A Population Based Convergence Criterion for Self-Organizing Maps

Lutz Hamel and Benjamin Ott
Department of Computer Science and Statistics,
University of Rhode Island,
Kingston, RI 02881,
USA.
Email: hamel@cs.uri.edu or benott@my.uri.edu

***Abstract*—Self-organizing maps are a type of artificial neural network extensively used as a data mining and analysis tool in a broad variety of fields including bioinformatics, financial analysis, signal processing, and experimental physics. They are attractive because they provide a simple yet effective algorithm for data clustering and visualization via unsupervised learning. A fundamental question regarding self-organizing maps is the question of convergence or how well the map models the data after training. Here we introduce a population based convergence criterion: the neurons of the map represent one population and the training data represents another population. The map is said to be converged if the neuron and the training data populations appear to be drawn from the same probability distribution. This can easily be tested with standard two-sample tests. This paper develops the underpinnings of this approach and then applies this new convergence criterion to real-world data sets. We demonstrate that our convergence criterion can be considered an appropriate model selection criterion.**

I. Introduction

Self-organizing maps (SOMs) are a type of artificial neural network extensively used as a data mining and analysis tool in a broad variety of fields including bioinformatics, financial analysis, signal processing, and experimental physics [9]. The straight forward nature of the SOM training algorithm and the way in which the visualization of a SOM can be easily and intuitively interpreted make it appealing as an analysis tool. However, as with any analysis tool, and especially with competitive learning-based tools, questions pertaining to the reliability and the convergence of the tool naturally emerge. Here we view convergence as a measure of how well a model represents the underlying data space. In SOMs, convergence, and therefore the quality of the produced visualization, critically depends on the number of neurons selected and the number of training iterations applied to those neurons. If we view SOMs as models of the training data then any convergence criterion essentially becomes a model selection criterion allowing us to distinguish "good models" from "poor models."

Several measures have been developed in order to analyze the convergence of a given SOM. One such measure, the quantization error, is the error function proposed by Kohonen and is the *de facto standard* measure of the convergence of a given SOM [9]. The quantization error of a given training observation is the smallest distance between that training observation and any neuron in the SOM. The quantization error of a training set is typically the mean sum squared quantization error of all training instances. The goal of the SOM algorithm then, is to minimize this value during training. Attempting to minimize the quantization error in a radical fashion (minimize it to zero, for example) can lead to overfitted models which may be ineffective at representing the data at hand because they may end up modelling noise in the training data which is not characteristic of the general population from which the training data sample was drawn. Since one can make the quantization error arbitrarily small by increasing the complexity of the model by adding neurons to the map and by increasing the training iterations, it is clear that the quantization error does not lend itself as a model selection criterion, since it cannot answer the question "When is the quantization error good enough?"

Another approach to obtaining measurable convergence criteria is to modify the SOM training algorithm itself so that statistical measures or other objective analysis techniques can be imposed. Bishop's generative topographic mapping (GTM) [3] and Verbeek's generative self-organizing map (GSOM) [11] seem to fall into this category. The GTM and GSOM attempt to model the probability density of a data set using a smaller number of latent variables (i.e. the dimensionality of the latent space is less than or equal to that of the data space). A non-linear mapping is then generated which maps the latent space into the data space. The parameters of this mapping are learned using using an expectation-maximization (EM) algorithm. Algorithms such as the GTM and the GSOM should be viewed as alternates to the SOM as opposed to modifications of it, even though they share properties similar to the SOM. Other scholars have taken an energy function approach, imposing energy functions on the SOM and then attempting to minimize these energy functions [8, 6]. Both of these approaches, namely altering the algorithm or imposing energy functions on the SOM, seem to take away from the SOM's appeal as a simple, fast algorithm for visualizing high dimensional data, especially since the alterations tend to be much more complex than the SOM learning algorithm itself.

Yet another approach is to calculate the significance of neighborhood stabilities in order to analyze whether or not data points close together in the input space remain close when projected onto the SOM. By analyzing many maps which were trained with bootstrap samples drawn from the training data,

this approach by Cottrell *et al* [4] provides a sound set of statistical tools to analyze SOMs while leaving the original SOM algorithm unchanged. However, this stability based approach is computationally unwieldy drastically increasing the amount of time associated with the analysis of a given data set. The increased time cost is due to the creation of many maps using bootstrapped samples of the training data (typically 100-200 maps; Efron recommends using at least 200 samples when bootstrapping statistics [5]) and the analysis of each pair of training data over each map after all of the maps have been created.

In this paper, we propose a population based approach for analyzing SOM convergence. Under some minor simplifying assumptions, Yin and Allison [12] showed that, in the limit, the neurons of a SOM will converge on the probability distribution of the training data. This seems to validate Kohonen's claim that a SOM will in effect model the probability distribution of the training data [9]. Hence, a simple two-sample test can be performed in order to see if the SOM has effectively modelled the probability distribution formed by the training data or not. This population based approach lends to a fast convergence criterion, based on standard statistical methods, which does not modify the original algorithm, hence preserving its appeal as a simple and fast analysis tool.

The remainder of this paper is structured as follows. Section II describes the basic SOM training algorithm. In Section III we investigate convergence properties of this algorithm in the limit following [12]. We look at model selection and quantization error in Section IV and we introduce our population based convergence criterion in Section V. We illustrate our convergence criterion with examples in Sections VI and VII. Finally, we state our conclusions and further work in Section VIII.

II. The Basic Algorithm

Here we describe the training algorithm for SOM as introduced by Kohonon [9] following the notation used by Yin and Allison [12]. The canonical training algorithm for SOM uses a set of neurons, $\mathbf{Y}$, usually arranged in a rectangular grid formation to form topology preserving discrete mappings of an N-dimensional input space $\mathbf{X} \subset \mathbb{R}^N$. Each element $\mathbf{x} \in \mathbf{X}$ is considered a training instance and is a vector $\mathbf{x} = [x_1, x_2, \ldots, x_N]^T$ and each neuron indexed by $c \in \mathbf{Y}$ is a vector $\mathbf{w}_c(t) \in \mathbb{R}^N$ with

$$\mathbf{w}_c(t) = [w_{c,1}(t), w_{c,2}(t), \ldots, w_{c,N}(t)]^T, \tag{1}$$

where t is a time step index. Each neuron is a weight vector of the same dimensionality as the input space and the weights are adjusted at each discrete time step t during training with $t \geq 0$. At each time step t a randomly selected input vector $\mathbf{x}(t) \in \mathbf{X}$ is chosen and is used to compute a winning neuron $\omega(t)$,

$$\omega(t) = \arg\min_{c \in \mathbf{Y}} \parallel \mathbf{x}(t) - \mathbf{w}_c(t) \parallel . \tag{2}$$

Here the winning neuron $\omega(t)$ has the smallest Euclidean distance $\parallel \mathbf{x}(t) - \omega(t) \parallel$ from the training instance $\mathbf{x}(t)$. Once the winning neuron has been found the weights of the neurons on the map are updated according to the following rule,

$$\mathbf{w}_c(t+1) = \mathbf{w}_c(t) + \alpha(t)\, h_{c,\omega(t)}(t)\, [\mathbf{x}(t) - \mathbf{w}_c(t)], \tag{3}$$

for all $c \in \mathbf{Y}$. Here $\alpha(t)$ is the *learning rate* at time step t (typically a small constant with $0 < \alpha(t) < 1$ for all t decaying as t grows large) and $h_{c,\omega(t)}(t)$ is the *neighborhood function* at time step t indexed by c and the winning neuron $\omega(t)$. The neighborhood of a winning neuron, $\mathbf{N}_{\omega(t)}(t)$, is defined as a set of neurons in proximity of and centered around the winning neuron according to the grid formation of the map. The neighborhood is time step sensitive and usually starts out as a large set,

$$\mathbf{N}_{\omega(t)}(t) \approx \mathbf{Y}, \tag{4}$$

at $t = 0$ and shrinks over time,

$$\mathbf{N}_{\omega(t)}(t) = \{\omega(t)\}, \tag{5}$$

with $t \gg 0$. With this we define the neighborhood function as the step function,

$$h_{c,\omega(t)}(t) = \begin{cases} 1 \text{ if } c \in \mathbf{N}_{\omega(t)}(t) \\ 0 \text{ otherwise} \end{cases} \tag{6}$$

This implies that the neighborhood function controls which neurons on the map are updated according to the rule in (3) and which are not. This "focussing" of neuron updating is crucial in the successful training of SOMs. Training usually continues for a fixed number of time steps.

III. Probabilistic Convergence

Some observations on the training of a SOM. Training instances are drawn from the input space $\mathbf{X}$ randomly and independently and are presented to the map; at time step t we can view the randomly chosen training instances as the set,

$$\mathbf{X}(t) = \{\mathbf{x}(i) \in \mathbf{X} \mid i = 0, \ldots, t\}, \tag{7}$$

with $\mathbf{X}(t) \overset{t\to\infty}{\longrightarrow} \mathbf{X}$. At time step t, each neuron $c \in \mathbf{Y}$ is trained with a subset of instances $\mathbf{X}_c(t) \subseteq \mathbf{X}(t)$ where,

$$\bigcup_{c \in \mathbf{Y}} \mathbf{X}_c(t) = \mathbf{X}(t). \tag{8}$$

At the beginning of the training process, when the neighborhoods are still large, i.e., $\mathbf{N}_{\omega(t)}(t) \approx \mathbf{Y}$, these subsets are maximally overlapped with each other. As training progresses and the neighborhoods shrink to a single element, i.e., $\mathbf{N}_{\omega(t)}(t) = \{\omega(t)\}$, these subsets eventually become mutually separated with,

$$\mathbf{X}_c(t) \bigcap \mathbf{X}_{c'}(t) \overset{t\to\infty}{\longrightarrow} \emptyset, \tag{9}$$

for all $c, c' \in \mathbf{Y}$ and $c \neq c'$. Furthermore, as time tends to infinity we have,

$$\mathbf{X}_c(t) \overset{t\to\infty}{\longrightarrow} \mathbf{X}_c, \tag{10}$$

the *final* subsets.

Now, let $p(\mathbf{x})$ be the probability density function over the input space $\mathbf{X}$, then the probability of a training instance $\mathbf{x}(t)$ belonging to a subset $\mathbf{X}_c(t)$ is,

$$P(\mathbf{X}_c(t)) = \int_{\mathbf{x}\in\mathbf{X}_c(t)} p(\mathbf{x})d\mathbf{x}, \tag{11}$$

for all $c \in \mathbf{Y}$. As $t \to \infty$ this becomes,

$$P(\mathbf{X}_c) = \int_{\mathbf{x}\in\mathbf{X}_c} p(\mathbf{x})d\mathbf{x}. \tag{12}$$

Yin and Allison [12] have shown under some mild assumptions that for a given map, $\mathbf{Y}$, the neurons will converge in the limit on the centroids $\mathbf{m}_c$ of the final subsets $\mathbf{X}_c$,

$$\mathbf{w}_c(t) \overset{t\to\infty}{\longrightarrow} \mathbf{m}_c = \frac{1}{P(\mathbf{X}_c)} \int_{\mathbf{x}\in\mathbf{X}_c} \mathbf{x}\, p(\mathbf{x})d\mathbf{x}, \tag{13}$$

for all $c \in \mathbf{Y}$.

IV. Model Selection and Quantization Error

We define the *quantization error* for a map $\mathbf{Y}$ at time step t as,

$$E_{\mathbf{Y}}(t) = \sum_{c\in\mathbf{Y}} \sum_{\mathbf{x}\in\mathbf{X}_c(t)} \parallel \mathbf{w}_c(t) - \mathbf{x} \parallel^2, \tag{14}$$

and it is easy to see given (13) that the learning algorithm converges on the minimal quantization error as time tends to infinity,

$$E_{\mathbf{Y}}(t) \overset{t\to\infty}{\longrightarrow} E_{\mathbf{Y}} = \sum_{c\in\mathbf{Y}} \sum_{\mathbf{x}\in\mathbf{X}_c} \parallel \mathbf{m}_c - \mathbf{x} \parallel^2 . \tag{15}$$

If we view map construction as a model building process and if we view the quantization error as a model selection criterion (as suggested in [9]) we run into problems in that optimality is defined only in the limit. There is no statistical measure on the quantization error that suggests when a quantization error is "good enough." To complicate things even further, adding neurons to the map and rerunning the training algorithm will likely reduce the quantization error because now the algorithm will split the training set into a larger number of final subsets (one per neuron) with fewer training instances in them which will give rise to a lowered quantization error. Again, no statistical measure based on the quantization error exists that would suggest an optimal number of nodes.

Given what we have developed so far and assuming that the topology of the input space $\mathbf{X}$ can be projected onto two dimensions with minimal distortion, then it is possible to reduce the quantization error to zero by constructing a large enough map. To see this, let n_x be the number of elements in the input space $\mathbf{X}$ and let n_y be the number of neurons in the map $\mathbf{Y}$. First assume that the map only consists of a single neuron, $n_y = 1$. That means in the limit we have $P(\mathbf{X}_c) = P(\mathbf{X}) = 1$ and therefore the single neuron will converge on the mean $\mathbf{m}_x$ of the input space,

$$\mathbf{w}_c(t) \overset{t\to\infty}{\longrightarrow} \mathbf{m}_x = \int_{\mathbf{x}\in\mathbf{X}} \mathbf{x}\, p(\mathbf{x})d\mathbf{x}, \tag{16}$$

for the only element $c \in \mathbf{Y}$. This implies of course a large quantization error,

$$E_{\mathbf{Y}}(t) \overset{t\to\infty}{\longrightarrow} E_{\mathbf{Y}} = \sum_{\mathbf{x}\in\mathbf{X}} \parallel \mathbf{m}_x - \mathbf{x} \parallel^2 . \tag{17}$$

Now assuming that we have as many neurons in our map as there are training instances, $n_y = n_x$, and making use of our assumption that the topology of the input space can be projected onto two dimensions with minimal distortion, then our final subsets could be singleton sets $\mathbf{X}_c = \{\mathbf{x}_c\}$ for all $c \in \mathbf{Y}$where,

$$\bigcup_{c\in\mathbf{Y}} \mathbf{X}_c = \mathbf{X}, \tag{18}$$

and

$$\bigcap_{c\in\mathbf{Y}} \mathbf{X}_c = \emptyset. \tag{19}$$

In the limit, each neuron of the map will then converge on the training instance $\mathbf{x}_c$ in its final subset,

$$\mathbf{w}_c(t) \overset{t\to\infty}{\longrightarrow} \mathbf{x}_c = \frac{1}{P(\{\mathbf{x}_c\})} \int_{\{\mathbf{x}_c\}} \mathbf{x}\, p(\mathbf{x})d\mathbf{x}, \tag{20}$$

for all $c \in \mathbf{Y}$. It is easy to see that the quantization error in the limit will be zero in this case. This means that quantization error as a model selection criterion dictates that the neurons of the SOM have to model the training data precisely. But statistical theory tells us that models that fit their training data precisely are usually overfitted since by modeling training data precisely these models also model any inherent noise.

V. Population Based Convergence

As we have seen in the previous section, the quantization error is not an adequate model selection criterion because we can make the errors as small as desired by increasing the complexity of the model. It is required of a model selection criterion that it allows us to determine when a model is "good enough" which the quantization error does not allow us to determine.

However, by slightly shifting perspective, another look at equation (20) reveals something interesting about SOMs: given enough neurons the SOM training algorithm attempts to recreate the training samples. This insight allows us to formulate a new convergence criterion:

> *A SOM is converged if its neurons appear to be drawn from the same distribution as the training instances.*

This formulation naturally leads to a two-sample test [10] as a convergence criterion. We can view the training data as one sample from the probability space $\mathbf{X}$ having probability density function $p(x)$ and we can treat the neurons of the SOM as another sample. We can then test to see whether or not the two samples appear to be drawn from the same probability space. If we operate under the simplifying assumption that each of the N coordinates (or features) of the input space $\mathbf{X} \subset \mathbb{R}^N$ are normally distributed and independent of the others, we can test each of the features separately. This assumption

lends to a fast algorithm for identifying SOM convergence which is based on statistically analyzing similarites between the features as expressed by the training data and as expressed by the neurons in the SOM. We define a feature as converged if the variance and the mean of that feature appear to be drawn from the same distribution for both the training data and the neurons. If all the features are converged then we say that the map is converged.

The following is the formula for the $(1-\alpha)*100\%$ confidence interval for the ratio of the variances from two random samples [10],

$$\frac{s_1^2}{{s_2}^2}\cdot\frac{1}{f_{\frac{\alpha}{2},n_1-1,n_2-1}}<\frac{\sigma_1^2}{\sigma_2^2}<\frac{s_1^2}{{s_2}^2}\cdot f_{\frac{\alpha}{2},n_1-1,n_2-1}, \qquad (21)$$

where s_1^2 and s_2^2 are the values of the variance from two random samples of sizes n_1 and n_2 respectively, and where $f_{\frac{\alpha}{2},n_1-1,n_2-1}$ is an F distribution with n_1-1 and n_2-1 degrees of freedom. To test for SOM convergence, we let s_1^2 be the variance of a feature in the training data and we let s_2^2 be the variance of that feature in the neurons of the map. Furthermore, n_1 is the number of training samples (i.e. the cardinality of the training data set) and n_2 is the number of neurons in the SOM. We say that the variance of a particular feature has converged (or appears to be drawn from the same probability space) if 1 lies in the confidence interval denoted by equation (21), that is, the ratio of the underlying variance as modeled by input space and the neuron space, respectively, is approximately equal to one, $\sigma_1^2/\sigma_2^2\approx 1$, up to the confidence interval.

In the case where $\bar{x}_1$ and $\bar{x}_2$ are the values of the means from two random samples of size n_1 and n_2, and the known variances of these samples are σ_1^2 and σ_2^2 respectively, the following formula provides $(1-\alpha)*100\%$ confidence interval for the difference between the means [10],

$$\mu_1-\mu_2 \quad > \quad (\bar{x}_1-\bar{x}_2)-z_{\frac{\alpha}{2}}\cdot\sqrt{\frac{\sigma_1^2}{n_1}+\frac{\sigma_2^2}{n_2}}, \qquad (22)$$

$$\mu_1-\mu_2 \quad < \quad (\bar{x}_1-\bar{x}_2)+z_{\frac{\alpha}{2}}\cdot\sqrt{\frac{\sigma_1^2}{n_1}+\frac{\sigma_2^2}{n_2}}. \qquad (23)$$

We will say that the mean of a particular feature has converged if 0 lies in the confidence interval denoted by equations (22) and (23). That is, we say the mean of a particular feature has converged if the difference of the means as modeled by the input space and the neuron space, respectively, is approximately equal to zero, $\mu_1-\mu_2\approx 0$, up to the confidence interval.

We will say that a SOM has converged on a feature, or that a feature has converged, if both the mean and variance converged in accordance with the above criteria. We can then form a measure for SOM convergence as follows for N features,

$$convergence=\frac{\sum_{i=1}^{N}\rho_i}{N}, \qquad (24)$$

where

$$\rho_i=\begin{cases}1 & \text{if feature } i \text{ has converged,}\\ 0 & \text{otherwise.}\end{cases}$$

The convergence score (24) proposed here is essentially a fraction of the number of features which actually converged (i.e. whose mean and variance were adequately modelled by the neurons in the SOM).

VI. Example Using Iris Data

The convergence measure proposed in the previous section was applied to the Fisher / Anderson iris data set [1] using 95% confidence intervals for both the mean and the variance tests. In the following plots fConv represents the convergence measure as defined in the previous sections, ssMean is the quantization error as defined in (14), and iterations represents the number of iterations that the SOM training algorithm was run. Each data point represents the data associated with a unique randomly initialized SOM which was trained for the number of iterations indicated on the plot. For instance, when fConv is plotted against iterations (Figure 1), the data points on the map, which are connected via lines for visualization purposes, represent the convergence score for unique, randomly initialized SOMs which were trained for the indicated number of iterations.

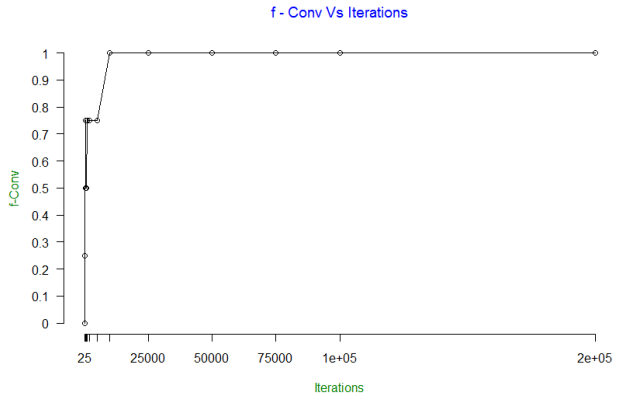

Fig. 1: Convergence measure, fConv, plotted as a function of iterations.

In Figure 1, we can see that the our convergence measure increases as the amount of training increases. This is expected for any valid convergence criterion. We can also see that SOM has fully converged after about 20,000 iterations. Figure 2 shows the convergence measure related to the quantization error. We can see that as the convergence score increases, the quantization error decreases. Most interesting is the fact that fully converged maps (points at the bottom right) do not necessarily have a quantization error of zero making our convergence score a suitable model selection criterion as the neurons in these converged maps adequately model the input data without overfitting. This also implies that if the convergence score falls substantially short of the value 1, then

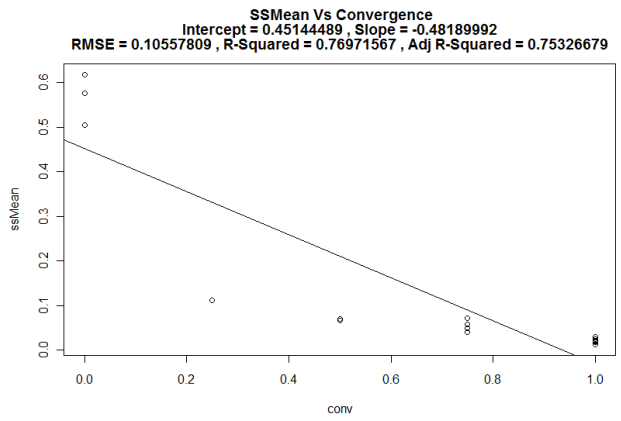

Fig. 2: ssMean plotted as a linear function of fConv.

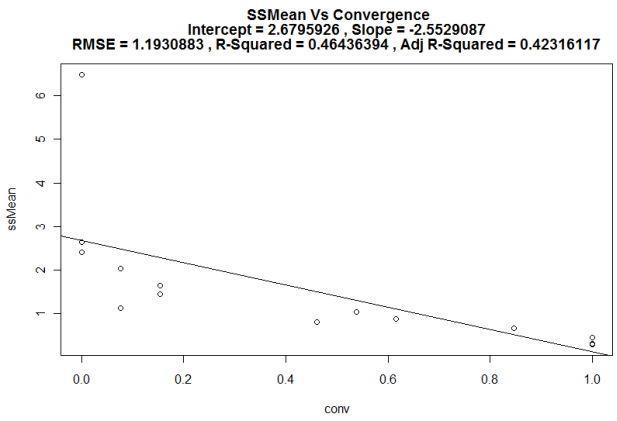

Fig. 4: ssMean plotted as a linear function of fConv.

we can improve the models by either training longer, adding neurons, increasing the learning rate, or all of the above. Each of these steps enables the SOM to more easily capture the variance of the training data, thereby increasing the possibility of convergence.

VII. Example Using Wine Data

In this section, our convergence measure was applied to Stefan Aeberhard's wine data set [2] using 95% confidence intervals for both the mean and the variance tests. The data was normalized before training the SOM. This data has thirteen dimensions and consists of chemical characteristics for three types of wine and we would expect three clusters to be shown on converged maps. Again as above, in the following plots fConv represents the convergence measure as defined in the previous sections, ssMean is the quantization error as defined in (14), and iterations represents the number of iterations that the SOM algorithm was run. Each data point represents the data associated with a unique randomly initialized SOM which was trained for the number of iterations indicated by the data point.

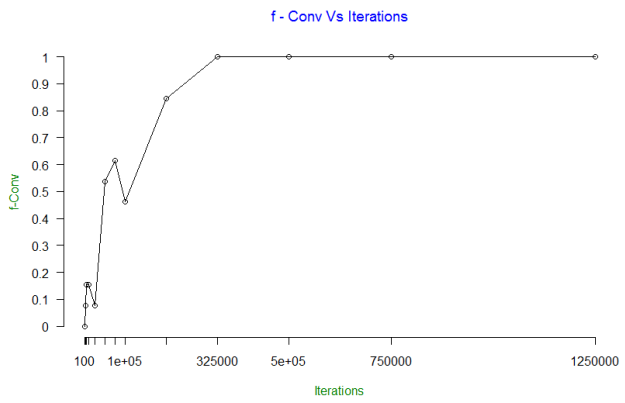

Fig. 3: Convergence measure, fConv, plotted as a function of iterations.

In Figure 3, we can see that, as expected, our convergence score trends upwards as the amount of training increases. Note also that this occurs even though each of the maps are randomly initialized. We can also see that maps fully converge after about 325,000 iterations. Figure 4 shows us that the convergence measure is inversely related to the quantization error. We can see that as convergence increases, the quantization error decreases. As before, fully converged maps (points at the bottom right) do not necessarily have a quantization error of zero. The neurons in these maps adequately model the input data without overfitting.

In Figures 5 and 6 we present two SOMs constructed using the wine data set, one with a low convergence score and one with a high convergence score, respectively.

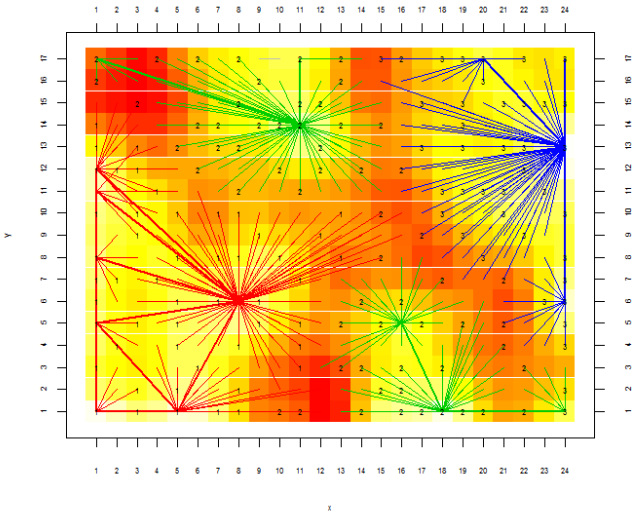

Fig. 5: Map trained using the wine data set for 5,000 iterations achieving a convergence score of 15.8%.

The map in Figure 5 has a convergence score of about 15% after 5,000 training iterations and one can easily see from the

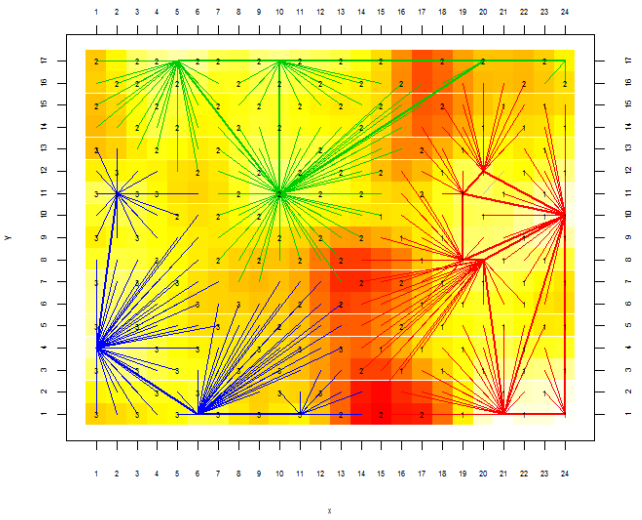

Fig. 6: Map trained using the wine data set for 500,000 iterations achieving a convergence score of 100%.

map that the cluster with elements '2' was broken into two parts: one part can be seen in the top left corner and the other in the bottom right. This is contrary to our expectation of being able to identify three contiguous clusters. In Figure 6 the map achieved a 100% score with 500,000 training iterations and produced the expected clusters. Here we can identify all three contiguous clusters. The cluster representations were created using a slightly modified version of the connected components approach as given in [7]. As expected, these maps seem to indicate that the quality of a map is directly correlated to its convergence score: the higher the convergence score, the better the map. That means that our convergence score is an appropriate model selection criterion as desired.

VIII. Conclusion and Further Work

Self-organizing maps are a popular data analysis and visualization tool. However, attempts to provide a convergence criterion for SOMs either resulted in a modified training algorithm or computationally complex constructions. In the case of the quantization error we have shown that it is not suited to be considered a convergence criterion. Here we presented an efficient alternative that treats the neurons as a data sample and convergence of the SOM is established if the neuron sample appears to be drawn from the same distribution as the training data. This two-sample test can be efficiently computed based on the features of the training data. Our examples demonstrated that our convergence criterion is inversely related to the quantization error; convergence increases as quantization error decreases. However, convergence does not necessarily imply a zero quantization error which means that our convergence criterion avoids the overfitting tendencies of quantization error based modeling approaches. Furthermore, our examples seem to indicate that the quality of the maps produced is directly correlated to our convergence score making it an appropriate model selection criterion.

Our next step is to compare our convergence criterion to established convergence criteria such as Cottrell *et al*'s stability measure [4]. Some preliminary studies are encouraging in that our convergence criterion always implies Cottrell's stability criterion. Our goal is to incorporate this new convergence criterion into a comprehensive SOM toolkit R package under development at the University of Rhode Island to be made available to the public.

References

[1] UCI machine learning repository: Iris data set. http://archive.ics.uci.edu/ml/datasets/Iris.

[2] UCI machine learning repository: Wine data set. http://archive.ics.uci.edu/ml/datasets/Wine.

[3] C. Bishop, M. Svensn, and C. Williams. Gtm: A principled alternative to the self-organizing map. *Artificial Neural NetworksICANN 96*, pages 165–170.

[4] M. Cottrell, E. De Bodt, and M. Verleysen. A statistical tool to assess the reliability of self-organizing maps. *Advances in self-organising maps*, page 714, 2001.

[5] B. Efron and R. J. Tibshirani. *An Introduction to the Bootstrap*. Chapman & Hall, New York, 1993.

[6] E. Erwin, K. Obermayer, and K. Schulten. Self-organizing maps: ordering, convergence properties and energy functions. *Biological cybernetics*, 67(1):47–55, 1992.

[7] L. Hamel and C.W. Brown. Improved interpretability of the unified distance matrix with connected components.

[8] T. Heskes. Energy functions for self-organizing maps. *Kohonen maps*, page 303316.

[9] T. Kohonen. *Self-organizing maps*. Springer series in information sciences. Springer, 2001.

[10] Irwin Miller and Marylees Miller. *John E. Freund's Mathematical Statistics with Applications (7th Edition)*. Prentice Hall, 7 edition, October 2003.

[11] J. J. Verbeek and N. Vlassis. The generative self-organizing map.

[12] H. Yin and N. M. Allinson. On the distribution and convergence of feature space in self-organizing maps. *Neural computation*, 7(6):1178–1187, 1995.