Neuron Labeling for Self-Organizing Maps Using a Novel Example-Centric Algorithm with Weight-Centric Finalization

Willem S. van Heerden[a]
Department of Computer Science, University of Pretoria, Pretoria, South Africa

Keywords: Unsupervised Neural Networks, Self-Organizing Maps, Neuron Labeling, Data Science, Data Mining, Exploratory Data Analysis.

Abstract: A self-organizing map (SOM) is an unsupervised artificial neural network that models training data using a map structure of neurons, which preserves the local topological structure of the training data space. An important step in the use of SOMs for data science is the labeling of neurons, where supervised neuron labeling is commonly used in practice. Two widely-used supervised neuron labeling methods for SOMs are example-centric neuron labeling and weight-centric neuron labeling. Example-centric neuron labeling produces high-quality labels, but tends to leave many neurons unlabeled, thus potentially hampering the interpretation or use of the labeled SOM. Weight-centric neuron labeling guarantees a label for every neuron, but often produces less accurate labels. This research proposes a novel hybrid supervised neuron labeling algorithm, which initially performs example-centric neuron labeling, after which missing labels are filled in using a weight-centric approach. The objective of this algorithm is to produce high-quality labels while still guaranteeing labels for every neuron. An empirical investigation compares the performance of the novel hybrid approach to example-centric neuron labeling and weight-centric neuron labeling, and demonstrates the feasibility of the proposed algorithm.

1 INTRODUCTION

Self-organizing maps (SOMs) are unsupervised learning neural networks (Kohonen, 2001), which have been widely investigated in the literature (Kaski et al., 1998; Oja et al., 2003; Poll ¨ a et al., 2009). SOMs ¨ have seen wide use in data science, data mining, and exploratory data analysis (van Heerden, 2017). Some specific application areas include business analytics (Bowen and Siegler, 2024), healthcare (Javed et al., 2024), pandemic analysis (Galvan et al., 2021), astronomy (Vantyghem et al., 2024), and environmental research (Rosa et al., 2024). SOMs are discussed in more detail within Section 2.

SOMs are made up of neurons, which together model a training data set. Several SOM neuron labeling techniques exist, each of which attaches typically text-based characterizations to a subset of a SOM's neurons. Neuron labeling often plays an essential part in SOM-based data analysis (van Heerden and Engelbrecht, 2008). The quality of SOM-based data science therefore depends heavily on the performance of the labeling algorithm that has been chosen. Section 3 elaborates upon neuron labeling approaches.

Two of the most commonly used labeling methods are example-centric neuron labeling and weight-centric neuron labeling (Kohonen, 1989). Example-centric neuron labeling is very accurate but leaves some neurons unlabeled, while weight-centric neuron labeling lacks accuracy but guarantees a label for every neuron (van Heerden, 2017).

Section 4 introduces a novel SOM neuron labeling algorithm, namely example-centric neuron labeling with weight-centric finalization. This algorithm combines the best aspects of example-centric and weight-centric neuron labeling, by guaranteeing high-quality labels for every neuron of a SOM.

To demonstrate the advantages of the new algorithm, an empirical analysis was conducted. Section 5 presents the experimental results, which compare the performance of the novel algorithm to example-centric and weight-centric neuron labeling.

Finally, Section 6 concludes with a summary of this work's most important findings. Additionally, several avenues are suggested for future investigations stemming from the research presented here.

[a] https://orcid.org/0000-0002-9736-7268

van Heerden, W.
Neuron Labeling for Self-Organizing Maps Using a Novel Example-Centric Algorithm with Weight-Centric Finalization.
DOI: 10.5220/0013072000003837

In *Proceedings of the 16th International Joint Conference on Computational Intelligence (IJCCI 2024)*, pages 508-519
ISBN: 978-989-758-721-4; ISSN: 2184-3236

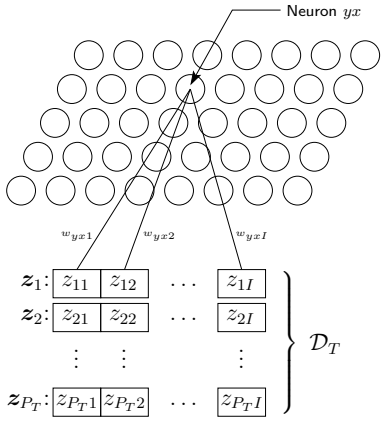

Figure 1: The architectural components making up a SOM.

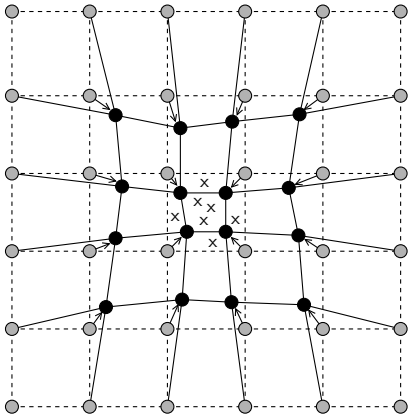

Figure 2: SOM weight updates for two-dimensional data.

2 SELF-ORGANIZING MAPS

Teuvo Kohonen introduced the SOM, which is an unsupervised neural network based on the associative nature of human cerebral cortices (Kohonen, 1982). In contrast to supervised neural networks, the training of a SOM does not require data classifications.

Figure 1 illustrates the architectural components of a SOM. Map training uses a training data set of P_T training examples, $\mathcal{D}_T = \{\vec{z}_1, \vec{z}_2, \ldots, \vec{z}_{P_T}\}$. Each training example is represented by an I-dimensional vector of attribute values, $\vec{z}_s = (z_{s1}, z_{s2}, \ldots, z_{sI})$. Every attribute value is denoted z_{sv} and is a real value.

The central map structure of a SOM consists of a grid of neurons with Y rows and X columns. Each neuron, denoted n_{yx}, is positioned at row y and column x of the grid, and has a linked weight vector, $\vec{w}_{yx} = (w_{yx1}, w_{yx2}, \ldots, w_{yxI})$. Every real valued weight, w_{yxv}, corresponds to z_{sv} across every $\vec{z}_s$ in $\mathcal{D}_T$.

The objective of SOM training is to adapt every $\vec{w}_{yx}$ in the map structure, so that they collectively model $\mathcal{D}_T$. The model is a simplified representation because $\mathcal{D}_T$ is I-dimensional, while the map is two-dimensional. The map has two characteristics:

1. The map models the probability density function of the data space represented by $\mathcal{D}_T$. Weight vectors move towards denser parts of the data space during training. Neurons thus model subsets of mutually similar data examples after training.
2. Data examples that are close together in the data space are represented by neurons that are positioned near each other in the map grid. The map model therefore maintains the local topological structure of the data space represented by $\mathcal{D}_T$.

Figure 2 shows the result of SOM training on synthetic two-dimensional data. Gray circles represent the original uniformly distributed positions of weight vectors in the map space, and dashed lines connect the weight vectors of adjacent neurons. Crosses show the positions of training examples in the data space. After training, the weight vectors shift to the locations of the black circles, where solid lines connect the weight vectors of adjacent neurons. Characteristic 1 is demonstrated by the drift of weight vectors to the dense group of training examples. Training examples are also positioned close to the weight vectors of adjacent neurons, illustrating characteristic 2.

Several SOM training algorithms exist (Engelbrecht, 2007), which are all usable with the proposed labeling approach. The experiments of Section 5 used the original stochastic training procedure (Kohonen, 1982), shown in Algorithm 1, which is commonly used and a good baseline for comparisons.

Stochastic training repeatedly iterates over each $\vec{z}_s \in \mathcal{D}_T$. The best matching unit (or BMU), denoted n_{ba}, with $\vec{w}_{ba}$ closest to $\vec{z}_s$, is defined as follows:

$$\|\vec{z}_s - \vec{w}_{ba}\|_2 = \min_{\forall yx} \left\{ \|\vec{z}_s - \vec{w}_{yx}\|_2 \right\} \quad (1)$$

At the current training iteration, t, every $\vec{w}_{yx}$ on the map is updated relative to the BMU, as follows:

$$\vec{w}_{yx}(t+1) = \vec{w}_{yx}(t) + \Delta\vec{w}_{yx}(t) \quad (2)$$

The change applied to a weight vector is composed of an update for each weight, as follows:

$$\Delta\vec{w}_{yx}(t) = \big(\Delta w_{yx1}(t), \Delta w_{yx2}(t), \ldots, \Delta w_{yxI}(t)\big) \quad (3)$$

Finally, the change computed for a weight is defined according to the following equation:

$$\Delta w_{yxv}(t) = h_{ba,yx}(t) \cdot \big(z_{sv} - w_{yxv}(t)\big) \quad (4)$$

Here, $h_{ba,yx}(t)$ is the neighborhood function at iteration t, which is usually the smooth Gaussian kernel:

$$h_{ba,yx}(t) = \eta(t) \cdot \exp\left(-\frac{\|c_{ba} - c_{yx}\|_2^2}{2 \cdot \big(\sigma(t)\big)^2} \right) \quad (5)$$

initialize map grid
set current training iteration, $t = 0$
randomly shuffle $\mathcal{D}_T$
repeat
  choose unselected $\vec{z}_s \in \mathcal{D}_T$
  use Equation (1) to find BMU, n_{ba}, for $\vec{z}_s$
  forall $\vec{w}_{yx}$ *in the map* **do**
    update $\vec{w}_{yx}$ using Equation (2)
  end
  update current training iteration, $t = t + 1$
  if all $\vec{z}_s \in \mathcal{D}_T$ selected, shuffle $\mathcal{D}_T$
until *a stopping condition is satisfied*

Algorithm 1: Stochastic SOM training procedure.

In this equation, $\eta(t)$ is the learning rate hyperparameter and $\sigma(t)$ is the kernel width hyperparameter, both at iteration t, while $\|c_{ba} - c_{yx}\|_2$ is the distance between the map coordinates of n_{ba} and n_{yx}.

For SOM training to converge, the learning rate and kernel width must both be monotonically decreasing functions of t. The reported experiments used an exponential decay function for the learning rate:

$$\eta(t) = \eta(0) \cdot e^{-t/\tau_1} \tag{6}$$

Here, $\eta(0)$ is the initial learning rate at the start of training, and τ_1 is a constant governing the decay rate. Similarly, the kernel width decays as follows:

$$\sigma(t) = \sigma(0) \cdot e^{-t/\tau_2} \tag{7}$$

The initial kernel width is $\sigma(0)$, and τ_2 is a constant affecting the rate of kernel width decay.

3 SOM NEURON LABELING

Neuron labeling attaches textual descriptions to neurons, to represent neuron characteristics. Neuron labels are often used during SOM-based exploratory data analysis, and are essential to automated rule extraction from SOMs (van Heerden and Engelbrecht, 2016). Neuron labeling approaches are either supervised or unsupervised (Azcarraga et al., 2008).

Supervised labeling uses classified examples from a labeling data set (either the training set or separate data). Algorithms in this category include example-centric neuron labeling (Kohonen, 1989), example-centric cluster labeling (Samarasinghe, 2007), and weight-centric neuron labeling (Kohonen, 1989).

Unsupervised labeling is not based on classified labeling data. In the simplest instance, a human analyst manually assigns neuron labels (Corradini and Gross, 1999). Algorithmic methods also exist, which base labels either on the weight vectors of the map, or

train a SOM, *map*, with $Y \times X$ neurons
forall n_{yx} *in map* **do**
  define empty mapped example set, M_{yx}
end
forall *labeling examples* $\vec{z}_s$ **do**
  use Equation (1) to find BMU, n_{ba}, for $\vec{z}_s$
  add $\vec{z}_s$ to M_{ba}
end
forall n_{yx} *in map* **do**
  find most common class, $\mathcal{A}_{cls}$, within M_{yx}
  label n_{yx} with $\mathcal{A}_{cls}$
end

Algorithm 2: Example-centric neuron labeling.

the labeling data in relation to the map neurons. These unsupervised algorithms include unique cluster labeling (Deboeck, 1999), unsupervised weight-based labeling (Serrano-Cinca, 1996; Lagus and Kaski, 1999; van Heerden and Engelbrecht, 2012), and unsupervised example-based labeling (Rauber and Merkl, 1999; Azcarraga et al., 2008).

Unsupervised labeling techniques are more flexible than their supervised counterparts. However, unsupervised labelings are difficult to interpret objectively, posing a problem for their empirical analysis. As a consequence, most research focuses on supervised labeling algorithms, as this paper does.

Example-centric cluster labeling performs poorly, particularly in the presence of heterogeneous clusters of weight vectors (van Heerden, 2017). This research therefore focuses only on example-centric neuron labeling and weight-centric neuron labeling, which are elaborated upon in Sections 3.1 and 3.2.

3.1 Example-Centric Neuron Labeling

Algorithm 2 represents example-centric neuron labeling. An initially empty set of mapped labeling examples is associated with each neuron on the map. Each labeling example, $\vec{z}_s$, is then mapped to its BMU using Equation (1), and $\vec{z}_s$ is added to the labeling example set of the BMU. Finally, each neuron is labeled using the most common classification appearing in the corresponding labeling example set.

Example-centric neuron labeling leaves neurons unlabeled when the associated labeling example sets are empty. Such neurons are often referred to as interpolating units, and represent a large proportion of the map in some instances (van Heerden, 2023).

Despite these unlabeled neurons, example-centric neuron labeling has been shown to outperform the other supervised labeling methods when used as a basis for simple classification tasks (van Heerden, 2017). It has also been observed that the presence of

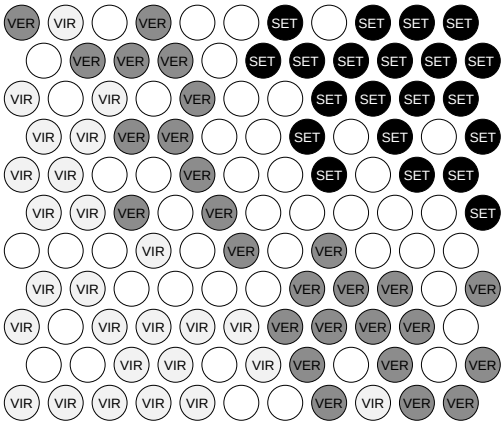

Figure 3: A SOM trained on the Iris data set and labeled using example-centric neuron labeling.

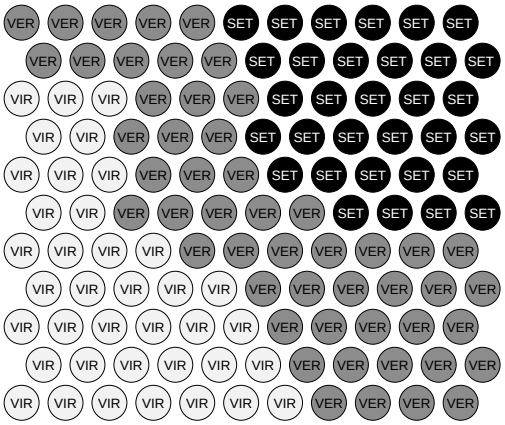

Figure 4: A SOM trained on the Iris data set and labeled using weight-centric neuron labeling.

unlabeled neurons is not strongly correlated to poor label quality (van Heerden and Engelbrecht, 2016).

While unlabeled neurons do not negatively impact the quality of map characterization, the missing labels often hamper SOM-based exploratory data analysis performed by human experts (van Heerden and Engelbrecht, 2016). This is because the labeled map is broken up by uncharacterized areas, which obscure a broader overview of the model characteristics.

The outcomes of the labeling algorithms under investigation are shown using an example SOM that was trained on the well-known Iris data set (Fisher, 1936). Figure 3 shows the example SOM labeled using example-centric neuron labeling. The labels "SET", "VER", and "VIR" respectively represent the Iris Setosa, Iris Versicolor, and Iris Virginica classifications present in the data set. The abundance of empty circles in this visualization clearly illustrates the large number of unlabeled neurons left by the algorithm, which account for 38.8% of the map.

3.2 Weight-Centric Neuron Labeling

Weight-centric neuron labeling is outlined in Algorithm 3. In contrast to example-centric neuron labeling, weight-centric neuron labeling maps neurons to data examples from a labeling set. Each neuron's weight vector is mapped to the closest labeling example, $\vec{z}_e$, which is called the best matching example (or BME), and is defined as follows:

$$\|\vec{w}_{yx} - \vec{z}_e\|_2 = \min_{\forall s}\{\|\vec{w}_{yx} - \vec{z}_s\|_2\} \quad (8)$$

train a SOM, *map*, with $Y \times X$ neurons
forall n_{yx} *in map* **do**
  use Equation (8) to find BME, $\vec{z}_e$, for $\vec{w}_{yx}$
  find class, $\mathcal{A}_{cls}$, associated with $\vec{z}_e$
  label n_{yx} with $\mathcal{A}_{cls}$
end

Algorithm 3: Weight-centric neuron labeling.

The neuron currently under consideration is then labeled with the classification of its BME.

Weight-centric neuron labeling is simpler than example-centric neuron labeling, and does not require the storage and searching of mapped labeling example sets. Additionally, the algorithm guarantees a label for every neuron on a map, facilitating easier interpretation for human data analysts.

Unfortunately, neurons will receive poor weight-centric labels if the BME match is not close. This results in a tendency for weight-centric neuron labeling to produce less accurate labels than example-centric neuron labeling (van Heerden, 2017).

Figure 4 depicts the result of performing weight-centric neuron labeling on the previously-mentioned example SOM trained on the Iris data set. It is clear that every neuron on the map has received a label, making the map more interpretable when compared to the example-centric neuron labeling in Figure 3.

4 THE PROPOSED ALGORITHM

The proposed algorithm is referred to as example-centric neuron labeling with weight-centric finalization, and hybridizes example-centric neuron labeling with weight-centric neuron labeling. The objective is to generate labels that are accurate, while also guaranteeing labels for every neuron on a map.

Algorithm 4 represents the proposed technique, which is a two-phase process. The first phase performs a normal example-centric neuron labeling, in which neurons that are BMUs at least once are labeled. This phase will typically produce high-quality

train a SOM, *map*, with $Y \times X$ neurons
forall n_{yx} *in map* **do**
  define empty mapped example set, M_{yx}
end
forall *labeling examples* $\vec{z}_s$ **do**
  use Equation (1) to find BMU, n_{ba}, for $\vec{z}_s$
  add $\vec{z}_s$ to M_{ba}
end
forall n_{yx} *in map* **do**
  find most common class, $\mathcal{A}_{cls}$, within M_{yx}
  label n_{yx} with $\mathcal{A}_{cls}$
end
forall n_{yx} *in map* **do**
  if n_{yx} *has no associated label* **then**
    use Equation (8) to find BME, $\vec{z}_e$, for $\vec{w}_{yx}$
    find class, $\mathcal{A}_{cls}$, associated with $\vec{z}_e$
    label n_{yx} with $\mathcal{A}_{cls}$
  end
end

Algorithm 4: Example-centric neuron labeling with weight-centric finalization.

labels for only a subset of the map neurons. The second phase then iterates over only the neurons that have remained unlabeled, assigning a weight-centric label to each. For each of these neurons, the classification of its BME becomes the neuron label. The labels produced by the second phase will be of lower quality, but will ensure a complete set of labels.

The labeling for the example Iris data set SOM, which is produced by the proposed algorithm, is shown in Figure 5. When comparing this labeling to the standard weight-centric neuron labeling shown in Figure 4, it is clear that the labelings are very similar. The maps only differ in terms of two neuron labels: the second from the left on the top row, and the third from the right on the bottom row. By consulting Figure 3 it is clear that these differences are due to the example-centric neuron labeling performed during the first labeling phase. It is hypothesized that such differences constitute more accurate labels than those produced by example-centric neuron labeling.

5 EXPERIMENTAL RESULTS

To explore the hypothesized advantages of example-centric neuron labeling with weight-centric finalization, a series of experiments were conducted. These experiments compared the performance of the novel algorithm against basic example-centric and weight-centric neuron labeling, when the algorithms were used in SOM-based data classification tasks.

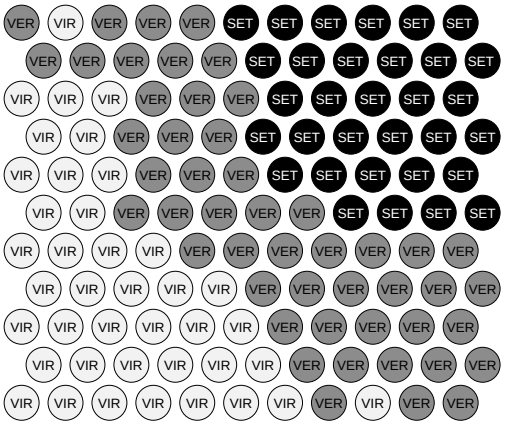

Figure 5: A SOM trained on the Iris data set and labeled using the proposed algorithm.

The standard stochastic training algorithm produced SOMs for several benchmark data sets. Six data sets were drawn from the UCI Machine Learning Repository, namely the Iris data set (Fisher, 1936), the ionosphere data set (Sigillito et al., 1989), the three monk's problems data sets (Wnek, 1993), and the glass identification data set (German, 1987). It should be noted that the third monk's problem set has 5% classification noise added to the training data. The data sets were preprocessed by using one-hot encoding for nominal attributes (Bishop, 2006), and scaling all attribute values to a $[0.0, 1.0]$ range using min-max normalization (Han et al., 2012).

The classification task was executed using the author's open-source SOM and labeling algorithm collection (van Heerden, 2024), which is based on Kohonen's original SOM_PAK reference implementation (Kohonen et al., 1996). Square maps were used throughout, and all weight vectors were initialized using hypercube initialization (Su et al., 1999). The SOM's main stopping condition was a limit of 0.0001 on a 30-iteration moving average of the standard deviation computed for the map's quantization error (van Heerden, 2017). To ensure termination, training was also stopped after 100 000 iterations.

Following map training, the appropriate neuron labeling algorithm was applied. Finally, classifications were performed by mapping a data example to its BMU. The label of the BMU was then applied as the classification of the data example. Each classification was compared to the actual classes of the data example, and a misclassification occurred if a match was not found. If the BMU was unlabeled, the data example remained unclassified.

Hyperparameter tuning was performed separately for the three neuron labeling algorithms applied to each of the benchmark data sets. The tuning procedure (van Heerden, 2017; Franken, 2009) used Sobol'

Table 1: Optimal hyperparameters per data set for example-centric neuron labeling.

Parameter	Iris	Ionosphere	Monks 1	Monks 2	Monks 3	Glass
Y and X	5	7	14	15	11	6
$\eta(0)$	5.488	3.848	6.055	8.867	9.941	0.449
τ_1	1 432.617	1 209.961	849.609	1 365.234	577.148	430.664
$\sigma(0)$	2.119	1.818	10.445	1.348	3.029	1.948
τ_2	77.539	59.570	9.766	31.641	83.008	7.617

Table 2: Optimal hyperparameters per data set for weight-centric neuron labeling.

Parameter	Iris	Ionosphere	Monks 1	Monks 2	Monks 3	Glass
Y and X	12	17	14	15	19	11
$\eta(0)$	4.609	0.918	6.563	8.867	9.629	0.508
τ_1	574.219	594.727	46.875	1 365.234	61.523	1 236.328
$\sigma(0)$	10.781	14.045	4.813	1.348	4.639	1.904
τ_2	96.094	87.305	34.375	31.641	67.383	5.859

Table 3: Optimal hyperparameters per data set for example-centric neuron labeling with weight-centric finalization.

Parameter	Iris	Ionosphere	Monks 1	Monks 2	Monks 3	Glass
Y and X	11	15	15	16	14	13
$\eta(0)$	7.266	8.945	8.867	2.480	1.836	7.910
τ_1	363.281	908.203	1 365.234	1 391.602	1 060.547	1 163.086
$\sigma(0)$	6.273	1.934	1.348	0.719	5.414	1.082
τ_2	16.406	58.984	31.641	2.930	55.078	75.195

sequences (Sobol', 1967) to generate configurations that sample the parameter space uniformly. Tables 1 to 3 present the optimal hyperparameters for example-centric neuron labeling, weight-centric neuron labeling, and example-centric neuron labeling with weight-centric finalization, respectively.

For each algorithm and data set, a 30-fold cross-validation was performed, and performance measures were recorded. To test whether performance measure differences were statistically significant, two-tailed non-parametric Wilcoxon signed-rank hypothesis tests (Wilcoxon, 1945) were performed with a confidence level of 0.05. A Bonferroni correction (Miller, 1981) was applied, to counteract the multiple comparisons problem. The results tables that follow report the mean and standard deviation for each performance measure, as well as a hypothesis test p-value. When a p-value indicates a significant performance difference, the mean and standard deviation of the better-performing algorithm are marked in bold.

Tables 4 and 5 respectively compare the overall training set error performance of the proposed approach against example-centric neuron labeling and weight-centric neuron labeling. These errors could be due to a combination of misclassified and unclassified training set data examples. The tables clearly illustrate that the proposed algorithm outperforms example-centric neuron labeling in all instances, and weight-centric neuron labeling in all but one case (for the third monk's problem, which included training data noise, weight-centric neuron labeling outperformed the proposed algorithm). The proposed approach generally did not outperform example-centric neuron labeling by a very large margin, except in the case of the ionosphere and glass identification data sets. In contrast, weight-centric neuron labeling underperformed by a substantial amount in the cases of the ionosphere, glass identification, and the first two monk's problems data sets.

To gain a deeper insight into whether any type of classification error was more prevalent, Tables 6 and 7 summarize the training set errors due only to misclassified data examples, when comparing the proposed algorithm to example-centric and weight-centric neuron labeling, respectively. The performance differences were exactly the same as those observed for the overall training set error, indicating that all classification errors were due to incorrect classifications.

Tables 8 and 9 respectively present the differences in performance when comparing the novel algorithm

Table 4: Overall training error comparison between the proposed algorithm and example-centric neuron labeling.

	EC-WC		EC		p-value
	$\mathcal{E}_T$	$\mathcal{S}_T$	$\mathcal{E}_T$	$\mathcal{S}_T$	
Iris	**1.724**	**0.594**	3.655	0.705	3.725×10^{-9}
Ionosphere	**3.647**	**0.845**	10.137	1.221	1.863×10^{-9}
Monks 1	**17.049**	**1.108**	18.126	1.084	0.001
Monks 2	**15.056**	**0.854**	15.766	0.657	0.001
Monks 3	**22.225**	**1.224**	23.828	1.792	0.001
Glass	**11.449**	**1.518**	26.28	1.531	1.863×10^{-9}

Table 5: Overall training error comparison between the proposed algorithm and weight-centric neuron labeling.

	EC-WC		WC		p-value
	$\mathcal{E}_T$	$\mathcal{S}_T$	$\mathcal{E}_T$	$\mathcal{S}_T$	
Iris	**1.724**	**0.594**	2.575	0.700	2.265×10^{-6}
Ionosphere	**3.647**	**0.845**	15.216	1.626	1.863×10^{-9}
Monks 1	**17.049**	**1.108**	27.974	2.416	1.863×10^{-9}
Monks 2	**15.056**	**0.854**	23.947	2.693	1.863×10^{-9}
Monks 3	22.225	1.224	**19.769**	**1.873**	5.307×10^{-6}
Glass	**11.449**	**1.518**	27.778	2.720	1.863×10^{-9}

Table 6: Training misclassified error comparison between the proposed algorithm and example-centric neuron labeling.

	EC-WC		EC		p-value
	$\mathcal{E}_{TM}$	$\mathcal{S}_{TM}$	$\mathcal{E}_{TM}$	$\mathcal{S}_{TM}$	
Iris	**1.724**	**0.594**	3.655	0.705	3.725×10^{-9}
Ionosphere	**3.647**	**0.845**	10.137	1.221	1.863×10^{-9}
Monks 1	**17.049**	**1.108**	18.126	1.084	0.001
Monks 2	**15.056**	**0.854**	15.766	0.657	0.001
Monks 3	**22.225**	**1.224**	23.828	1.792	0.001
Glass	**11.449**	**1.518**	26.28	1.531	1.863×10^{-9}

Table 7: Training misclassified error comparison between the proposed algorithm and weight-centric neuron labeling.

	EC-WC		WC		p-value
	$\mathcal{E}_{TM}$	$\mathcal{S}_{TM}$	$\mathcal{E}_{TM}$	$\mathcal{S}_{TM}$	
Iris	**1.724**	**0.594**	2.575	0.700	2.265×10^{-6}
Ionosphere	**3.647**	**0.845**	15.216	1.626	1.863×10^{-9}
Monks 1	**17.049**	**1.108**	27.974	2.416	1.863×10^{-9}
Monks 2	**15.056**	**0.854**	23.947	2.693	1.863×10^{-9}
Monks 3	22.225	1.224	**19.769**	**1.873**	5.307×10^{-6}
Glass	**11.449**	**1.518**	27.778	2.720	1.863×10^{-9}

Table 8: Training unclassified error comparison between the proposed algorithm and example-centric neuron labeling.

	EC-WC		EC		p-value
	$\mathcal{E}_{TU}$	$\mathcal{S}_{TU}$	$\mathcal{E}_{TU}$	$\mathcal{S}_{TU}$	
Iris	0.000	0.000	0.000	0.000	N/A
Ionosphere	0.000	0.000	0.000	0.000	N/A
Monks 1	0.000	0.000	0.000	0.000	N/A
Monks 2	0.000	0.000	0.000	0.000	N/A
Monks 3	0.000	0.000	0.000	0.000	N/A
Glass	0.000	0.000	0.000	0.000	N/A

Table 9: Training unclassified error comparison between the proposed algorithm and weight-centric neuron labeling.

	EC-WC		WC		p-value
	$\mathcal{E}_{TU}$	$\mathcal{S}_{TU}$	$\mathcal{E}_{TU}$	$\mathcal{S}_{TU}$	
Iris	0.000	0.000	0.000	0.000	N/A
Ionosphere	0.000	0.000	0.000	0.000	N/A
Monks 1	0.000	0.000	0.000	0.000	N/A
Monks 2	0.000	0.000	0.000	0.000	N/A
Monks 3	0.000	0.000	0.000	0.000	N/A
Glass	0.000	0.000	0.000	0.000	N/A

to the standard example-centric and weight-centric methods, in terms of the training set error due only to unclassified data examples. Here, it is observable that no training examples were left unclassified by any of the algorithms. All the labeling approaches therefore fully modeled the underlying training data.

In terms of algorithm analysis, training set performance is not a realistic representation of the real-world behavior that can be expected from an algorithm. As a result, the next three sets of comparisons focus on test set error, computed from data examples that were not presented during SOM training.

The differences in overall test set classification performance, when contrasting the new algorithm and the basic example-centric and weight-centric labeling techniques, are illustrated in Tables 10 and 11. No statistically significant differences in test error performance were observed when comparing the proposed method to example-centric neuron labeling. This implies that example-centric neuron labeling with weight-centric finalization classifies examples as well as example-centric labeling alone. This is expected, because the first phase of the hybrid technique is based on example-centric labeling. The weight-centric finalization thus does not interfere with classification accuracy. Weight-centric labeling performed significantly worse than the novel algorithm by large margins in half of the data sets (the ionosphere and first two monk's problems). In the other three data sets, no significant performance difference was observed. The example-centric basis of the hybrid method is responsible for this behavior.

Tables 12 and 13 juxtapose the test set error due to only misclassifications. While very small performance differences were observable when comparing the novel approach to example-centric labeling, none were statistically significant. The comparison against weight-centric labeling produced very similar results to those observed for the overall test error.

The test set error due to unclassified examples is compared in Tables 14 and 15. No statistically significant differences were observed when comparing the new algorithm against either example-centric or weight-centric neuron labeling. This supports the observation that classification errors were due mostly to misclassifications, which was made when considering the training set error performance.

Finally, Tables 16 and 17 compare the percentage of neurons that were left unlabeled by the algorithms. The example-centric method produced significantly more unlabeled neurons than the proposed algorithm. The observed differences were generally large, with example-centric neuron labeling leaving nearly half the neurons for the first two monk's problems uncharacterized. The combined approach and weight-centric labeling of course both left no neurons unlabeled.

When viewed holistically, example-centric labeling with weight-centric finalization classified data ex-

Table 10: Overall test error comparison between the proposed algorithm and example-centric neuron labeling.

	EC-WC		EC		p-value
	$\mathcal{E}_G$	$\mathcal{S}_G$	$\mathcal{E}_G$	$\mathcal{S}_G$	
Iris	2.667	6.915	4.000	8.137	0.688
Ionosphere	8.788	9.086	11.515	10.923	0.124
Monks 1	19.286	12.178	20.000	12.358	0.532
Monks 2	19.762	10.559	20.238	10.116	0.866
Monks 3	26.190	9.807	26.429	12.178	0.959
Glass	25.714	14.236	31.905	18.639	0.187

Table 11: Overall test error comparison between the proposed algorithm and weight-centric neuron labeling.

	EC-WC		WC		p-value
	$\mathcal{E}_G$	$\mathcal{S}_G$	$\mathcal{E}_G$	$\mathcal{S}_G$	
Iris	2.667	6.915	3.333	7.581	1.000
Ionosphere	**8.788**	**9.086**	15.152	12.016	0.002
Monks 1	**19.286**	**12.178**	31.667	11.817	2.365×10^{-4}
Monks 2	**19.762**	**10.559**	29.762	9.395	3.052×10^{-5}
Monks 3	26.190	9.807	27.619	12.118	0.842
Glass	25.714	14.236	33.333	17.729	0.024

Table 12: Test misclassified error comparison between the proposed algorithm and example-centric neuron labeling.

	EC-WC		EC		p-value
	$\mathcal{E}_{GM}$	$\mathcal{S}_{GM}$	$\mathcal{E}_{GM}$	$\mathcal{S}_{GM}$	
Iris	2.667	6.915	3.333	7.581	1.000
Ionosphere	8.788	9.086	11.212	10.319	0.191
Monks 1	19.286	12.178	18.810	12.226	0.965
Monks 2	19.762	10.559	19.524	9.177	0.971
Monks 3	26.190	9.807	25.952	12.082	0.734
Glass	25.714	14.236	30.476	19.031	0.314

Table 13: Test misclassified error comparison between the proposed algorithm and weight-centric neuron labeling.

	EC-WC		WC		p-value
	$\mathcal{E}_{GM}$	$\mathcal{S}_{GM}$	$\mathcal{E}_{GM}$	$\mathcal{S}_{GM}$	
Iris	2.667	6.915	3.333	7.581	1.000
Ionosphere	**8.788**	**9.086**	15.152	12.016	0.002
Monks 1	**19.286**	**12.178**	31.667	11.817	2.365×10^{-4}
Monks 2	**19.762**	**10.559**	29.762	9.395	3.052×10^{-5}
Monks 3	26.190	9.807	27.619	12.118	0.842
Glass	25.714	14.236	33.333	17.729	0.024

Table 14: Test unclassified error comparison between the proposed algorithm and example-centric neuron labeling.

	EC-WC		EC		p-value
	$\mathcal{E}_{GU}$	$\mathcal{S}_{GU}$	$\mathcal{E}_{GU}$	$\mathcal{S}_{GU}$	
Iris	0.000	0.000	0.667	3.651	1.000
Ionosphere	0.000	0.000	0.303	1.660	1.000
Monks 1	0.000	0.000	1.190	3.294	0.125
Monks 2	0.000	0.000	0.714	2.179	0.250
Monks 3	0.000	0.000	0.476	1.812	0.500
Glass	0.000	0.000	1.429	4.359	0.250

Table 15: Test unclassified error comparison between the proposed algorithm and weight-centric neuron labeling.

	EC-WC		WC		p-value
	$\mathcal{E}_{GU}$	$\mathcal{S}_{GU}$	$\mathcal{E}_{GU}$	$\mathcal{S}_{GU}$	
Iris	0.000	0.000	0.000	0.000	N/A
Ionosphere	0.000	0.000	0.000	0.000	N/A
Monks 1	0.000	0.000	0.000	0.000	N/A
Monks 2	0.000	0.000	0.000	0.000	N/A
Monks 3	0.000	0.000	0.000	0.000	N/A
Glass	0.000	0.000	0.000	0.000	N/A

Table 16: Unlabeled neuron percentage comparison between the proposed algorithm and example-centric neuron labeling.

	EC-WC		EC		p-value
	$\mathcal{E}_{U}$	$\mathcal{S}_{U}$	$\mathcal{E}_{U}$	$\mathcal{S}_{U}$	
Iris	**0.000**	**0.000**	19.467	3.104	1.863×10^{-9}
Ionosphere	**0.000**	**0.000**	3.810	1.912	3.725×10^{-9}
Monks 1	**0.000**	**0.000**	43.469	3.065	1.863×10^{-9}
Monks 2	**0.000**	**0.000**	49.807	1.885	1.863×10^{-9}
Monks 3	**0.000**	**0.000**	32.700	4.967	1.863×10^{-9}
Glass	**0.000**	**0.000**	8.241	3.961	7.451×10^{-9}

Table 17: Unlabeled neuron percentage comparison between the proposed algorithm and weight-centric neuron labeling.

	EC-WC		WC		p-value
	$\mathcal{E}_{U}$	$\mathcal{S}_{U}$	$\mathcal{E}_{U}$	$\mathcal{S}_{U}$	
Iris	0.000	0.000	0.000	0.000	N/A
Ionosphere	0.000	0.000	0.000	0.000	N/A
Monks 1	0.000	0.000	0.000	0.000	N/A
Monks 2	0.000	0.000	0.000	0.000	N/A
Monks 3	0.000	0.000	0.000	0.000	N/A
Glass	0.000	0.000	0.000	0.000	N/A

amples no worse than basic example-centric labeling, and had better classification performance than normal weight-centric labeling. At the same time, example-centric labeling with weight-centric finalization clearly maintained a higher percentage of labeled neurons than example-centric neuron labeling.

The proposed algorithm underperformed in terms of training misclassification error on the single data set with injected noise (the third monk's problem), when compared to weight-centric neuron labeling. This type of performance shortfall is, however, not clear when considering test error. A more thorough investigation into the performance of the algorithms in the presence of noise is deferred to future studies.

6 CONCLUSIONS

This paper introduces a novel supervised neuron labeling algorithm that successfully combines the advantages of both example-centric and weight-centric neuron labeling. If only label quality is important, the proposed technique is largely equivalent to example-centric labeling, with the latter being preferable due to a less complex algorithm. However, if humans are to analyze a SOM's labels, the proposed hybrid approach is preferable because it maintains a fully labeled map while offering high-quality labels.

Future work will focus on alternative ways in which high-quality complete SOM labelings can be achieved. Three approaches are under consideration:

- An approach (Li and Eastman, 2006) that has been briefly proposed, but not experimentally explored, characterizes each unlabeled neuron using the mean distance between the weight vector of the unlabeled neuron and the weight vectors of neuron groups that are labeled with the same class. The unlabeled neuron receives the label of the class with the smallest mean distance.
- The propagation of labels from characterized neurons to adjacent uncharacterized neurons will be investigated. A possible basis for this is neuron proximity graphs (Herrmann and Ultsch, 2007).
- Semi-supervised SOMs learn the distribution of classification attributes across the map, without biasing training (Kiviluoto and Bergius, 1997). Labels will be based on the learned classification attributes for each neuron.

Finally, the general performance characteristics of the supervised labeling algorithms will be investigated in greater depth. The scalability of supervised labeling algorithms to very large datasets and maps will be of interest. It will also be informative to analyze the performance characteristics of the algorithms in the presence of classification and attribute noise.

REFERENCES

Azcarraga, A., Hsieh, M.-H., Pan, S.-L., and Setiono, R. (2008). Improved SOM labeling methodology for data mining applications. In *Soft Computing for Knowledge Discovery and Data Mining*, pages 45–75. Springer. doi:10.1007/978-0-387-69935-6_3.

Bishop, C. M. (2006). *Pattern Recognition and Machine Learning*. Springer.

Bowen, F. and Siegler, J. (2024). Self-organizing maps: A novel approach to identify and map business clusters. *Journal of Management Analytics*, 11(2):228–246. doi:10.1080/23270012.2024.2306628.

Corradini, A. and Gross, H.-M. (1999). A hybrid stochastic-connectionist architecture for gesture recognition. In *Proceedings of ICIIS*, pages 336–341. doi:10.1109/ICIIS.1999.810286.

Deboeck, G. (1999). Public domain vs. commercial tools for creating neural self-organizing maps. *PC AI*, 13(1):27–33.

Engelbrecht, A. P. (2007). *Computational Intelligence: An Introduction*. Wiley, 2nd edition. doi:10.1002/9780470512517.

Fisher, R. A. (1936). Iris data set. UCI Machine Learning Repository. doi:10.24432/C56C76.

Franken, N. (2009). Visual exploration of algorithm parameter space. In *Proceedings of CEC*, pages 389–398. doi:10.1109/CEC.2009.4982973.

Galvan, D., Effting, L., Cremasco, H., and Conte-Junior, C. A. (2021). The spread of the COVID-19 outbreak in Brazil: An overview by Kohonen Self-Organizing Map networks. *Medicina*, 57(3):1–19. doi:10.3390/medicina57030235.

German, B. (1987). Glass identification data set. UCI Machine Learning Repository. doi:10.24432/C5WW2P.

Han, J., Kamber, M., and Pei, J. (2012). *Data Mining: Concepts and Techniques*. Morgan Kaufmann, 3rd edition. doi:10.1016/C2009-0-61819-5.

Herrmann, L. and Ultsch, A. (2007). Label propagation for semi-supervised learning in Self-Organizing Maps. In *Proceedings of WSOM*. doi:10.2390/biecoll-wsom2007-113.

Javed, A., Rizzo, D. M., Lee, B. S., and Gramling, R. (2024). SOMTimeS: Self organizing maps for time series clustering and its application to serious illness conversations. *Data Mining and Knowledge Discovery*, 38:813–839. doi:10.1007/s10618-023-00979-9.

Kaski, S., Kangas, J., and Kohonen, T. (1998). Bibliography of Self-Organizing Map (SOM) papers: 1981–1997. *Neural Computing Surveys*, 1:102–350.

Kiviluoto, K. and Bergius, P. (1997). Analyzing financial statements with the Self-Organizing Map. In *Proceedings of WSOM*, pages 362–367.

Kohonen, T. (1982). Self-organizing formation of topologically correct feature maps. *Biological Cybernetics*, 43(1):59–69. doi:10.1007/BF00337288.

Kohonen, T. (1989). *Self-Organization and Associative Memory*. Springer, 3rd edition. doi:10.1007/978-3-642-88163-3.

Kohonen, T. (2001). *Self-Organizing Maps*. Springer-Verlag, 3rd edition. doi:10.1007/978-3-642-56927-2.

Kohonen, T., Hynninen, J., Kangas, J., and Laaksonen, J. (1996). SOM_PAK: The Self-Organizing Map program package. Technical Report A31, Helsinki University of Technology.

Lagus, K. and Kaski, S. (1999). Keyword selection method for characterizing text document maps. In *Proceedings of ICANN*, volume 1, pages 371–376. doi:10.1049/cp:19991137.

Li, Z. and Eastman, J. R. (2006). The nature and classification of unlabelled neurons in the use of Kohonen's Self-Organizing Map for supervised classification. *Transactions in GIS*, 10(4):599–613. doi:10.1111/j.1467-9671.2006.01014.x.

Miller, Jr, R. G. (1981). *Simultaneous Statistical Inference*. Springer-Verlag, 2nd edition. doi:10.1007/978-1-4613-8122-8.

Oja, M., Kaski, S., and Kohonen, T. (2003). Bibliography of Self-Organizing Map (SOM) papers: 1998–2001 addendum. *Neural Computing Surveys*, 3:1–156.

Pöllä, M., Honkela, T., and Kohonen, T. (2009). Bibliography of Self-Organizing Map (SOM) papers: 2002-2005 addendum. Technical Report TKK-ICS-R23, Helsinki University of Technology.

Rauber, A. and Merkl, D. (1999). The SOMLib digital library system. In *Proceedings of ECDL*, pages 323–342. doi:10.1007/3-540-48155-9_21.

Rosa, A. H., Stubbings, W. A., Akinrinade, O. E., Gontijo, E. S. J., and Harrad, S. (2024). Neural network for evaluation of the impact of the UK COVID-19 national lockdown on atmospheric concentrations of PAHs and PBDEs. *Environmental Pollution*, 341:122794. doi:10.1016/j.envpol.2023.122794.

Samarasinghe, S. (2007). *Neural Networks for Applied Sciences and Engineering: From Fundamentals to Complex Pattern Recognition*. Auerbach Publications, Boca Raton, Florida, United States of America. doi:10.1201/9780849333750.

Serrano-Cinca, C. (1996). Self organizing neural networks for financial diagnosis. *Decision Support Systems*, 17(3):227–238. doi:10.1016/0167-9236(95)00033-X.

Sigillito, V., Wing, S., Hutton, L., and Baker, K. (1989). Ionosphere data set. UCI Machine Learning Repository. doi:10.24432/C5W01B.

Sobol', I. M. (1967). On the distribution of points in a cube and the approximate evaluation of integrals. *USSR Computational Mathematics and Mathematical Physics*, 7(4):86–112. doi:10.1016/0041-5553(67)90144-9.

Su, M.-C., Liu, T.-K., and Chang, H.-T. (1999). An efficient initialization scheme for the self-organizing feature map algorithm. In *Proceedings of IJCNN*, volume 3, pages 1906–1910. doi:10.1109/IJCNN.1999.832672.

van Heerden, W. S. (2017). Self-organizing feature maps for exploratory data analysis and data mining: A practical perspective. Master's thesis, University of Pretoria.

van Heerden, W. S. (2023). Automatic distance-based interpolating unit detection and pruning in self-organizing maps. In *Proceedings of SSCI*, pages 1298–1303. doi:10.1109/SSCI52147.2023.10372025.

van Heerden, W. S. (2024). SOMLib. https://github.com/wvheerden/SOMLib.

van Heerden, W. S. and Engelbrecht, A. P. (2008). A comparison of map neuron labeling approaches for unsupervised self-organizing feature maps. In *Proceedings of IJCNN*, pages 2139–2146. doi:10.1109/IJCNN.2008.4634092.

van Heerden, W. S. and Engelbrecht, A. P. (2012). Unsupervised weight-based cluster labeling for self-organizing maps. In *Proceedings of WSOM*, pages 45–54. doi:10.1007/978-3-642-35230-0_5.

van Heerden, W. S. and Engelbrecht, A. P. (2016). An investigation into the effect of unlabeled neurons on Self-Organizing Maps. In *Proceedings of SSCI*. doi:10.1109/SSCI.2016.7849938.

Vantyghem, A. N., Galvin, T. J., Sebastian, B., O'Dea, C., Gordon, Y. A., Boyce, M., Rudnick, L., Polsterer, K., Andernach, H., Dionyssiou, M., Venkataraman, P., Norris, R., Baum, S., Wang, X. R., and Huynh, M. (2024). Rotation and flipping invariant self-organizing maps with astronomical images: A cookbook and application to the VLA Sky Survey QuickLook images. *Astronomy and Computing*, 47:100824. doi:10.1016/j.ascom.2024.100824.

Wilcoxon, F. (1945). Individual comparisons by ranking methods. *Biometrics Bulletin*, 1(6):80–83. doi:10.2307/3001968.

Wnek, J. (1993). Monk's problems data sets. UCI Machine Learning Repository. doi:10.24432/C5R30R.