDO GROWING SELF-ORGANIZING MAPS

RESUMO

Mapear o uso e a cobertura da Terra é crucial para entender a dinâmica agrícola. Recentemente, a análise de séries temporais extraídas de imagens de satélite de observação da Terra tem sido amplamente utilizada para produzir informações sobre uso e cobertura da terra. Na análise de séries temporais, o clustering é uma técnica utilizada para descobrir padrões em conjuntos de dados. Neste trabalho, avaliamos o algoritmo GSOM (Growing Self-Organizing Maps) para agrupar séries temporais de imagens de satélite e o comparamos com o algoritmo SOM (Self-Organizing Maps). Este artigo apresenta um estudo de caso utilizando séries temporais de imagens de satélite associadas a amostras de uso da terra e classes de cobertura, destacando a vantagem de fornecer um fator neutro (chamado spread factor) como parâmetro para o GSOM, no lugar do tamanho da grade do SOM. Nós iniciamos comparando o GSOM com o SOM tradicional, analisando a topologia da rede resultante, o tempo de execução dos algoritmos, a eficácia dos clusters e a manutenção da vizinhança. No segundo estudo de caso, nós modificamos o conjunto de dados, aumentando a quantidade de amostras e repetindo a análise. Nós terminamos concluindo que é possível fazer o clustering de séries temporais de imagens de satélite utilizando o GSOM e evitando o parametro adicional do tamanho da grade requerido pelo SOM. Além disso, o GSOM mantém a maioria das propriedades do SOM, e pode ser considerado como uma alternativa adequada ao SOM.

Keywords: Growing Self-Organized Map. Uso e Cobertura da Terra. Aprendizado de Máquinas. Clustering. Aprendizado Não Supervisionado.

LIST OF FIGURES

Page

LIST OF TABLES

CONTENTS

1 INTRODUCTION

Land use was always considered a local environmental issue, but nowadays, it is becoming a force of global importance. It encompasses a wide variety of activities that vary substantially in their intensity and consequences. Human actions like intensifying farmland production, clearing tropical forests, practicing subsistence agriculture, or expanding urban areas are changing the world's landscapes in pervasive ways (SPERA et al., 2020). After several decades of research, environmental impacts of land use throughout the globe were revealed, varying from changes on Earth's ecosystems to modifications in atmospheric composition (FOLEY et al., 2005).

Mapping and monitoring of land use and land cover is essential for planning and managing natural resources. Remote sensing is a useful technique to detect land use and land cover. There is a serious scientific effort to use Earth observation satellite imagery to evaluate land transformation, such as deforestation, on continental to global scales (PARENTE et al., 2017).

Technologies and methods of remote sensing and digital image processing play a crucial role in the identification, mapping, assessment and monitoring of land use and land cover. The use of remote sensing image time series analysis to produce land use and land cover information has increased greatly (GOMEZ et al., 2016). Time series derived from Earth observation satellite images are useful to facilitate detecting complex underlying processes that would be difficult to identify using traditional change detection approaches or bi-temporal analysis (PASQUARELLA et al., 2016).

In order to classify Earth observation image time series to produce land use and land cover maps, machine learning methods such as Support Vector Machine (SVM) and Random Forest (RF) have been used (PICOLI et al., 2018). Most of these methods are based on supervised learning algorithms, requiring a training step that uses labeled land use and land cover samples. An important challenge on this task is the selection of representative samples to obtain a good classification accuracy.

To better select land use and land cover samples from satellite image time series, Santos et al. (2019) propose a method based on Self-Organizing Maps (SOM) neural network (KOHONEN et al., 2001). SOM is a clustering method suitable for time series data sets. The proposed method uses SOM to produce metrics that indicate the quality of the land use and land cover samples. The goal is to evaluate which spectral bands and vegetation indexes are best suitable for the separability of land

1

use and land cover classes. This method explores two main features of SOM: (1) the capacity of mapping high-dimensional input space to a two-dimensional grid; and (2) the topological preservation of neighborhood, which generates spatial clusters of similar patterns in the output space (SANTOS et al., 2019).

Despite its advantages, SOM has a characteristic that difficult its use by regular users. It uses a predefined and fixed network architecture in terms of number and arrangement of neural processing elements. Simulations have to be run several times on different network sizes to find an appropriate network structure (FLEXER, 1999).

Several methods have been proposed to grow SOMs during training according to the input data. Most of these growing maps attempt to dynamically evolve the map structure, rather than train a grid with fixed data (SAMARASINGHE, 2006).

On the Growing Hierarchical Self-Organizing Map (GHSOM), the key idea is to use a hierarchical structure of multiple layers where each layer consists of a number of independent self-organizing maps. One SOM is used at the first layer of the hierarchy. For every unit in this map a SOM might be added to the next layer of the hierarchy (DITTENBACH et al., 2000).

The Growing Cell Structure of Map (GCS) is based on two-dimensional SOMs starting with a small grid size, usually three units forming a triangle (FRITZKE, 1993). The basic approach is to add neurons in areas that receive a high number of inputs, or add neurons in areas with the highest accumulated error of a neuron. After inserting new neurons, connections are adjusted between them so that the triangular connectivity is maintained.

On Growing Neural Gas (GNG) an incremental model similar to GCS adds neurons and connections trying to learn the data topological relations, but replaces the fixed unit growing with an increasing number of units during the self-organization process (FRITZKE, 1995b). To determine where to insert new units, local error measures are gathered during the adaptation phase.

The Growing Self-Organizing Map (GSOM) grows dynamically, and its growth can be controlled using a spread factor such that a smaller map showing broader (macro) cluster regions can initially be obtained (ALAHAKOON et al., 2000). If more accuracy is needed, the whole map can be grown further until desired accuracy is achieved in terms of the visibility of interesting clusters (SAMARASINGHE, 2006).

1.1 Objective

Determine the SOM grid size is not a straightforward task for regular users. Users regularly understand data set details, but are not usually specialists on clustering methods. The objective of this work is to propose and analyze an alternative to SOM, providing the same SOM clustering capabilities, but avoiding the task of previously determine the grid size. We intend to facilitate the user workload, providing a method to cluster time series data, based on a neutral parameter instead of the grid size, avoiding steps that require specific computational knowledge.

Our hypothesis is that it is possible to use an variation of the SOM algorithm, that adapts its size to the input data, avoiding the prior task of determine the SOM grid size.

In this work, we analyze several evolving SOMs proposals, studying different growing mechanisms to dynamically evolve the map structure. The differences between the several evolving SOMs are explored, looking at each solution benefits and penalties, and highlighting the advantages of Growing Self-Organizing Map (GSOM) over other approaches. After the analysis, the use of GSOM is proposed to be used as an alternative of traditional SOM for satellite image time series clustering.

This work also review the available implementation of GSOM solutions, testing the packages/scripts and validating the results. By the end of this phase, a Python package is described to showcase the GSOM algorithm implementation. By using this approach, the indirect knowledge discovery of unsupervised learning provided by GSOM is tested, and the benefits of the organic map growing are validated.

Finally, is also scope of this work to apply the GSOM on the clustering of satellite image time series, comparing its results with the obtained by Santos et al. (2019). By using the same samples and the same approach, but with a different algorithm, this work assure the accuracy of the GSOM time series clustering, comparing it with the traditional SOM and checking the characteristics of the generated map. We also apply the GSOM on another time series data set, in order to validate its clustering capabilities.

1.2 Main contributions

This work is part of the Brazil Data Cube Project. This project creates multidimensional data cubes from Brazilian territory and applies machine learning methods and satellite image time series analysis to generate land use and land cover infor-

mation from these data. In this project, the SOM method is being used for satellite image times series clustering Santos et al. (2019). So, this work contributes to land use and land cover research area by validating an approach that avoids SOM grid size parameter.

Despite using satellite images time series clustering on the whole work, the analysis can be expanded for generic time series cases. After reviewing the original GSOM algorithm, we propose a customization, in order to fine tune the clustering method to better fit the time series clustering. The proposed changes aim to better adapt the algorithm to the proposed problem.

After testing the available GSOM implementations, we also contribute releasing a new python package including the base algorithm and the above proposed changes. The source code is available on GitHub, as an open source package, granting users the rights to use, study, change and distribute the software to anyone and for any purpose. Further works are also able to retrieve the functionalities needed to reproduce and improve this work.

We applied the Growing SOM into a data set extracted from satellite images, and compare its results with the SOM results obtained by previous works. We also compare the network topology, clustering accuracy, performance and neighborhood of both algorithms, to contribute validating GSOM as a feasible alternative to SOM on the proposed scope.

This work also derived two publications. The first one is a short paper called "Evaluating Growing Self-Organizing Maps for Satellite Image Time Series Clustering" presenting the preliminary results of this study at GeoInfo 2019 - Brazilian Symposium on Geoinformatics (ADEU et al., 2019). The second one is a full paper accepted in MALSEOD 2020 workshop (Machine Learning for Space and Earth Observation Data) at ICCSA (International Conference on Computational Science and Applications) and waiting to be published in LNCS - Springer Lecture Notes in Computer Science (ADEU et al., 2020).

2 LITERATURE REVIEW

2.1 Land use and land cover

Land use and land cover relate to the usage and biophysical cover of the Earth's terrestrial surface, identifying bare soil, inland water, vegetation or human infrastructure and the changes between them. Accurate land use information is required for science, monitoring and reporting. Land cover changes naturally over time, as well as a result of human activities. Monitoring and mapping of land use and land cover in a consistent and robust manner over large areas is made possible with Earth Observation (EO) data (GOMEZ et al., 2016).

Satellite data can be used to map and monitor land cover over daily to weekly temporal scales and from continental to local scales. Advances in spatial analysis software, image processing libraries and database management tools have enhanced the ability to analyze these data for representing land cover and land use (TREITZ, 2004). Remote sensing and digital image processing enables observation, identification, mapping, assessment, and monitoring of land cover at a range of spatial, temporal, and thematic scales (ROGAN; CHEN, 2004). Remote sensing applications have taken advantage from recent developments, which were focused on interoperability with GIS, ease-of-use improvements and increased availability of algorithms for remote sensing data automated processing (FRANKLIN, 2001).

A time series of land cover maps can capture the complexities of Earth's changing surface. The inclusion of time series in the land cover mapping process provides information on class stability and informs on logical class transitions (both temporally and categorically) (LIU; CAI, 2012). Therefore, successful utilization of remotely sensed data for land use and land cover monitoring requires careful selection of an appropriate data set and image processing technique(s) (LUNETTA; ELVIDGE, 1998).

2.2 Satellite image time series

One of the recent leading research trends in land use science is the use of large time series data sets derived from Earth observation. Multiyear time series composed of multiple land surface attributes allow a broader and in deep view of land use and land cover (PICOLI et al., 2018).

Since remote sensing satellites revisit the same place, we can calibrate their images so that measures of the same place at different times are comparable (Figure 2.1(a)). These observations can be organized so that each measure from the sensor maps to a

three-dimensional array in space-time. From a data analysis perspective, each pixel location (x, y) at consecutive times, t1, ...,tm, makes up a satellite image time series, such as the one in Figure 2.1(b). From these time series, we can extract land use and land cover information.

Figure 2.1 - Deriving Time series from Earth observation satellite images.
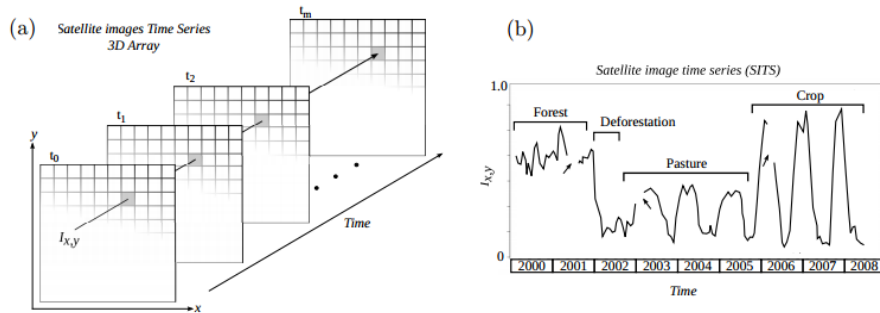
Based on a collection of satellite images, its possible to create a dimensional array of satellite images (a), and extract a vegetation index time series at a fixed (x,y) pixel location (b).

SOURCE: Maus et al. (2016)

A lot of studies have shown the interest of such multi-temporal satellite images to improve the accuracy of several classes of vegetation classification (PETITJEAN et al., 2010; SABOUR et al., 2007). In fact, different types of vegetation have different phenologies, i.e. different life cycles. As a consequence, the vegetation are easily discriminate by using multi-temporal satellite images than by analyzing a single image (GUYET; NICOLAS, 2015).

In the process to extract land use and land cover information from Earth observation data sets, Santos et al. (2019) proposes the use of SOM to improve the training step of the land cover classification. It is used to assess the quality of the land use and land cover samples and to evaluate which spectral bands and vegetation indexes are best suitable for the separability of land use and land cover classes.

2.3   Self-organizing Map (SOM)

A Self-Organizing Map (SOM) is an unsupervised neural network. SOM maps a high dimensional space onto a low-dimensional space preserving its neighborhood topology. SOM is composed by input and output layers, with the latter generally

being a two-dimensional grid.

Each element of a grid is called neuron. An important property of SOM is that the neurons are organized in a way that they maintain a similar neighborhood, that is, neurons that have similar characteristics are close in the output layer. Figure 2.2 shows the structure of SOM.

Figure 2.2 - SOM Structure.
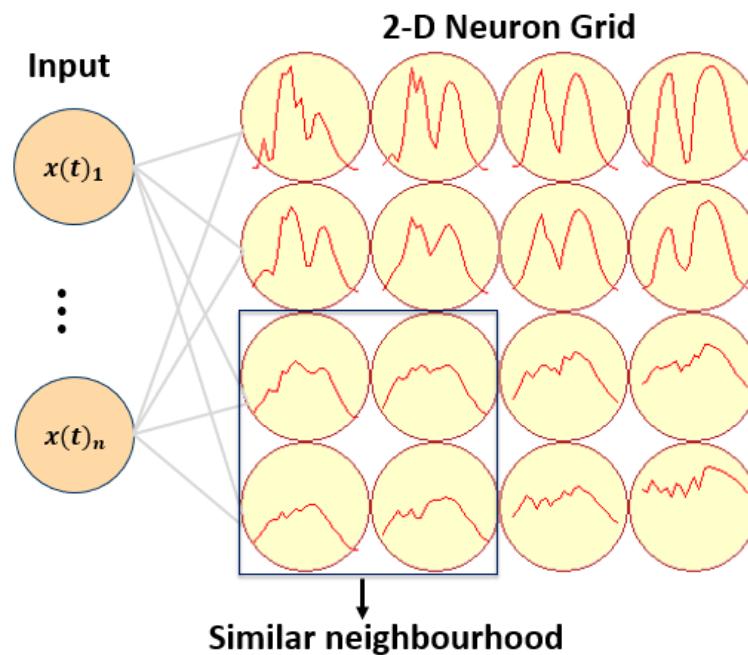
An example of a 4x4 SOM structure. The input vector is presented to the grid. The most similar neuron adapts its weight vector, trying to approximate the samples. The neighborhood of this neuron is also updated, generating a similar neighborhood.

SOURCE: Author

Each neuron j has a n-dimensional weight vector $w_j = [w_1, \ldots, w_n]$ associated to it. An input $x(t) = [x(t)_1, \ldots, x(t)_n]$ is associated with most similar neuron to it through distance metrics, as Euclidean distance. The distance D_j is computed between an input vector and each neuron j for all the neurons in the output layer (Equation 2.1).

$$D_j = \sum_{i=1}^{N} \sqrt{(x(t)_i - w_j)^2}. \tag{2.1}$$

Once we have with all distances between an input and all neurons, the minimum distance is determined the Best-Matching-Unit (BMU), i.e. the neuron d_b with weight vector closer to $x(t)$ (Equation 2.2):

$$d_b = min\left\{D_1, \ldots, D_j\right\}. \tag{2.2}$$

The BMU and its neighbors within a radius r must be updated. The weights are adjusted to increase the similarity with the input vector, the update is given by:

$$w_j(t) = w_j(t) + \alpha(t) \times h_{b,j}(t)[x(t)_i - w_j(t)], \tag{2.3}$$

In Equation 2.3, $\alpha(t)$ is the learning rate, set as $0 < \alpha(t) < 1$ and $h_{b,j}$ is a neighborhood function. The SOM mapping ends when all input vectors are presented to the output layer. During each time, the $\alpha(t)$ must be reduced and the neighborhood function reduces the radius of the neighborhood. There are several ways to reduce the value of $\alpha(t)$ and the radius of the neighbors, they can be found in Natita et al. (2016).

2.4 SOM on time series clustering

SOM has been widely used for time series clustering in various domains, such as meteorology and oceanography (LIAO, 2005; MWASIAGI, 2011; LIU et al., 2016; PEARCE et al., 2014). In many studies, authors point out the benefits of SOM to identify temporal patterns from time series and to visualize big amounts of time series. However, in land use and land cover domain, the neural network SOM has not been widely exploited for image time series analysis.

Santos et al. (2019) proposes a methodology that can be used as an exploratory analysis tool for land use and land cover samples from remote sensing image time series, based on SOM properties. It also intends to provide means to detect sample outliers textually and not only visually, using neighborhood analysis. In Figure 2.3 there are neurons labelled as Soy-Corn, Millet-Cotton and Soy-Sunflower in the middle of the region classified as Soy-Cotton. Since the classes of the samples in

these neurons match the input classes, two possibilities are considered. The first hypothesis is that these input samples are outliers, insofar as their classification has been mapped to different locations in the SOM map that other inputs of the same class. A competing hypothesis is that there is some inherent confusion between some of the classes that cannot be resolved by the SOM methods with the current input samples.

Figure 2.3 - SOM Clustering and Possible Outliers
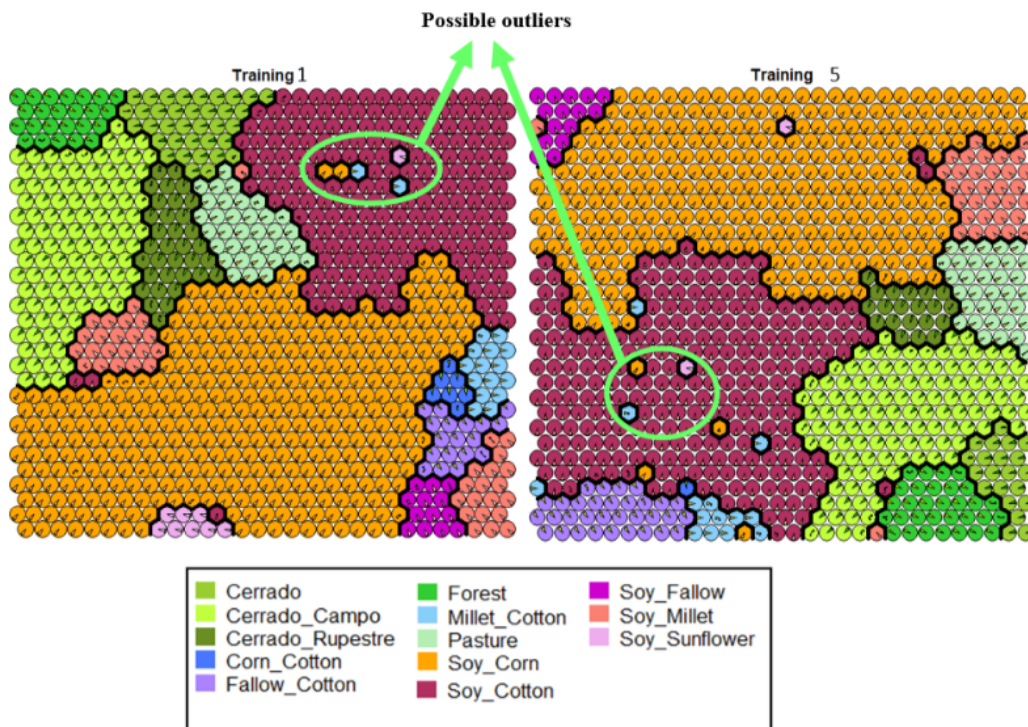
Two SOM results for the same dataset. Analyzing neighborhood and identifying neurons labeled with different classes than its neighbors highlights possible outliers.

SOURCE: Santos et al. (2019)

Similarity measure is a key aspect for achieving effectiveness in time series analysis and working with time series is very expensive in terms of processing cost (DING et al., 2008). It's important to observe that the distance function used by time series SOM comparisons, directly influences the clustering accuracy and the overall time spend on the clustering. For remote sensing image time series clustering, Euclidean

and Manhattan distances are more accurate than DTW (FERREIRA et al., 2019).

The literature shows that the determination of grid size is an empirical process. The amount of training samples can influence the size of the grid, for large data sets, reasonably large maps are better (FLEXER, 2001; KOHONEN et al., 2001). Besides that, super estimating the grid implies on empty neurons without any associated sample, or neurons without generalization capabilities, with a single sample associated. Underestimating the grid implies on generalist neurons and clustering errors. In the next sections, alternatives to evolving the grid dynamically are presented, in order to simplify the sizing of the grid and avoid generalization capability issues, while trying to preserve the neighborhood.

2.5 Evolving SOM methods

There are several proposed methods to address the traditional SOM fixed grid. Most of them rely on identifying the insertion point for a new neuron/map, and use a specific approach do grow the map. In the next sections, some of these methods are detailed.

2.5.1 Growing hierarchical self-organizing maps (GHSOM)

The key idea of the GHSOM is to use a hierarchical structure of multiple layers where each layer consists of a number of independent SOMs. One SOM is used at the first layer of the hierarchy. For every unit in this map a SOM might be added to the next layer of the hierarchy. This principle is repeated with the third and any further layers of the GHSOM (DITTENBACH et al., 2003). This generates a three-dimensional tree-structure in order to represent the hierarchical structure present in a data collection during an unsupervised training process.

This neural network architecture is capable of identifying the required number of units during its unsupervised learning process. Additionally, the data set is clustered hierarchically by relying on a layered architecture comprising a number of independent self-organizing maps within each layer. The actual structure of the hierarchy is determined dynamically to resemble the structure of the input data as accurately as possible (DITTENBACH et al., 2002).

A graphical representation of a GHSOM is given in Figure 2.4. The map in layer 1 consists of 2x2 units and provides a rather rough organization of the main clusters in the input data. The three independent maps in the second layer offer a more detailed view on the data. The input data for one map is the subset which has been mapped

on to the corresponding unit in the upper layer. One unit in the first layer map has not been expanded into a map in the second layer because the data representation quality was already accurate enough. It has to be noted that the maps have different sizes according to the structure of the data, which relieves us from the burden of predefining the structure of the architecture.

Figure 2.4 - Architecture of a GHSOM.
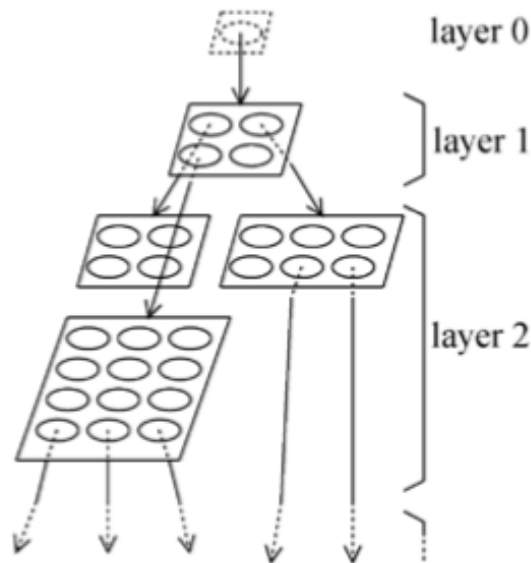
The GHSOM evolves to a structure of SOMs reflecting the hierarchical structure of the input data. Each neuron with accumulated error above a threshold generates a new layer with a new SOM associated to it.

SOURCE: Dittenbach et al. (2002)

2.5.2 Growing cell structure of map (GCS)

The Growing Cell Structure of Map presented here is proposed by Wu e Yen (2003) and provides the ability to automatically find a suitable network structure and size. This is achieved through a controlled growth process that also includes occasional removal of units. The method starts with a small grid size. The proposed approach suggests three units forming a triangle. The network structure can grow based on two different conditions: (1) add neurons in areas that receive a high number of inputs, or (2) add neurons in areas with the highest accumulated error of a neuron

11

(i.e., the largest sum of the distance between a neuron's codebook vector and all inputs belonging to that neuron) (SAMARASINGHE, 2006). The connections between the neurons are updated after the insertion of new neurons, so that the triangular connectivity is maintained.

In the Figure 2.5, the initial map is the three-neuron map (a). The cross indicates the position where a neuron needs to be added, based on the criteria stated above. The next image (b) in Figure 2.5 shows the new map with the new neuron added and the map reorganized to maintain a triangular structure. The third image (c) shows the map extended after adding a neuron at the position of the cross in the second map (b) and reorganized to maintain the triangular structure. The last image (d) in Figure 2.5 shows a map that has grown further. This process continues until it no longer is necessary to grow the map further (SAMARASINGHE, 2006).

Figure 2.5 - GCS growing example.
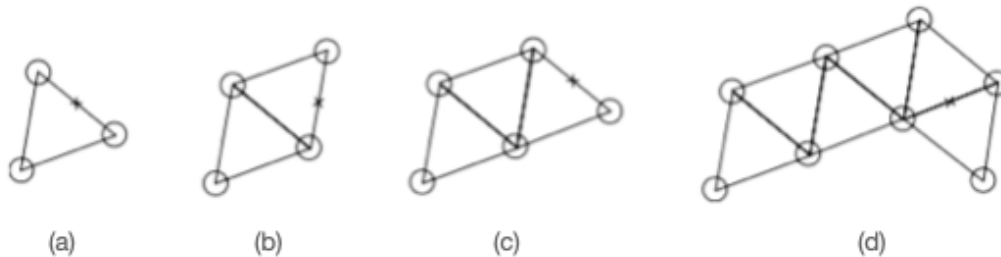
Schematic illustration of map growing in Growing Cell Structure (GCS) method. The initial map is a three-neuron map (a). The cross indicates the position where a new neuron needs to be added. The subsequent figures (b), (c) and (d) illustrates the map growing, with new neurons being added on the cross position.

SOURCE: Samarasinghe (2006)

With the addition of a new neuron, GCS intends to create a new Voronoi region for that neuron, where the inputs closer to this new neuron fall. The properties of the new neuron are interpolated from their neighbors. This redistribution of the inputs is reflected in the GCS method of redistributing the errors of the two existing neurons with the newly created one (FRITZKE, 1995a). This guarantee local order in the map and in nearly all cases also leads to global ordering.

2.5.3 Growing neural gas (GNG)

The Neural Gas is an algorithm to represent network topologies based on a Hebbian adaptation rule with winner-take-all like competition (MARTINETZ; SCHULTEN, 1991). Based on this concept, the Growing Neural Gas (GNG) is proposed, as an incremental network model that learns topological relations by adding neurons and connections, until a performance criterion has been met (FRITZKE, 1995b). It uses a similar approach as the Growing Cell Structure of Map, but it replaces the topology with a fixed dimensionality for a dimensionality which depends on the input data and may vary locally.

Growing Neural Gas has two different phases alternating until a threshold is met. The first one is the adaptation phase, where the neural network adapts itself to a random signal input. At this time, a connection between the input signal two nearest neurons is strengthened (or created if it does not exist), the nearest neuron and all its direct neighbors move towards the input signal and the nearest neuron error is increased. Also, an aging neuron mechanism is triggered, removing from the network neurons not strengthened for a long time. The second one is the growing phase, where new neurons are created and connected into the network. The neuron with the largest error in the whole network is found, and a new neuron is created between this neuron and its neighbor with the largest error (FISER et al., 2013). Figure 2.6 shows an example of how GNG network adapts to a signal distribution which has different dimensionalities in different areas of the input space.

Figure 2.6 - The Growing Neural Gas network growing
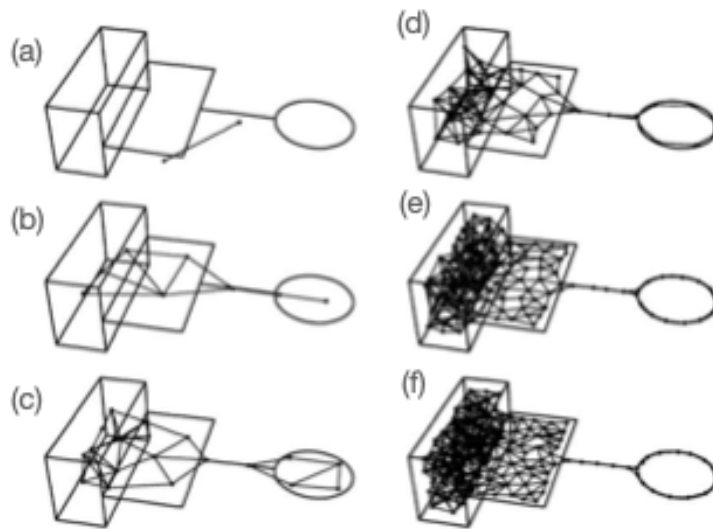
Schematic example of a Growing Neural Gas network adapting to a signal distribution. The initial network (a) starts with 2 randomly placed neurons and evolves, filling the space applying 600 (b), 1800 (c), 5000 (d), 15000 (e) and 20000(f) input signals

SOURCE: Fritzke (1995b)

2.5.4 Growing SOM (GSOM)

Growing SOM (GSOM) is an alternative to traditional SOM for satellite image time series clustering. It is originally proposed to address the SOM requirement of pre-determining the map size (ALAHAKOON et al., 2000). SOM attempts to fit a data set into a predefined structure by self-organizing its node weights as well as possible within its fixed borders, while in GSOM the network borders are expandable, generating new nodes whenever needed to expand the network outwards.

The GSOM is parameterized by a *spread factor*, a data dimensionality neutral factor. It can be used as a controlling measure for generating maps with different sizes, without previous knowledge about the data set number of samples or attributes.

Figure 2.7 - Initial GSOM grid with four neurons.
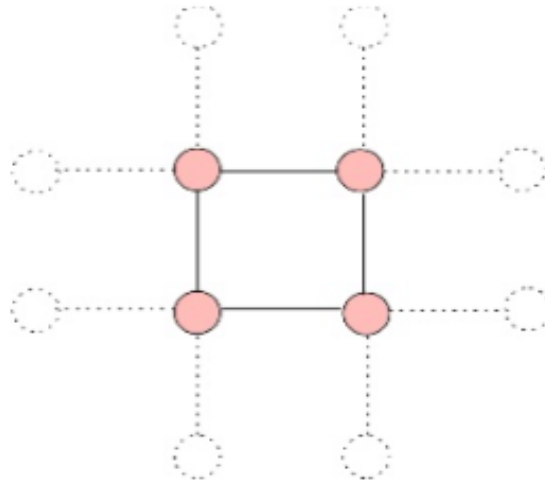
The GSOM grid starts with 4 nodes, and 8 available positions for new nodes. The available positions are located on the nearest neighborhood of the initial grid. The node with the highest accumulated error will start the growing on its nearest neighborhood.

SOURCE: Vasighi e Abbasi (2017)

The GSOM learning algorithm has three phases:

Initial phase: At the initialization phase, GSOM network starts with four neurons with randomly assigned weight vectors. All the initial neurons are boundary nodes and have the opportunity to grow. In Figure 2.7, the four initial neurons are connected with lines and the available positions are shown via dashed circles.

Growing phase: In this phase, the time series data are presented to the network. The weight vector that is closest to the input vector mapped (the winner neuron), is selected based on a distance function. The winner neuron accumulative error is calculated according to a distance function between the input vector and the winner neuron weight vector. This error indicates the distance between the input vector and the weight vector of neurons. When the accumulated error of a neuron exceeds a *growing threshold* (calculated based on the spread factor), and the candidate neuron is on the grid boundary, new neurons are added in the available free positions around the candidate neuron in the grid, as shown on Figure 2.8. If the winning

15

neuron is not on the grid boundary, the accumulated error of the neighbors are updated according to the winner's distance, giving the nonboundary nodes some ability in initiating node growth.

Smoothing phase: In order to fine-tune the weight vector position and improve the map smoothness, a smoothing phase is applied after the growing is completed (VASIGHI; ABBASI, 2017). No new nodes are added during this phase. The purpose is to smooth out any existing quantization error, especially in the nodes grown at the growing phase latter stages (ALAHAKOON et al., 2000). The smoothing phase is stopped when the node error values in the map become very small.

Figure 2.8 - New node generation from the network boundary.
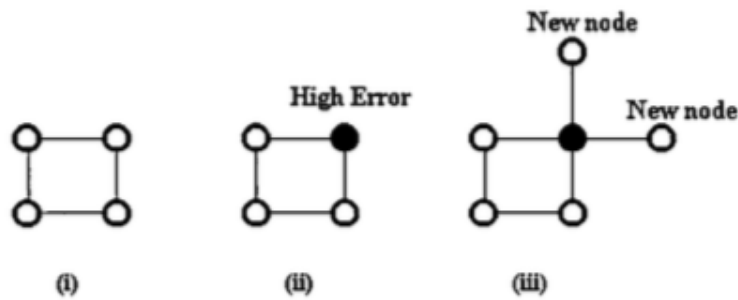
Any node with at least one of its immediate neighboring positions free is considered a boundary neuron. If the accumulated error of the node with the highest accumulated error exceeds a growing threshold, new nodes are generated on all free neighboring positions.

SOURCE: Alahakoon et al. (2000)

This process generates the GSOM, which develops into different shapes depending on the clusters present in the data. The GSOM shape represents the data grouping, and therefore, such grouping has a better opportunity of attracting the user attention for further investigation (ALAHAKOON et al., 2000).

2.6 Evolving SOM - methods evaluation

After reviewing the available Evolving SOM methods, and selecting the most referenced ones (GHSOM, GCS, GNG and GSOM), they are compared, to highlight

the differences between the methods. The objective is to select a candidate to be used in replacement of SOM on satellite image time series clustering. All the reviewed solutions address the fixed grid restriction imposed by traditional SOM in a different way.

On the GHSOM, the hierarchical relationship between the multiple SOMs, generated by the additional SOM layers, turns the data visualization confused. It is not possible to understand the relationship between the SOMs in the same layer, as this relationship is not explicit. The neighborhood concepts between the neurons are also not clear, since the samples are located in different neurons depending on the layer.

Although the concept behind the GCS of Map seems to be simple, there is some uncertainty on the algorithm. Fritzke (1993) highlights that the position for inserting the new neurons could be the neighbor with the maximum accumulated error, the neighbor with the most distant center position or even a randomly picked neighbor. Nonetheless, there is some arbitrariness in the definition of the conditions that have been suggested to define the directions of growing. Arbitrary branching criteria define arbitrary structures in the network (KOHONEN, 2013). This leads us to believe that GCS does not provide an organic growing of the map.

Fiser et al. (2013) states that GNG has a poor time performance, particularly on large scale problems, and this can be a limiting factor for further applications or a wider usage. This work validates the clustering data sets composed by time series derived from satellite images. An important characteristic of this data set is that its size tends to increase quickly with the time. Besides that, the growing mechanism provided by GNG is a complicating factor, interfering on the visual inspection of the similar clusters and turning the outliers identification into a complicated task.

Due to its flexible structure, the GSOM achieves the same amount of spread that traditional SOM, with a lesser number of nodes, and provide a useful advantage in mapping large data sets. In addition, such flexible structure provides a better visualization of the groups in the data and attracts attention to such groups (and outliers) by branching out. This characteristic also preservers the neighborhood, during the map growing. It should be added that the GSOM has preserved the simplicity and ease of use of the SOM and has expanded its usefulness by dynamically generating the map structure (ALAHAKOON et al., 2000). This characteristic, combined with the detailed specification presented on Alahakoon et al. (2000) turns the GSOM into the selected option as a SOM alternative into the satellite image time series clustering.

3 GROWING SOM FOR SATELLITE IMAGE TIME SERIES CLUSTERING

3.1 Review of GSOM implementations

As a starting point, some implementations of the algorithm were searched. The three implementations found were PyGSOM (LUDWIG, 2016), Kohonen-Maps (RALHAN, 2018) and GSOM (MENDIS, 2015). These implementations were tested, but the results were not satisfactory. In some cases the performance are not acceptable, or the algorithm specification is not precisely implemented. On PyGSOM, different possible approaches for the GSOM algorithm are used in the implementation, resulting in a mixed solution. Ludwig (2016) states that this implementation should not be taken as a reference. Kohonen-Maps implementation seems to be a study based on the GSOM specification, manually constructed to fit a specific data set. Any modification to adapt this solution to fit any larger time series data set could result in a huge manual job. The GSOM implemented by Mendis (2015) stores only the last sample associated to each neuron, instead of all the samples. As a consequence, visualization of the best matching unit is based on the last classified sample, not on the most common. Furthermore, after running examples, this solution seems to not respect the neighborhood while growing the grid. In Figure 3.1 is possible to notice neurons holding samples from the same cluster, but spatially distant on the generated grid.

Figure 3.1 - Python GSOM implementation.
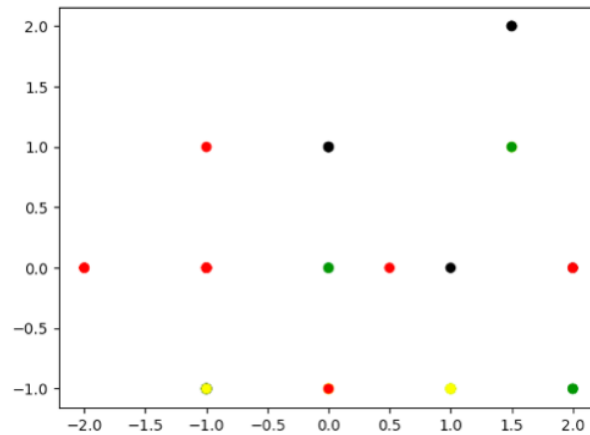
Python GSOM grid for Iris data set. Neurons labeled with the same class are spatially distant on the grid, indicating that the neighborhood similarity was not preserved.

SOURCE: Author

To establish the comparison between the use of SOM into the clustering of satellite image time series as specified on Santos et al. (2019) and the possible improvements provided by GSOM, a search for a GSOM implementation on R environment was started. The only GitHub available R package is GrowingSOM (HUNZIKER, 2018). Unfortunately, this package is not available as an official R package on CRAN. After installing and testing this package, was noticed that this implementation does not store the relationship between the samples and the neuron associated with them. The visualization features and the developed public interfaces are also limited. But the main concern of this implementation is the training performance. As this solution is fully implemented in R, and the main goal of this work is the clustering of satellite image time series, the performance is not acceptable.

3.2 GSOM customization for satellite image time series

In this work, we propose minor changes in the GSOM described by Alahakoon et al. (2000). These changes aim to fit the algorithm to work with satellite image time series. It is important to advise that in all the time series comparison made by GSOM, the Euclidian distance is used.

During the network weights adaptation, the described learning rate states that it

needs to be a function that gradually takes higher values as the map grows and the number of nodes becomes larger (ALAHAKOON et al., 2000). The suggested function is described by equation (1) with $R = 3.8$, and $n(t)$ is the number of neurons on a given iteration t.

$$(\frac{1-R}{n(t)}) \tag{3.1}$$

The use of this learning rate function result in a classification accuracy decrease. In order to improve the accuracy, this function is replaced by the function described by equation (2), that gradually takes minor values as the number of epochs increases:

$$e^{(\frac{-iteration}{epochs})}, \tag{3.2}$$

with *iteration* representing the current iteration number and *epochs* representing the total amount of epochs.

Alahakoon et al. (2000) states that the GSOM starting neighborhood selected for weight adaptation is smaller compared to the SOM, and weight adaptation is carried out by reducing neighborhood until neighborhood is unity. But a function for initializing and reducing the neighborhood influence during the growing and smoothing phase is not provided. To satisfy these requirements, after several tests, we defined the neighborhood influence as showed by Equation (3):

$$e^{(\frac{-d}{2 \times \sigma^2})}, \tag{3.3}$$

$$\sigma = ini \times e^{(\frac{-iteration}{epochs})}, \tag{3.4}$$

where *ini* is the initial neighborhood influence, *iteration* is the current iteration number, and *epochs* is the total amount of epochs. On all GSOM executions used in this work, the initial neighborhood influence varies between 0 and 1. Alahakoon et al. (2000) reinforces that the neighborhood influence in GSOM should be smaller when compared to regular SOM.

3.3 Python package development

After reviewing the available GSOM implementations, and looking at the algorithm description provided by Alahakoon et al. (2000), we decide to build an implementation from scratch. The intend is to better understand each step of the algorithm, and be able to modify any needed parameter to fine tune GSOM to fit the time series clustering approach.

The source code of the developed package is available on GitHub (GSOM..., 2020) and can be installed as follows:

```
python3 -m pip install
git+https://github.com/rodrigosales/GSOM.git
```

In order to use the package, all python source codes must use the import directive with the package name:

```
import GSOM
```

The package allows two different input methods, the first one starts a GSOM execution, with both Growing phase and Smoothing phase. On the above example, a GSOM grid is initialized with a Spread Factor configured as 1.0 and a 0.7 Learning Rate. After that, a Growing Phase with 5 epochs will be started, and followed by a Smoothing Phase with 15 epochs.

```
init_grid (input, sf = 1.0, alfa = 0.7)
start_growing_phase (input, 5)
start_smoothing_phase (input, 15)
```

As the SOM algorithm is very similar to a GSOM Smoothing phase, an alias for SOM execution is also provided as a second input method. In order to start a simple SOM, we need to provide the grid size and a Learning Rate. In the following example, a 25x25 grid is initialized with a 0.6 Learning Rate, and 15 epochs will run.

```
init_grid (input, 25, 25, 0.6)
start_som (input, 15)
```

We provide 5 output methods, to be able to retrieve GSOM results. The first one prints the best matching unit of each sample, indicating the neuron that best cluster each sample.

```
print_clustering(input)
```

In order to provide a classification based on the generated clusters, a method is developed to label a neuron, based on the most frequent samples associated to each neuron, this output can be retrieved as:

```
neuron_labels = get_neuron_labels(input, input_labels)
```

Based on the input labels, and the retrieved neuron labels, it is possible to get the accuracy for each neuron:

```
check_neuron_accuracy(input, neuron_labels, input_labels)
```

Based on the same data, we are able to generate the confusion matrix, and get the classification accuracy for each class:

```
generate_confusion_matrix(input, neuron_labels, input_labels)
```

Finally, providing e color for each label, we are able to plot a scatter and visually check the network topology:

```
labels_colors = {
    "Pasture":"#F77B01",
    "Cerrado":"#2B6490",
    "Forest":"#4EAF4A",
    "Fallow_Cotton":"#E41B1D",
    "Soy_Fallow":"#C26596",
    "Soy_Corn":"#BFC127",
    "Soy_Sunflower":"#CCEBC5",
    "Soy_Millet":"#949494",
    "Soy_Cotton":"#A65628"
}


plot_map(input, input_labels, neuron_labels, labels_colors,
show_samples = False)
```

All the GSOM results presented in this text were obtained using the above functions.

4 GSOM TESTS AND ANALYSIS

In order to check the accuracy of GSOM for satellite image time series clustering and to compare it with SOM, we implemented both SOM and GSOM in Python (PYTHON SOFTWARE FOUNDATION, 2020). GSOM implementation includes changes proposed in Section 3.2 of this work. This implementation was tested in two different data sets, providing case studies with different analysis.

4.1 Case study 01

As a first case study, we executed SOM and GSOM in the same time series data set described in Santos et al. (2019). In this case study, we run both algorithms and compare the results, analyzing the network topology, performance, cluster accuracy and neighborhood influence.

4.1.1 Data set

The methodology to extract the satellite image time series data set used in our case study is shown in Figure 4.1. This data set is composed of 2215 ground samples of land use and land cover classes, including natural vegetation and agricultural, from Mato Grosso state in Brazil. These samples refer to nine distinct land use and land cover classes. These classes and the samples are detailed on Table 4.1.

Table 4.1 - Ground samples used in case study 01.

Class Label	Count	Frequency
Cerrado	400	18.9%
Fallow-Cotton	34	1.6%
Forest	138	6.5%
Pasture	370	17.5%
Soy-Corn	398	18.8%
Soy-Cotton	399	18.9%
Soy-Fallow	88	4.2%
Soy-Millet	235	11.1%
Soy-Sunflower	53	2.5%

Each sample has a spatial location (latitude and longitude), start and end date that corresponds to the agricultural year (from August to September) and the corresponding sample class label. For each sample spatial location, we got the time series associated to that location or pixel from a satellite image collection ordered in

time, as shown in Figure 4.1. In this work, we used the product MOD13Q1, Collection 6 from MODIS (Moderate-Resolution Imaging Spectroradiometer) sensor of the NASA satellite Terra, including samples from 2001 to 2016, extracted every 16 days at 250 meter spatial resolution. Each time series has multiple attributes that are generated by EVI (enhanced vegetation index), NDVI (normalized difference vegetation index), NIR (near-infrared) and MIR (mid-infrared) attribute concatenation. These ground samples were retrieved from DIDAN (2015).

Figure 4.1 - Extracting satellite image time series from land use and land cover samples.
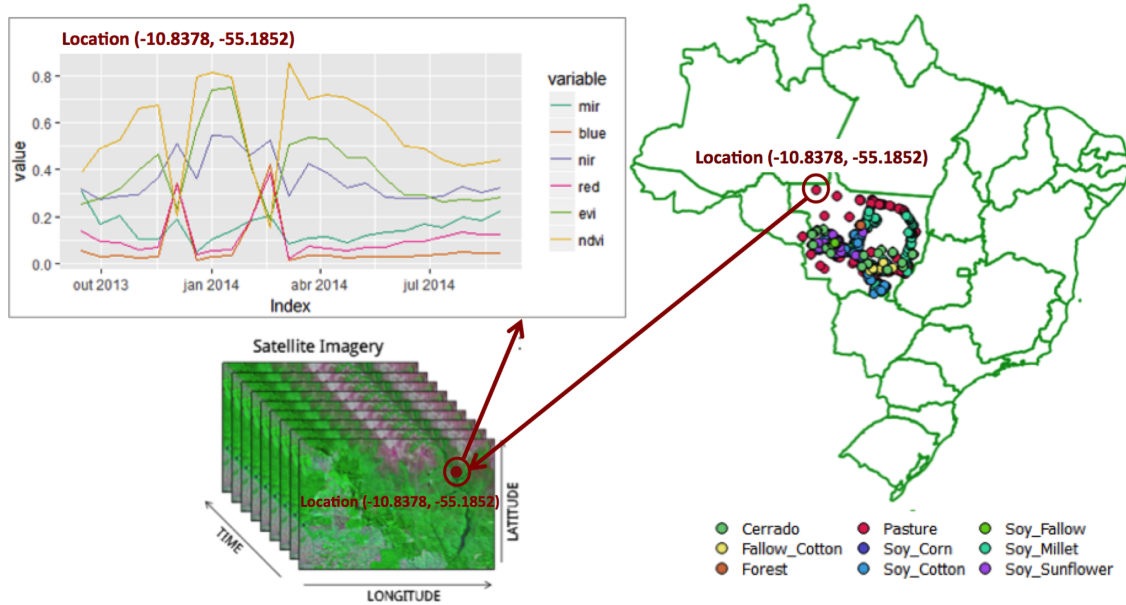
An example of a fixed spatial location in Mato Grosso do Sul state in Brasil extracted from the dataset, and the multiple satellite images associated to it. The multiple parameters collected by the satellite generates multiple time series.

SOURCE: Ferreira et al. (2019)

4.1.2 Network topology

SOM uses a fixed network architecture in terms of number and arrangement of neural processing elements which have to be predefined. To better estimate the SOM network size, a preliminary study about the number of attributes, classes separability and number of data samples needs to be done. On the other hand, GSOM requires only a spread factor as an input parameter. The spread factor is a neutral number between 0 and 1 that defines how much the network needs to grow.

Figure 4.2 shows the networks created by SOM (a) using a fixed 25 x 25 grid and GSOM (b) using a spread factor 1, learning rate 1, initial neighborhood influence 0.6 and number of epochs 15; 10 epochs for growing and 5 epochs for smoothing. Both algorithms achieved similar classification accuracy, but with different network topologies. In the SOM case, as the grid size is always prefixed, the cluster distribution can differ, but the grid will always have the same size. On the other hand, in the GSOM case, we can observe the grid growing to fit the data.

Figure 4.2 - SOM x GSOM network topology.
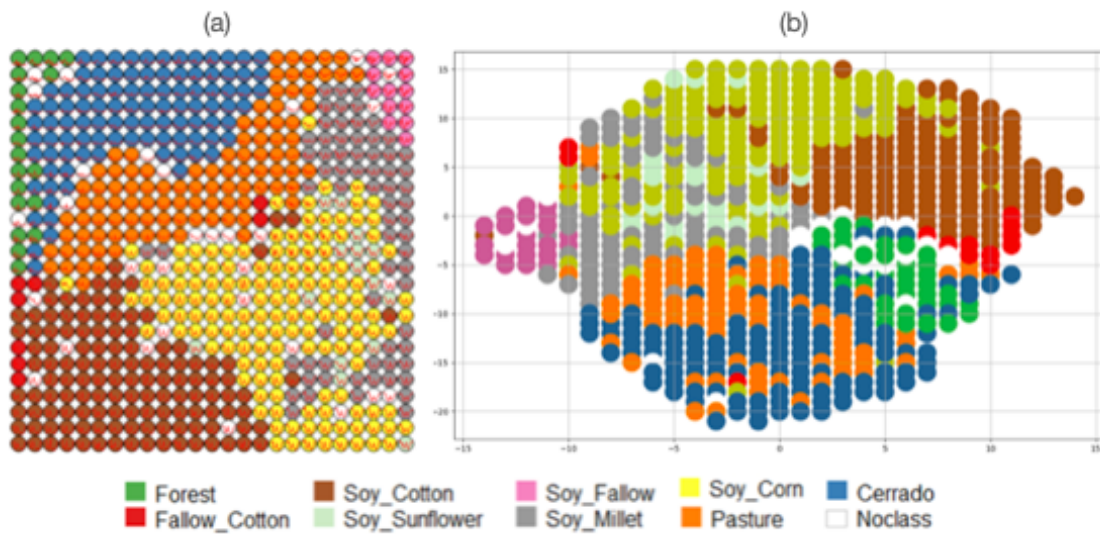
The differences between the network topology generated by SOM (a) always showing a fixed and pre-determined grid and GSOM (b) showing a variable structure depending on the input data.

SOURCE: Author

4.1.3 Running time

Trying to approximate SOM classification accuracy, several GSOM running experiments were made, testing different learning rates, spread factors and number of epochs. We noticed that, for the given data set, the GSOM map increased significantly on the number of neurons between epochs 1 and 5. Between epochs 6 and 10, the map had a small growing of the neuron number. After epoch 11, we faced almost no growing. Analyzing the smoothing phase, we noticed that after 5 epochs, the difference between the neuron weights after each epoch were above 0.001. This fact leads us to fix the number of epochs on growing phase in 10, and the number of epochs on smoothing phase in 5.

As the SOM algorithm has a fixed grid, each epoch takes approximately the same time to run. On the GSOM growing phase, the increasing number of neurons implies on a variable time spent on each epoch. Figure 4.3 illustrates the time spend by SOM (with 25 x 25 fixed grid) and GSOM parametrized with 10 Growing epochs and 5 Smoothing epochs. Both algorithms were executed 10 times, using the same learning rate and the same neighborhood update functions. Thanks to its increasing number of neurons during the growing phase, GSOM running time is smaller than SOM running time on the first 5 epochs. After that, both algorithms had similar running times.

Figure 4.3 - SOM x GSOM running time comparison.
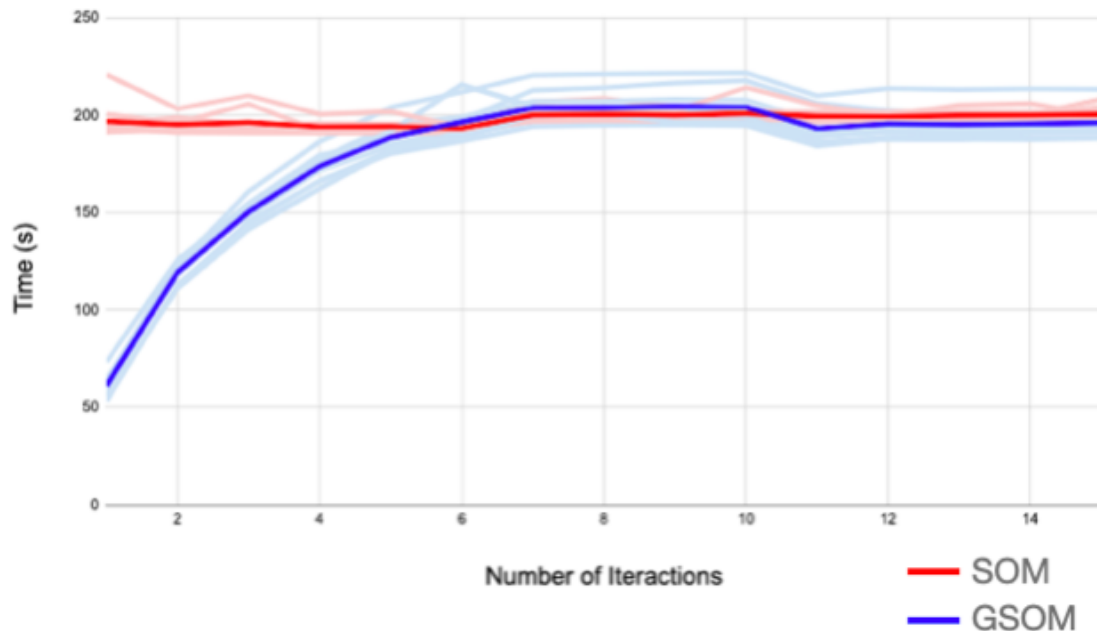
During the growing phase, GSOM has a small advantage over SOM running time, thanks to the smaller number of nodes. After the growing phase (epochs 5) both algorithms have similar running time.

SOURCE: Author

4.1.4 Cluster accuracy

After the unsupervised clustering provided by SOM and GSOM, each neuron is analyzed, and the samples associated to this neuron are counted. All neurons are labeled, using the label of the majority samples associated with it. It is important to notice that, when each neuron is labeled with the class of the majority samples, the samples associated to other classes are considered as mistakes. We are able to

calculate the resulting accuracy, but actually we are calculating the purity of the neurons, as the following statement:

> When each map node was labeled according to the majority of the subsections in the node and the abstracts belonging to the other subsections were considered as misclassifications, the resulting accuracy (actually, the purity of the nodes) was 64% (KOHONEN et al., 2000).

As a consequence, considering each neuron as a single cluster, we were able to calculate the accuracy (the purity) of the whole map.

In order to better understand the algorithm result's variability, we ran GSOM on the same dataset 10 times, calculating the cluster accuracy on each execution, using the majority sample rule. The box plot shown in Figure 4.4 summarizes the GSOM clustering accuracy for each class and for the whole dataset.

Figure 4.4 - GSOM 10 executions blox plot
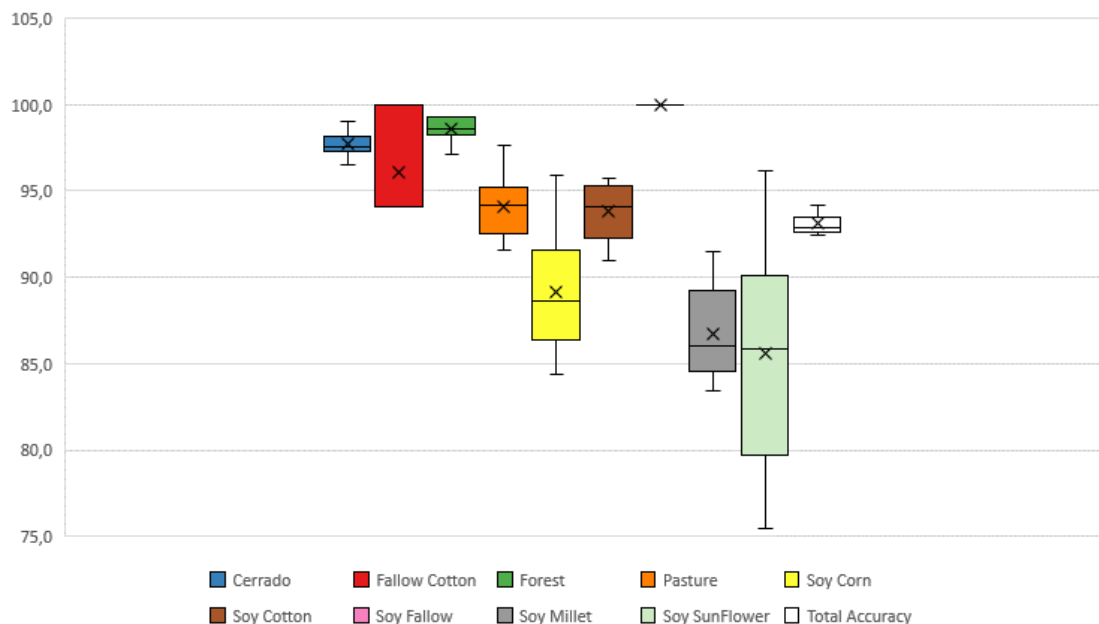
Box plot for 10 GSOM executions, showing the average accuracy (with an x), median, lower quartile, upper quartile, minimum and maximum value for each class.

SOURCE: Author

In order to compare the cluster accuracy, the same implementation of SOM and GSOM algorithms were executed 10 times, using the same learning rate and the same neighborhood update functions. As the Python implementation slows down the overall solution performance, only 15 epochs of each algorithm were ran. Before that, several GSOM executions were ran in order to generate a grid map with almost the same neurons quantity of a 25 x 25 SOM. The results are shown in Table 4.2. We noticed that, besides a small classification increment, GSOM presents significant results of +11.9% and +15.3% on Fallow-Cotton and Soy-Sunflower clustering respectively.

Table 4.2 - 15 SOM epochs x 15 GSOM epochs - Accuracy for each cluster.

Cluster	SOM Accuracy	GSOM Accuracy	GSOM − SOM
Cerrado	96.4	97.7	1.3
Fallow-Cotton	84.1	96.0	11.9
Forest	98.4	98.6	0.2
Pasture	88.9	94.1	5.2
Soy-Corn	86.1	89.2	3.1
Soy-Cotton	91.1	93.9	2.8
Soy-Fallow	99.9	100.0	0.1
Soy-Millet	82.1	86.7	4.6
Soy-Sunflower	70.3	85.6	15.3
Overall Accuracy	90.0	93.1	3.1

In the case study provided by Santos et al. (2019), the best classification scenario is obtained with the time series presented to a SOM parameterized with grid size = 25 x 25, learning rate = 1 and number of epochs = 100. The SOM implementation used in these experiments is the Kohonen R package (WEHRENS; BUYDENS, 2007). It provides the original SOM functionality with good performance due to its Rcpp implementation (EDDELBUETTEL; FRANÇOIS, 2011). The use of C++ code mixed with R code provides a significant reduction on the algorithm running time, allowing an increase on the number of epochs without a huge impact on the overall execution time. After 100 epochs, the overall classification accuracy was 93%.

The same time series attributes were presented to a GSOM, parameterized with spread factor = 1, learning rate = 1, initial neighborhood influence = 0.6 and number of epochs = 15 (10 epochs for growing and 5 epochs for smoothing). The cluster accuracy for each cluster is presented in Table 4.3. We can notice that, despite of a

small classification decrease on Soy-Millet and Pasture clusters, GSOM presents a significant increment of the Fallow-Cotton and Soy-SunFlower clustering accuracy of +10.3% and +8.7% respectively. Also, the overall sample cluster was pretty similar and stated as 93.0% for SOM and 93.1% for GSOM.

Table 4.3 - 100 SOM epochs x 15 GSOM epochs - Accuracy for Each Cluster.

Cluster	SOM Accuracy	GSOM Accuracy	GSOM − SOM
Cerrado	97.3	97.7	0.4
Fallow-Cotton	85.7	96.0	10.3
Forest	99.3	98.6	−0.7
Pasture	97.3	94.1	−3.2
Soy-Corn	84.0	89.2	5.2
Soy-Cotton	95.5	93.9	−1.6
Soy-Fallow	100.0	100.0	0.0
Soy-Millet	90.3	86.7	−3.6
Soy-Sunflower	76.9	85.6	8.7
Overall Accuracy	93.0	93.1	0.1

4.1.5   Neighborhood analysis

A relevant SOM property that needs to be validated during the GSOM tests is the neighborhood topography maintenance. On a regular SOM, similar satellite image time series are grouped into nearness neighborhoods, even if these time series are labeled as different classes. The distribution on the SOM grid over the epochs tends to cluster satellite image time series of a specific sample on the same closer neighborhood. In the specific case of satellite image time series associated with land use and land cover ground samples, the neighborhood maintenance in the clustering process is useful to identify sampling outliers.

In this case study, we observed that GSOM also keeps this property when looking at the nearest neighbors, as shown in Figure 4.5. The map highlights the nearest neighbors of Neuron 1 (N1). We also point out some random distant neighbors in blue (N593, N607, N610, and N615). Analyzing the distance between the weight vectors of Neuron 1 and the weight vectors of the nearest and some distant random neighbors, we can notice that the distance between the neuron weights grows as we move away from the comparing source. We can visually inspect the time series generated by each of this neuron weights, to check the distance between the nearest neighbors and the distant ones, as shown in Figure 4.6.

31

Figure 4.5 - GSOM grid sample extract - neighborhood analysis.
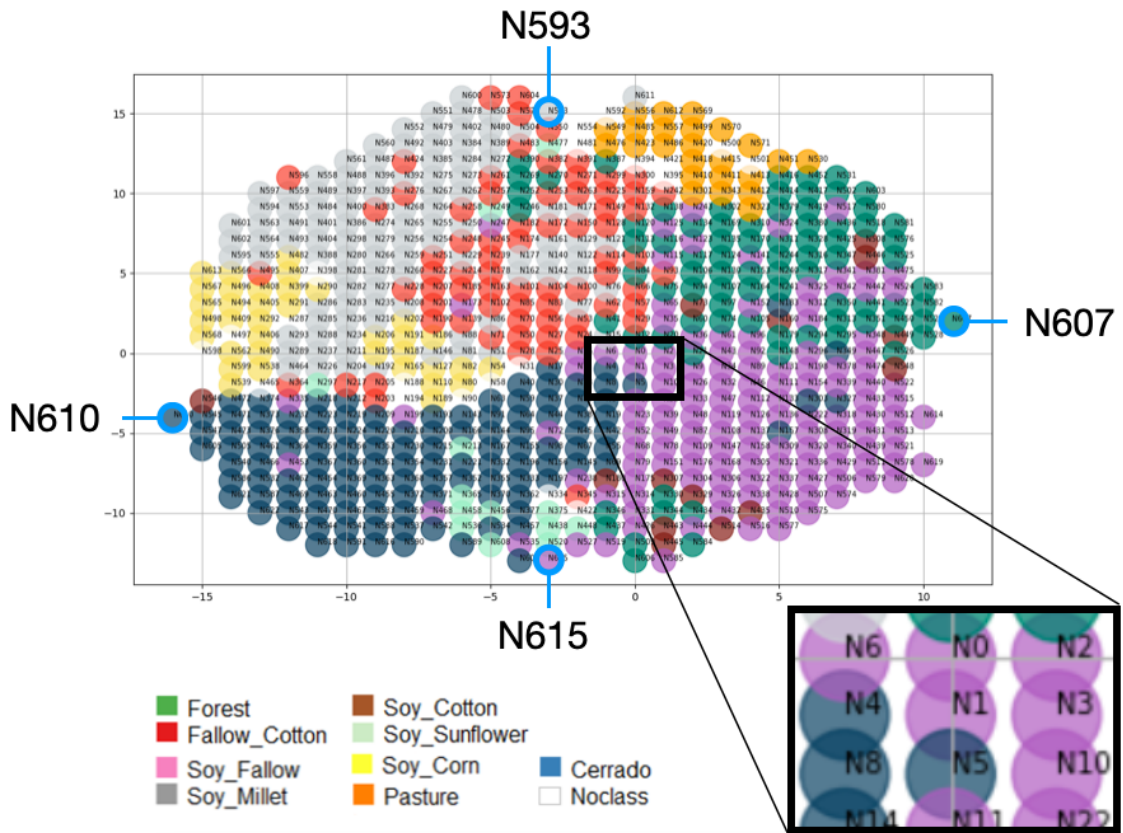
Nearest neighborhood of neuron N1 highlighted, including neurons N0, N3, N5 and N4. Random distant from neuron N1 highlighted, including neurons N593, N607, N615 and N610.

SOURCE: Author

Figure 4.6 - Neighborhood Analysis - Nearest x Random Distant Neighbors.
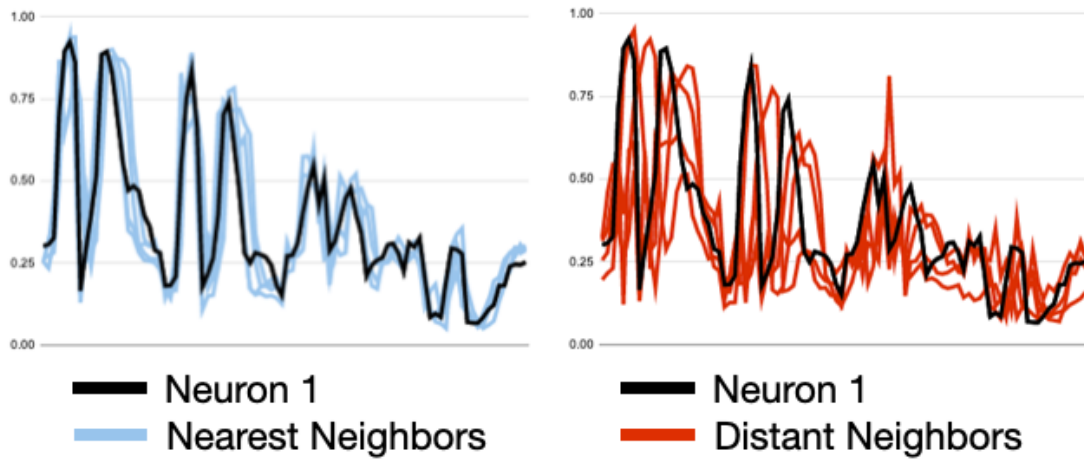
SOURCE: Author

The sum of the difference between Neuron 1 weights of its nearest neighbors is 26.94. As we move away from Neuron 1, this distance grows. As an example, the distance between the selected distant random neuron weights and Neuron 1 weights is 57.15. These distances are summarized on Table 4.4. This property was analyzed for other neurons in the map and similar results were found.

Table 4.4 - Distance between Neuron 1 weights and its nearest/farther neighbors.

N1 - Nearest	Distance	N1 - Farther	Distance
N1 - N0	7.27	N1 - N593	14.80
N1 - N3	5.67	N1 - N607	16.43
N1 - N5	6.28	N1 - N615	13.95
N1 - N4	7.72	N1 - N610	11.97
Neighborhood Distance	26.94	Neighborhood Distance	57.15

4.2 Case study 02

As a tentative to validate the GSOM algorithm applied to time series clustering, we propose the use of the algorithm in another data set. Trying to compare the results with the Chapter 4 outcome, the GSOM was initially executed with the same learning rate, neighborhood influence function and spread factor. In order to provide detailed neighborhood analysis, we then vary the initial neighborhood influence and measure the results.

4.2.1 Data set

This second data set is composed of 50360 ground samples of land use and land cover classes from Cerrado biome, on multiple states in Brazil, detailing natural vegetation and agricultural. These samples were also retrieved from product MOD13Q1, Collection 6 from MODIS (Moderate-Resolution Imaging Spectroradiometer) sensor of the NASA satellite Terra (DIDAN, 2015). These data set classes are the result of a merge between visual interpretation of high resolution images by remote sensing specialists and field observations provided by INPE (National Institute for Space Research). These classes and the samples are detailed on Table 4.5.

Table 4.5 - Ground samples used in case study 02.

Class Label	Count	Frequency
Cerrado	9172	18.2%
Cerradao	11659	23.2%
Soy-Cotton	4124	8.2%
Fallow-Cotton	630	1.3%
Silvicultura	423	0.8%
Soy-Fallow	1712	3.4%
Soy-Corn	5557	11.0%
Pasture	6602	13.1%
Araguaia	2699	5.4%
Cerrado Rupestre	6916	13.7%
Dunas	550	1.1%
Millet-Cotton	316	0.6%

4.2.2 Results

Comparing with the first data set, the increase on the data set number of samples had a direct effect on the final grid size. After one growing and five smoothing epochs

with Spread Factor configured as 1.0, the grid size was expanded from the initial 4 to 2543 neurons. Despite that, we still observing the grid growing to fit the input data, instead of an undetermined growing. Figure 4.7 shows the resultant network topology.

Figure 4.7 - GSOM - Larger data set / Larger Map (Spread Factor = 1.0)
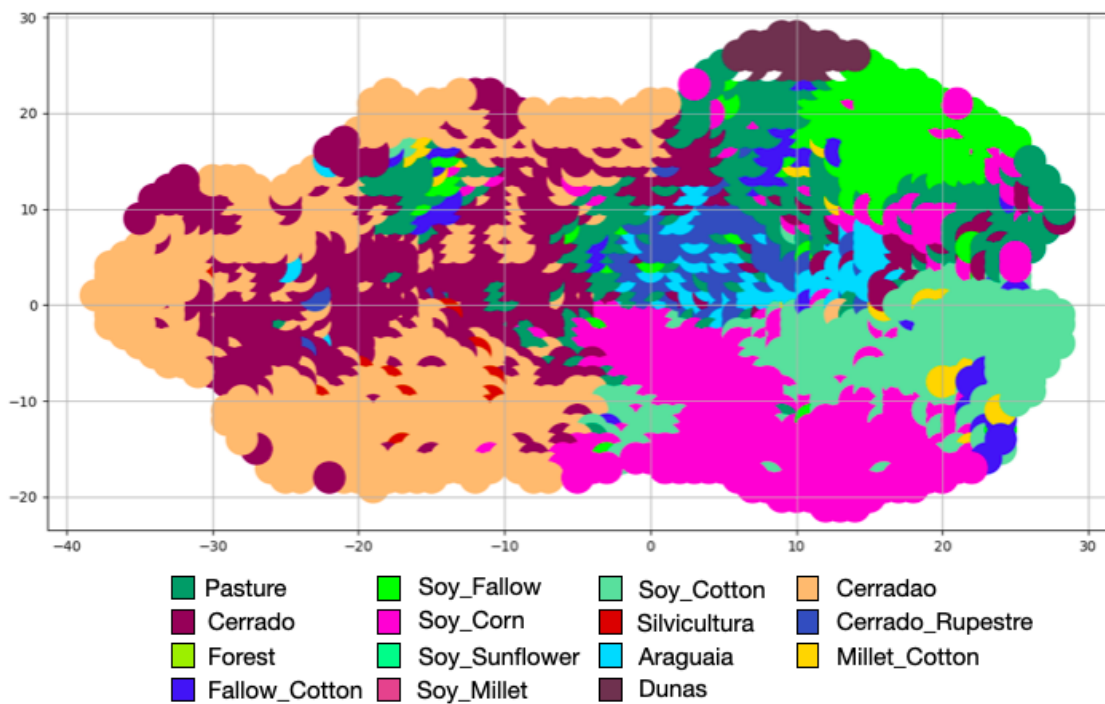
Pasture	Soy_Fallow	Soy_Cotton	Cerradao
Cerrado	Soy_Corn	Silvicultura	Cerrado_Rupestre
Forest	Soy_Sunflower	Araguaia	Millet_Cotton
Fallow_Cotton	Soy_Millet	Dunas	

Resulting GSOM topology using the larger dataset. After 6 epochs, with Spread Factor configured as 1.0, 2543 neurons were generated.

SOURCE: Author

To provide the classification capabilities and validate the clustering accuracy, the same methodology used in the first case study was used. The neurons were labeled, based on the majority samples associated with it, and the map accuracy was calculated. The results are shown on the middle column of Table 4.6.

Despite the 95.4% classification accuracy, the large network implied on a huge increase of the overall time to run the GSOM algorithm. Each of the 50360 samples need to be shown to all the 2543 neurons in order to find the best matching unit,

resulting in an expressive increase on the execution time.

Trying to decrease the whole execution time and improve the generalization capability of the network, we reduced the spread factor, in order to generate a smaller map. After two growing and five smoothing epochs with 0.7 spread factor, we were able to obtain the network showed in Figure 4.8.

Figure 4.8 - GSOM - Larger data set / Smaller Map(Spread Factor = 0.7)

Resulting GSOM topology using the larger dataset. After 6 epochs, with Spread Factor configured as 0.7, 559 neurons were generated.
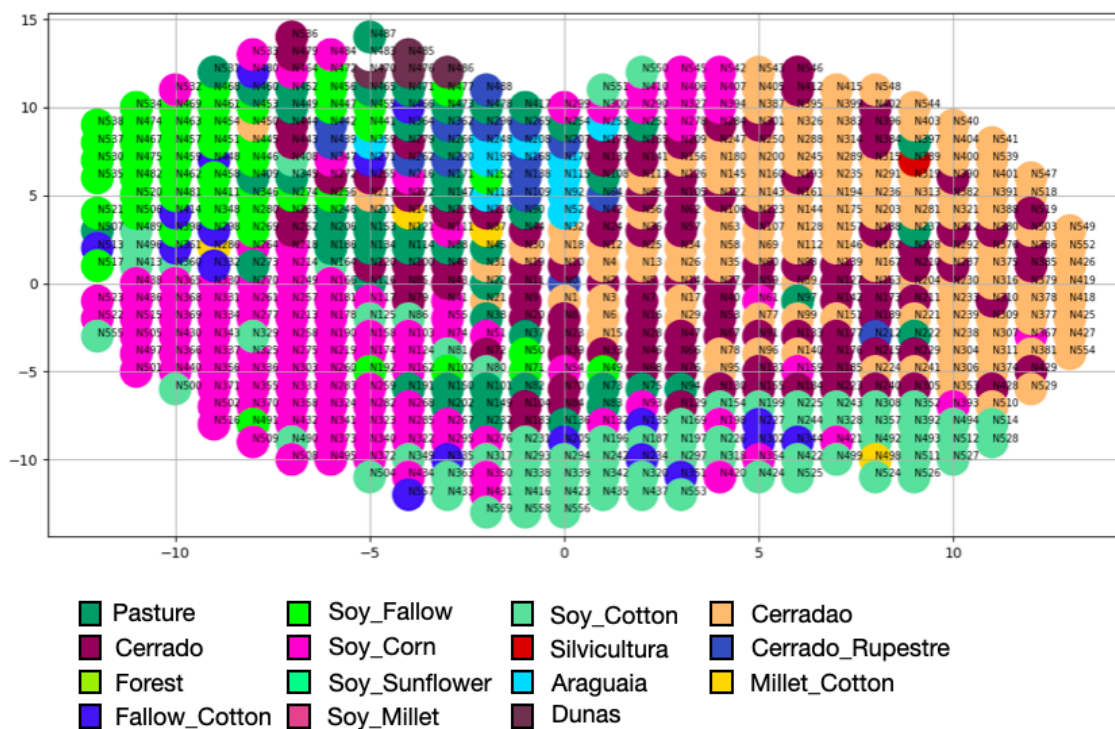
SOURCE: Author

This map finished the growing phase with 559 neurons. It results on a smaller map than the first resultant map from Case Study 1. The map showed on Figure 4.6 classified 2215 samples on 615 neurons, resulting on 3.6 samples per neuron average density. The larger map of this second case study classified 50360 samples using 2543 neurons, resulting in 19,8 samples per neuron average density. The last map showed

36

on Figure 4.8 classified 50360 on 559 neurons, resulting on 90.0 samples per neuron average density.

Table 4.6 last column summarizes the classification results of this last experiment. We can notice that, despite the overall accuracy of 91.3%, Silvicultura samples were classified with poor accuracy (23.2%) on the smaller map.

Table 4.6 - GSOM Accuracy Table - Larger Map x Smaller Map

Cluster	GSOM Larger Map Accuracy	GSOM Smaller Map Accuracy
Cerrado	95.3	92.9
Cerradao	93.9	90.6
Soy-Cotton	98.5	92.8
Fallow-Cotton	91.5	74.6
Silvicultura	74.7	23.2
Soy-Fallow	97.8	94.2
Soy-Corn	98.3	96.2
Pasture	94.91	88.8
Araguaia	91.5	92.8
Cerrado Rupestre	97.5	92.9
Dunas	100.0	100.0
Millet-Cotton	76.5	62.6
Overall Accuracy	95.4	91.3

4.2.3   Neighborhood analysis

Observing the results presented in the neighborhood analysis of the first case study, we concluded that GSOM keeps the neighborhood influence for the nearest neighbors. But looking at the whole map, and comparing it with a similar SOM (as presented in Figure 4.2), we can notice that GSOM overall neighborhood was compromised. Neurons labeled with the same class seem to create more concise groups on SOM than on GSOM. Despite the recommendations from Alahakoon et al. (2000) for keeping the neighborhood influence smaller on GSOM, we ran experiments and compared maps with initial neighborhood influence 0.6 and 1.0.

In order to expand the neighborhood analysis done on Section 4.1.5, we calculated the distance between a single neuron and all the other neurons on the grid. These distances were normalized and expressed in grayscale, with lighter pixels expressing smaller distances and darker pixels expressing larger distances. In Figure 4.9 we

present this distance for 3 different maps. The first one (a) is the SOM map, used as a reference. The second one (b) is a GSOM with initial neighborhood influence configured as 0.6. The last one (c) had initial neighborhood influence configured as 1. In all cases, the distance is relative to Neuron 0, showed in the map as a white dot at [0, 0].

Figure 4.9 - Distance grey map
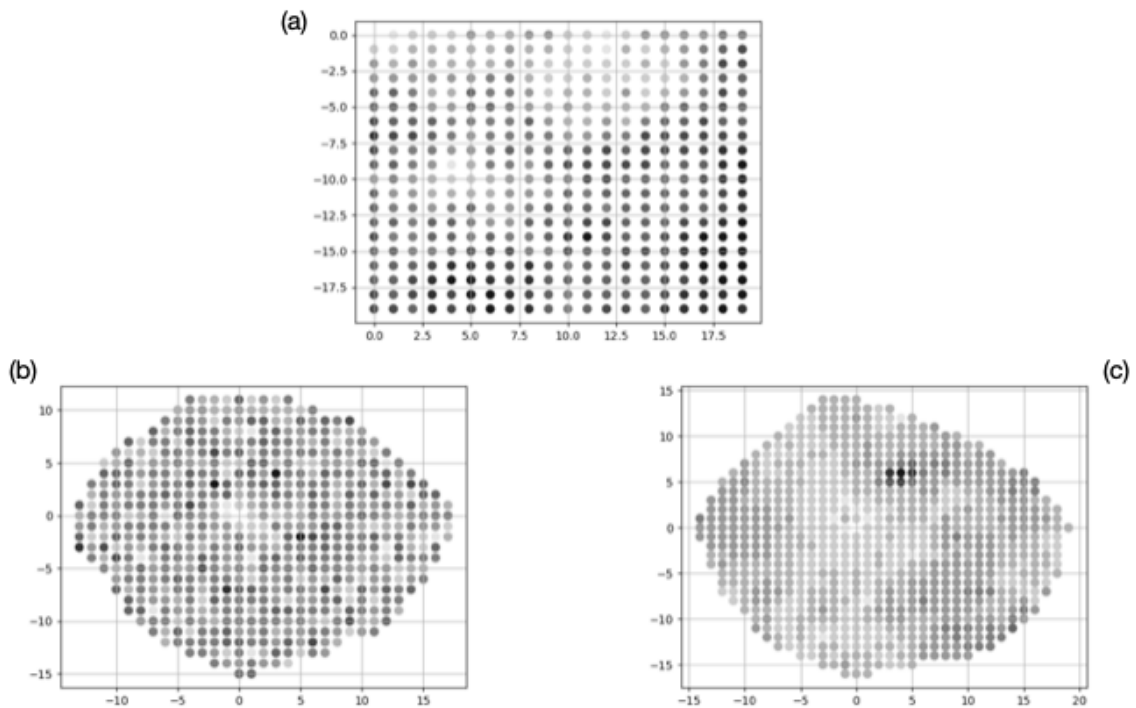
Grey map representing the distance between neuron N0 and all the other Neurons for SOM (a), GSOM with initial neighborhood influence configured as 0.6 (b) and GSOM with initial neighborhood influence configured as 1.0 (c).
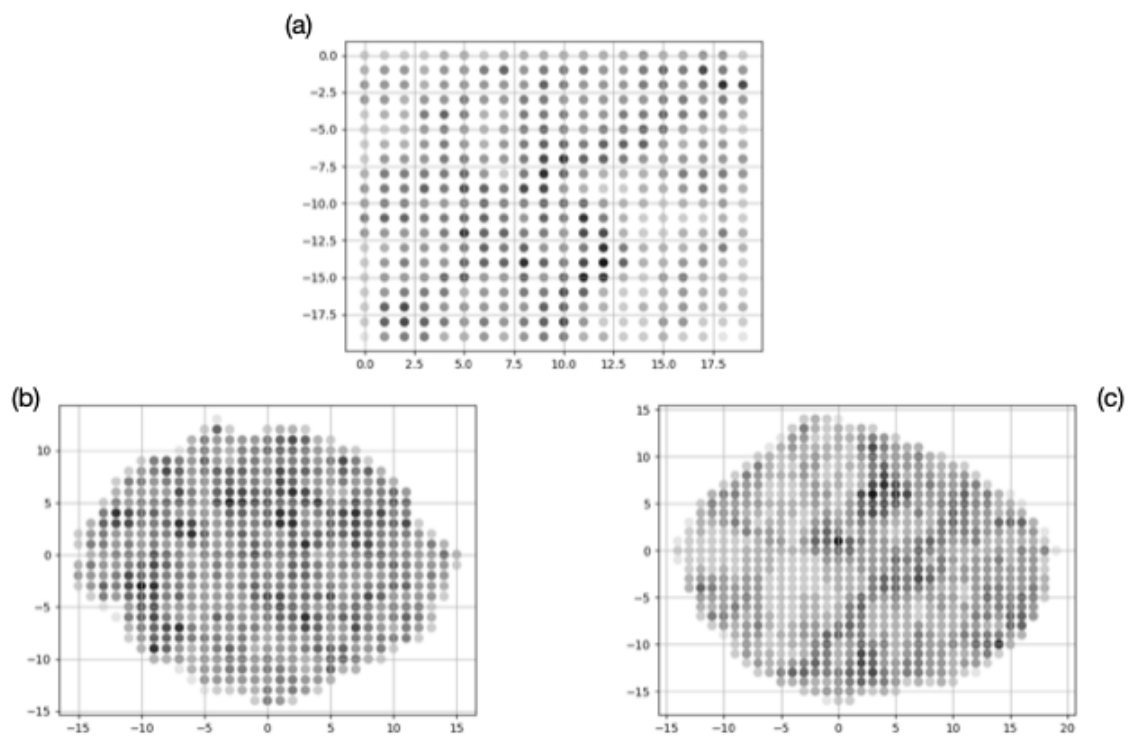
SOURCE: Author

In Figure 4.9 we can observe that a larger initial neighborhood influence generates a more homogeneous map. Higher initial neighborhood influence implies on neuron weights distance increasing with the distance between the neurons. On the several experiments done, varying the reference neuron from Neuron 0 to any other neuron, the results were similar.

Trying to expand this analysis and generate a global metric for the whole map, we calculated the unified distance matrix (U-matrix) of the GSOM map. The U-matrix has become a standard visualization of self-organizing feature maps (LOTSCH; ULTSCH, 2014) and express the distance between the weight vector of a neuron and its nearest neighbors. After calculating this value for all the map neurons, we present this value in a gray scale, where lighter pixels represents smaller distances and darker pixel represents greater distances.

Figure 4.10 shows the U-matrix for a SOM (a), for a GSOM with initial neighborhood influence configured as 0.6 (b) and for a GSOM with initial neighborhood influence 1.0 (c). In general, groups of light pixels are considered clusters, and groups of dark pixels are considered boundaries between these clusters.

Figure 4.10 - U-matrix for SOM and GSOM

U-matrix map representing the distance between a single neuron and its nearest neighbors for SOM (a), GSOM with initial neighborhood influence configured as 0.6 (b) and GSOM with initial neighborhood influence configured as 1.0 (c).

SOURCE: Author

The topology map, including each neuron labeled according the dominant classes is presented in Figure 4.11, for SOM (a), GSOM with initial neighborhood influence 0.6 (b) and 1.0 (c) respectively. We can observe more concise and grouped clusters in GSOM when higher initial neighborhood influence was used. On the other hand, with smaller initial neighborhood influence, GSOM seems to spread the neurons labeled with the same classes, creating smaller and diffused clusters.

Figure 4.11 - Neighborhood Clusters and Boundaries
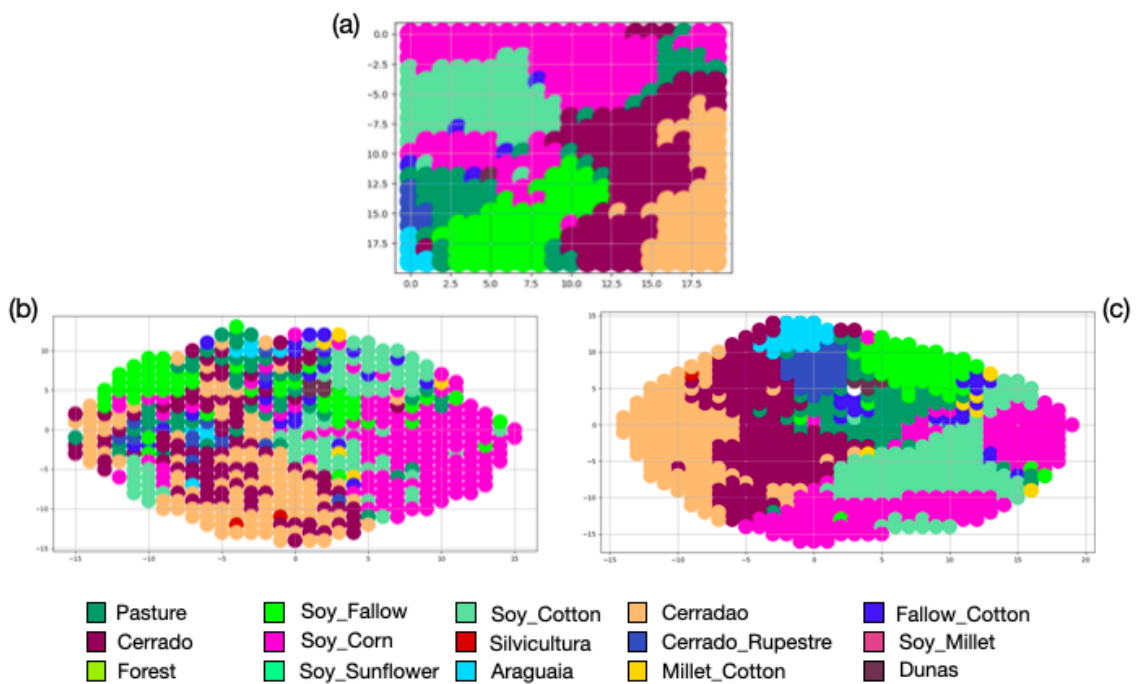
The topology map for SOM (a), GSOM with initial neighborhood influence 0.6 (b) and 1.0 (c). Increasing GSOM initial neighborhood influence parameter results on more concise and grouped clusters.

SOURCE: Author

Despite the difference on the cluster grouping, the accuracy of the map seems not to be affected by the neighborhood influence. Both GSOM executions achieve almost the same classification accuracy, regardless of difference on each single class accuracy. Table 4.7 shows the classification accuracy for each single class and for the whole GSOM maps (with initial neighborhood influence 0.6 and 1.0 respectively).

Table 4.7 - Cluster accuracy comparison between GSOM with different initial neighborhood influences.

Cluster	GSOM 0.6 ini	GSOM 1.0 ini
Cerrado	83.8	87.0
Cerradao	92.6	92.5
Soy-Cotton	94.9	93.2
Fallow-Cotton	87.9	73.6
Silvicultura	79.3	79.6
Soy-Fallow	94.4	89.4
Soy-Corn	95.6	95.9
Pasture	86.5	87.2
Araguaia	78.0	92.2
Cerrado Rupestre	97.4	94.1
Dunas	100.0	100.0
Millet-Cotton	76.0	77.2
Overall Accuracy	89.7	90.4

5 CONCLUSIONS

For the given satellite image time series clustering problem, on the given data set (Mato Grosso state in Brazil and Cerrado biome vegetation ground samples), GSOM seems to be a suitable alternative to SOM. After small customization and adaptation, the GSOM generated map grew as expected, reaching the same amount of neurons of the equivalent SOM, but with a different network topology. GSOM also keeps an important characteristic of SOM, the neurons neighborhood influence.

The cluster accuracy reached by GSOM was similar to the equivalent SOM. It also had a small advantage on the overall clustering time, thanks to its growing phase, when the neuron number is smaller than SOM. We were also able to test the main GSOM characteristic that initiates this work, which was the capability of clustering the data set without specifying the grid size. The Spread Factor usage was a simple and effective task, allowing data analysts without previous algorithm knowledge to cluster the data.

Another important conclusion was the changes provided by the initial neighborhood influence. Despite the recommendation the keep it as lower values, using greater values provided the same cluster accuracy and more concise clusters.

The source code used in this work and the data set files are available on GitHub (GSOM..., 2020). The current implementation is a prototype, that will be evolved in order to become a consolidated product in the future.

Suggested future works include changes on the GSOM algorithm in order to incorporate the geo-targeting information of the samples, and consider it during the clustering. We believe that the spatial location of the sample can influence the clustering and improve the classification accuracy. We also include in the future works the combined use of GSOM with other methods like Bayesian classifiers, in order to improve the classification accuracy.

REFERENCES

ADEU, R. S. S.; FERREIRA, K. R.; ANDRADE, P. R.; SANTOS, L. Evaluating growing self-organizing maps for satellite image time series clustering. In: BRAZILIAN SYMPOSIUM ON GEOINFORMATICS (GEOINFO) 2019. **Proceedings...** São José dos Campos, Brazil, 2019. 4

_____. Assessing satellite image time series clustering using growing som. In: MACHINE LEARNING FOR EARTH OBSERVATION WORKSHOP 2020. **Proceedings...** Cagliari, Italy, 2020. 4

ALAHAKOON, D.; HALGAMUGE, S. K.; SRINIVASAN, B. Dynamic self-organizing maps with controlled growth for knowledge discovery. **IEEE Transactions on Neural Networks**, v. 11, p. 601–614, 2000. 2, 14, 16, 17, 20, 21, 22, 37

DIDAN, K. **MOD13Q1 MODIS/Terra Vegetation Indices 16-Day L3 Global 250m SIN Grid V006**. 2015. Disponível em: <https://doi.org/10.5067/modis/mod13q1.006>. 26, 34

DING, H.; TRAJCEVSKI, G.; SCHEUERMANN, P.; WANG, X.; KEOGH, E. Querying and mining of time series data: experimental comparison of representations and distance measures. **Proceedings of the VLDB Endowment**, p. 1542–1552, 2008. 9

DITTENBACH, M.; MERKL, D.; RAUBER, A. The growing hierarchical self-organizing map. In: CONFERENCE ON NEURAL NETWORKS 2000. **Proceedings...** [S.l.], 2000. p. 15 – 19. 2

_____. Uncovering hierarchical structure in data using the growing hierarchical self-organizing map. **Neurocomputing**, v. 48, p. 199–216, 2002. 10, 11

_____. Organizing and exploring high-dimensional data with the growing hierarchical self-organizing map. **IEEE Transactions on Neural Networks**, p. 1331 – 1341, 2003. 10

EDDELBUETTEL, D.; FRANÇOIS, R. Rcpp: Seamless R and C++ integration. **Journal of Statistical Software**, 2011. 30

FERREIRA, K. R.; SANTOS, L.; PICOLI, M. C. A. Evaluating distance measures for image time series clustering in land use and cover monitoring. **Machine Learning for Earth Observation Workshop**, v. 2466, 2019. 10, 26

FISER, D.; FAIGL, J.; KULICH, M. Growing neural gas efficiently. **Neurocomputing Letters**, v. 104, p. 72–82, 2013. 13, 17

FLEXER, A. On the use of self-organizing maps for clustering and visualization. In: ZYTKOW, J. M.; RAUCH, J. (Ed.). **Principles of Data Mining and Knowledge Discovery**. Berlin: Springer, 1999. p. 80–88. 2

_____. On the use of self-organizing maps for clustering and visualization. **Journal Intelligent Data Analysis**, v. 5, p. 373 – 384, 2001. 10

FOLEY, J. A.; DEFRIES, R.; ASNER, G. P.; BARFORD, C.; BONAN, G.; CARPENTER, S. R.; CHAPIN, F. S.; COE, M. T.; DAILY, G. C.; GIBBS, H. K.; HELKOWSKI, J. H.; HOLLOWAY, T.; HOWARD, E. A.; KUCHARIK, C. J.; MONFREDA, C.; PATZ, J. A.; PRENTICE, I. C.; RAMANKUTTY, N.; SNYDER, P. K. Global consequences of land use. **Science**, v. 309, n. 5734, p. 570–574, 2005. 1

FRANKLIN, S. E. **Remote sensing for sustainable forest management**. [S.l.]: CRC Press, 2001. 5

FRITZKE, B. Growing cell structures: a self-organizing network for unsupervised and supervised learning. **Neural Networks**, v. 7, n. 9, p. 1441–1460, 1993. 2, 17

_____. Growing grid - a self-organizing network with constant neighborhood range and adaptation strength. **Neural Processing Letters**, v. 2, p. 9–13, 1995. 12

_____. A growing neural gas network learns topologies. **Advances in Neural Information Processing Systems**, v. 7, p. 625–632, 1995. 2, 13, 14

GOMEZ, C.; WHITE, J. C.; WULDER, M. A. Optical remotely sensed time series data for land cover classification: a review. **ISPRS Journal of Photogrammetry and Remote Sensing**, v. 116, p. 55–72, 2016. 1, 5

GSOM Python Implementation Repository. 2020. Disponível em: <https://github.com/rodrigosales/GSOM>. Acesso em: May, 20th 2020. 22, 43

GUYET, T.; NICOLAS, H. Long term analysis of time series of satellite images. **Pattern Recognition Letters**, v. 70, p. 17–23, 2015. 6

HUNZIKER, A. **GrowingSOM R Package**. 2018. Disponível em: <https://github.com/alexhunziker/GrowingSOM>. 20

KOHONEN, T. Essentials of the self-organizing map. **Neural Networks**, v. 37, p. 52 – 65, 2013. 17

KOHONEN, T.; KASKI, S.; LAGUS, K.; SALOJARVI, J.; HONKELA, J.; PAATERO, V.; SAARELA, A. Self organization of a massive document collection. **IEEE Transactions on Neural Networks**, p. 574–85, 2000. 29

KOHONEN, T.; SCHROEDER, M. R.; HUANG, T. S. **Self-organizing maps**. New York: Springer-Verlag, 2001. 1, 10

LIAO, T. W. Clustering of time series data - a survey. **Pattern Recognition**, v. 38, p. 1857–1874, 2005. 8

LIU, D.; CAI, S. A spatial-temporal modeling approach to reconstructing land-cover change trajectories from multi-temporal satellite imagery. **Annals of the Association of American Geographers**, v. 102, p. 1329–1347, 2012. 5

LIU, Y.; WEISBERG, R. H.; VIGNUDELLI, S.; MITCHUM, G. T. Patterns of the loop current system and regions of sea surface height variability in the eastern gulf of mexico revealed by the self-organizing maps. **Journal of Geophysical Research: Oceans**, v. 121, n. 4, p. 2347–2366, 2016. 8

LOTSCH, J.; ULTSCH, A. Exploiting the structures of the u-matrix. In: VILLMANN, T.; SCHLEIF, F.-M.; KADEN, M.; LANGE, M. (Ed.). **Advances in self-organizing maps and learning vector quantization**. Berlim: Springer, Cham, 2014. v. 295, p. 249–257. 39

LUDWIG, P. **PyGsom**. 2016. Disponível em: <https://github.com/philippludwig/pygsom>. 19

LUNETTA, R. S.; ELVIDGE, C. D. **Remote sensing change detection**: environmental monitoring methods and applications. [S.l.]: Ann Arbor Press, 1998. 5

MARTINETZ, T.; SCHULTEN, K. A neural gas network learns topologies. **Neural Networks**, p. 397–402, 1991. 13

MAUS, V.; CAMARA, G.; CARTAXO, R.; SANCHEZ, A.; RAMOS, M.; QUEIROZ, G. A time-weighted dynamic time warping method for land-use and land-cover mapping. **IEEE Journal of Selected Topics in Applied Earth Observations and Remote Sensing**, v. 9, n. 8, p. 3729 – 3739, 2016. 6

MENDIS, L. **GSOM**. 2015. Disponível em:
<`https://github.com/AnantaData/GSOM`>. 19

MWASIAGI, J. I. **Self organizing maps - applications and novel algorithm design**. [S.l.]: InTech, 2011. 8

NATITA, W.; WIBOONSAK, W.; DUSADEE, S. Appropriate learning rate and neighborhood function of self-organizing map (som) for specific humidity pattern classification over southern thailand. **International Journal of Modeling and Optimization**, v. 6, p. 61–65, 2016. 8

PARENTE, L.; FERREIRA, L.; FARIA, A.; NOGUEIRA, S.; ARAÚJO, F.; TEIXEIRA, L.; HAGEN, S. Monitoring the brazilian pasturelands: a new mapping approach based on the landsat 8 spectral and temporal domains. **International Journal of Applied Earth Observation and Geoinformation**, p. 135–143, 2017. 1

PASQUARELLA, V. J.; HOLDEN, C. E.; KAUFMAN, L.; WOODCOCK, C. E. From imagery to ecology: leveraging time series of all available landsat observations to map and monitor ecosystem state and dynamics. **Remote Sensing in Ecology and Conservation**, v. 2, 2016. 1

PEARCE, J. L.; WALLER, L. A.; CHANG, H. H.; KLEIN, M.; MULHOLLAND, J. A.; SARNAT, J. A.; SARNAT, S. E.; STRICKLAND, M. J.; TOLBERT, P. E. Using self-organizing maps to develop ambient air quality classifications: a time series example. **Environmental Health: a Global Access Science Source**, v. 13, p. 56, 2014. 8

PETITJEAN, F.; GANÇARSKI, P.; MASSEGLIA, F.; FORESTIER, G. Analysing satellite image time series by means of pattern mining. In: FYFE, C.; TINO, P.; CHARLES, D.; GARCIA-OSORIO, C.; YIN, H. (Ed.). **Intelligent Data Engineering and Automated Learning**. Berlin: Springer, 2010. 6

PICOLI, M.; CAMARA, G.; SANCHES, I.; SIMOES, R.; CARVALHO, A.; MACIEL, A.; COUTINHO, A.; ESQUERDO, J.; ANTUNES, J.; BEGOTTI, R.; ARVOR, D.; ALMEIDA, C. Big earth observation time series analysis for monitoring brazilian agriculture. **ISPRS Journal of Photogrammetry and Remote Sensing**, v. 145, p. 328–339, 2018. 1, 5

PYTHON SOFTWARE FOUNDATION. **Python language reference, version 3.7.2.** 2020. Disponível em: <`http://www.python.org`>. Acesso em: May, 20th 2020. 25

RALHAN, A. **Kohonen-maps**. 2018. Disponível em:
<https://github.com/abhinavralhan/kohonen-maps>. 19

ROGAN, J.; CHEN, D. Remote sensing technology for mapping and monitoring
land-cover and land-use change. **Progress in Planning**, v. 61, p. 301–325, 2004. 5

SABOUR, S. T.; LOHMANN, P.; SOERGEL, U. Monitoring agricultural activities
using asar envisat data. In: INTERNATIONAL ARCHIVES OF
PHOTOGRAMMETRY, 2007. **Proceedings...** [S.l.], 2007. 6

SAMARASINGHE, S. **Neural networks for applied sciences and
engineering**: from fundamentals to complex pattern recognition. [S.l.]: Auerbach
Publications, 2006. 2, 12

SANTOS, L. A.; FERREIRA, K. R.; PICOLI, M.; CAMARA, G. Self-organizing
maps in earth observation data cubes analysis. In: INTERNATIONAL
WORKSHOP ON SELF-ORGANIZING MAPS, 2019. **Proceedings...** [S.l.]:
Springer, Cham, 2019. p. 70–79. 1, 2, 3, 4, 6, 8, 9, 20, 25, 30

SPERA, S.; WINTER, J.; PARTRIDGE, T. Brazilian maize yields negatively
affected by climate after land clearing. **Nature Sustainability**, p. 1–8, 2020. 1

TREITZ, P. Remote sensing for mapping and monitoring land-cover and land-use
change. **Progress in Planning**, v. 61, 2004. 5

VASIGHI, M.; ABBASI, S. Multiple growing self-organizing map for data
classification. In: INTERNATIONAL SYMPOSIUM ON ARTIFICIAL
INTELLIGENCE AND SIGNAL PROCESSING, 2017. **Proceedings...** [S.l.],
2017. 15, 16

WEHRENS, R.; BUYDENS, L. Self and super-organizing maps in r: the kohonen
package. **Journal of Statistical Software**, 2007. 30

WU, Z.; YEN, G. A som projection technique with the growing structure for
visualizing high dimensional data. **International Journal of Neural Systems**,
v. 13, n. 5, p. 353 – 365, 2003. 11

PUBLICAÇÕES TÉCNICO-CIENTÍFICAS EDITADAS PELO INPE

Teses e Dissertações (TDI)

Teses e Dissertações apresentadas nos Cursos de Pós-Graduação do INPE.

Notas Técnico-Científicas (NTC)

Incluem resultados preliminares de pesquisa, descrição de equipamentos, descrição e ou documentação de programas de computador, descrição de sistemas e experimentos, apresentação de testes, dados, atlas, e documentação de projetos de engenharia.

Propostas e Relatórios de Projetos (PRP)

São propostas de projetos técnico-científicos e relatórios de acompanhamento de projetos, atividades e convênios.

Publicações Seriadas

São os seriados técnico-científicos: boletins, periódicos, anuários e anais de eventos (simpósios e congressos). Constam destas publicações o Internacional Standard Serial Number (ISSN), que é um código único e definitivo para identificação de títulos de seriados.

Pré-publicações (PRE)

Todos os artigos publicados em periódicos, anais e como capítulos de livros.

Manuais Técnicos (MAN)

São publicações de caráter técnico que incluem normas, procedimentos, instruções e orientações.

Relatórios de Pesquisa (RPQ)

Reportam resultados ou progressos de pesquisas tanto de natureza técnica quanto científica, cujo nível seja compatível com o de uma publicação em periódico nacional ou internacional.

Publicações Didáticas (PUD)

Incluem apostilas, notas de aula e manuais didáticos.

Programas de Computador (PDC)

São a seqüência de instruções ou códigos, expressos em uma linguagem de programação compilada ou interpretada, a ser executada por um computador para alcançar um determinado objetivo. Aceitam-se tanto programas fonte quanto os executáveis.