A computational study on contexutal self-organizing maps for subset data integration

by

Holly Deann Baiotto

A thesis submitted to the graduate faculty

in partial fulfillment of the requirements for the degree of

MASTER OF SCIENCE

Co-majors: Mechanical Engineering; Human Computer Interaction

Program of Study Committee:
Eliot Winer, Major Professor
Soumik Sarkar
Stephen Gilbert

The student author, whose presentation of the scholarship herein was approved by the program of study committee, is solely responsible for the content of this thesis. The Graduate College will ensure this thesis is globally accessible and will not permit alterations after a degree is conferred.

Iowa State University

Ames, Iowa

2020

TABLE OF CONTENTS

Page

LIST OF FIGURES

Page

NOMENCLATURE

SOM	Self-Organizing Map
CSOM	Contextual Self-Organizing Map
SOMVIZ	Self-Organizing Map Software

ABSTRACT

As sensing and data collection capabilities have dramatically increased in recent years, many areas from medicine to entertainment to engineering have to rethink how products are designed, delivered and maintained. In engineering fields data is everywhere and its use as a decision aid, in the constant stream of tradeoff decisions, is critical to delivering more robust products and services accurately and efficiently. Thus, the need to develop intelligent methods to analyze and visualize large datasets, to enable human understanding, is critical. One method that has been proven effective in this endeavor is the self-organizing map (SOM). However, SOMs require substantial computational resources and time to train, making them impractical for large datasets or datasets that may be added to over time. If this issue could be overcome, this approach could be widely adopted. This thesis studies the concept of using a subset of data to represent the characteristics of a full data set via a SOM. The correlation of a subset and full dataset SOM was studied on two different test cases. The percent difference of node weights was used to compare map representations between the partial and full datasets. A node alignment process was designed and implemented to enable a more accurate comparison of two SOMs.

The methodology was evaluated on two test cases. A hundred comparisons of node weights from subset and full datasets maps were completed per test case. Results showed that pairing node weights by row and column designation did not accurately compare two different SOMs. The alignment process was then performed on ten samples of map comparisons per test case. Results of the aligned nodes provided a much more accurate comparison of SOMs from partial and full datasets.

The results of this study show that with a good representative subset of data very similar nodal weights can be reached through map training compared to using the full dataset. This

allows a trained SOM to be available as a decision aid in a fraction of the training time compared to using the full dataset.

CHAPTER 1. INTRODUCTION

Advancements in the information age have allowed for large amounts of data to be collected from engineering systems. Engineers in just about any area (e.g., research, product development, manufacturing, etc.) are continually expected to use "big data" when making impactful decisions. While the propensity to acquire data has grown exponentially in recent years, so has the complexity with which to analyze it. Research has shown that humans cannot visualize and comprehend the complexities of "big data" without a physical representation of information [1]. To overcome this, methods have been developed for humans to visualize and analyze "big data" to detect hidden patterns and trends. One method, self-organizing maps (SOM), has been developed to categorize similar data within a set and help humans visualize n-dimensional relationships in a simplified manner. Kohonen's SOM is a proven unsupervised method known to the science community [2]. It is typically a 3-layer neural network that preserves the topology structure of the dataset in nodes. The training of the map begins by including all the nodes and then converges to a localized area. A winner takes all approach is applied to the node weight that is most similar to an incoming input data point. The output of a SOM is a clustered or categorized map of similar data.

Previous work by Nekolny explored the visualization of optimization problems using a contextual SOM (CSOM). CSOMs allow an additional parameter, or "label" to be added to the final clustered data to help users interpret additional properties of a dataset [3]. Richardson et al. enhanced Nekolny's work by separating the original CSOM map into three separate maps showcasing statistical differences in each node, which allowed design engineers to study nodal properties for unique data combinations [4]. Kohl et al. further developed this CSOM method by adding a clustering technique to create "meta-clusters" to segment higher-dimensional data and

provide users with an intermediate map representation above individual nodes [5]. SOMs have also been used to visualize a variety of large datasets in areas such as finance, cyber security, and image classification [6–8].

However, SOMs may be an infeasible option in many practical environments where time is limited, because very large datasets require long computation times for training the map. Training time is especially long for datasets that are never complete. These “evolving” datasets are continuously added to over time. For example, a SOM might be trained on 100,000 points of product data. Six months or a year later, 50,000 new points have been created. Traditionally, the SOM would have to be trained on the entire 150,000 point dataset. So, the original dataset is trained, then after a period of time, a subset is added to the original dataset and trained again. This process is applicable for revisions of engineering design or product development over years of innovation, but it is an inefficient process because the original dataset is trained multiple times.

Richardson, Holub, and Winer published a study on CSOM computational time with serial CPU and parallel GPU methodologies [9]. Based on their results for a serial CPU method, 10,000 data points with 10 dimensions took twenty minutes of computation time on state-of-the-art hardware in 2014. A million data points in 50 dimensions took over a week to train. Even with CPU and GPU advancements, significant computational resources and time are needed to train a large dataset. Consider this example, a manufactured product design produces an original dataset of 500,000 points with ten dimensions. It takes anywhere from 1-3 days to compute and train a CSOM. A year later, two product design revisions have produced an additional 250,000 points. The subset is added to the original data and the computation time now takes somewhere in the range of 3-5 days. Another year passes and an additional subset of 250,000 points has been

created. Training a CSOM on this data (i.e. 1M points) will now take 5-7 days to complete. This process repeats and makes the use of CSOMs less and less feasible due to the time required to create a map. Essentially, this process trains the original dataset three times.

This thesis presents a computational study of an essential component of a SOM, the nodal weights during and after training. It was theorized that final nodal weights of a SOM from a partial dataset could be used as the initial weight values for subsequent partial datasets in new SOMs. The resulting maps would be almost identical, albeit with less data, when compared to a map trained on the full dataset (i.e. summation of all partial datasets). If this theory proved true, it would provide substantial evidence that SOMs could be created in sequence from one partial dataset to another rather than continually adding to and training on the full dataset.

Thesis Organization

This remainder of the thesis is organized as follows. Chapter 2 provides background information on the basic of SOMS, applications across a broad range of fields, and methods of comparing SOMs. Chapter 3 is a submitted journal article that details the methodology of comparing node weights for partial and full SOMs, along with aligning maps. Results from two test cases are also presented. Chapter 4 summarizes all the research, presents conclusions, and outlines future work.

CHAPTER 2. BACKGROUND

This section explains the research of SOM applications and the fundamental logic used in the codebase for this computational study.

Basic Self-Organizing Map Method

Kohonen's SOM structure is derived from a multi-layer neural network that projects clustered results onto a two-dimensional map. The size of the SOM is pre-defined by the user at the beginning of the method. SOM methodology uses a neighborhood function that depends on input data being distributed on a map through a winning node calculation as shown in Figure 2.1 [2]. The first layer manages the input of the data for the initialization of the map and the distribution of data points in each training calculation. The middle of the neural network is the computation layer where Euclidean distance, neighborhood influence, and a winning node is calculated and trained. Clustered results are projected onto the final layer map that keeps the topological integrity of the n-dimensional data.

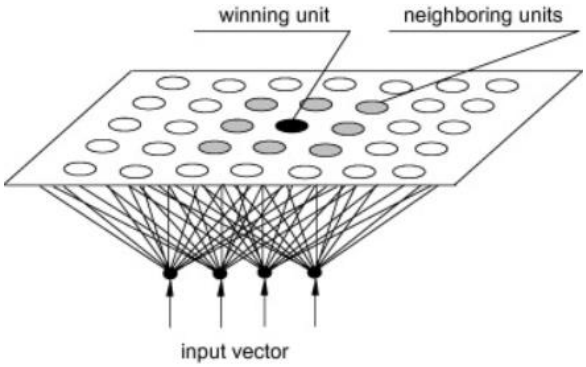

Figure 2.1. SOM Training Methodology [10]

The computation layer is comprised of nodes with an input vector, Eq. (1), and k-dimensional weight vector, Eq. (2). After all the weight vectors are used to fit the data in the map, it is projected to the output layer. Each computation update iterates through a neighborhood of nodes that change depending on the training stage.

$$x =< x_1, x_2, \dots, x_k > \qquad (1)$$

$$w_j =< w_{j1}, w_{j2}, \dots, w_{jk} > \qquad (2)$$

There are two stages in the training phase of the map; ordering, and convergence. The ordering phase trains the coarse topological structure, while the convergence phase fine-tunes the data. In the ordering phase, the neighborhood influence starts with the entire map and then gets smaller after each update iteration. The convergence phase has neighborhood updates based on formulas and time (i.e. iterations).

A single update begins with random data point selection from the dataset. The Euclidean distance, Eq. (3), is calculated for every node in the map and the node with the smallest Euclidean distance is the winner of the data point.

$$i(x) = \min\|x - w_j\|, \; j = 1,2, \dots, l \qquad (3)$$

The data point is distributed to the winning node, and a Gaussian function updates the weight vectors of the nodes in the influence area around the winning node as shown in Eq. (4).

$$w_j(n+1) = \; w_j(n) + \eta(n) * \; h_{j,i(x)}(n) * (x - w_j(n)) \qquad (4)$$

This update equation is used for all points in the dataset, one at a time. The influence range on the neighborhood changes after each update occurs and exponentially gets smaller, as shown in Eq. (5). This allows the map to become localized for fine adjustments. The number of iterations (sometimes referred to as training time) also causes the map to focus on a localized area. The rate of area change is influenced by a training time-varying neighborhood by using Eq. (6).

$$h_{j,i(x)}(n) = exp\left(-\frac{d_{j,i}^2}{2\sigma^2(n)}\right), \; n = 0,1,2, \dots \qquad (5)$$

$$\sigma(n) = \sigma_0 exp\left(-\frac{-n}{\tau_1}\right), n = 0,1,2, \dots \qquad (6)$$

$$\eta(n) = \eta_0 exp\left(-\frac{-n}{\tau_1}\right), n = 0,1,2, \dots \qquad (7)$$

The learning rate of a map exponentially decreases by training time to avoid overtraining by using Eq. (7). The variable η_o gradually decreases as training time continues. During the convergence phase, η_o stays constant to allow localized updates to occur. When the SOM training is complete, a clustered map is displayed on the output layer.

A Contextual Self Organizing Map (CSOM) adds one more step to a SOM by adding contextual labels on the output layer [3]. The Euclidean distance and winning node are again calculated for each data point. An update puts a label on the winning node but does not update any surrounding nodes. This label, a number or text, is not used in the training of the SOM, but is there to describe the contents of each nodal cluster. Figure 2.2 shows an example of an animal contextual map, where the clusters are mammal predators, mammal prey, and birds. In this example, the label is the name of the species. Characteristics such as number of legs, presence of feathers or fur, and type of diet (i.e. omnivore, herbivore, etc.) were used to train the CSOM.

dog	dog	fox	fox	fox	cat	cat	cat	eagle	eagle
dog	dog	fox	fox	fox	cat	cat	cat	eagle	eagle
wolf	wolf	wolf	fox	cat	tiger	tiger	tiger	owl	owl
wolf	wolf	lion	lion	lion	tiger	tiger	tiger	hawk	hawk
wolf	wolf	lion	lion	lion	tiger	tiger	tiger	hawk	hawk
wolf	wolf	lion	lion	lion	owl	dove	hawk	dove	dove
horse	horse	lion	lion	lion	dove	hen	hen	dove	dove
horse	horse	zebra	cow	cow	cow	hen	hen	dove	dove
zebra	zebra	zebra	cow	cow	cow	hen	hen	duck	goose
zebra	zebra	zebra	cow	cow	cow	duck	duck	duck	goose

Figure 2.2. Contextual SOM of clustered animals [11]

The CSOM codebase called SOMViz, has been used in previous research and has additional features. Research features not used in this thesis are meta-clustering, parallelization with GPU computing, and a novel convergence heuristic [5,9,12]. A trained map is displayed with a hexagonal u-matrix [13], and three different statistical graphics are shown in the interface [4] as shown in Figure 2.3.

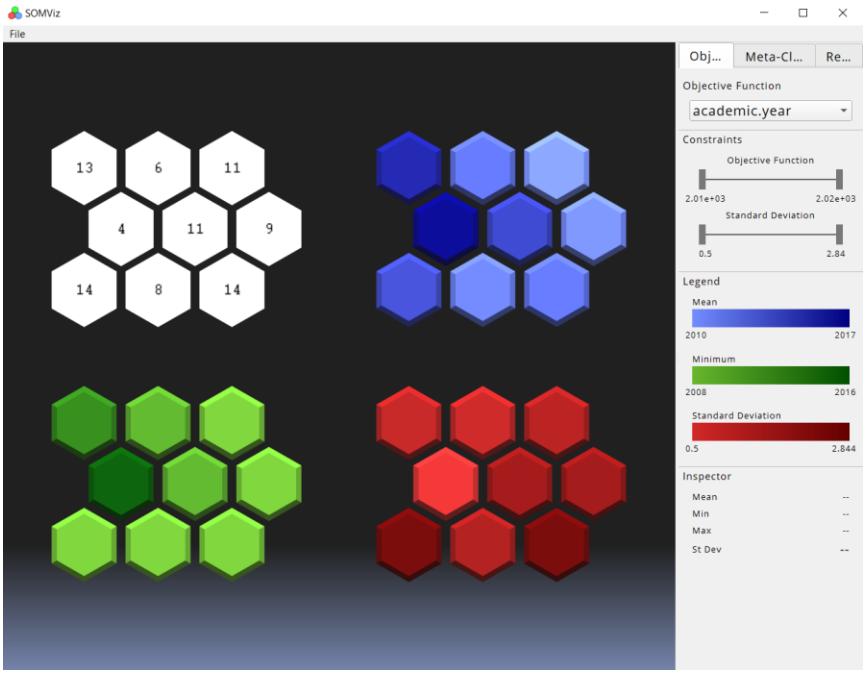

Figure 2.3. SOMViz software user interface developed by Richardson et al.

A feature of SOMViz is the colored or labeled nodes based on the contextual calculation with Euclidean distance in respect to other nodes. The colored legend shows the upper and lower bounds of three statistical properties (i.e. mean, minimum, and standard deviation) each corresponding to a specific map in the interface. SOMViz helps a user visualize underlying trends that may be occurring within, or between, clusters. It is a visual variation of a SOM that allows another value (i.e. the contextual label) to more fully describe the dataset and inherent relationships within.

SOM Research

Kohonen's SOM Foundation

SOM has been an established research method since it was developed by Tuevo Kohonen in 1982 [2]. The idea was inspired by the cerebral cortex and how it efficiently processes different types of sensory signals in separate locations. Unexplored data is clustered into similar categories through a competitive unsupervised neural network algorithm as described in the

previous section. In 1997 Kohonen used a large dataset to further understand the strengths and weaknesses of SOM methodology [14]. The software called WEBSOM was developed to categorize a million documents based on similar verbiage in each document. The purpose was to create an organized database where new documents could be categorized with similar documents, essentially performing like modern day internet search engines. Kohonen's logic was to compare similar data and add new data to the map through an update of vectors. This logic is a similar premise for adding partial datasets onto a completed map. Kohonen's computation time was eight to nine weeks for 31 million datapoints. While modern technology vastly reduces this time, a comparable computer in today's world would still take many days to perform the computations for these many points. Kohonen concluded computation for the winner search and map updates in large maps are time-consuming tasks. He proposed improving the initial map weights and using vector pointers of the winning node location as time-saving solutions. These fundamental ideas and technological advancements are still researched in the scientific community today. Many publications have cited Kohonen's work to showcase the strength of the methodology and researched improvements. Ahmed et al. explored optimizing SOM through a genetic algorithm and Akinduko et al. investigated methods for improving the initialization of the map [15,16]. Many variations of SOM have been developed with other machine learning methods to expand the variety of applications. Wickramasinghe et al. developed a parallelizable SOM on top of a supervised machine learning method for image classification [17]. Abaei et al. developed a two stage semi-supervised hybrid SOM to detect software fault prediction [18]. These projects are further evidence that research in SOMs is very active, but none of the publications have definitively shown how using, or reusing, nodal weights can benefit SOM training.

Applications of SOMs in Engineering Design

The SOM is a popular approach that has been used in a variety of engineering applications. Utility and environmental engineering applications such as water distribution systems, performance of power utilities, and landslide susceptibility are examples where SOMs have made an impact. Mustonen et al. studied deterioration in water quality due to disturbances in water distribution systems [19]. They collected data from an artificial system with pressure shock waves that stirred up deposits and biofilms in the water. Mustonen et al. analyzed the data with SOMs and were able to detect abrupt changes in water quality due to certain conditions. The authors attempted to develop an online monitoring and alert system for water quality based on patterns discovered using SOMs. Leder et al. utilized a SOM methodology to propose an increase of performance for power utilities [20]. Power companies want to maximize the capability of the electrical grid, but also need to do so in safe operating conditions. Operators in control centers manage the electrical systems and can only respond to a single issue at a time. Using SOMs allowed for the exploration of optimized conditions while providing a quick analysis for operators. Theoretical results showed promising opportunity for SOMs to be used as a tool on real power systems. Huang et al. evaluated landslide susceptibility in the Three Gorges Reservoir Area in China [21]. The authors used a two-stage system called a SOM network based extreme learning machine (ELM) to calculate a landslide susceptibility index and compared these results to a traditional prediction method. They concluded the SOM-ELM model had a higher success and prediction rate with efficiency over the traditional method.

Manufacturing applications of SOMs include Khanzadeh et al.'s work on clustering complex geometries to increase the accuracy of additive manufacturing [22]. Ruping et al used SOMs to aid in the fabrication process of integrated circuits in semi-conductors [23]. Germen et al. studied sound data from induction motors to predict faulty conditions [24]. Induction motors

are everywhere in manufacturing and early diagnosis of motor defects is key to avoid production down time. Acoustic data with different kinds of faults were classified in a SOM. Clustering results allowed the authors to determine the difference between a healthy and faulty motor. Additionally, the type of faulty condition was distinguishable from the analysis, which allowed for scheduled maintenance instead of interrupted production. Li et al. performed a similar study for fault diagnosis of bearings; another essential component in manufacturing [25]. The authors used SOMs to determine a condition monitor index based on load levels and motor speed. The clustering results were then used in a Bayesian model to accurately predict faults. A simulation and industrial case study validated the approach over conventional methods. While SOMs have been used by many, two specific projects are discussed below to demonstrate the strengths and weaknesses of this class of methods. They are non-destructive material testing and exploration of design solutions for supersonic wing design.

Pérez-Benítez et al. used SOM clustering to understand microstructure signal effects on Magnetic Barkhausen Noise non-destructive testing [26]. This micro-scale testing technique can be influenced by multiple source signals like carbon content, plastic deformation, or residual stresses in materials. It is hard to identify the exact sources when all the signals are intertwined, so the researchers used SOMs to sample a variety of signals to see if a pattern with non-destructive analysis existed. The results identified the simultaneous influence on two microstructural sources within the method. Since the analysis was successful with a SOM, all the microstructural structures superimposed on each would be beneficial to understand the influence, but that would be a very large dataset and take a lot of computation time, so the researchers did not attempt it.

A second notable project completed by Obayashi and Sasaki was to apply SOMs to 766 possible solutions of supersonic wing designs to identify the solutions that best met requirements and user preferences [27]. The researchers performed two SOM iterations: one for looking at all 766 designs and a second simplified clustering based on subgroups to narrow down the solutions. They claim SOMs are a versatile way to explore tradeoffs in aeronautical engineering. However, this research also demonstrated the fundamental shortcoming of using SOMs, the time and resources needed to train a map to aid in decision making. If there were 776 solutions for a wing design, how many solutions for an entire aircraft design with all of its subsystems? The answer is potentially thousands to millions. While a SOM has the capability to explore the data for an entire aircraft design, the computational time it would take to produce a map often makes using the method impractical. If a large analysis could be done in multiple SOM iterations or using a subset of the full dataset, the applicability and practicality of using SOMs in everyday engineering tasks could increase dramatically.

Comparing Self-Organizing Maps

To explore the feasibility of using partial datasets to represent a full dataset, multiple SOMs will have to be compared. Thus, it was important to understand what research was already done in this area. Mayer et al. explored the methodology of comparing two or more SOM maps trained by the same data [28]. They analyzed data, cluster, and multi-faceted shifts for two datasets: iris and a synthetic datamining dataset. Data shifts were defined as different orientations of data on the same 2-dimensional map structure. This is common in final SOM representations as they are dimensionless. As a SOM is based on a neural network topology, the number of inputs can scale directly with the number of dimensions in a dataset. The resultant map displays clusters of data through the training process as described earlier. However, the final 2D map is dimensionless. There are no x and y descriptors as is common in visualization techniques that

employ some form of dimensional reduction. Instead, the horizontal and vertical direction on a trained SOM only denote the row and column number for a particular node as shown in Figure 2.3. Mayer et al matched data points between two similarly sized SOMs by calculating data point vectors. Traditionally, different size datasets require different sizes of SOMs and finding matching clusters of data in two different map sizes was difficult. The researchers matched data points and calculated the vectors between multiple maps. A challenge with different size SOMs is clusters tend to be spread out on more nodes compared to consolidated clusters in smaller maps. Meyer et al. addressed this by calculating a confidence interval for each matching cluster to indicate the strength of the map correlations. They verified their methodology on a combined case study of multi-SOM, where the data was in a different orientation dispersed on different map sizes. Results showed they were able to match data points and clusters through a multi-SOM analysis, but the methodology does not verify if there are similar node weights amongst the maps to be able to train from each other. In addition, the alignment process was complex and validating accuracy was difficult.

Aligning SOM maps was also researched by Pampalk for music applications [29]. The goal was to categorize perceptual similarities for music, images, or videos based on musical components, such as tempo, rhythm, and volume, through an interactive user experience. The author recognized SOMs as a powerful tool to visualize music similarity, but wanted a user to gradually experience individual dimensions without losing orientation from different SOMs. Pampalk divided all the dimensions of the dataset into projections on multiple layers (tempo on one layer, rhythm on another, etc.). The SOM algorithm was performed, but each dimension of a data point was distributed on a different layer and the overall summation of all the layers was calculated by weights. This methodology is an interesting take on multiple SOM maps with

additional weights to manage, but it showed clusters of data aligned on different layer projections. The methodology does not explore node weights throughout the clusters and multiple layers of dimensions.

There are not many publications on SOM methodologies using subsets of a dataset or reuse of nodal weights to train another SOM. A methodology that trains multiple sets of smaller data and aligns similar clusters, without having to retrain previous data would save immense amounts of computation and allow maps to be trained much faster. This would enable SOMs to be used more readily for the many tradeoff decisions in product and process design.

CHAPTER 3. CONTEXTUAL SELF-ORGANIZING MAPS FOR SUBSET DATA INTEGRATION

Holly Baiotto,[1] Adam Kohl,[2] and Eliot Winer[3]
Iowa State University, Ames, IA, 50010, USA

A manuscript submitted to the AIAA Journal of Aerospace Information Systems

Abstract

As sensing and data collection capabilities have dramatically increased in recent years, many areas from medicine to entertainment to engineering have to rethink how products are designed, delivered and maintained. In engineering fields data is everywhere and its use as a decision aid, in the constant stream of tradeoff decisions, is critical to delivering more robust products and services accurately and efficiently. Thus, the need to develop intelligent methods to analyze and visualize large datasets, to enable human understanding, is critical. One method that has been proven effective in this endeavor is the self-organizing map (SOM). However, SOMs require substantial computational resources and time to train, making them impractical for large datasets or datasets that may be added to over time. If this issue could be overcome, this approach could be widely adopted. This paper studies the concept of using a subset of data to represent the characteristics of a full data set via a SOM. The correlation of a subset and full dataset SOM was studied on two different test cases. The percent difference of node weights was used to compare map representations between the partial and full datasets. A node alignment process was designed and implemented to enable a more accurate comparison of two SOMs. The results of this study show that with a good representative subset of data very similar nodal weights can be reached through map training compared to using the full dataset. This allows a trained SOM to be available as a decision aid in a fraction of the training time compared to using the full dataset.

---

[1] Graduate Assistant, Virtual Reality Application Center.
[2] Graduate Assistant, Virtual Reality Application Center, and AIAA Member Grade (if any) for third author.
[3] Director, Virtual Reality Application Center and AIAA Associate Fellow.

Introduction

ADVANCEMENTS in the information age have allowed for large amounts of data to be collected from engineering systems. Engineers in just about any area (e.g., research, product development, manufacturing, etc.) are continually expected to use "big data" when making impactful decisions. While the propensity to acquire data has grown exponentially in recent years, so has the complexity with which to analyze it. Research has shown that humans cannot visualize and comprehend the complexities of "big data" without a physical representation of information [1]. To overcome this, methods have been developed for humans to visualize and analyze "big data" to detect hidden patterns and trends. One method, self-organizing maps (SOM), has been developed to categorize similar data within a set and help humans visualize n-dimensional relationships in a simplified manner. Kohonen's SOM is a proven unsupervised method known to the science community [2]. It is typically a 3-layer neural network that preserves the topology structure of the dataset in nodes. The training of the map begins by including all the nodes and then converges to a localized area. A winner takes all approach is applied to the node weight that is most similar to an incoming input data point. The output of a SOM is a clustered or categorized map of similar data.

Previous work by Nekolny explored the visualization of optimization problems using a contextual SOM (CSOM). CSOMs allow an additional parameter, or "label" to be added to the final clustered data to help users interpret additional properties of a dataset [3]. Richardson et al. enhanced Nekolny's work by separating the original CSOM map into three separate maps showcasing statistical differences in each node, which allowed design engineers to study nodal properties for unique data combinations [4]. Kohl et al. further developed this CSOM method by adding a clustering technique to create "meta-clusters" to segment higher-dimensional data and provide users with an intermediate map representation above individual nodes [5]. SOMs have also been used to visualize a variety of large datasets in areas such as finance, cyber security, and image classification [7,8,31].

However, SOMs may be an infeasible option in many practical environments where time is limited, because very large datasets require long computation times for training the map. Training time is especially long for datasets that are never complete. These "evolving" datasets are continuously added to over time. For example, a SOM might be trained on 100,000 points of product data. Six months or a year later, 50,000 new points have been created. Traditionally, the SOM would have to be trained on the entire 150,000 point dataset. So, the original dataset is trained, then after a period of time, a subset is added to the original dataset and trained again. This process is applicable for

revisions of engineering design or product development over years of innovation, but it is an inefficient process because the original dataset is trained multiple times.

Richardson, Holub, and Winer published a study on CSOM computational time with serial CPU and parallel GPU methodologies [9]. Based on their results for a serial CPU method, 10,000 data points with 10 dimensions took twenty minutes of computation time on state-of-the-art hardware in 2014. A million data points in 50 dimensions took over a week to train. Even with CPU and GPU advancements, significant computational resources and time are needed to train a large dataset. Consider this example, a manufactured product design produces an original dataset of 500,000 points with ten dimensions. It takes anywhere from 1-3 days to compute and train a CSOM. A year later, two product design revisions have produced an additional 250,000 points. The subset is added to the original data and the computation time now takes somewhere in the range of 3-5 days. Another year passes and an additional subset of 250,000 points has been created. Training a CSOM on this data (i.e. 1M points) will now take 5-7 days to complete. This process repeats and makes the use of CSOMs less and less feasible due to the time required to create a map. Essentially, this process trains the original dataset three times.

This paper presents a computational study of an essential component of a SOM, the nodal weights during and after training. It was theorized that final nodal weights of a SOM from a partial dataset could be used as the initial weight values for subsequent partial datasets in new SOMs. The resulting maps would be almost identical, albeit with less data, when compared to a map trained on the full dataset (i.e. summation of all partial datasets). If this theory proved true, it would provide substantial evidence that SOMs could be created in sequence from one partial dataset to another rather than continually adding to and training on the full dataset.

Background

This section explains the research of SOM applications and the fundamental logic used in the codebase for this computational study.

Basic Self-Organizing Map Method

Kohonen's SOM structure is derived from a multi-layer neural network that projects clustered results onto a two-dimensional map. The size of the SOM is pre-defined by the user at the beginning of the method. SOM methodology uses a neighborhood function that depends on input data being distributed on a map through a winning node calculation

as shown in Figure 3.1 [2]. The first layer manages the input of the data for the initialization of the map and the distribution of data points in each training calculation. The middle of the neural network is the computation layer where Euclidean distance, neighborhood influence, and a winning node is calculated and trained. Clustered results are projected onto the final layer map that keeps the topological integrity of the n-dimensional data.

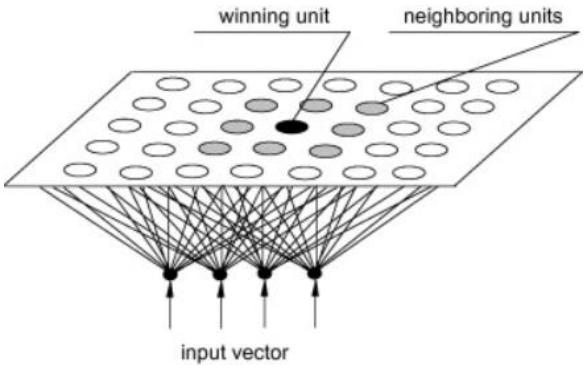

Figure 3.1 SOM training methodology [10]

The computation layer is comprised of nodes with an input vector, Eq. (1), and k-dimensional weight vector, Eq. (2). After all the weight vectors are used to fit the data in the map, it is projected to the output layer. Each computation update iterates through a neighborhood of nodes that change depending on the training stage.

$$x =< x_1, x_2, \dots, x_k > \qquad (1)$$

$$w_j =< w_{j1}, w_{j2}, \dots, w_{jk} > \qquad (2)$$

There are two stages in the training phase of the map; ordering, and convergence. The ordering phase trains the coarse topological structure, while the convergence phase fine-tunes the data. In the ordering phase, the neighborhood influence starts with the entire map and then gets smaller after each update iteration. The convergence phase has neighborhood updates based on formulas and time (i.e. iterations).

A single update begins with random data point selection from the dataset. The Euclidean distance, Eq. (3), is calculated for every node in the map and the node with the smallest Euclidean distance is the winner of the data point.

$$i(x) = \min\left\|x - w_j\right\|, \; j = 1,2, \dots, l \qquad (3)$$

The data point is distributed to the winning node, and a Gaussian function updates the weight vectors of the nodes in the influence area around the winning node as shown in Eq. (4).

$$w_j(n+1) = \; w_j(n) + \eta(n) * \; h_{j,i(x)}(n) * (x - w_j(n)) \qquad (4)$$

This update equation is used for all points in the dataset, one at a time. The influence range on the neighborhood changes after each update occurs and exponentially gets smaller, as shown in Eq. (5). This allows the map to become localized for fine adjustments. The number of iterations (sometimes referred to as training time) also causes the map to focus on a localized area. The rate of area change is influenced by a training time-varying neighborhood by using Eq. (6).

$$h_{j,i(x)}(n) = exp\left(-\frac{d_{j,i}^2}{2\sigma^2(n)}\right),\ n = 0,1,2,\ldots \quad (5)$$

$$\sigma(n) = \sigma_0 exp\left(-\frac{-n}{\tau_1}\right), n = 0,1,2,\ldots \quad (6)$$

$$\eta(n) = \eta_0 exp\left(-\frac{-n}{\tau_1}\right), n = 0,1,2,\ldots \quad (7)$$

The learning rate of a map exponentially decreases by training time to avoid overtraining by using Eq. (7). The variable η_o gradually decreases as training time continues. During the convergence phase, η_o stays constant to allow localized updates to occur. When the SOM training is complete, a clustered map is displayed on the output layer.

A Contextual Self Organizing Map (CSOM) adds one more step to a SOM by adding contextual labels on the output layer [3]. The Euclidean distance and winning node are again calculated for each data point. An update puts a label on the winning node but does not update any surrounding nodes. This label, a number or text, is not used in the training of the SOM, but is there to describe the contents of each nodal cluster. Figure 3.2 shows an example of an animal contextual map, where the clusters are mammal predators, mammal prey, and birds. In this example, the label is the name of the species. Characteristics such as number of legs, presence of feathers or fur, and type of diet (i.e. omnivore, herbivore, etc.) were used to train the CSOM.

dog	dog	fox	fox	fox	cat	cat	cat	eagle	eagle
dog	dog	fox	fox	fox	cat	cat	cat	eagle	eagle
wolf	wolf	wolf	fox	cat	tiger	tiger	tiger	owl	owl
wolf	wolf	lion	lion	lion	tiger	tiger	tiger	hawk	hawk
wolf	wolf	lion	lion	lion	tiger	tiger	tiger	hawk	hawk
wolf	wolf	lion	lion	lion	owl	dove	hawk	dove	dove
horse	horse	lion	lion	lion	dove	hen	hen	dove	dove
horse	horse	zebra	cow	cow	cow	hen	hen	dove	dove
zebra	zebra	zebra	cow	cow	cow	hen	hen	duck	goose
zebra	zebra	zebra	cow	cow	cow	duck	duck	duck	goose

Figure 3.2 Contextual SOM of clustered animals, mammal predator an prey and birds [11]

The CSOM codebase called SOMViz, has been used in previous research and has additional features. Research features not used in this paper are meta-clustering, parallelization with GPU computing, and a novel convergence heuristic[5,9,12]. A trained map is displayed with a hexagonal u-matrix [13], and three different statistical graphics are shown in the interface [4] as shown in Figure 3.3.

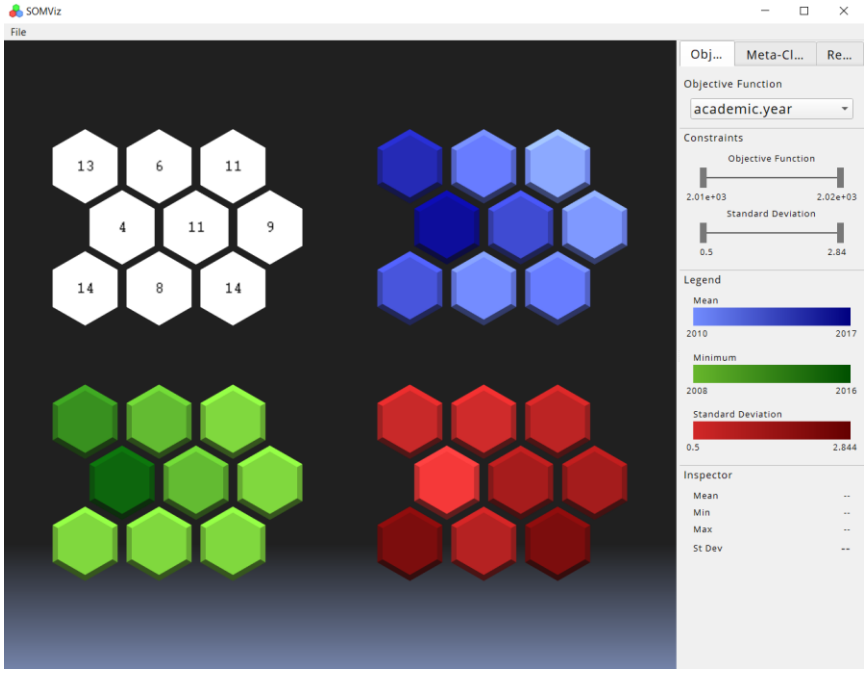

Figure 3.3 SOMViz software user interface developed by Richardson et al.

A feature of SOMViz is the colored or labeled nodes based on the contextual calculation with Euclidean distance in respect to other nodes. The colored legend shows the upper and lower bounds of three statistical properties (i.e. mean, minimum, and standard deviation) each corresponding to a specific map in the interface. SOMViz helps a user visualize underlying trends that may be occurring within, or between, clusters. It is a visual variation of a SOM that allows another value (i.e. the contextual label) to more fully describe the dataset and inherent relationships within.

SOM Research

Kohonen's SOM Foundation

SOM has been an established research method since it was developed by Tuevo Kohonen in 1982 [2]. The idea was inspired by the cerebral cortex and how it efficiently processes different types of sensory signals in separate

locations. Unexplored data is clustered into similar categories through a competitive unsupervised neural network algorithm as described in the previous section. In 1997 Kohonen used a large dataset to further understand the strengths and weaknesses of SOM methodology [14]. The software called WEBSOM was developed to categorize a million documents based on similar verbiage in each document. The purpose was to create an organized database where new documents could be categorized with similar documents, essentially performing like modern day internet search engines. Kohonen's logic was to compare similar data and add new data to the map through an update of vectors. This logic is a similar premise for adding partial datasets onto a completed map. Kohonen's computation time was eight to nine weeks for 31 million datapoints. While modern technology vastly reduces this time, a comparable computer in today's world would still take many days to perform the computations for these many points. Kohonen concluded computation for the winner search and map updates in large maps are time-consuming tasks. He proposed improving the initial map weights and using vector pointers of the winning node location as time-saving solutions. These fundamental ideas and technological advancements are still researched in the scientific community today. Many publications have cited Kohonen's work to showcase the strength of the methodology and researched improvements. Ahmed et al. explored optimizing SOM through a genetic algorithm and Akinduko et al. investigated methods for improving the initialization of the map [15,16]. Many variations of SOM have been developed with other machine learning methods to expand the variety of applications. Wickramasinghe et al. developed a parallelizable SOM on top of a supervised machine learning method for image classification [17]. Abaei et al. developed a two stage semi-supervised hybrid SOM to detect software fault prediction [18]. These projects are further evidence that research in SOMs is very active, but none of the publications have definitively shown how using, or reusing, nodal weights can benefit SOM training.

Applications of SOMs in Engineering Design

The SOM is a popular approach that has been used in a variety of engineering applications. Utility and environmental engineering applications such as water distribution and resource systems, landslide susceptibility, and management of river basins [19-22]. Applications of manufacturing engineering include additive manufacturing, integrated circuit fabrication, faulty detection of induction motors and bearings [23-26]. While SOMs have been used by many, two specific projects are discussed below to demonstrate the strengths and weaknesses of this class of methods. They are non-destructive material testing and exploration of design solutions for supersonic wing design.

Pérez-Benítez et al. used SOM clustering to understand microstructure signal effects on Magnetic Barkhausen Noise non-destructive testing [27]. This micro-scale testing technique can be influenced by multiple source signals like carbon content, plastic deformation, or residual stresses in materials. It is hard to identify the exact sources when all the signals are intertwined, so the researchers used SOMs to sample a variety of signals to see if a pattern with non-destructive analysis existed. The results identified the simultaneous influence on two microstructural sources within the method. Since the analysis was successful with a SOM, all the microstructural structures superimposed on each would be beneficial to understand the influence, but that would be a very large dataset and take a lot of computation time, so the researchers did not attempt it.

A second notable project completed by Obayashi and Sasaki was to apply SOMs to 766 possible solutions of supersonic wing designs to identify the solutions that best met requirements and user preferences [28]. The researchers performed two SOM iterations: one for looking at all 766 designs and a second simplified clustering based on subgroups to narrow down the solutions. They claim SOMs are a versatile way to explore tradeoffs in aeronautical engineering. However, this research also demonstrated the fundamental shortcoming of using SOMs, the time and resources needed to train a map to aid in decision making. If there were 776 solutions for a wing design, how many solutions for an entire aircraft design with all of its subsystems? The answer is potentially thousands to millions. While a SOM has the capability to explore the data for an entire aircraft design, the computational time it would take to produce a map often makes using the method impractical. If a large analysis could be done in multiple SOM iterations or using a subset of the full dataset, the applicability and practicality of using SOMs in everyday engineering tasks could increase dramatically.

Comparing Self-Organizing Maps

To explore the feasibility of using partial datasets to represent a full dataset, multiple SOMs will have to be compared. Thus, it was important to understand what research was already done in this area. Mayer et al. explored the methodology of comparing two or more SOM maps trained by the same data [29]. They analyzed data, cluster, and multi-faceted shifts for two datasets: iris and a synthetic datamining dataset. Data shifts were defined as different orientations of data on the same 2-dimensional map structure. This is common in final SOM representations as they are dimensionless. As a SOM is based on a neural network topology, the number of inputs can scale directly with the number of dimensions in a dataset. The resultant map displays clusters of data through the training process as described

earlier. However, the final 2D map is dimensionless. There are no x and y descriptors as is common in visualization techniques that employ some form of dimensional reduction. Instead, the horizontal and vertical direction on a trained SOM only denote the row and column number for a particular node as shown in Figure 3. 3.3. Mayer et al matched data points between two similarly sized SOMs by calculating data point vectors. Traditionally, different size datasets require different sizes of SOMs and finding matching clusters of data in two different map sizes was difficult. The researchers matched data points and calculated the vectors between multiple maps. A challenge with different size SOMs is clusters tend to be spread out on more nodes compared to consolidated clusters in smaller maps. Meyer et al. addressed this by calculating a confidence interval for each matching cluster to indicate the strength of the map correlations. They verified their methodology on a combined case study of multi-SOM, where the data was in a different orientation dispersed on different map sizes. Results showed they were able to match data points and clusters through a multi-SOM analysis, but the methodology does not verify if there are similar node weights amongst the maps to be able to train from each other. In addition, the alignment process was complex and validating accuracy was difficult.

Aligning SOM maps was also researched by Pampalk for music applications [30]. The goal was to categorize perceptual similarities for music, images, or videos based on musical components, such as tempo, rhythm, and volume, through an interactive user experience. The author recognized SOMs as a powerful tool to visualize music similarity, but wanted a user to gradually experience individual dimensions without losing orientation from different SOMs. Pampalk divided all the dimensions of the dataset into projections on multiple layers (tempo on one layer, rhythm on another, etc.). The SOM algorithm was performed, but each dimension of a data point was distributed on a different layer and the overall summation of all the layers was calculated by weights. This methodology is an interesting take on multiple SOM maps with additional weights to manage, but it showed clusters of data aligned on different layer projections. The methodology does not explore node weights throughout the clusters and multiple layers of dimensions.

There are not many publications on SOM methodologies using subsets of a dataset or reuse of nodal weights to train another SOM. A methodology that trains multiple sets of smaller data and aligns similar clusters, without having to retrain previous data would save immense amounts of computation and allow maps to be trained much faster. This would enable SOMs to be used more readily for the many tradeoff decisions in product and process design.

Methodology

This section of the paper will discuss the methodology used in the foundation of the computational study. First, the changes made to the SOMViz software are discussed. Second, the test datasets are described. Third, a description of the nodal weight analysis that was developed for the full and partial datasets is provided. Finally, the data point tracking and node alignment process that was developed to assess the experimental runs is discussed.

SOMViz modifications for the computational study

The internal algorithm and operation of the SOMViz software started the same with opening a dataset, selecting the map size, and training the map. Nodes were indexed starting from zero as shown in Figure 3.4.

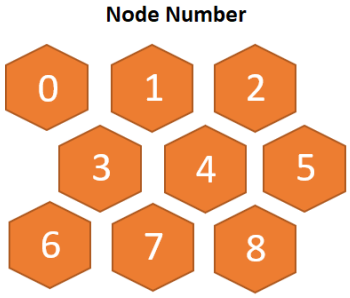

Figure 3.4 SOMViz map node index

Data points clustered in each node, and the associated weights, were stored after the map was trained. A button was added to the user interface to export the clustered data points and node weights to CSV files. The data was compiled for each dataset of CSOM and analyzed.

A preliminary map initialization method was added to SOMViz for the computational study. The initialization method took node weights from a previously trained SOM and initialized a new map with the node weights. Testing of the new initialization method with the same data, resulted in similar clusters from the previous trained map. The initialization method is not presented in this paper.

Description of Test Cases

Two datasets were chosen to develop and test the study and analysis procedures. The first test case was the Harvard Tuition dataset, which includes tuition costs for a variety of colleges over a span of 32 years [32]. A subset

of this data was created by only using approximately nine years (i.e. 2008 to 2017) of the tuition costs. This produced ninety data points with three dimensions. The second test case was the wine dataset [33], a standard dataset used to evaluate machine learning algorithms. This test case has 178 data points with 13 dimensions.

For the study, partial datasets were created from the original, full data. These partial datasets were created by randomly selecting half of the original data. For each test case ten partial datasets were created, randomly, to study if data selection had an effect on the final clustering. All eleven sets of data were trained on a CSOM (one full and ten halves), with node weights and data points exported for each. A total of 10 comparisons occurred between the full dataset and the ten halves. In order to make certain the results were reproducible, CSOMs were also trained for nine more full sets of data to showcase the impact of the random node weight initialization. Overall, there were 100 comparisons performed per test case.

The size of the CSOM map was pre-determined, using guidance from the original method, and stayed the same for the full and half datasets. Traditionally, the size of a SOM is dependent upon the size, and dimensions, of the dataset to represent the topological properties accurately. This research focuses on how accurately a subset of data can capture a full dataset, without introducing additional variables from changing the size of the map. The Harvard tuition test case was trained on a 3x3 nodal map, while the Wine test case was trained on a 4x4 map.

Node Weight Analysis

Each comparison between a full and half dataset calculates the percent difference of the full and half node weights. This is an indicator of how similar the weights are between nodes on separate maps. The goal is to have a lower percentage as this indicates the nodal pairs are very similar to each other and that a trained dataset from a partial dataset is essentially the same as those for the full dataset. The percent difference equation is the absolute value of full node weight minus half node weight divided by the full node weight times 100, for each node weight and dimension, see Eq. (8).

$$|\%\, Difference| = \frac{Full\ Node\ Weight\ x - Half\ Node\ Weight\ x}{Full\ Node\ Weight\ x},\ \ x = 0,1,2,\dots,\#\ of\ nodes \quad (8)$$

Figure 3. 3.5 shows the process of computing the percent differences for a sample of data for illustrative purposes. The percent difference is calculated for three nodes and all the dimensions within. In the full testing, this will be done for all nodes. Figure 3.5a shows the full dataset node weights for each dimension, while Figure 3.5b shows the half dataset node weights. The sequential node index of each dataset is paired together to compute the percent difference,

for example node 0 dimension 0 from the full dataset is matched with node 0 dimension 0 from the half dataset. Figure 3.5c shows the percent difference results between the two datasets from Eq. 8. The coloring of cells in Figure 3.5c shows four different percent difference categories: 0-40% (white), 41-80% (green), 81-120% (yellow), and 121+% (red).

Full Wine

	Node 0	Node 1	Node 2
Dimension 0	0.282	0.306	0.320
Dimension 1	0.188	0.173	0.247
Dimension 2	0.507	0.443	0.458
Dimension 3	0.540	0.499	0.480
Dimension 4	0.208	0.209	0.254
Dimension 5	0.348	0.418	0.550
Dimension 6	0.292	0.347	0.450
Dimension 7	0.607	0.421	0.326
Dimension 8	0.315	0.343	0.452
Dimension 9	0.134	0.131	0.155
Dimension 10	0.497	0.479	0.440
Dimension 11	0.508	0.589	0.654
Dimension 12	0.162	0.129	0.144

Half Wine

	Node 0	Node 1	Node 2
Dimension 0	0.557	0.541	0.441
Dimension 1	0.468	0.578	0.551
Dimension 2	0.454	0.470	0.480
Dimension 3	0.548	0.555	0.533
Dimension 4	0.331	0.241	0.195
Dimension 5	0.199	0.208	0.230
Dimension 6	0.098	0.065	0.092
Dimension 7	0.563	0.732	0.836
Dimension 8	0.285	0.255	0.199
Dimension 9	0.631	0.558	0.369
Dimension 10	0.162	0.199	0.337
Dimension 11	0.113	0.169	0.215
Dimension 12	0.287	0.295	0.275

	Node 0	Node 1	Node 2
Dimension 0	98%	77%	38%
Dimension 1	150%	233%	123%
Dimension 2	10%	6%	5%
Dimension 3	1%	11%	11%
Dimension 4	59%	16%	23%
Dimension 5	43%	50%	58%
Dimension 6	67%	81%	79%
Dimension 7	7%	74%	157%
Dimension 8	10%	26%	56%
Dimension 9	370%	325%	139%
Dimension 10	67%	58%	23%
Dimension 11	78%	71%	67%
Dimension 12	78%	129%	91%

Figure 3.5 a) Full wine node weights, b) Half wine node weights, c) Percent difference results

These percent difference categories were purposely selected with large ranges as it was unknown if a partial dataset could represent a full dataset at all. Further breakdown of an error category would be performed, if necessary. The dimension percent was distributed into the respective categories for each node as shown in Figure 3.6. A comparison summary of all the nodes between two maps is shown in Figure 3.7.

		Node 0	Node 1	Node 2
Dimension % Difference Categories	0-40%	4	4	5
	41-80%	6	5	4
	81-120%	1	1	1
	120+%	2	3	3

Figure 3.6 Percent difference for each node between a half and full dataset

		Node Weights	
		Count	Probability
Dimension % Difference Categories	0-40%	123	56%
	41-80%	59	27%
	81-120%	8	4%
	120+%	30	14%

Figure 3.7 Summary of node weight comparisons between two SOMViz maps

The node weight probability is the likelihood of two comparing nodes being in a percent difference category. The goal is to have the most amount of node weights in the lowest percent difference range. A low percent difference ensures the weights are similar, and the maps accurately represent each other. If this does not occur, then the maps are not similar. This analysis process was applied to the remaining 99 comparisons of node weight data.

Node Weight Alignment

The SOMViz clustering process normalizes input data and initializes the nodes with random samples of data. Training of the SOM is completed by a random order of input data. Similar clusters result from a trained map, but the orientation of the clusters (i.e. final row and column position) can change as the map is dimensionless. To say it plainly, with many factors changing (i.e. node weight initialization, order of input data), clusters on two maps, from the same data, will often be in different row and columns. All the same, or nearly the same, clusters will exist, just in different locations on the map.

However, the study described in this paper provides evidence that cluster locations are influenced, at least partially, by the nodal weights. Although researchers have assumed the node location on a final two-dimensional SOM was not fixed or consistent, this might not be the case. Further investigation needs to be performed to understand the relationship.

Multiple CSOMs were analyzed by comparing pairs of nodal weights. Since the sizes of the datasets were both equal the issue of what nodal pairs to select arose. For example, node 0 in a full dataset typically does not have the same data points as node 0 in a half dataset. In order to accommodate this shortcoming for the analysis, the nodes needed to be aligned by matching data points within. This process will not perfectly align all nodes because each SOM map is clustered with random sequencing. However, the process will generally align similar clusters, which will potentially produce a more accurate comparison of nodal pairs between maps.

Since the map sizes were the same, a complex alignment process as discussed in the background section was not deemed necessary. A simplified manual alignment process was developed for this research. This procedure (data point tracking, node alignment, and node weight comparison as described next) was labor-intensive and time-consuming, so the comparison was completed for one full and ten half datasets. If needed on a larger scale in the future, an automated version of this method should be developed.

Data points from each map were compiled into a list, labeled, and colored a key based on the input values. Points were distributed into their corresponding nodes for the half and full maps. Figure 3.8 shows the beginning of the alignment process with labeled data points for each map. The greyed out points in Figure 3.8 are the full data points not in the half dataset (i.e. the "other half" of the data). The manual procedure was to visually find each data point in both maps and note the location. Nodes with the highest quantity of matching data points were paired together. For example, node 13 in the half dataset would align with node 2 in the full dataset, because there are four matching data points.

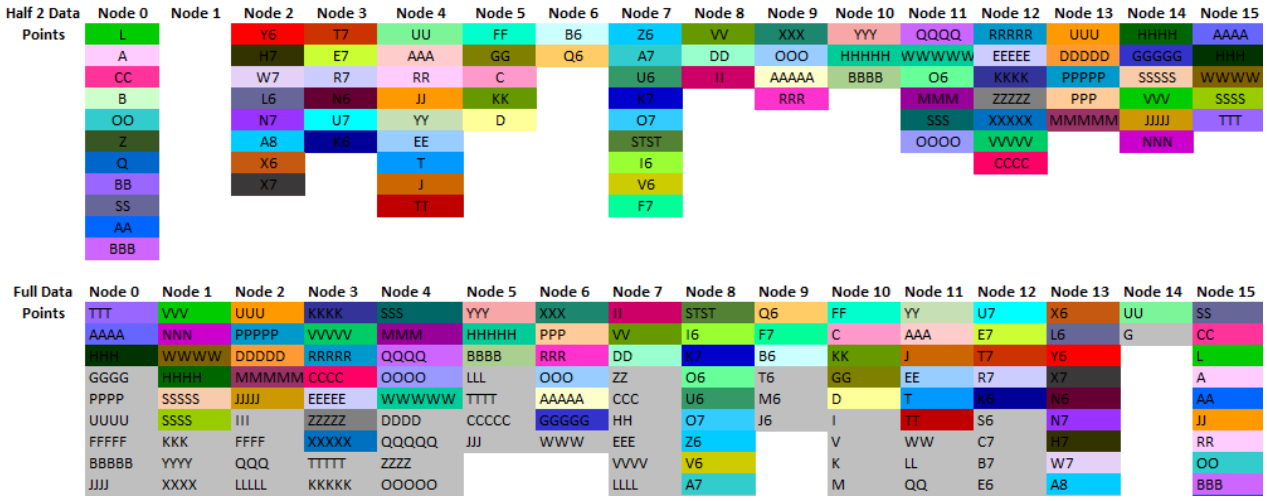

Figure 3.8 Datapoint setup for alignment of one full and one half dataset

Figure 3.9 shows the node lists after manually being aligned. The full nodes were reordered to match the half order. A series of trends emerged through the alignment proccess that seem to impact the results. Grouped data points stayed together and relocated to a different node index. Occasionally, grouped data points in one map would split between multiple nodes in the other map or vice-versa. The greatest impact on results can be contributed towards the end of the alignment process. After a majority of the nodes are matched, the remaining nodes may not contain equivalent data points and have to be matched as there are no more options remaining. This developed alignment process is not perfect but is a crucial first step to properly comparing two different SOM maps.

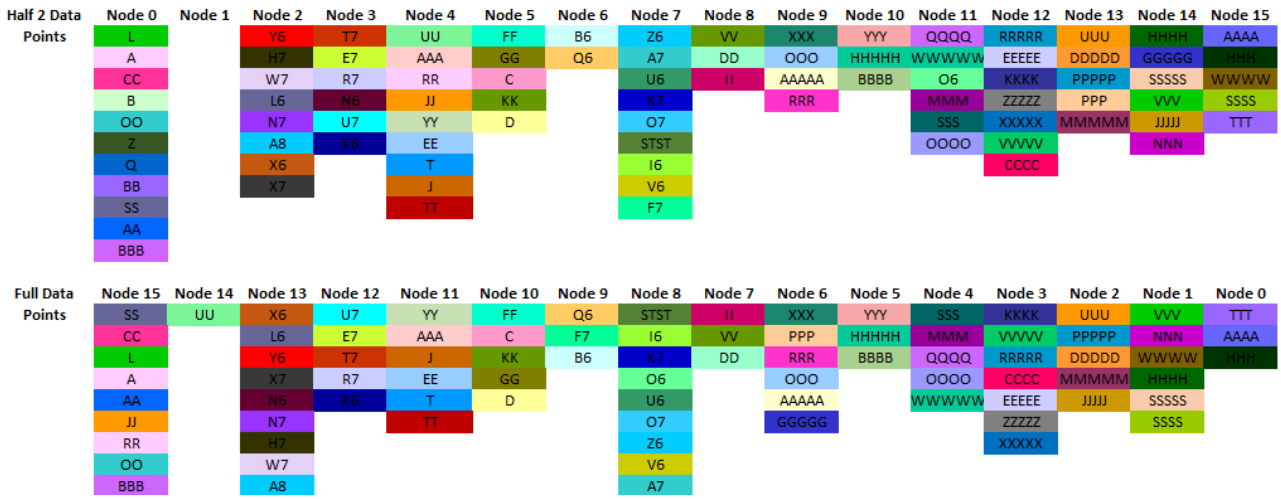

Figure 3.9 Aligned nodes for one full and one half dataset

The node weights were also reordered to match the alignment pairings, followed by the previous node weight comparison. In order to see the impact of the alignment procedure on the node weights, the lowest percent difference category 0-40% was broken down into smaller increments. Data from the initial node weight comparison was referred to as non-aligned data in this analysis, and the new aligned data is referred to as aligned data. Comparing the non-aligned summary (Figure 3.10a) to the aligned summary (Figure 3.10b) shows more nodes aligned with 10% error or less. This shows a strong comparison between these maps.

		Node Weights	
		Count	Probability
Dimension % Difference Categories	0-10%	31	15%
	11-20%	27	13%
	21-30%	27	13%
	31-40%	38	18%

		Node Weights	
		Count	Probability
Dimension % Difference Categories	0-10%	92	44%
	11-20%	54	26%
	21-30%	31	15%
	31-40%	13	6%

Figure 3.10 a) Lowest 40% breakdown for non-aligned nodes, b) Lowest 40% breakdown for aligned nodes

The number of data points in each node was tracked in the non-aligned and aligned comparisons. There is not an expectation for the nodes to have the same number of data points, but the distribution could show a correlation between map alignment. Figure 3.11 shows the data point distribution for full, half, and half aligned datasets. The total data points between the full and half dataset are not the same, because only half of the data is represented in the subset data. All of the distribution percentages are small, but the overall trend and changing intervals of data in each node

produces a general summary of a map. In the results section graphs created from this data are shown to demonstrate trends that suggest map alignment is a necessary step for this analysis.

Wine Full		Node 0	Node 1	Node 2	Node 3	Node 4	Node 5	Node 6	Node 7	Node 8	Node 9	Node 10	Node 11	Node 12	Node 13	Node 14	Node 15
Data	Count	12	12	10	10	10	7	7	9	16	6	10	15	13	14	2	25
Points	Percent	7%	7%	6%	6%	6%	4%	4%	5%	9%	3%	6%	8%	7%	8%	1%	14%

Wine Half		Node 0	Node 1	Node 2	Node 3	Node 4	Node 5	Node 6	Node 7	Node 8	Node 9	Node 10	Node 11	Node 12	Node 13	Node 14	Node 15
Data	Count	9	7	8	5	5	1	1	6	4	7	4	8	10	6	3	5
Points	Percent	10%	8%	9%	6%	6%	1%	1%	7%	4%	8%	4%	9%	11%	7%	3%	6%

Wine Half Aligned		Node 7	Node 11	Node 15	Node 10	Node 3	Node 5	Node 6	Node 9	Node 2	Node 4	Node 8	Node 13	Node 1	Node 0	Node 14	Node 12
Data	Count	6	8	5	4	5	1	1	7	8	5	4	6	7	9	3	10
Points	Percent	7%	9%	6%	4%	6%	1%	1%	8%	9%	6%	4%	7%	8%	10%	3%	11%

Figure 3.11 Datasets with data point quantity per node

Results

The following section discusses the results obtained from the above methods on the test cases.

Test Case 1 - Harvard Tuition Dataset

After 100 comparisons of one full to one half dataset on unaligned data, 20% of the nodal comparisons were in the in the 81-121+% categories, while over half of the node weights had a percent difference between 0-40% as shown in Figure 3.12a. This breakdown suggests some similarity between the half and full dataset SOMs exists, but further analysis was needed. Figure 3.12b has the breakdown of the 0-40% percent difference category. In this analysis, it is shown that 40% of the overall node pairs have a percent difference of less than 20%, suggesting at least a moderate level of accuracy in the map comparisons. The remaining 16% in the 40% breakdown further supports map accuracy. This also provides the first evidence that node locations via row and column may not be truly random from map to map created on the same, or similar, data.

Tuition 10 Full vs 10 Halves		Node Weights Count	Node Weights Probability
Dimensional % Difference Categories	0-40%	1006	56%
	41-80%	438	24%
	81-120%	139	8%
	121+%	217	12%

Tuition 40% Breakdown 10 Full vs 10 Halves		Node Weights Count	Node Weights Probability
Dimensional % Difference Categories	0-10%	438	24%
	11-20%	282	16%
	21-30%	174	10%
	31-40%	112	6%

Figure 3.12 a) Tuition data results for 100 comparisons, b) Tuition data 40% data breakdown

The manual node alignment process was then completed on a sample of this data. One full dataset trained SOM was compared to ten random half dataset maps. First, the same analysis as just shown was done on this sample with the data not aligned. The results are shown in Figures 3. 3.13a and 3.13b.

Tuition - Not Aligned 1 Full vs 10 Halves		Node Weights	
		Count	Probability
Dimension % Difference Categories	0-40%	113	63%
	41-80%	40	22%
	81-120%	9	5%
	120+%	18	10%

Tuition 40% Breakdown 1 Full vs 10 Halves		Node Weights	
		Count	Probability
Dimension % Difference Categories	0-10%	49	27%
	11-20%	35	19%
	21-30%	15	8%
	31-40%	14	8%

Figure 3.13 a) Non-aligned tuition data results, b) Non-aligned 40% tuition breakdown

Figures 3.14a and 3.14b show the overall results of the same sampled data after being aligned. Comparing Figures 3.13 and 3.14, the aligned results are 31% more accurate in the 0-40% category than the non-aligned results. About a quarter of the node weights in the non-aligned data have a percent difference between 0-10%, whereas half of the aligned node weights appear in this category.

Tuition - Aligned 1 Full vs 10 Halves		Node Weights	
		Count	Probability
Dimension % Difference Categories	0-40%	170	94%
	41-80%	9	5%
	81-120%	1	1%
	120+%	0	0%

Tuition 40% Breakdown - Aligned 1 Full vs 10 Halves		Node Weights	
		Count	Probability
Dimension % Difference Categories	0-10%	101	56%
	11-20%	57	32%
	21-30%	12	7%
	31-40%	0	0%

Figure 3.14 a) Aligned tuition data results, b) Aligned 40% tuition breakdown

This analysis shows that node alignment is potentially very important when comparing multiple SOM maps. When aligned, only 6% of the node weights in the aligned data were outside the 40% percent difference range. The 6% from the highest three categories could be attributed to the last couple of nodes paired in the alignment process not having matching data points as described earlier.

The distribution of data points in the nodes was also examined. This can serve as a visual representation of node alignment. Figure 3.15a displays the number of data points in each node for each dataset in a non-aligned comparison, whereas Figure 3.15b shows the aligned comparison. The general trend between these plots shows more uniformity in the aligned data versus the non-aligned data. Although, the non-aligned plot does have a general

pattern, which supports the previous notion of influence on the cluster locations in the SOM. Datasets 9H (i.e. dataset 9 half) and 9H-A (i.e. dataset 9 half-aligned) appear to be slight outliers.

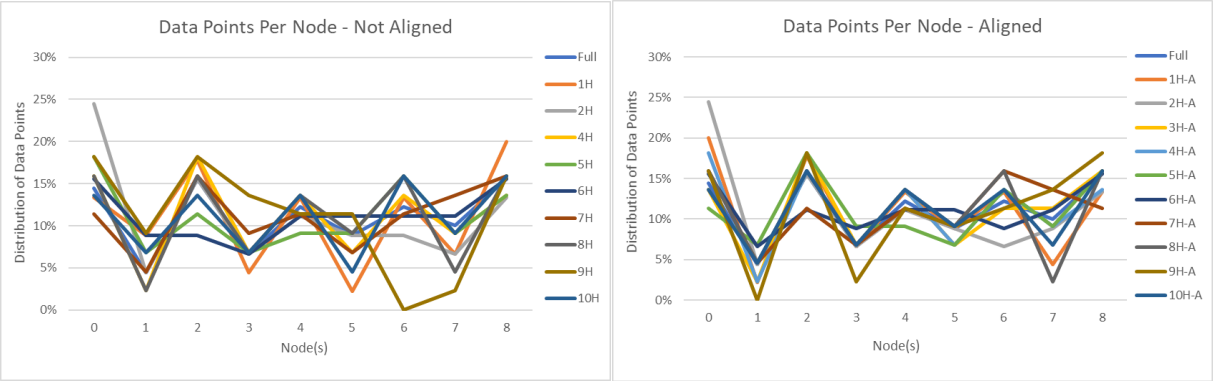

Figure 3.15 a) Distribution of non-aligned tuition data points, b) Distribution of aligned tuition data points

Further investigation into this outlier dataset shows the complexities that arise with the node alignment process shown in Figure 3.16. In this comparison, five of the nodes have the exact same data points in each map. Four of the nodes have similar data, but the quantity of data points is not evenly distributed amongst the nodes. Node 2 in both maps should match up since eight of the data points are the same, but two of the data points from node 2 in the full dataset appear in node 1 of the half dataset. Thus, node 2's weights may be slightly different because of the additional data points in the full map. In turn, node 1 in the half and node 5 in the full have different weights because some data points were split apart. The most inaccurate node weight representation will be node 6 in the half data set paired with node 1 in the full. This inaccuracy is due to data point KKK grouping with node 4 in the half dataset. This pairing was the last match in the node alignment process, so no other suitable pairings were left.

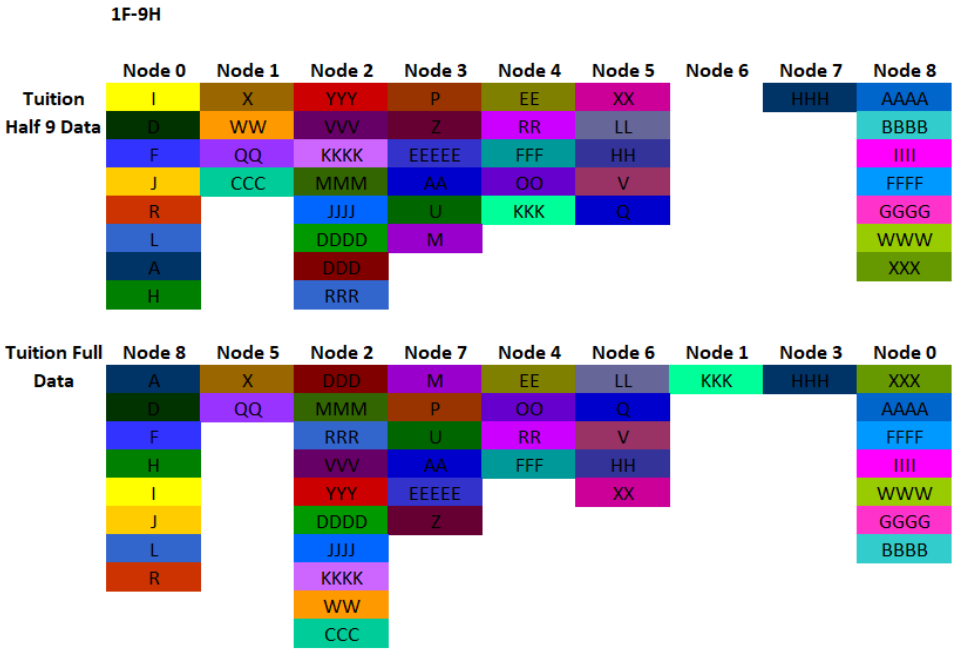

Figure 3.16 Node alignment process for tuition 9 dataset and full dataset

A look at the data point distribution further validates these observations. Figure 3.17a shows the non-aligned data only having five nodes with a similar trend; 0 through 2, 4 and 8. Figure 3.17b showcases the aligned data with seven nodes having the same trend. Node 7 had a different directional trend between the half and full dataset, when a few nodes split data points. This outlier dataset demonstrates that depending on the makeup of the partial dataset, node alignment may not yield that much improvement due to the data available and where it clustered in the map. However, map alignment for the entire test case did show that the half datasets accurately represented the full dataset for nine of the ten cases.

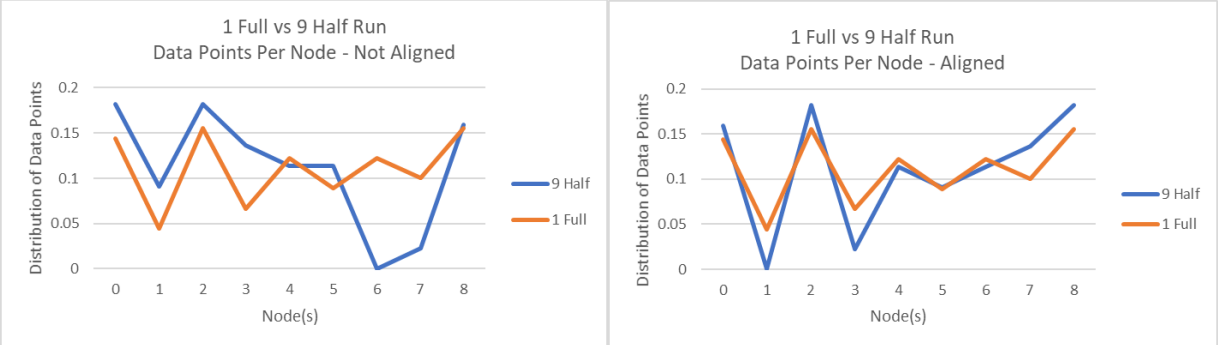

Figure 3.17 a) Non-aligned data point distribution, b) Aligned data point distribution

Test Case 2 - Wine Dataset

A hundred comparisons on test case 2, the wine dataset, showed similar trends as test case 1. The overall results are presented in Figure 3.18a and the lower 40% breakdown is shown in Figure 3.18b. Over 60% of the node weights have a percent difference less than 40%, which suggests the subset data has a better representation of the full dataset as compared to test case 1 in the non-aligned scenario. The lower 40% has a moderate distribution ranging between 10 and 22%. This again supports that node location via row and column is not entirely random.

Wine 10 Full vs 10 Halves		Node Weights	
		Count	Probability
Dimensional % Difference Categories	0-40%	12860	62%
	41-80%	5025	24%
	81-120%	1126	5%
	121+%	1789	9%

Wine 40% Breakdown 10 Full vs 10 Halves		Node Weights	
		Count	Probability
Dimensional % Difference Categories	0-10%	4547	22%
	11-20%	3638	17%
	21-30%	2527	12%
	31-40%	2148	10%

Figure 3.18 a) Wine data results for 100 comparisons, b) Wine data 40% data breakdown

The manual node alignment process was then completed on a sample of this data as was done in test case 1. One full dataset trained SOM was compared to the ten random half dataset maps. Figure 3.19a shows the sampled results of non-aligned data and Figure 3.19b analyzes the error in the lower 40% breakdown. Over 60% in the 0-40% category and almost 40% in the 0-20% category for the sampled non-aligned data.

Wine - Not Aligned 1 Full vs 10 Halves		Node Weights	
		Count	Probability
Dimension % Difference Categories	**0-40%**	1261	61%
	41-80%	510	25%
	81-120%	128	6%
	120+%	181	9%

Wine 40% Breakdown - Not Aligned 1 Full vs 10 Halves		Node Weights	
		Count	Probability
Dimension % Difference Categories	**0-10%**	440	21%
	11-20%	346	17%
	21-30%	274	13%
	31-40%	201	10%

Figure 3.19 a) Non-aligned wine data results, b) Non-aligned wine data 40% breakdown

Figures 3.20a and 3.20b show the results of the same sample with alignment performed. With alignment the percent differences show that the partial and full maps are extremely similar. This representation is based on the 95% probability of a node weight being in the 0-40% percent difference category. The 40% breakdown shows 75% of the node weights have a percent difference between 0-20%. This further displays there is an accurate representation of

subset data to full data. There still is some error as 5% of the data being outside a reasonable percent difference in the categories of 41-120+%.

Wine - Aligned 1 Full vs 10 Halves		Node Weights	
		Count	Probability
Dimension % Difference Categories	0-40%	1967	95%
	41-80%	87	4%
	81-120%	18	1%
	120+%	8	0%

Wine 40% Breakdown - Aligned 1 Full vs 10 Halves		Node Weights	
		Count	Probability
Dimension % Difference Categories	0-10%	991	48%
	11-20%	554	27%
	21-30%	286	14%
	31-40%	136	7%

Figure 3.20 a) Aligned wine data results, b) Aligned wine data 40% breakdwon

The data point count in each node again supports that alignment is necessary. Figure 3.21a shows the non-aligned datasets with a chaotic appearance, while Figure 3.21b shows the aligned datasets have a defined trend. It appears partial datasets 3H (i.e. dataset 3 half) and 9H (i.e. dataset 9 half) might be outliers in the alignment process based on the data point distribution.

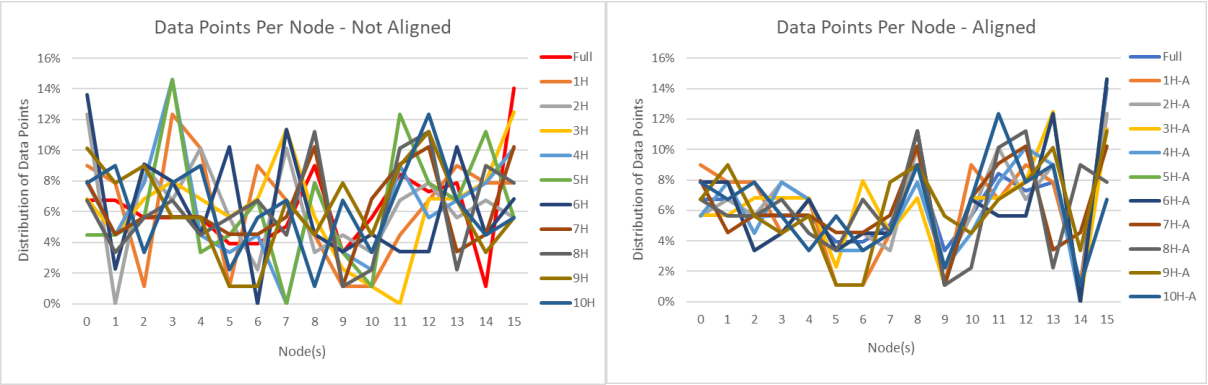

Figure 3.21 a) Distribution of non-aligned wine data points, b) Distribution of aligned wine data points

Further investigation into 9H shows the complication of the alignment process as in test case 1. After looking at all of the node matches, none of them have the same grouping of data points as shown in Figure 3.22. They have all been split apart and distributed throughout other nodes. The pairings that do not have matching data points and could be a source of error in map initialization are nodes 5, 6, and 14, but they are also the least populated nodes. It is interesting how those three nodes in both maps are directly in the same position, so the data has a correct topological representation. However, the data points in those six nodes are all dispersed to different nodes in the full dataset. A majority of the remaining nodes kept a portion of the same data points clustered together.

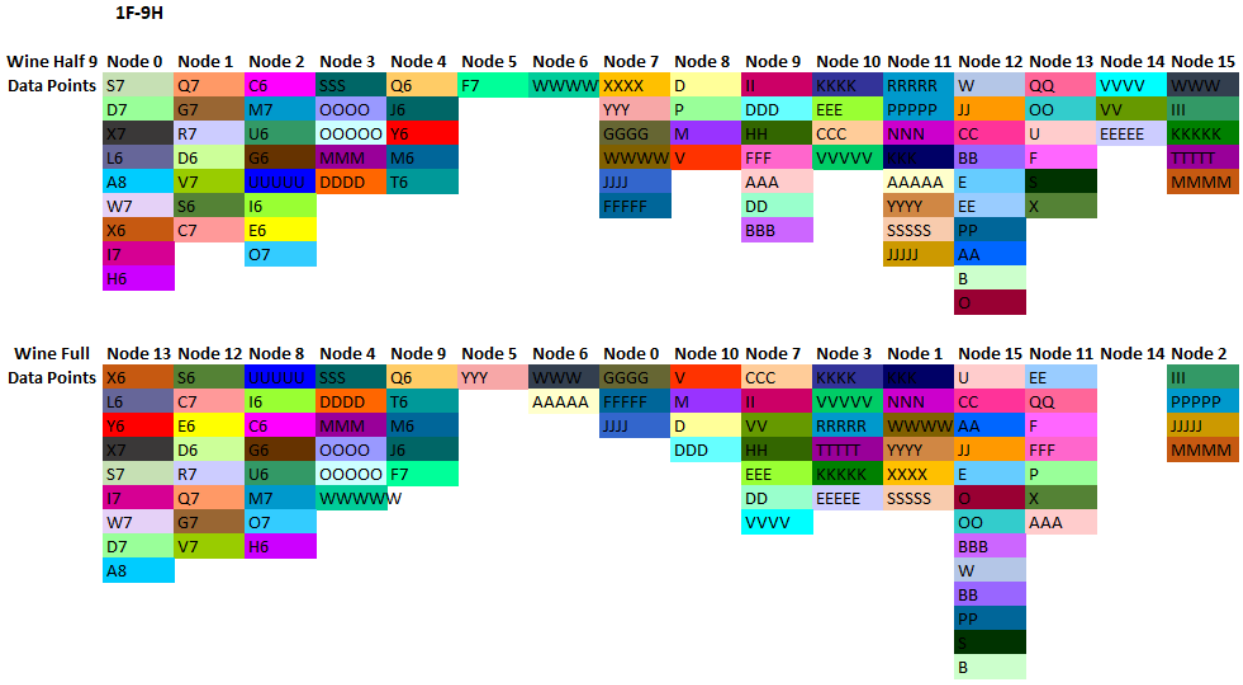

Figure 3.22 Node alignment process for wine 9H half and full dataset

Examination of the datapoint distribution for non-aligned data in Figure 3.23a and the aligned data in Figure 3.23b, shows some improvement for the 9H dataset after alignment. Non-aligned data had ten nodes with the same directional trend, whereas the aligned data had twelve nodes. So, even with 9H providing some error, it was still minimized by the alignment process.

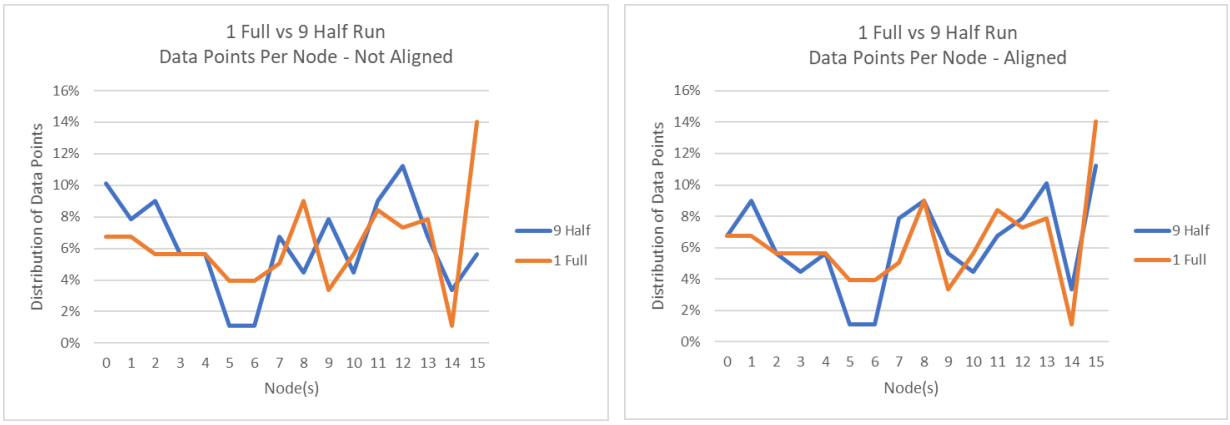

Figure 3.23 a) Graph of data point count in non-aligned data, b) Graph of data point count in aligned data

Conclusions and Future Work

SOMViz is a proven method of organizing n-dimensional data and helps the end-user process “big data”. However, due to computational resources and algorithmic development, SOMViz can take days to weeks depending on the size of the dataset. If it were possible to train new, partial, data as it becomes available, as opposed to always having to create a new full dataset, SOMs could become a much more widespread and useful method. This paper evaluated full and subset datasets through their node weights, an essential SOM component describing the connections between nodes, to enable comparisons of multiple SOMs. Initialized node weights are the critical component to setup an accurate representation between a subset and full dataset SOMs. An initial comparison of node weights between a full and subset data showed the subset data only partially represented a full dataset. The error could be attributed to the random training order of a SOM. A proof of concept, manual node alignment process, was created to further analyze this situation. Nodes were aligned by matching data points from two different SOM maps trained on subsets, or a full dataset. Reordering and pairing the nodes with their weights showed that, generally, the subsets created SOMs very similar to those created from the full dataset. Aligned maps of subset and full datasets had very similar node weights. These results suggest that partial datasets could be trained using node weights from a previous SOM run of similar data. This would dramatically reduce the computational time for developing a SOM.

There are several future work tasks suggested based on this research. Automating the node alignment process would allow for faster evaluation of partial to full datasets. The general procedure would identify pairs of nodes with the highest number of similar data points. Additional procedures are needed to manage when two nodes merge into one, or the remaining nodes are not similar at the end of comparisons. Further investigation into how the node weights affect the alignment process would also be beneficial.

Once the alignment process is automated and integrated, then SOMViz features would need to be added to handle training a new map from an already trained map. Additional node weights and data points would have to be exported and compared with previous data for validity.

While the two datasets tested are benchmarks, and a large number if comparisons were done, additional datasets need to be tested with the node alignment process. The results would validate what has been found in this research. Datasets to consider should include large ones as well as nonlinear data with outliers or holes. The size of the data subset could be a limiting factor in the success of the method. Additionally, the size of the map needs to be evaluated in relationship to alignment accuracy. If the map cannot correctly cluster the data in the given amount of space, then

maybe a different size or growing map will make improvements. Evaluation of computing time would give a hard number to say this method has decreased computation time by a certain amount.

References

[1] Gershon, N., Eick, S. G., and Card, S. “Information Visualization.” Interactions, Vol. 5, No. 2, 1998, pp. 9–15. https://doi.org/10.1145/274430.274432.

[2] Kohonen, T. “Self-Organized Formation of Topologically Correct Feature Maps.” Biological Cybernetics, Vol. 43, No. 1, 1982, pp. 59–69. https://doi.org/10.1007/BF00337288.

[3] Nekolny, B. “Contextual Self-Organizing Maps for Visual Design Space Exploration.” Graduate Theses and Dissertations, 2010. https://doi.org/10.31274/etd-180810-2599.

[4] Richardson, T. “A Software Environment for Visualizing High-Dimensional Data Using Contextual Self-Organizing Maps Linked with Immersive Virtual Reality.” Graduate Theses and Dissertations, 2013.

[5] Kohl, A. “Visualizing Engineering Design Data Using a Modified Two-Level Self-Organizing Map Clustering Approach.” Graduate Theses and Dissertations, 2017. https://doi.org/10.31274/etd-180810-5786.

[6] Deboeck, G., and Kohonen, T. Visual Explorations in Finance: With Self-Organizing Maps. Springer Science & Business Media, 2013.

[7] Mitrokotsa, A., and Douligeris, C. Detecting Denial of Service Attacks Using Emergent Self-Organizing Maps. Presented at the Proceedings of the Fifth IEEE International Symposium on Signal Processing and Information Technology, 2005., 2005.

[8] Wickramasinghe, C. S., Amarasinghe, K., and Manic, M. “Deep Self-Organizing Maps for Unsupervised Image Classification.” IEEE Transactions on Industrial Informatics, Vol. 15, No. 11, 2019, pp. 5837–5845. https://doi.org/10.1109/TII.2019.2906083.

[9] Richardson, T., Holub, J., and Winer, E. H. Improving Contextual Self-Organizing Map Solution Times Using GPU Parallel Training. In 15th AIAA/ISSMO Multidisciplinary Analysis and Optimization Conference, American Institute of Aeronautics and Astronautics, 2014.

[10] “SOM Training Methodology Image” 1-S2.0-S0925753511000865-Gr2.Jpg (352×219). https://origin-ars.els-cdn.com/content/image/1-s2.0-S0925753511000865-gr2.jpg. Accessed Jul. 5, 2020.

[11] Kubat, M. “Neural Networks: A Comprehensive Foundation by Simon Haykin, Macmillan, 1994, ISBN 0-02-352781-7.” The Knowledge Engineering Review, Vol. 13, No. 4, 1999, pp. 409–412. https://doi.org/10.1017/S0269888998214044.

[12] Richardson, T., and Winer, E. H. Increasing Feasibility of the Self-Organizing Map as a Design Tool through a Novel Convergence Heuristic. In 16th AIAA/ISSMO Multidisciplinary Analysis and Optimization Conference, American Institute of Aeronautics and Astronautics, 2015.

[13] Ultsch, A. “U*-Matrix: A Tool to Visualize Clusters in High Dimensional Data.” 2003, p. 11.

[14] Kohonen, T. Exploration of Very Large Databases by Self-Organizing Maps. In Proceedings of International Conference on Neural Networks (ICNN’97), No. 1, 1997, pp. PL1- PL6 vol.1.

[15] Ahmed, R. F. M., Salama, C., and Mahdi, H. Optimizing Self-Organizing Maps Parameters Using Genetic Algorithm: A Simple Case Study. Cham, 2020.

[16] Akinduko, A. A., Mirkes, E. M., and Gorban, A. N. “SOM: Stochastic Initialization versus Principal Components.” Information Sciences, Vols. 364–365, 2016, pp. 213–221. https://doi.org/10.1016/j.ins.2015.10.013.

[17] Wickramasinghe, C. S., Amarasinghe, K., and Manic, M. Parallalizable Deep Self-Organizing Maps for Image Classification. Presented at the 2017 IEEE Symposium Series on Computational Intelligence (SSCI), 2017.

[18] Abaei, G., Selamat, A., and Fujita, H. "An Empirical Study Based on Semi-Supervised Hybrid Self-Organizing Map for Software Fault Prediction." Knowledge-Based Systems, Vol. 74, 2015, pp. 28–39. https://doi.org/10.1016/j.knosys.2014.10.017.

[19] Mustonen, S. M., Tissari, S., Huikko, L., Kolehmainen, M., Lehtola, M. J., and Hirvonen, A. "Evaluating Online Data of Water Quality Changes in a Pilot Drinking Water Distribution System with Multivariate Data Exploration Methods." Water Research, Vol. 42, No. 10, 2008, pp. 2421–2430. https://doi.org/10.1016/j.watres.2008.01.015.

[20] Clark, S., Sisson, Scott. A., and Sharma, A. "Tools for Enhancing the Application of Self-Organizing Maps in Water Resources Research and Engineering." Advances in Water Resources, 2020, p. 103676. https://doi.org/10.1016/j.advwatres.2020.103676.

[21] Huang, F., Yin, K., Huang, J., Gui, L., and Wang, P. "Landslide Susceptibility Mapping Based on Self-Organizing-Map Network and Extreme Learning Machine." Engineering Geology, Vol. 223, 2017, pp. 11–22. https://doi.org/10.1016/j.enggeo.2017.04.013.

[22] Jeong, K.-S., Hong, D.-G., Byeon, M.-S., Jeong, J.-C., Kim, H.-G., Kim, D.-K., and Joo, G.-J. "Stream Modification Patterns in a River Basin: Field Survey and Self-Organizing Map (SOM) Application." Ecological Informatics, Vol. 5, No. 4, 2010, pp. 293–303. https://doi.org/10.1016/j.ecoinf.2010.04.005.

[23] Khanzadeh, M., Rao, P., Jafari-Marandi, R., Smith, B. K., Tschopp, M. A., and Bian, L. "Quantifying Geometric Accuracy With Unsupervised Machine Learning: Using Self-Organizing Map on Fused Filament Fabrication Additive Manufacturing Parts." Journal of Manufacturing Science and Engineering, Vol. 140, No. 3, 2018. https://doi.org/10.1115/1.4038598.

[24] Ruping, S., and Muller, J. "Analysis of IC Fabrication Processes Using Self-Organizing Maps." 1999, pp. 631–636. https://doi.org/10.1049/cp:19991181.

[25] Germen, E., Başaran, M., and Fidan, M. "Sound Based Induction Motor Fault Diagnosis Using Kohonen Self-Organizing Map." Mechanical Systems and Signal Processing, Vol. 46, No. 1, 2014, pp. 45–58. https://doi.org/10.1016/j.ymssp.2013.12.002.

[26] Li, Z., Fang, H., Huang, M., Wei, Y., and Zhang, L. "Data-Driven Bearing Fault Identification Using Improved Hidden Markov Model and Self-Organizing Map." Computers & Industrial Engineering, Vol. 116, 2018, pp. 37–46. https://doi.org/10.1016/j.cie.2017.12.002.

[27] Pérez-Benítez, J. A., Espina-Hernández, J. H., and Martínez-Ortiz, P. "Unwrapping the Influence of Multiple Parameters on the Magnetic Barkhausen Noise Signal Using Self-Organizing Maps." NDT & E International, Vol. 54, 2013, pp. 166–170. https://doi.org/10.1016/j.ndteint.2012.10.006.

[28] Obayashi, S., and Sasaki, D. Self-Organizing Map of Pareto Solutions Obtained from Multiobjective Supersonic Wing Design. Presented at the 40th AIAA Aerospace Sciences Meeting & Exhibit, Reno,NV,U.S.A., 2002.

[29] Mayer, R., Neumayer, R., Baum, D., and Rauber, A. Analytic Comparison of Self-Organising Maps. In Advances in Self-Organizing Maps (J. C. Príncipe and R. Miikkulainen, eds.), Springer Berlin Heidelberg, Berlin, Heidelberg, 2009, pp. 182–190.

[30] Pampalk, E. "Aligned Self-Organizing Maps." Proc. 4th Workshop on Self-Organizing Maps, 2003, p. 6.

[31] Chen, N., Ribeiro, B., Vieira, A., and Chen, A. "Clustering and Visualization of Bankruptcy Trajectory Using Self-Organizing Map." Expert Systems with Applications, Vol. 40, No. 1, 2013, pp. 385–393. https://doi.org/10.1016/j.eswa.2012.07.047.

[32] Harvard Tuition. https://kaggle.com/harvard-university/harvard-tuition. Accessed Jun. 22, 2020.

[33] UCI Machine Learning Repository: Wine Data Set. https://archive.ics.uci.edu/ml/datasets/wine. Accessed Jun. 25, 2020.

CHAPTER 4. CONCLUSIONS AND FUTURE WORK

SOMViz is a proven method of organizing n-dimensional data and helps the end-user process “big data”. However, due to computational resources and algorithmic development, SOMViz can take days to weeks depending on the size of the dataset. If it were possible to train new, partial, data as it becomes available, as opposed to always having to create a new full dataset, SOMs could become a much more widespread and useful method. This thesis evaluated full and subset datasets through their node weights, an essential SOM component describing the connections between nodes, to enable comparisons of multiple SOMs. Initialized node weights are the critical component to setup an accurate representation between a subset and full dataset SOMs. An initial comparison of node weights between a full and subset data showed the subset data only partially represented a full dataset. The error could be attributed to the random training order of a SOM. A proof of concept, manual node alignment process, was created to further analyze this situation. Nodes were aligned by matching data points from two different SOM maps trained on subsets, or a full dataset. Reordering and pairing the nodes with their weights showed that, generally, the subsets created SOMs very similar to those created from the full dataset. Aligned maps of subset and full datasets had very similar node weights. These results suggest that partial datasets could be trained using node weights from a previous SOM run of similar data. This would dramatically reduce the computational time for developing a SOM.

There are several future work tasks suggested based on this research. Automating the node alignment process would allow for faster evaluation of partial to full datasets. The general procedure would identify pairs of nodes with the highest number of similar data points. Additional procedures are needed to manage when two nodes merge into one, or the remaining

nodes are not similar at the end of comparisons. Further investigation into how the node weights affect the alignment process would also be beneficial.

Once the alignment process is automated and integrated, then SOMViz features would need to be added to handle training a new map from an already trained map. Additional node weights and data points would have to be exported and compared with previous data for validity.

While the two datasets tested are benchmarks, and a large number if comparisons were done, additional datasets need to be tested with the node alignment process. The results would validate what has been found in this research. Datasets to consider should include large ones as well as nonlinear data with outliers or holes. The size of the data subset could be a limiting factor in the success of the method. Additionally, the size of the map needs to be evaluated in relationship to alignment accuracy. If the map cannot correctly cluster the data in the given amount of space, then maybe a different size or growing map will make improvements. Evaluation of computing time would give a hard number to say this method has decreased computation time by a certain amount.

REFERENCES

[1] Gershon, N., Eick, S. G., and Card, S. "Information Visualization." *Interactions,* Vol. 5, No. 2, 1998, pp. 9–15. https://doi.org/10.1145/274430.274432.

[2] Kohonen, T. "Self-Organized Formation of Topologically Correct Feature Maps." *Biological Cybernetics*, Vol. 43, No. 1, 1982, pp. 59–69. https://doi.org/10.1007/BF00337288.

[3] Nekolny, B. "Contextual Self-Organizing Maps for Visual Design Space Exploration." *Graduate Theses and Dissertations*, 2010. https://doi.org/10.31274/etd-180810-2599.

[4] Richardson, T. "A Software Environment for Visualizing High-Dimensional Data Using Contextual Self-Organizing Maps Linked with Immersive Virtual Reality." *Graduate Theses and Dissertations*, 2013.

[5] Kohl, A. "Visualizing Engineering Design Data Using a Modified Two-Level Self-Organizing Map Clustering Approach." *Graduate Theses and Dissertations,* 2017. https://doi.org/10.31274/etd-180810-5786.

[6] Deboeck, G., and Kohonen, T. *Visual Explorations in Finance: With Self-Organizing Maps*. Springer Science & Business Media, 2013.

[7] Mitrokotsa, A., and Douligeris, C. Detecting Denial of Service Attacks Using Emergent Self-Organizing Maps. Presented at the Proceedings of the Fifth IEEE International Symposium on Signal Processing and Information Technology, 2005., 2005.

[8] Wickramasinghe, C. S., Amarasinghe, K., and Manic, M. "Deep Self-Organizing Maps for Unsupervised Image Classification." *IEEE Transactions on Industrial Informatics*, Vol. 15, No. 11, 2019, pp. 5837–5845. https://doi.org/10.1109/TII.2019.2906083.

[9] Richardson, T., Holub, J., and Winer, E. H. Improving Contextual Self-Organizing Map Solution Times Using GPU Parallel Training. In *15th AIAA/ISSMO Multidisciplinary Analysis and Optimization Conference*, American Institute of Aeronautics and Astronautics, 2014.

[10] SOM Training Methodology Image. https://origin-ars.els-cdn.com/content/image/1-s2.0-S0925753511000865-gr2.jpg. Accessed Jul. 5, 2020.

[11] Kubat, M. "Neural Networks: A Comprehensive Foundation by Simon Haykin, Macmillan, 1994, ISBN 0-02-352781-7." *The Knowledge Engineering Review*, Vol. 13, No. 4, 1999, pp. 409–412. https://doi.org/10.1017/S0269888998214044.

[12] Richardson, T., and Winer, E. H. Increasing Feasibility of the Self-Organizing Map as a Design Tool through a Novel Convergence Heuristic. In *16th AIAA/ISSMO Multidisciplinary Analysis and Optimization Conference*, American Institute of Aeronautics and Astronautics, 2015.

[13] Ultsch, A. "U*-Matrix: A Tool to Visualize Clusters in High Dimensional Data." 2003, p. 11.

[14] Kohonen, T. Exploration of Very Large Databases by Self-Organizing Maps. In *Proceedings of International Conference on Neural Networks (ICNN'97)*, No. 1, 1997, pp. PL1- PL6 vol.1.

[15] Ahmed, R. F. M., Salama, C., and Mahdi, H. Optimizing Self-Organizing Maps Parameters Using Genetic Algorithm: A Simple Case Study. Cham, 2020.

[16] Akinduko, A. A., Mirkes, E. M., and Gorban, A. N. "SOM: Stochastic Initialization versus Principal Components." *Information Sciences*, Vols. 364–365, 2016, pp. 213–221. https://doi.org/10.1016/j.ins.2015.10.013.

[17] Wickramasinghe, C. S., Amarasinghe, K., and Manic, M. Parallalizable Deep Self-Organizing Maps for Image Classification. Presented at the 2017 IEEE Symposium Series on Computational Intelligence (SSCI), 2017.

[18] Abaei, G., Selamat, A., and Fujita, H. "An Empirical Study Based on Semi-Supervised Hybrid Self-Organizing Map for Software Fault Prediction." *Knowledge-Based Systems*, Vol. 74, 2015, pp. 28–39. https://doi.org/10.1016/j.knosys.2014.10.017.

[19] Mustonen, S. M., Tissari, S., Huikko, L., Kolehmainen, M., Lehtola, M. J., and Hirvonen, A. "Evaluating Online Data of Water Quality Changes in a Pilot Drinking Water Distribution System with Multivariate Data Exploration Methods." *Water Research*, Vol. 42, No. 10, 2008, pp. 2421–2430. https://doi.org/10.1016/j.watres.2008.01.015.

[20] Leder, C., and Rehtanz, C. Electric Power System's Stability Assessment and Online-Provision of Control Actions Using Self-Organizing Maps. Berlin, Heidelberg, 2001.

[21] Huang, F., Yin, K., Huang, J., Gui, L., and Wang, P. "Landslide Susceptibility Mapping Based on Self-Organizing-Map Network and Extreme Learning Machine." *Engineering Geology*, Vol. 223, 2017, pp. 11–22. https://doi.org/10.1016/j.enggeo.2017.04.013.

[22] Khanzadeh, M., Rao, P., Jafari-Marandi, R., Smith, B. K., Tschopp, M. A., and Bian, L. "Quantifying Geometric Accuracy With Unsupervised Machine Learning: Using Self-Organizing Map on Fused Filament Fabrication Additive Manufacturing Parts." *Journal of Manufacturing Science and Engineering*, Vol. 140, No. 3, 2018. https://doi.org/10.1115/1.4038598.

[23] Ruping, S., and Muller, J. “Analysis of IC Fabrication Processes Using Self-Organizing Maps.” 1999, pp. 631–636. https://doi.org/10.1049/cp:19991181.

[24] Germen, E., Başaran, M., and Fidan, M. “Sound Based Induction Motor Fault Diagnosis Using Kohonen Self-Organizing Map.” *Mechanical Systems and Signal Processing*, Vol. 46, No. 1, 2014, pp. 45–58. https://doi.org/10.1016/j.ymssp.2013.12.002.

[25] Li, Z., Fang, H., Huang, M., Wei, Y., and Zhang, L. “Data-Driven Bearing Fault Identification Using Improved Hidden Markov Model and Self-Organizing Map.” *Computers & Industrial Engineering*, Vol. 116, 2018, pp. 37–46. https://doi.org/10.1016/j.cie.2017.12.002.

[26] Pérez-Benítez, J. A., Espina-Hernández, J. H., and Martínez-Ortiz, P. “Unwrapping the Influence of Multiple Parameters on the Magnetic Barkhausen Noise Signal Using Self-Organizing Maps.” *NDT & E International*, Vol. 54, 2013, pp. 166–170. https://doi.org/10.1016/j.ndteint.2012.10.006.

[27] Obayashi, S., and Sasaki, D. Self-Organizing Map of Pareto Solutions Obtained from Multiobjective Supersonic Wing Design. Presented at the 40th AIAA Aerospace Sciences Meeting & Exhibit, Reno,NV,U.S.A., 2002.

[28] Mayer, R., Neumayer, R., Baum, D., and Rauber, A. Analytic Comparison of Self-Organising Maps. In *Advances in Self-Organizing Maps* (J. C. Príncipe and R. Miikkulainen, eds.), Springer Berlin Heidelberg, Berlin, Heidelberg, 2009, pp. 182–190.

[29] Pampalk, E. “Aligned Self-Organizing Maps.” *Proc. 4th Workshop on Self-Organizing Maps*, 2003, p. 6.