M A S T E R A R B E I T

Visualisations for Comparing Self-organising Maps

ausgeführt am Institut für
Softwaretechnik und Interaktive Systeme
der Technischen Universität Wien

unter der Anleitung von
ao.Univ.Prof. Dipl.-Ing. Dr.techn. Andreas Rauber

durch

Doris Baum
Matrikelnummer: 0426038
Friedrich-Rückert-Weg 10
90547 Stein
Deutschland

Wien, am 16. März 2007 ______________________

Acknowledgements

I'd like to thank Andreas Rauber for being the best supervisor I could have wished for, always encouraging, always envisioning new possibilities, and always asking for one more check-box in the GUI.
I thank Thomas Lidy, Rudolf Mayer, Georg Pölzlbauer, Robert Neumayer, and Angela Roiger for indispensable help, encouragement, and lunches.

I thank my parents for their unquestioning support,
both financial and emotional,
and all my friends for their patience and understanding,
especially Florian Ruckeisen and Judith Oliva.

Zusammenfassung

Self-organising Maps (SOMs) sind ein sehr nützliches Werkzeug um große Datensammlungen zu untersuchen und zu analysieren: Sie projizieren hochdimensionale Daten in einen niedrigdimensionalen Raum, so dass die Daten leichter von Menschen analysiert werden können als in ihrer ursprünglichen Form. Zum Zweck der Analyse existieren viele Visualisierungen, die verschiedene Aspekte und Eigenschaften der SOMs und der Daten darstellen. Es gibt allerdings sehr wenige Visualisierungen um zwei oder mehr SOMs direkt mit einander zu vergleichen. Um dies zu beheben stellt diese Arbeit drei Visualisierungen vor, die SOMs vergleichen, die auf dem gleichen Datenset mit unterschiedlichen Parametern trainiert wurden. Damit soll herausgefunden werden, wo die Daten jeweils auf den Karten zu liegen kommen und die Stabilität und Qualität der Projektionen eingeschätzt werden.

Abstract

Self-organising Maps (SOMs) are a very useful method for exploring and analysing large data collections: They project high-dimensional data into a low-dimensional output space so that it is easier to analyse for humans than the original data. For the purpose of analysis, plenty of visualisations exist which display different aspects and properties of the maps and the data. There are, however, very few visualisations for directly comparing two or more SOMs with each other. This work tries to rectify that by introducing three visualisations to compare SOMs trained on the same dataset with different parameters, to find out where the data comes to lie on each map and assess the stability and quality of the projections.

Contents

Chapter 1

Introduction

Self-organising Maps (SOMs) are a very useful method for data exploration: They project high-dimensional data onto a low-dimensional output space, a two-dimensional grid, so that it can be more easily analysed by humans than the original data. For the purpose of analysis, plenty of visualisations exist which display different aspects and properties of the maps and the data. There are, however, very few approaches for directly comparing two or more SOMs with each other. This work tries to rectify that by introducing three visualisations to compare SOMs trained on the same dataset with different parameters. Two of them display where the data comes to lie on each map, showing the differences and similarities of two maps, while the third compares "arbitrarily many" SOMs with each other (where the number is of course limited by computing power). All three methods can be used to asses the stability and quality of the projection the SOM performs, as well as the influence of the parameters used. The visualisations were implemented in an existing framework for SOM training and visualisation, the SOMViewer project, and experiments on synthetic and real-world datasets were conducted. The tests lead to the conclusion that the proposed visualisations can indeed be useful in data exploration.
Chapter 2 gives an introduction to Self-organising Maps and shortly describes related work: SOM visualisations, cluster analysis on SOMs, quality measures, and comparing multiple SOMs. Chapter 3 explains the visualisations introduced in this work, while Chapter 4 describes the datasets used in the experiments. The actual experiments are detailed in Chapter 5, a few possible improvements of the visualisations are proposed in Chapter 6, and Chapter 7 closes this work.

Chapter 2

Self-organising Maps (SOMs)

2.1 Introduction

Self-organising Maps (SOMs) were introduced by Teuvo Kohonen [5], [6], therefore sometimes also called Kohonen maps. They are a type of artificial neural network and an unsupervised machine learning method (see, for example, [2] for a detailed explanation of these terms).
SOMs serve as an instrument for topology-preserving, low-dimensional visualisation of high-dimensional data, similar to Multidimensional Scaling [7]. That is, they project data from a high-dimensional input space onto a low-dimensional (usually 2-dimensional) output space, trying to preserve the closeness relations of the data samples. Visualisations of the resulting output space can be very useful in data exploration to get an impression of the properties of the data.
SOMs consist of artificial neurons placed in a low-dimensional (usually 2-dimensional because this is easiest to visualise) grid with a rectangular or hexagonal lattice. Throughout the experiments in this work, a rectangular grid is used. Through their position on the grid, the neurons' distance relation is defined like on a map; for example, adjacent units on the grid are considered "close", whereas units in opposing corners of the grid are considered "far apart".
The units are represented by prototype vectors m_k in the input space. Prior to the training phase, the units' prototype vectors are initialised to random values in many SOM implementations. To shorten the training phase, the prototype vectors may also be initialised to non-random values constructed from the data.
During the training phase, a simple training step is repeated many times to make the map of prototype vectors fit the input data closely, so as to give a good projection of the data:

1. Select a data sample $x(t)$ from the input data, for example randomly.

2. Find the prototype vector c closest to the data sample. The unit it represents is called the Best Matching Unit (BMU) for that data sample.

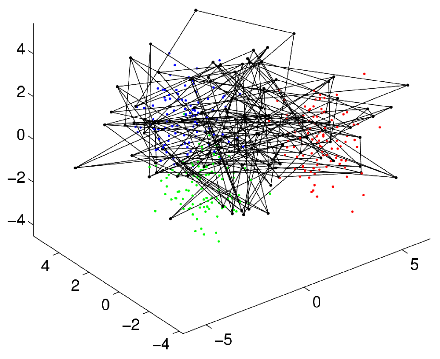


Figure 2.1: A SOM's prototype vectors in the input space before training.

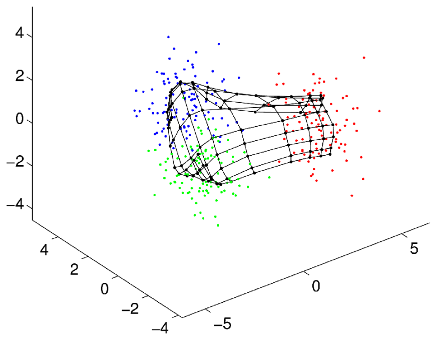


Figure 2.2: A SOM's prototype vectors in the input space after training.

3. Move all prototype vectors towards the data sample, proportionally to their closeness to the BMU. That is, if a unit is close to the BMU, move its prototype vector more than if the unit is far away from the BMU. The rule for the new position of the prototype vectors is

$$m_k(t+1) = m_k(t) + \alpha(t)h_{ck}(t)(x(t) - m_k(t)) \tag{2.1}$$

   where $m_k(t+1)$ is the new position of the prototype vector (at step $t+1$), $m_k(t)$ is the old position (at step t), $x(t)$ is the selected data sample, $a(t)$ is the learning rate and $h_{ck}(t)$ is a neighbourhood kernel around BMU c. The neighbourhood kernel defines the measure for the units' "closeness" to the BMU; often, a Gaussian kernel depending on the grid lattice distance to the BMU is used.

   Both learning rate and neighbourhood kernel decrease monotonically with time, so that learning starts quick and on a global scale and gradually becomes slower and more local. The SOMViewer software used in this work to generate and analyse SOMs uses negative exponential functions to determine the current learning rate and size of neighbourhood.

Through the training, the SOM folds onto the data like a flexible net, giving a 2-dimensional projection of the data on the grid, trying to preserve topological properties of the data. Because of the neighbourhood kernel, similar prototype vectors end up in each other's neighbourhood, so similar data vectors are mapped onto units close to each other on the grid. Thus, similarity is mirrored by closeness in a SOM.
Figures 2.1 and 2.2 show a SOM's prototype vectors and their grid in the input data space (the black dots and lines), together with the input data (the red, green, and blue dots, each colour marks a class). Both figures were produced with the Matlab

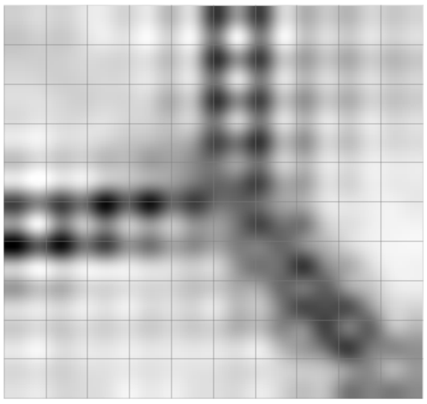
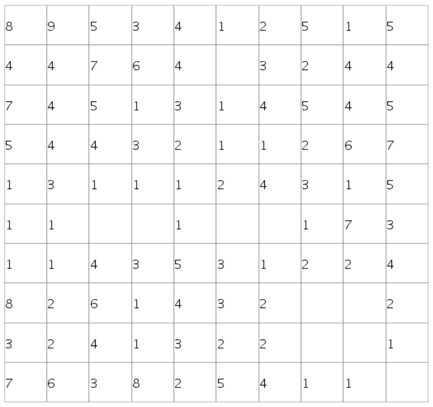


Figure 2.3: Example of a U-matrix.



8	9	5	3	4	1	2	5	1	5
4	4	7	6	4		3	2	4	4
7	4	5	1	3	1	4	5	4	5
5	4	4	3	2	1	1	2	6	7
1	3	1	1	1	2	4	3	1	5
1	1			1			1	7	3
1	1	4	3	5	3	1	2	2	4
8	2	6	1	4	3	2			2
3	2	4	1	3	2	2			1
7	6	3	8	2	5	4	1	1	

Figure 2.4: Example of a hit histogram.

SOM Toolbox[1]. Figure 2.1 shows the SOM randomly initialised but yet untrained. Figure 2.2 shows the same SOM after training – with the SOM adjusted to the shape of the data.

2.2 SOM Visualisations

There are numerous visualisation methods for SOMs, the basic and most frequently used ones summed up in [17].

Maybe the most prominent one is the U-matrix [14] which shows each prototype vector's/unit's euclidean distance to its neighbours in shades of grey, different hues of colour, by shape, or in some other way. An example for a U-matrix is depicted in Figure 2.3; it shows a SOM made with the SOMViewer software [3] from dataset "6" described in Section 4.1 (page 31). The dataset contains three equidistant clusters of samples taken from Gaussian distributions. The SOM has 10×10 units drawn as squares with grey borders. High distance of prototype vectors is indicated by dark colour here – there clearly is a Y-shaped black "rift" through the SOM, which separates three light areas. Looking at the data (see Figure 4.1, page 32) reveals that there are indeed three well-separated clusters. So the U-matrix can be used to discern clusters and cluster borders during data exploration.

Another often used SOM visualisation is the data histograms or hit histogram. It shows for each unit for how many data samples it was the Best Matching Unit – that is, how many hits of data vectors it scored. The number can be indicated by the size of a marker on the unit's grid cell, by colour, by 3-dimensional bar charts, or by simply printing the number into the unit's cell. Figure 2.4 is an example for

[1]Matlab: `http://www.mathworks.com/`; SOM Toolbox: `http://www.cis.hut.fi/projects/somtoolbox/`.

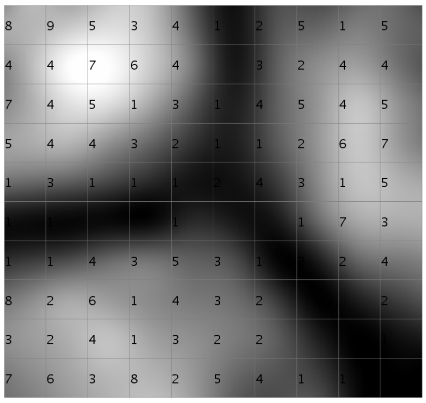


Figure 2.5: Example of a Smoothed Data Histogram with $s = 10$

a hit histogram; it was made from the same data and the same SOM as Figure 2.3. When comparing Figure 2.4 with Figure 2.3, one can see that the units in the black rift in the U-matrix get very few hits or no hits at all in the hit histogram: they are the Best Matching Unit for at most one or two data vectors. This is to be expected along the cluster borders because the input space is sparse with data there.

The Smoothed Data Histograms (SDH) visualisation [10] is an extension of the data histogram visualisation. It is used to visualise clusters in the data on the SOM. It estimates the probability density of the input data on the map by smoothing the results of the normal data histogram. Instead of assigning a data vector only to a single Best Matching Unit, it is assigned to be a member of those s units whose prototype vectors are closest to the data vector, i.e. the s Best Matching Units. The degree to which a data vector is a member of a unit is determined by the rank of closeness between the data vector and the unit's prototype vector: The membership degree is highest for the closest unit, second highest for the second closest unit, and so on. The units of the SOM are coloured in the SDH according to the summed up membership degrees of the data vectors assigned to them. Figure 2.5 shows an example of a Smoothed Data Histogram with $s = 10$; it was made from the same data and the same SOM as the examples above. The clusters already detected in the U-Matrix can also be seen in the SDH, there is a similar Y-shaped rift between the clusters. The SDH additionally shows the density of the data on the map.

Although the SOM is an unsupervised learning method and is not trained on information about different classes of data, existing class information can be used in a further visualisation. If there is information on which classes the input data vectors belong to, a class information visualisation can be drawn showing for each unit the classes of the data vectors for which it is Best Matching Unit. Figure 2.6 shows an example of this: The data vectors are mapped onto their Best Matching Units, and the units get coloured markers according to the classes of their data vectors. The data and the SOM for Figure 2.6 are the same as for the other examples, and here one can see the three different classes that make up the three clusters already

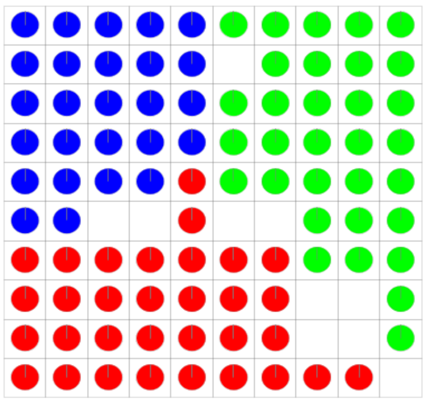


Figure 2.6: Example of class information visualisation.

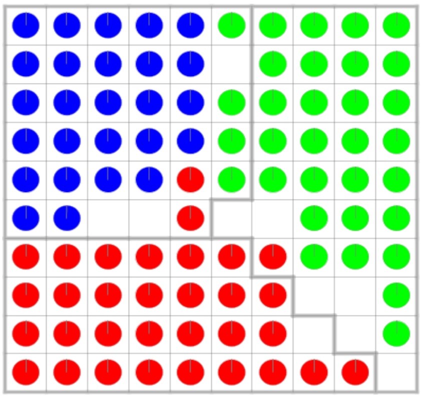


Figure 2.7: Example of SOM clustering.

detected in the other two visualisations. This, of course, is only possible if class information on the data is available, which is not necessarily the case when using a SOM for data exploration.

This section gives just a few examples of SOM visualisations but there are, of course, more. The main part of this work is dedicated to introducing three new variations of SOM visualisations, which compare two or more SOMs with each other. They are explained in Chapter 3 and tested on data in Chapter 5.

2.3 Cluster Analysis on SOMs

Two of the visualisations introduced in Chapter 3 make use of the SOMViewer's cluster analysis methods described in [12]. Cluster analysis is used to find clusters / groups / partitions in the data, which is an essential task in data mining and especially data exploration, as is explained in depth for example in [13]. The clusters found may represent the actual structure of the data or may be used to summarise or compress it. The SOM algorithm can be seen as a kind of clustering algorithm itself – a unit's prototype vector representing the cluster centre for the data vectors mapped on the unit. Thus an application of cluster analysis on a SOM is really an application of clustering on a compressed / reduced data set. This can reduce the computational load needed for clustering greatly without losing too much on clustering accuracy, compared with cluster analysis on the original data, as is described in [18].
Figure 2.7 shows a SOM class information visualisation (on the same data and SOM as the other figures) with cluster borders drawn in grey, which were obtained by applying the Ward's linkage clustering [20] and setting the number of clusters to 3. The borders overlap with the rift from Figure 2.3 and separate the classes in Figure 2.6, though not perfectly. As can be seen, cluster analysis can be used to find a real structure underlying the data.

2.4 Measuring the Quality of a SOM's Projection

A number of measures have been described for computing and comparing the quality of a SOM's projection (summed up in [11]), such as the Topographic Error, the Topographic Product [1], Trustworthiness and Neighbourhood Preservation [16], and special decompositions of the SOM Distortion Measure [19]. They all measure, in one way or another, how well the topology is preserved in the projection onto the SOM's two-dimensional grid.

The Topographic Error is a simple measure: For all input data vectors, the nearest weight vector (Best Matching Unit) and the second-nearest weight vector (Second Best Matching Unit) are computed. If they are not adjacent on the SOM-grid, this is counted as a local error. For the global topographic error measure, the number of local errors is summed up and divided by the overall number of data samples.

The Topographic Product [1] is based on two measures, P_1 and P_2, which indicate for each unit whether its k nearest neighbour units coincide, regardless of their order. P_1 assesses the input space, using the distances between the units' prototype vectors, while P_2 assesses the output space, using the euclidean grid distance between the units. Both are combined into P_3, which differs from 1 if there are violations of neighbourhood and indicates whether the dimensionality of the output space is too large ($P_3 > 1$) or too small ($P_3 < 1$). The P_3 values of the units on the map are averaged to get a global measure P for the whole map, which also indicates whether the dimensionality of the map fits or is too large or too small.
Because the Topographic Product uses distances of units and prototype vectors, the input dataset is not needed for its computation.

The Neighbourhood Preservation [16] measure is similar to the Topographic Product in that it indicates how well the neighbourhood of the k nearest neighbours in the input space is preserved in the projection onto the output space. A difference is that it works with input data vectors instead of prototype vectors. However, the more important measure introduced in the paper is the Trustworthiness of the neighbourhoods in the output space. It assesses whether the k nearest neighbours of a data vectors in the output space (the SOM-grid) are also close in the input space, thus giving an indication of the expressiveness and reliability of the output space view on the data.

The decomposition of the SOM Distortion measure introduced in [19] gives another possibility to assess the quality of the projection. The SOM Distortion measure is a local energy function of the SOM if the data set is discrete and the neighbourhood kernel is fixed; the SOM training rule approximates the Distortion measure's gradient. The decomposition introduced by Vesanto et al. splits the SOM Distortion into three parts: local data variance, neighbourhood variance, and neighbourhood bias. The local data variance gives the quantisation quality. The neighbourhood variance assesses the trustworthiness of the map topology, similar to the Trustworthiness measure from [16]. The neighbourhood bias links the other two and can be seen as the stress between them.

All of these measures can be used to compare SOMs and (to varying success) to select the SOM that best fits the data. However, none of these measures compares two SOMs with each other or directly shows their differences.

2.5 Comparing Multiple SOMs

There is – to the current knowledge of this work's author – only one approach to comparing and visualising multiple SOMs trained on the same data: The concept of Aligned Self-organising Maps, explained in [8] and [9]. It is used to train several SOMs on the same data with slightly different feature extraction parameter settings in order to explore the effects of these settings. The problem in comparing such SOMs is that the data vectors and clusters will come to lie on different areas in the individual SOMs due to different initialisation and parameters. To alleviate this problem, the Aligned SOMs are trained in such a way that each data vector can be found in the same region on different maps. This requires an adaption of the SOM training algorithm: The SOMs are stacked on top of each other, so that each SOM is a layer in the stack, and a distance measure between the layers is defined to control the smoothness of the transition between the SOMs. With this distance measure, a distance between units in different layers can be derived, which is used in training in the same way that the distance between units in one SOM is used to preserve the topology of the data. That is, for the training step a data vector and a layer are selected, the Best Matching Unit in the layer is found, and the units in the other layers are adapted according to their distance from the BMU. The units within the layer are adapted according to the normal SOM training step. The result is an Aligned SOM stack which "morphs" the first layer in the stack to the last.
However, this method can't be used on maps trained with the unmodified SOM algorithm, or on maps with different sizes. The methods introduced in the next chapter try to alleviate this.

2.6 Summary

In this chapter, the concept of the Self-organising Map (SOM) was explained briefly: The SOM is a method for topology-preserving projection of high-dimensional data into a low-dimensional output space for the purpose of visualisation and data exploration. It is an artificial neural network whose neurons are represented by prototype vectors in the input space. The neurons are placed in a lattice which defines a dis-

tance relation between them. During the training phase, the prototype vectors are shifted to fit to the data, and each time a specific unit is shifted the units close to it on the lattice are shifted as well, relative to their closeness.
Several visualisations for the SOM were shortly described: The U-Matrix, the data histograms, the Smoothed Data Histograms (SDH), and the class information visualisation, which can all be used to gain information about the data and the SOM in question.
Also, cluster analysis on SOMs was touched on as a method to detect existing clusters in the data or to summarise and compress it. The clusters found by cluster analysis algorithms can also be visualised to gain insight into the structure of the data.
Several quality measures were described: the Topographic Error, the Topographic Product, Trustworthiness and Neighbourhood Preservation and decompositions of the SOM Distortion Measure. They can all be used to assess and compare the quality of a SOM's projection. However, they all measure the fit of a SOM to its input data, they don't compare two SOMs directly.
Finally, a method for comparing multiple SOMs trained on the same data was explained: Aligned Self-organising Maps. It can be used to train and compare a stack of maps with slightly different feature parameter settings to explore the effects of these settings. However, the method can't be used on maps trained with the original SOM algorithm or maps with different sizes or number of training steps.
All in all, this chapter gave an overview of the underlying concepts used in the next chapters.

Chapter 3

Visualisations

This chapter introduces three new variations of SOM visualisations – the main difference to other visualisations being that they are applied to two or more SOMs. They compare the SOMs and show were the data comes to lie on different SOMs, which indicates how stable the data projection is, and which properties of the SOMs are just artefacts of different parameter settings. Parameters comprise such settings as the size of the SOM, the learning rate, the number of iterations, etc.
The first visualisation, described in section 3.1, shows differences in the projection of the data vectors. The second one, explained in Section 3.2, compares cluster analysis done on two SOMs. The third one, detailed in Section 3.3, compares "arbitrarily many" SOMs (limited, of course, by the computing power needed for a comparison of a large number of SOMs) and shows the average distances of data vectors in the SOMs' projections.

3.1 Data Shifts Visualisation

The data shifts visualisation is applied to two SOMs trained on the same data with different parameters. It can be used to detect differences between these SOMs: It displays changes in the position of the data vectors relative to neighbouring data vectors on the SOMs. More precisely, the data shifts visualisation shows for a specific data vector how many other vectors mapped onto neighbouring units in the first SOM are also mapped onto neighbouring units in the second SOM. This can be used to find out how stable the mapping is, how steadily a data vector is put into a neighbourhood of similar vectors on different SOMs. Or more abstractly: How much of the data topology on the map really is caused by attributes of the data, and how much of it is simply an effect of different SOM parameters or initialisations.
The moving data vectors are displayed as arrows drawn between the grid visualisations of the two maps – see Figure 3.1 to get a first impression of what it looks like. The thickness of the arrows corresponds to the number of neighbours that are the same in both maps, while the colour stands for the type of data shift (see

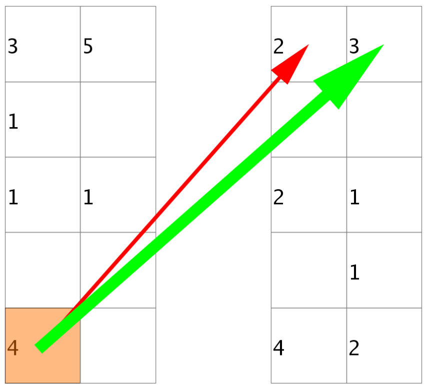


Figure 3.1: Comparison of the position of data vectors on two different SOMs trained on the same data with different parameters; The SOMs are visualised by the two rectangular grids, where each square depicts a SOM unit and each number is the number of data vectors mapped to that unit. The vectors on the unit marked in orange in the left SOM split up and move to two different units in the right SOM. The majority of vectors (3 of 4) moves to the unit marked with the green arrow, while one of the vectors moves to a different unit, marked with the red arrow (this unit also has another vector mapped to it).

Section 3.1.2 for details).

Of course, the visualisation inherently relies on the definition of the neighbourhood used. Definition and adjustment of the neighbourhood in the data shifts visualisation are described in Section 3.1.1.

To visualise the number of neighbours of a data vector that are the same in both compared SOMs, the notion of a "data shift" is defined together with three different types of shifts, all described in Section 3.1.2.

3.1.1 Definition of the Neighbourhood

The distance function used in this visualisation is the SOM-grid distance in the output space. The distance between two data vectors is defined by the euclidean grid distance between the SOM units they are mapped to. For example, in Figure 3.1, the distance between the starting points of the arrows is 0 (as they lie on the same unit in the left SOM), and the distance between the end points is 1 (as they lie on adjacent units in the right SOM). Based on this, the neighbourhood of a vector is defined as the vectors lying within a certain radius around it. From this definition of neighbourhood it follows (maybe a little counter-intuitively) that a vector is always neighbour to itself. So each vector will have at least one neighbour – itself.

Figure 3.2 illustrates which units around a selected centre unit are within the neighbourhood for a (very limited) number of radiuses. The vectors mapped to the units within the radius naturally all are neighbours to the vectors on the centre unit. The units are marked with Roman numerals, and each radius comprises the units marked with the corresponding numeral and all lesser numerals. A neighbourhood of radius 0 only contains (vectors on) unit I; a neighbourhood with radius 1 contains unit I

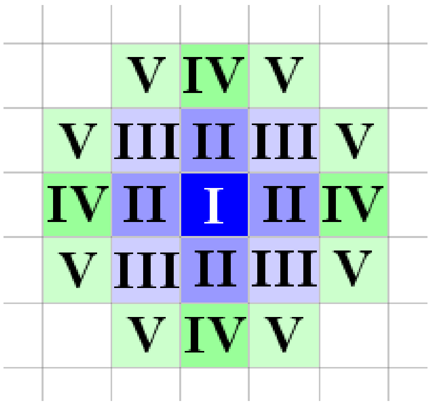


Figure 3.2: Units that lie within a certain neighbourhood radius around a centre.

and all units marked with II; a radius of $\sqrt{2} \approx 1.41$ holds units marked with I, II, and III (all bluish units); a radius of 2 holds units marked with IV and below; a radius of $\sqrt{5} \approx 2.24$ holds all units with V and below (i.e. all coloured units), and so on.

The neighbourhood radius can be separately adjusted for both SOMs; that is, while the neighbourhood radius in the first SOM (called source threshold) may be set to 0, the radius in the second SOM (called target threshold) can be set to 5. This can be useful if the map sizes of the SOMs differ: In the larger of the two SOMs, there are more units for the data vectors to spread over, so vectors on the same unit in the smaller SOM may scatter over adjacent units in the larger. If one is not interested in this effect, the thresholds can be adjusted to compensate it.
Additionally, there are two "modes" of the visualisation, "cumulative" and "non-cumulative". The "non-cumulative" mode changes the neighbourhood radiuses for a part of the calculation of the data shift types (see next section for a detailed explanation of the types and the changes). In the "cumulative" mode the neighbourhood radiuses remain unchanged.

3.1.2 Data Shift Definition and Types

The "data shifts" used to build this visualisation correspond to the data vectors' property to either stay with their neighbouring data vectors on both SOMs, or to leave their neighbours from the first SOM and move to a new area in the second SOM. They are represented by arrows of different colours and thickness, which are drawn between the grid visualisations of the two SOMs (see Figure 3.1 for an example). Different arrow types display where the majority of vectors from a neighbourhood on the first SOM moved to on the second SOM, and where the separating outliers come to lie.
During the calculation process of the visualisation, a shift is produced for each data vector on the maps, unnecessary doublets are discarded later on. A data shift has several attributes:

Count / Thickness: The data shift's "neighbour count" or just "count", which is represented as the arrow's thickness.

This is the number of data vectors that are both in the "old" neighbourhood and in the "new" neighbourhood of the data vector in question. This count is always at least one because each vector is neighbour to itself.

If the "non-cumulative" mode is used, the neighbourhood radiuses are changed. They are set to 0 for determining the count (but not for the adjacent shifts, see below). This effectively means that only vectors that move together *from* a certain unit on the first SOM *to* the same unit on the second SOM, are considered neighbours and increase the count. If the "cumulative" mode is used, the neighbourhood radiuses remain unchanged, and vectors within the radius also count as neighbours.

That is, if both neighbourhood radiuses are 5 and the "non-cumulative" mode is used only the data vectors mapped onto the same unit with the data vector in question are regarded as neighbours. However, if the "cumulative" mode is used, all data vectors mapped to units with a euclidean distance of 5 or less from the specified data vector's Best Matching Unit are regarded as neighbours.

If the thresholds that determine the data shift type (see below) are given as absolute values, the neighbour count is visualised by the thickness of the arrow that represents the data shift.

Percentage / Thickness: The percentage of old neighbours which are also new neighbours. This is just the count divided by the total number of neighbours in the old neighbourhood. The percentage may alternatively be used to determine the arrow's thickness.

If the thresholds that determine the data shift type (see below) are given as percent values, the percentage is visualised by the thickness of the arrow that represents the data shift.

So, how the thickness of the arrow is calculated depends on whether the thresholds are absolute values or percentages.

Coordinates: The coordinates of the shift. That is, the SOM grid coordinates of the two units the vector in question is mapped to.

These positions are, of course, visualised as the start and end points of the arrow representing the data shift.

Type / Colour: The shift type, one of "stable", "adjacent", or "outlier", which determines the colour of the arrow. The type is determined by measuring the neighbour count or percentage against two thresholds which can be adjusted by the user: The stable count threshold and the outlier count threshold. The thresholds are either given as absolute values ("The shift must have a count greater than or equal to 5"), or as percentages ("The percentage must be 75% or more, i.e. 75% of the old neighbours must also be new neighbours"). If the thresholds are given as percentages, a further threshold, the minimum absolute count, can be used to discard all shifts that have a count of less than this minimum threshold. This can be used to sort out shifts of vectors that are very few or single in their source neighbourhood and thus produce a "100%"-shift; "100% of the vectors moved from the old to the new neighbourhood" may sometimes not be seen as useful information if 100% are only one vector.

The types are calculated as follows:

Stable / Green: If the count/percentage is greater than or equal to the "stable count threshold", the shift is considered "stable". This indicates that a high number of vectors moved from the old neighbourhood into the new neighbourhood together with the vector in question.

Stable shifts are drawn as green arrows.

Adjacent / Cyan: If a shift is not stable itself but has its start and end points within the respective neighbourhoods of a stable shift, it is called "adjacent". (The "non-cumulative" mode has no direct effect on adjacent shifts.)

Adjacent shifts are drawn as cyan arrows.

Outlier / Red: If a data shift is neither a stable shift, nor an adjacent shift, and if its count/percentage is greater than or equal to the "outlier count threshold", then it is an outlier shift. Following from the above definition, an outlier shift is more than a (respective) neighbourhood radius away from the start or end point of a stable shift (otherwise it would be an adjacent shift). More intuitively: an outlier shift signifies a small bunch of vectors that moved far away from their old neighbours.

Outlier shifts are drawn as red arrows.

None / Not displayed: If a data shift is more than a neighbourhood radius away from a stable shift but does not have a high enough count to qualify as an outlier, it is not considered (and drawn) at all.

For example, if the outlier count threshold is 1 this applies to single vectors mapped outside their former neighbourhood.

After a data shift is calculated for each data vector on the maps, double data shifts (which would only produce the same arrows multiple times in the visualisation) are discarded. A double shift is one that matches another shift in all its attributes.

Double shifts are quite easy to find, because if the shifts coordinates match (the shifts go from the same unit to the same unit), all other attributes will match as well. The count and percentage must be the same because the number of neighbours is the same, and the type must be the same because the types' conditions mutually exclude each other and cannot be fulfilled for more than one type at the same time.

3.1.3 Mathematical Formalisation

The data shifts and their types can be mathematically formalised as follows:
Let r_1 and r_2 be the radiuses of the source and target neighbourhoods (source and target threshold), and let d_1 and d_2 be the distance functions in the output space of the source and target SOM. Let c_s be the stable count threshold and c_o be the outlier count threshold. x_i is the data vector in question. Then x_i's source and target neighbourhoods U_{1i} and U_{2i} contain other data vectors as follows:

$$U_{1i} = \{x_j | d_1(x_j, x_i) \leq r_1\} \tag{3.1}$$

$$U_{2i} = \{x_j | d_2(x_j, x_i) \leq r_2\} \tag{3.2}$$

The set of neighbours that are in both neighbourhoods, S_i can easily be found, as well as the set of vectors that are neighbours in the first SOM but not in the second, O_i:

$$S_i = U_{1i} \cap U_{2i} \tag{3.3}$$

$$O_i = U_{1i} \setminus U_{2i} \tag{3.4}$$

If the count thresholds are given as absolute values, vector x_i's data shift is stable if $|S_i| \geq c_s$. On the other hand, if the count thresholds are given as percentages, the data shift is a stable shift if $\frac{|S_i|}{|U_{1i}|} \geq c_s$.

If the data shift is not a stable shift, it is an adjacent shift if there is another data vector x_s whose data shift is stable and

$$d_1(x_i, x_s) \leq r_1 \wedge d_2(x_i, x_s) \leq r_2. \tag{3.5}$$

Finally, if the shift is neither stable nor adjacent, it is an outlier shift if $|O_i| \geq c_o$ and the count thresholds are given as absolute values. In case the count thresholds are given as percentages, the shift is an outlier if $\frac{|O_i|}{|U_{1i}|} \geq c_o$.

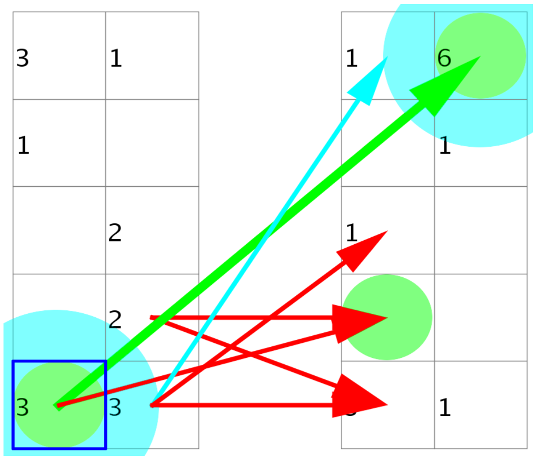


Figure 3.3: Part of a data shifts visualisation focusing on the vectors on the unit marked with the blue box. The source and target thresholds (neighbourhood radiuses) are 1, the "non-cumulative" mode is active, and the stable and outlier count thresholds are 2 and 1 respectively. The circles represent the neighbourhood radiuses for determining the neighbour count (green) and the adjacent shifts (cyan) of the vectors in question.

3.1.4 Example

The definitions above may seem quite abstract, but what does the visualisation look like and what does it mean? See Figure 3.3 for an example of different data shift types explained on two very small maps.

The parameters for this visualisation are a source threshold (neighbourhood radius on the left map) of 1 and a target threshold (neighbourhood radius on the right map) of 1. The "non-cumulative" mode is active, so the neighbourhood radiuses are set to 0 for determining the counts; therefore only vectors that go from the same unit to the same unit count towards the neighbour count. The stable count threshold is 2, while the outlier count threshold is 1.

The focus is on the three vectors on the unit marked with the blue box and their neighbours. Two of the three vectors move to the unit in the top right corner of the right map, one moves away from the other two (to the unit with the third green circle). The green circles mark the neighbourhoods used for determining the count of the three data shifts. So the vectors that move to the top right corner each have two neighbours that move with them (one of the neighbours being themselves), their count is 2. The other vector has only one neighbour that moves with it – itself. This yields two stable shifts and one outlier shift. The two stable shifts would produce exactly the same arrow, so one of them is discarded and the remaining one is shown as the green arrow.

As we have a stable shift now, we can check whether it has any adjacent shifts that move from its source neighbourhood to its target neighbourhood (but don't have a high enough count to be stable). The neighbourhood radiuses are both 1, the cyan circles mark the relevant areas on the SOM. If there is a vector that moves from the left cyan circle to the right cyan circle, it yields an adjacent shift. There is one indeed, which moves from the bottom right corner of the left map to the top left corner of the right map, marked with the cyan arrow. All the other vectors have a count of only 1 and don't start and end in the cyan circles, so they are outliers, drawn as red arrows.

3.2 Cluster Shifts Visualisation

The cluster shifts visualisation is conceptionally similar to the data shifts visualisation introduced in section 3.1. However, it works on clusters instead of units or neighbourhoods. It operates on two SOMs, which it compares to find out how similar the results of clustering on both maps are.
Using the clustering functionality of the SOMViewer [12], the SOMs are clustered with Ward's linkage [20], and the same (user-adjustable) number of clusters is extracted from the cluster trees for both SOMs. The clusters of both maps are identified with each other, and these mappings are then visualised. In the style of the data shifts visualisation, blue arrows are used to indicate which cluster in the first SOM is mapped onto which cluster in the second SOM. Additionally, the data vectors which move from a cluster in the first SOM to the assigned cluster in the second SOM is considered "stable" shift, and a vector which moves to another cluster is considered an "outlier". Again, these are drawn as green and red arrows. Figure 3.4 shows an example of a cluster shift visualisation of two SOMs which were both trained on synthetic data. The data was generated to contain two slightly overlapping Gaussian clusters. The number of clusters to find was set to two by the user.

3.2.1 Cluster Mapping

Obviously, a mapping from the clusters in the first SOM onto the clusters in the second SOM must be found for this visualisation to work. The method used is described below, although other methods are conceivable – for ideas see Chapter 6.

The mapping used in the data shifts visualisation is determined from the agreement of data vectors in the clusters of the first and second SOM: The more data vectors from cluster A_i in the first SOM are mapped into cluster B_i in the second SOM, the higher the confidence p_{ji} that cluster A_i corresponds to cluster B_j.
To put it mathematically: Let set M_{ij} contain all data vectors x which are in A_i and in B_j. To compute the confidence p_{ij} that A_i should be mapped onto B_j, the cardinality of M_{ij} is divided by the cardinality of A_i.

$$M_{ij} = \{x | x \in A_i \wedge x \in B_j\} \tag{3.6}$$

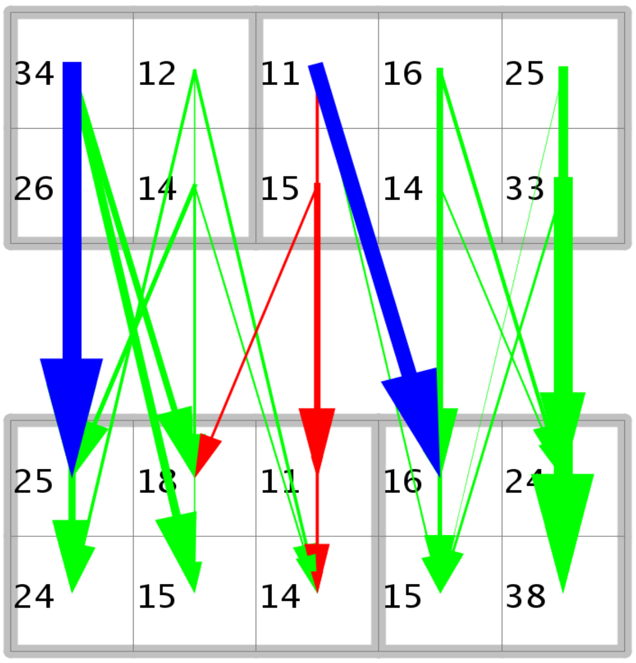


Figure 3.4: A cluster shifts visualisation, showing two SOMs clustered into two clusters each. The cluster borders are drawn as grey lines. The blue arrows indicate which cluster has been mapped onto which, whereas the green arrows depict the data vectors that stayed within clusters mapped onto each other, and the red arrows show data vectors not mapped into corresponding clusters.

$$p_{ij} = \frac{\mid M_{ij} \mid}{\mid A_i \mid} \tag{3.7}$$

To compute all confidence values, a table with entries for all possible mappings is prepared and initialised to 0s. Then the list of data vectors is traversed and the table entries are updated: When a vector moved from cluster A_i to cluster B_j, the table entry for $A_i \mapsto B_j$ is increased by one. After that, the table entries are divided by the number of data vectors in cluster A_i. This yields the percentage of data vectors from cluster A_i that moved to cluster B_j, the measure for confidence, for all possible mappings.

From the table of confidences, the final mapping is derived: The entries are sorted according to confidence, and the mapping $A_i \mapsto B_j$ from the highest entry is taken as the first definite cluster assignment. Then all table entries concerning either A_i or B_j are excluded from the further search, and the next highest entry is picked as the next definite assignment. This step is repeated until all clusters in both SOMs have been assigned.

When the mapping is determined, the visualisation can easily be drawn. The cluster mappings are indicated by blue arrows, whose thickness corresponds to the confidence of the assignment. The stable (green) shifts are shifts from a cluster in the first SOM to the assigned cluster in the second SOM. The outlier (red) shifts are shifts into other clusters.

Cluster in 1st SOM	**Cluster in 2nd SOM**	**Confidence**
3	2	90%
~~2~~	~~2~~	~~90~~%
0	3	85%
1	0	54%
~~1~~	~~1~~	~~46~~%
~~0~~	~~1~~	~~15~~%
2	1	10%
3	3	6%
3	1	3%
1	2	1%
3	0	0%
2	3	0%
2	0	0%
1	3	0%
0	2	0%
0	0	0%

Table 3.1: Table of confidences for two SOMs trained on the data set described in Section 4.2.

Cluster in 1st SOM	**Cluster in 2nd SOM**	**Confidence**
0	3	85%
1	0	54%
2	1	10%
3	2	90%

Table 3.2: Final cluster assignment resulting from Table 3.1.

3.2.2 Example

As an example, two SOMs were trained on three partially overlapping Gaussian clusters, with a different number of samples for each cluster. Clustering both SOMs into 4 clusters yielded the table of confidences given in Table 3.1, which is sorted according to confidences. Clusters are named 0 through 3 in both SOMs. To derive the cluster mappings, the table entry with the highest confidence is selected, then other entries concerning one of the two clusters in questions are striked out (and thus can't be selected later on). This step is repeated until all clusters are mapped. The grey table entries are not used because a full mapping is found within the first seven lines.

Through picking the cluster mappings with the highest confidentialities, the table is reduced to Table 3.2. These are the lines from Table 3.1 that are not striked out or grey. The final mappings are $0 \mapsto 3$, $1 \mapsto 0$, $2 \mapsto 1$, and $3 \mapsto 2$.

Figure 3.5 shows a simplified visualisation with no stable and outlier shifts and the cluster names added.

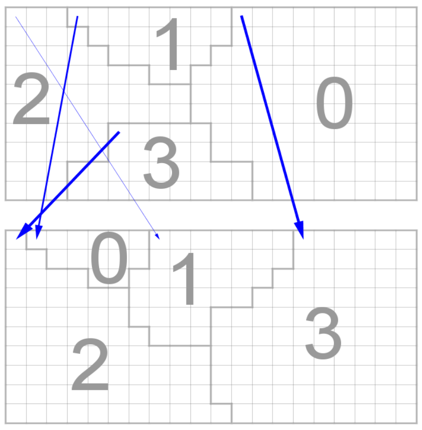


Figure 3.5: Simplified visualisation of the example with no stable and outlier shifts drawn in and the cluster names added.

3.3 Comparison Visualisations for Multiple SOMs

The comparison visualisation compares "arbitrarily many" SOMs trained on the same data set – where "arbitrarily many" of course depends on the computing power available. It focuses on one specific SOM, the "main SOM", and compares it to a set of other SOMs. To be more precise, the visualisation colours each unit in the main SOM according to the average pairwise distance of the unit's data vectors in the other SOMs. The distance function used is, as before, the euclidean grid distance in the SOMs' output space. A threshold can be used to determine how large a pairwise distance must be to contribute to the average.
There are four variations of the comparison visualisation's basic concept, combinations of different statistical moments with different distance functions:

The "Comparison – Mean" visualisation displays the mean euclidean distance over the set of other SOMs.

The "Comparison – Variance" visualisation displays the variance of the euclidean distance over the set of other SOMs.

The "Comparison – Cluster mean" visualisation displays the mean of a cluster distance (defined in Section 3.3.3) over the set of other SOMs.

The "Comparison – Cluster variance" visualisation displays the variance of the same cluster distance over the set of other SOMs.

3.3.1 Computing the Comparison Visualisations

All four variations are computed in the same fashion, from the mapping of the data vectors onto the main SOM M and a data vector distance matrix for each other SOM S_x.

Basically, each SOM S_x is compared to the main SOM M, all data vector distances from all comparisons are accumulated and then divided by the number of SOMs s. It is cost-effective to always compute the mean and variance for the same distance function in one go and then cache them, as the necessary calculations are very similar and the additional cost is minimal. So in the author's implementation, mean and variance for the same distance function are always computed together.

For the computation, a mean matrix and a variance matrix with the same size as SOM M are initialised to zero. Both matrices are used as an accumulator and in the end contain the mean and variance of the distances over all SOMs S_x.

To start the comparison between M and S_x, a distance matrix is derived for S_x, containing the pairwise distance between all data vectors in S_x. See Section 3.3.3 for a detailed explanation of distance functions and the distance matrix. Only the distance matrix makes the difference between the mean / variance and the cluster mean / cluster variance visualisations – all other steps of the computation are the same.

Then, M is traversed unit by unit, and for each unit, the pairwise distances on the various other SOMs between the data vectors on it are summed up and divided by the number of pairs. If the threshold value is used, only pairwise distances greater than the threshold are summed up, but the sum is still divided by the same number of pairs. This effectively sets the pairwise distances less than or equal to the threshold to 0.

In any case, this yields a local average of distances in S_x for each unit of M. This average is added to the mean matrix for the unit in question. In a similar fashion, the variance matrix is updated with the average of the squared pairwise distances. For a mathematical explanation, see Section 3.3.2.

When these steps are done for all SOMs S_x, each entry of the mean matrix and the variance matrix is divided by the number of SOMs s, and the variance matrix entry is reduced by the squared mean matrix entry (why this is necessary is explained in the next section). Now the matrices contain the final mean and variance over all SOMs.

When calculating the colours corresponding to mean or variance values in the visualisation, it is desirable to use the whole available palette. The values are therefore normalised and stretched to cover the full palette. Thus the colours in the comparison visualisation don't indicate the absolute mean or variance, but are relative to the maximum value from this comparison. So different comparison visualisations can't be compared directly by colours. Rather one has to remember that for example black in one visualisation may stand for a totally different value than black in another. Comparisons between colours are only meaningful within one visualisation.

3.3.2 Mathematical Summary of the Computation

The computations done can be mathematically expressed as follows: If s is the number of SOMs to be compared with the main SOM; and k is the number of vector pairs on unit u of the main SOM, then

$$meanMatrix(u) = \frac{\sum_{j=0}^{s} \frac{\sum_{i=0}^{k} d_j(i)}{k}}{s} \tag{3.8}$$

$$varianceMatrix(u) = \frac{\sum_{j=0}^{s} \frac{\sum_{i=0}^{k} d_j(i)^2}{k}}{s} - meanMap(u)^2 \tag{3.9}$$

where $d_j(i)$ denotes the distance between the vectors of pair i in the output space of SOM j.
(The formulas may look similar, but note that for the variance the distance is squared.)
The variance $Var(X)$ is defined as

$$Var(X) = E((X - \mu)^2) \tag{3.10}$$

where $E(.)$ is the expected value (mean), X is the random variable and μ is the random variable's expected value (mean).
The variance can also be calculated in another way, using an alternative variance formula (which follows from the linearity of expected values):

$$Var(X) = E(X^2) - (E(X))^2 \tag{3.11}$$

It is derived as follows:

$$\begin{aligned} Var(X) &= E((X - \mu)^2) = E((X - E(X))^2) \\ &= E(X^2 - 2 \cdot X \cdot E(X) + (E(X))^2) \\ &= E(X^2) - 2(E(X))^2 + (E(X))^2 \\ &= E(X^2) - (E(X))^2 \end{aligned}$$

3.3.3 Calculation of the Distance Matrix

For the calculation of mean and variance of vector distances, one needs to know the pairwise distances of all vectors in all compared SOMs. This information can be conveniently represented by a *distance matrix* for each SOM. The distance matrix holds the pairwise distances between all vectors in a particular SOM.

Euclidean Distance

For a euclidean distance, the distance matrix is easily computed from the information which units a vector pair comes to lie on. The distance between two vectors is simply defined as the euclidean distance between the two units onto which the vectors are mapped.

$$d(a, b) = \sqrt{(x_a - x_b)^2 + (y_a - y_b)^2}. \tag{3.12}$$

Cluster Distance

But of course other distance measures can be used, for example a "cluster distance", where it is important on which clusters a vector pair comes to lie. The distance between two vectors is then equal to the distance between the two clusters onto which the vectors are mapped. This, of course, requires the definition of a distance measure between two clusters.

In hierarchical clustering, for example, there are several metrics for cluster distance, called linkage functions. Among the most commonly used are single linkage, complete linkage, and average linkage (see, for example, [13]).
The single linkage distance function is defined by the minimum pairwise distance between the elements of the two clusters. That is: The distance between two clusters is the distance of their closest elements:

$$d_{single}(r, s) = min(dist(x_{ri}, x_{sj})), \tag{3.13}$$

where $d_{single}(r, s)$ is the single linkage distance between clusters r and s, x_{ri} are the elements of r, x_{sj} are the elements of cluster s, and $dist(.)$ is the distance function for cluster elements (which in turn must be defined).
Likewise, complete linkage is the maximum pairwise distance between the elements of the two clusters, and average linkage is the mean pairwise distance between the elements of the two clusters.

The distance function for the "Comparison - Cluster Mean" visualisation is very similar to single linkage: the distance between two vectors is equal to the distance between the clusters they have been mapped to. The distance between two clusters is the minimum pairwise distance between the units in the clusters. The distance between units is plain vanilla euclidean distance.
Special cases of cluster distances are: 0, if both vectors lie in the same cluster, and 1, if the vectors lie in neighbouring clusters (clusters that share a piece of border).
Figure 3.6 illustrates the distance function definition: The vectors we're looking at are mapped onto the units marked with the red and green diamond. The cluster borders are shown in grey. The distance between the vectors is defined as the distance between the clusters they fall into, cluster 1 and cluster 2. The distance between two clusters is the euclidean distance between their closest units – for cluster 1 and cluster 2 this is the distance indicated by the blue line.

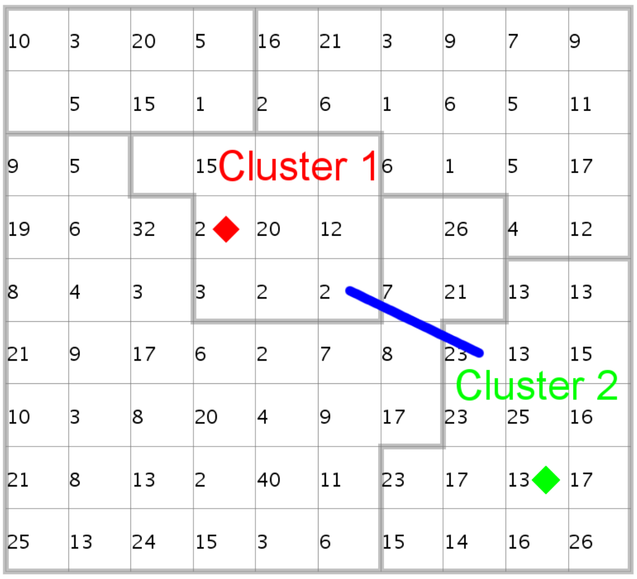


Figure 3.6: The cluster distance function: The distance between the vectors marked with the red and green diamond is actually the distance marked with the blue line.

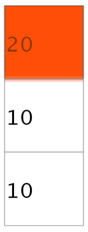


Figure 3.7: Example of a "Comparison – Mean" visualisation.

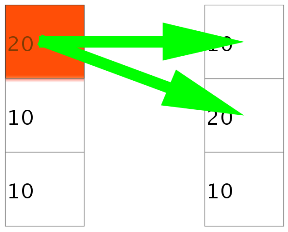


Figure 3.8: Combined comparison and data shifts visualisation.

Of course, one could base other comparison visualisations on other distance definitions than the ones given here, see Chapter 6 for ideas.

3.3.4 Example

Figure 3.7 gives an example of a "Comparison – Mean" visualisation. It was made from two very small maps with only 3 units, the maps were trained on synthetic data with four distinct clusters which had 10 samples each. The visualisation reveals that the lower two units have a mean pairwise distance of 0, but that the topmost unit has a higher mean pairwise distance. This means that all data vectors from each of the lower two units must be mapped onto the same unit respectively in the other SOM. The data vectors from the topmost unit, however, split up and move to different units in the other SOM. Figure 3.8, a combined comparison and data shifts visualisation of both SOMs, shows how the data vectors split up and where they move to.

This chapter introduced the three new visualisations and gave small examples how they work. However, to better demonstrate their possibilities and functionality, they were tested in more detail on selected datasets. The datasets used will be described in the next chapter, while Chapter 5 contains the actual tests.

3.4 Summary

In this chapter, three new SOM visualisations were introduced and described which can be used to compare two or more SOMs.
The data shifts visualisation compares the input data vector's position on two maps relative to the neighbouring data vectors. Stable regions and outliers are identified and visualised.
The cluster shifts visualisation compares clusterings of two SOMs. It uses cluster analysis methods to split the SOMs into a given number of clusters and then identifies clusters from both maps with each other. Data vectors that are mapped to corresponding clusters are considered stable, while vectors that aren't are regarded as outliers.
The comparison visualisation compares two or more SOMs and displays the average pairwise distance of each unit's data vectors in the compared SOMs.
The visualisations are intended to gain insight on the quality of the SOMs' projection and to find out which properties of the SOMs are just artefacts of their parameter settings.

Chapter 4

Datasets Used in the Experiments

The datasets used in the experiments in Chapter 5 were created or selected to test different properties of the visualisations.
The first three datasets were artificially produced with Matlab[1] to have a certain form, contain a given number of clusters, or impose specific difficulties on the SOMs' projection.
The last dataset is a real-world example: the Iris dataset, which is well-known and frequently used for testing new data exploration or machine learning techniques. Thus, the visualisations introduced here can be compared with other methods tested on the Iris dataset.

4.1 Equidistant Gaussian Clusters

The equidistant Gaussian cluster datasets are intended for testing the cluster analysis related features of the visualisations. They were created synthetically to contain clusters which overlap in various degrees.
The "test case" consists of eleven datasets, each of which was produced by taking 100 samples each from three Gaussian distributions. The Gaussian distributions were 2-dimensional, had a standard deviation of 1, and their centres all had the same distance from each other. The distance was varied between the datasets, starting from 0 (all Gaussians overlap) in dataset "0", increasing by 1 for each following dataset, and ending with 10 in dataset "10" (all Gaussians are clearly separated from each other). The samples from the different Gaussians are assigned different class labels (numbers from 1 to 3) and thus yield clusters in the datasets, which are more or less separated, depending on the distance of the Gaussian centres. This, of course, can be used to test how well a clustering algorithm can separate clusters in the data.

[1]`http://www.mathworks.com/`

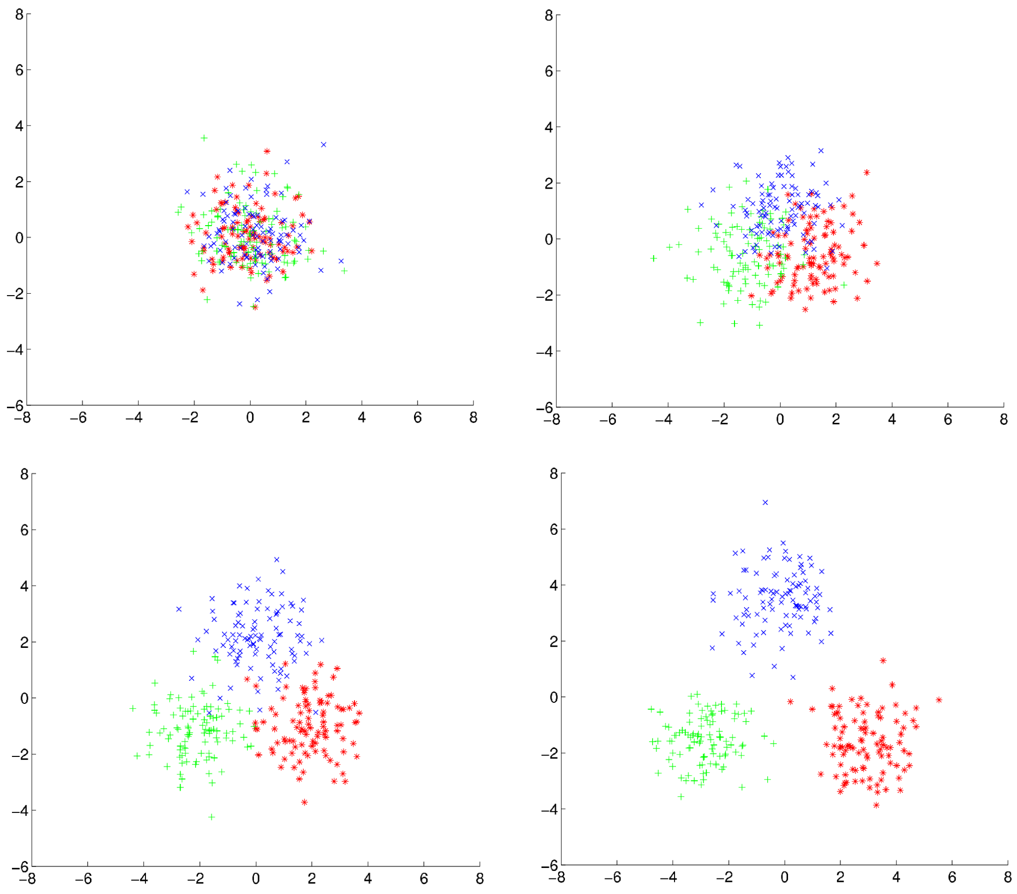


Figure 4.1: Equidistant Gaussian cluster datasets “0”, “2”, “4”, and “6” (from left to right and top to bottom). Different classes are displayed by different colours and symbols.

Dataset	**Red cluster** ($*$)	**Green cluster** ($+$)	**Blue cluster** ($\times$)
0	(0.00, 0.00)	(0.00, 0.00)	(0.00, 0.00)
1	(0.50, -0.29)	(-0.50, -0.29)	(0.00, 0.58)
2	(1.00, -0.58)	(-1.00, -0.58)	(0.00, 1.15)
3	(1.50, -0.87)	(-1.50, -0.87)	(0.00, 1.73)
4	(2.00, -1.15)	(-2.00, -1.15)	(0.00, 2.31)
5	(2.50, -1.44)	(-2.50, -1.44)	(0.00, 2.89)
6	(3.00, -1.73)	(-3.00, -1.73)	(0.00, 3.46)
7	(3.50, -2.02)	(-3.50, -2.02)	(0.00, 4.04)
8	(4.00, -2.31)	(-4.00, -2.31)	(0.00, 4.62)
9	(4.50, -2.60)	(-4.50, -2.60)	(0.00, 5.20)
10	(5.00, -2.89)	(-5.00, -2.89)	(0.00, 5.77)

Table 4.1: Coordinates of the cluster centres.

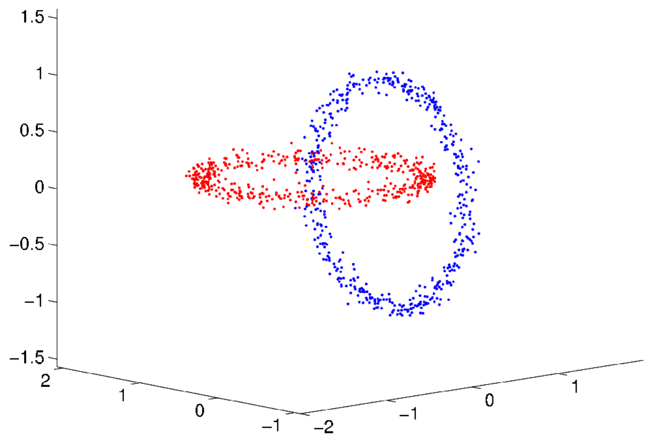


Figure 4.2: Intertwined Rings Dataset.

Figure 4.1 shows plots of the datasets "0", "2", "4", and "6" (from left to right and top to bottom). The different classes are marked by different colours and symbols. Table 4.1 shows the coordinates of the Gaussian centres for all distances/datasets.
Note that from dataset "6" on, with a centre distance of 6 or more, the clusters appear nicely separated. This is to be expected, because the centres of the clusters are $2 \times 3\sigma$ apart (σ, the standard deviation, being 1). So the clusters' 3σ radiuses, into which 99.7% of their samples should fall according to the Gaussian cumulative distribution function, don't overlap. This makes the probability of the clusters overlapping very small.
The experiments done with these datasets can be found in Sections 5.2 and 5.4.

4.2 Intertwined Rings Dataset

The intertwined rings dataset is very similar to, but not exactly the same as, the "Chain-link Benchmark" in [15]. It is used in the experiments to provoke breaches in topology in the SOMs trained on it. The breaches should then be detected by the visualisations.
The dataset consists of 1000 3-dimensional samples which make up two perpendicular, intertwined rings. The position of the samples along the rings is taken from a Gaussian distribution, as is the distance of the samples from the centre circle of the ring (the standard deviation for the distance is 0.05). The centres of the rings are at $(0, 1, 0)$ and $(0, 0, 0)$. A sample's class is defined by the ring it belongs to, so the dataset has two classes. Figure 4.2 shows a plot of the dataset. As was said before, it can be used to provoke a breach in topology in the SOM's projection and see how this breach is detected by the visualisation methods. The breach will occur because the rings are intertwined and the SOM can't project them onto a 2-dimensional space without in one way or the other violating the topology. The experiments and results are detailed in Section 5.3.

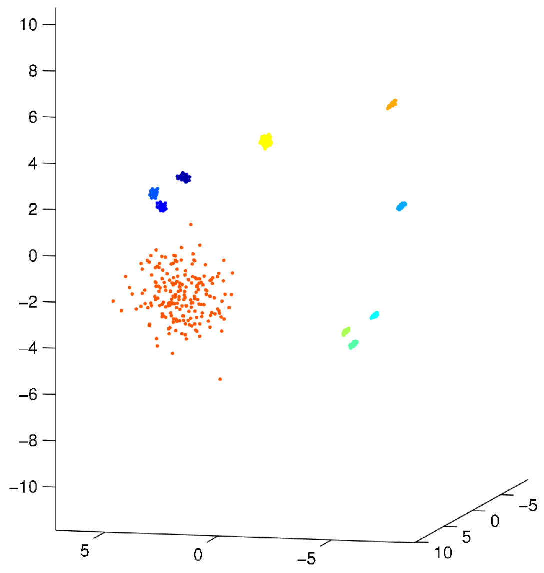


Figure 4.3: Complex Artificial Cluster Dataset.

4.3 Complex Artificial Cluster Dataset

The third artificial dataset was specifically constructed to have ten clusters (named "1" through "10"), which form super-clusters and have varying distances. It can be used to test how the clusters and subclusters show up in SOMs of different sizes and how the visualisations depict them.

The dataset is ten-dimensional, a plot of the first three components is given in Figure 4.3. Although the dataset itself is ten-dimensional, some of the clusters only spread in two dimensions, like flat discs in ten-dimensional space. The clusters were produced by sampling from Gaussian distributions.

Six of the clusters form two super-clusters with three equidistant clusters in each super-cluster. The sub-clusters contain 50 samples each and have a standard deviation of 0.1 and a euclidean distance from each other of 2. One of the super-clusters uses all ten dimensions (marked in dark blue in the plot), the sub-clusters are named "1", "2", and "3". The other super-cluster uses only two dimensions so that its sub-clusters are discs lying in a plane in ten-dimensional space (marked in greens); they are named "5", "6", and "7".

Furthermore, there are four single clusters, each of which is a different combination of the properties number of samples n, dimensionality dim, and standard deviation σ as follows.

- Cluster "4": $n = 150$, $dim = 2D$, $\sigma = 0.1$, marked in light blue.
- Cluster "8": $n = 150$, $dim = 10D$, $\sigma = 0.1$, marked in yellow.
- Cluster "9": $n = 50$, $dim = 2D$, $\sigma = 0.1$, marked in light orange.
- Cluster "10": $n = 200$, $dim = 10D$, $\sigma = 1$, marked in dark orange.

All super-clusters and single clusters have a euclidean distance from each other of at least 7.
The experiments done with this dataset are described in Section 5.1.

4.4 Iris Data Set

A non-artificial, “real-world” dataset the visualisations were tested on is the well-known Iris dataset [4]. It consists of 150 measurements of characteristics of Iris flowers from three different species: Iris setosa, Iris versicolor, and Iris virginica. The characteristics are sepal length, sepal width, petal length, and petal width, each measured in centimetres. The species of a sample serves as its class, each class has 50 samples. While Iris setosa is linearly separable from the other two classes, the distributions of Iris versicolor and Iris virginica overlap, and a new sample could not necessarily be classified as one of the two classes with certainty.
The experiments conducted on this dataset can be found in Section 5.5.

4.5 Summary

This chapter introduced the datasets used in the experiments described in the next chapter. The datasets were constructed or selected to have certain properties which should be detected by the visualisations.
The ten equidistant Gaussian cluster datasets each contain three Gaussian clusters which all have the same distance from each other. This distance varies between the datasets, starting from 0 (clusters overlap totally) and ending at 10 (clusters are clearly separated). The datasets can be used to test the clustering related visualisations.
The intertwined rings dataset contains two rings of data vectors which are intertwined. This property leads to breaches in topology in the SOMs trained on the dataset. So this dataset can be used to test the capability to detect breaches in topology.
The complex artificial cluster dataset contains ten clusters with varying number of samples, variance and dimensionality; six of the clusters are grouped into two super-clusters. It can be used to test in how far visualisations detect super-clusters and sub-clusters and how the different properties of the clusters are reflected by different SOMs.
Finally, the well-known Iris dataset and its characteristics were shortly described. As a “real-world” dataset, it is used to test the visualisations on data that is not artificially constructed.

Chapter 5

Experiments

The experiments in this chapter were made to demonstrate the functionality and capabilities of the proposed visualisations. The first three of the following sections will describe experiments on specially constructed datasets. There is a focus for each visualisation on one specific dataset which has the properties that the visualisation is supposed to detect. The findings of the visualisation in question are then compared with the results of the other two visualisations. The last section gives an analysis of a real-world example, the Iris dataset, using all three visualisations.

5.1 Data Shifts Visualisation on the Complex Artificial Cluster Dataset

The data shifts visualisation was tested mainly on the "complex artificial cluster dataset" (see Section 4.3). Two maps were trained, both with an initial learning rate $\alpha_0 = 0.75$, 10,000 iterations and a random seed for prototype vector initialisation of 42. The size and initial neighbourhood size for both maps differed, the first map having 2×3 units and $\sigma_0 = 3$ and the second map having 6×9 units and $\sigma_0 = 9$.
The resulting maps were analysed and compared with the data shifts and the class information visualisation, to find out where the data is situated on both maps and how stable the mappings are. The parameters for the data shifts visualisation were: source threshold: 0, target threshold: 0, cumulative "off", absolute count thresholds: stable: 10, outlier: 1.
Figure 5.1 shows four snapshots of the visualisations, each focusing on a different unit of the smaller SOM. Note that the positions of the data vectors in the larger SOM are mirrored with respect to their position in the smaller SOM: Data from the left-hand side of the small map moves to the right side of the large map and vice versa, data from the top of the small map moves to the bottom of the large map and vice versa.
The first snapshot (top left) shows where the data vectors from the top right unit in the small map move to on the large map. They split up into three equally sized

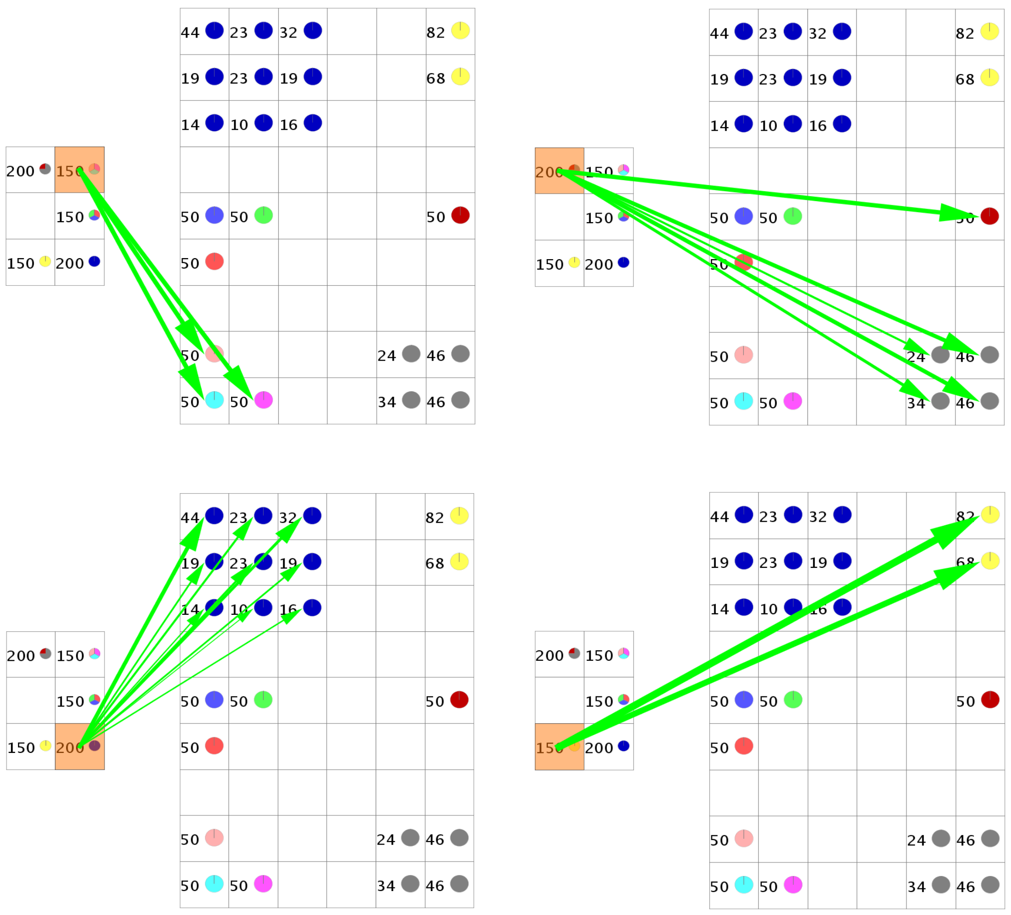


Figure 5.1: Data shifts visualisation of two SOMs trained on the complex artificial cluster dataset.

groups which come to lie on three adjacent units. Looking at the colour coding of the class information, one can see that the data vectors concerned are from clusters “5”, “6”, and “7”, which were constructed to form a super-cluster. Obviously, the super-cluster was mapped onto one unit in the smaller SOM and split up into its sub-clusters in the larger SOM. Just the same goes for clusters “1”, “2”, and “3”, which lie on the right middle unit in the smaller SOM and in the middle of the lefthand side of the larger SOM (marked in red, green, and blue). There is no snapshot of that because it is equivalent to the first snapshot.

The second snapshot (top right) shows something similar: Data vectors that lie on one unit in the smaller SOM split up and spread over more units in the larger SOM. This case, however, is a little different because a part of the data vectors move away from the others onto an isolated unit (marked in dark red). The rest of the data spreads over four adjacent units (marked in grey). Apparently, there are two distinct clusters in the dataset which get mapped onto the same unit in the small SOM and which separate in the large SOM because there is enough space. Possibly the grey cluster has sub-clusters which split up in the larger SOM. The class information visualisation confirms that there are indeed two distinct clusters, cluster “8” (marked in grey) and cluster “9” (marked in dark red). Cluster “8”, however, doesn’t have constructed sub-clusters, rather it has three times more samples than cluster “9” and these are spread on the large map according to the inner topology of the cluster.

The last two snapshots (bottom-half of Figure 5.1) also show data vectors from one unit in the small SOM splitting up and spreading over a larger area in the large SOM. However, in the third snapshot (bottom left) the vectors spread over a much larger area with fewer vectors per unit than in the forth snapshot (bottom right). Although there are more vectors in the dark blue cluster (200) than in the yellow cluster (150), this can’t account alone for the much larger spread of the dark blue cluster. It can be assumed that the variance in the dark blue cluster is larger than in the yellow cluster. Checking with the class information, this is actually the case: The Gaussian distribution from which cluster “10” (dark blue) was drawn, has a standard deviation of 1, while the one cluster “4” (yellow) was taken from only has a standard deviation of 0.1.

From this, another question arises: Why does cluster “8” (grey) spread over four units in the larger SOM, while cluster “4” (yellow) only spreads over two units, indicating a lower variance in the data? They have the same number of samples (150) and were both drawn from a Gaussian distribution with standard deviation 0.1. The only difference between the two clusters is their dimensionality: Cluster “8” uses all ten dimensions while cluster “4” only uses two dimensions. Thus the variance in cluster “4” is indeed lower because in the unused dimensions there is no variance at all.

The variance and spread can also be checked by changing the visualisation parame-

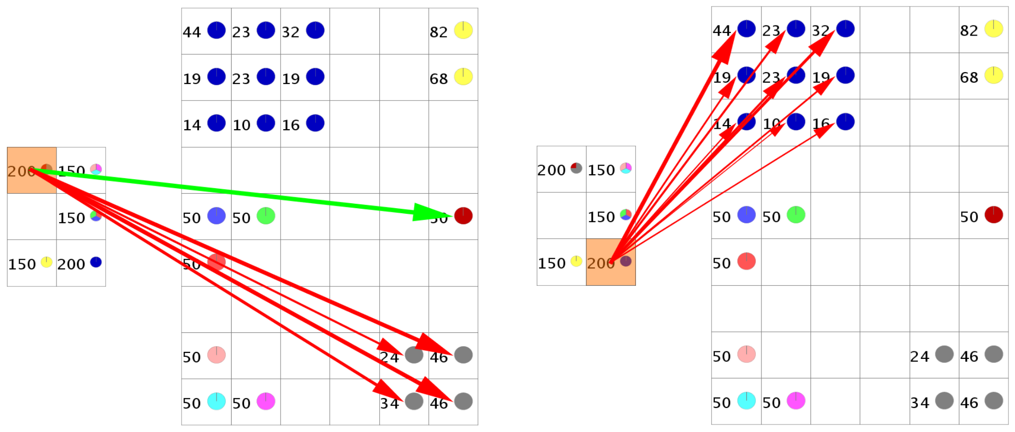


Figure 5.2: Data shifts visualisation of two SOMs trained on the complex artificial cluster dataset with different stable count threshold.

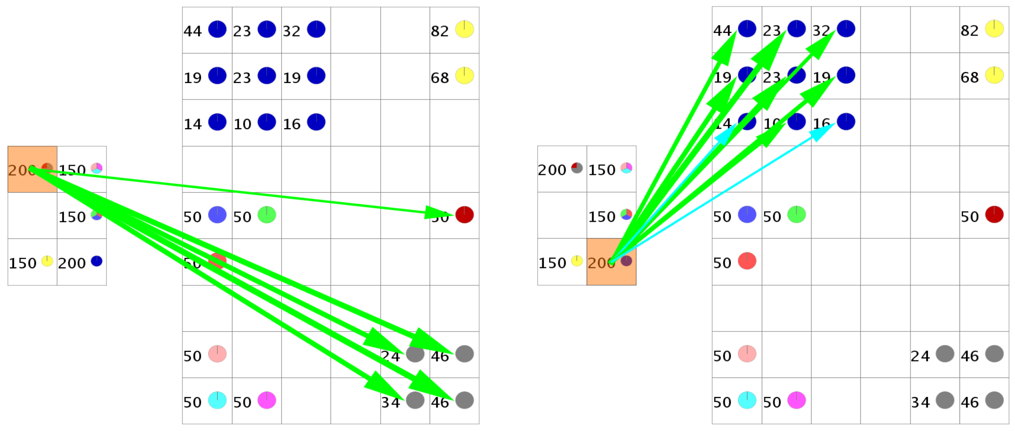


Figure 5.3: Data shifts visualisation of two SOMs trained on the complex artificial cluster dataset with stable count threshold of 50, cumulative mode on, and a target threshold of 1.

ters: If, as is the case, one knows that the smallest class / cluster in the input data has 50 members, the absolute stable count threshold can be set to this minimum (while leaving the outlier threshold at 1). This changes the results from Figure 5.1 in two regions: Cluster "8" (grey) and cluster "10" (dark blue), see Figure 5.2, all other data shifts stay the same. So, for all clusters but "8" and "10" stable shifts with 50 or more neighbours occur between the small and the large SOM, showing the high coherence of these clusters. Clusters "8" and "10", however, spread over more units in the large SOM, and each unit has less than 50 data vectors on it, which leads to the outlier shifts marked in red in Figure 5.2. The two clusters break apart into smaller portions because of their high variance.

By changing the visualisation parameters further, the bunches of outlier shifts in Figure 5.2 can again be recognised as coherent clusters: If the cumulative mode is switched on and the target threshold is set to 1, the outlier shifts are changed to stable or adjacent shifts, see Figure 5.3. The choice of a target threshold of 1 is

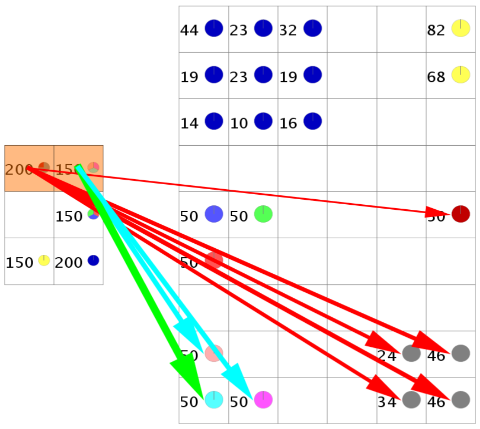


Figure 5.4: Data shifts visualisation of two SOMs trained on the complex artificial cluster dataset with stable count percentage 100%, cumulative mode on, and a target threshold of 1.

reasonable because the target map is larger than the source map. If the cumulative mode is not switched on (but the target threshold is set to 1), the outlier shifts don't change at all because in non-cumulative mode the target threshold only influences adjacent shifts. Adjacent shifts, however, can only occur within the neighbourhood of a stable shift, and there are no stable shifts for clusters “8” and “10” in Figure 5.2.

Using the target threshold in another way, one can switch the stable count threshold from an absolute value of 50 to a percentage of 100%, while leaving the cumulative mode on and the target threshold at 1. The visualisation now shows vectors as stable shifts that have all vectors from their unit in the small SOM within a neighbourhood of radius 1 in the large SOM. Most of the clusters now produce either a stable or at least an adjacent shift, but clusters “8” and “10” and additionally cluster “9” (dark red) yield outliers again. In case of clusters “8” and “10” this again results from their large variance, in case of cluster “9” it comes from being mapped onto the same unit with a entirely different cluster (cluster “8”) in the small SOM. Figure 5.4 shows an exemplary part of this visualisation.

As can be seen, there are countless possibilities to set the parameters for the data shifts visualisation, only a small number of which could be shown here for the sake of brevity.

A number of conclusions can be drawn from the analysis with the data shifts visualisation: Sub-clusters can be identified and differentiated from clearly separated clusters that are only mapped onto the same unit if the SOM is too small. More importantly, the comparison between a large and a small map gives an impression of the variance of clusters and a rough idea of their shape. Additionally, the relative orientation of the maps can be easily detected.

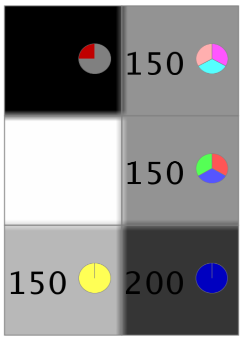


Figure 5.5: "Comparison – Mean" visualisation of the two SOMs trained on the complex artificial cluster dataset.

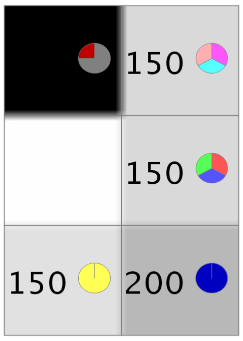


Figure 5.6: "Comparison – Variance" visualisation of the two SOMs trained on the complex artificial cluster dataset.

5.1.1 Checking Results with Cluster Shifts Visualisation and Comparison Visualisation

To check the results from the previous section, the cluster shifts visualisation and the comparison visualisation were applied to the same two maps of the complex artificial cluster dataset.

The "Comparison – Mean" visualisation which compares the small map with the large map is given in Figure 5.5. The top left unit (with clusters "8" (grey), "9" (dark red)) and the bottom right unit (with cluster "10" (dark blue)) have a high mean pairwise distance which confirms that the clusters on them have a higher variance than the other clusters. The high values result from the high variance of cluster "8" and "10" and from cluster "8" and "9" splitting up and moving to different areas in the large SOM.
The corresponding "Comparison – Variance" visualisation (see Figure 5.6) shows that the variance of the pairwise distances is highest on the top left unit. This is to be expected because this unit contains two clusters which move to clearly separated areas in the large SOM. The variance of the bottom right unit is also higher than that of the other units, again because the spread of cluster "10" in the large SOM is higher than that of the other clusters.

The cluster shifts visualisation of the two maps is given in Figure 5.7. It contains six clusters – more clusters don't make sense because each cluster must have at least one unit in it, and the small SOM has only six units. So in the small SOM, each cluster consists of one unit only, and one of the clusters is completely empty. At first sight it might look like the large SOM has seven clusters, but the empty cluster in the lower left part of the map and the cluster in the middle of the right side are actually one cluster.
As can be seen in the two snapshots at the top, the clustering algorithm correctly finds cluster "4" (yellow) and the super-cluster consisting of clusters "5", "6", and

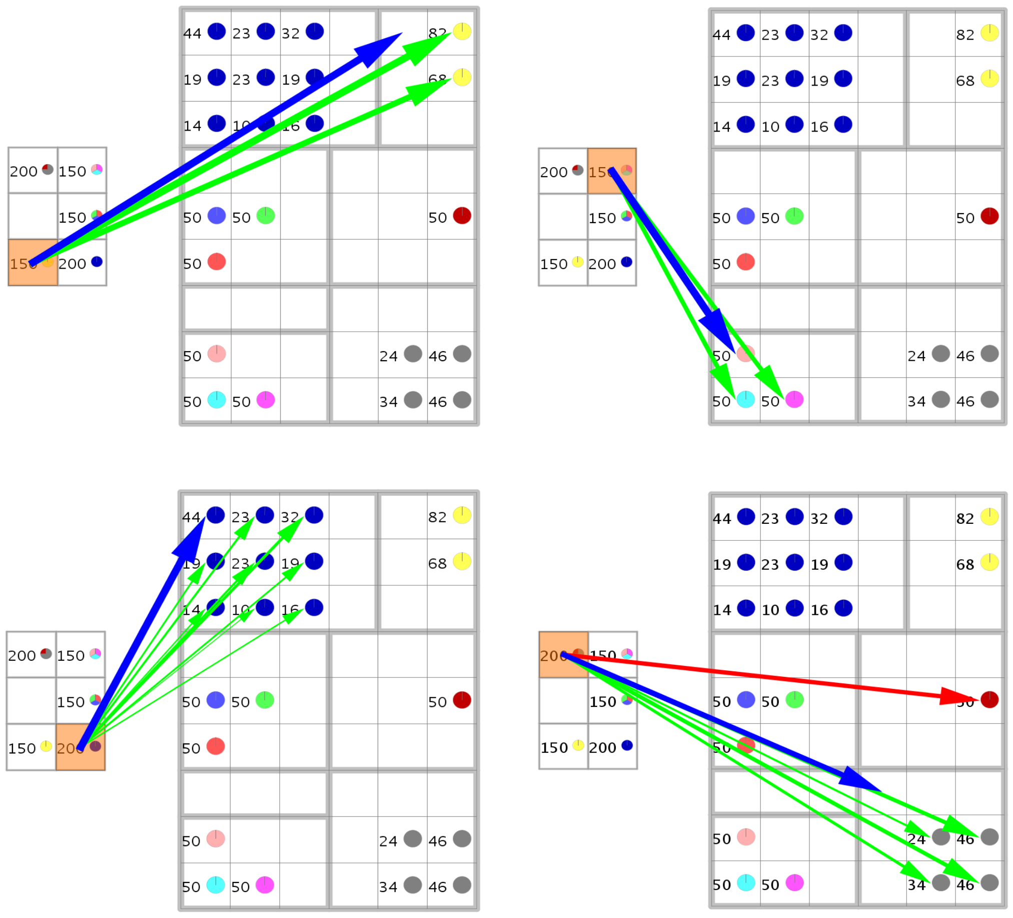


Figure 5.7: Cluster shifts visualisation of two SOMs trained on the complex artificial cluster dataset.

“7”. It also finds the super-cluster of clusters “1”, “2”, and “3”, which is not shown in Figure 5.7 because it looks very similar to the top right snapshot. Cluster “10” is also identified, see the bottom left snapshot. However, there is a cluster shift outlier in the bottom right snapshot: The vectors from clusters “8” and “9” come to lie in one unit / cluster in the small SOM, but are recognised as separate clusters in the large SOM. The data vectors from cluster “9” are marked as outliers because their cluster is identified with the empty cluster in the small SOM. A correct identification isn’t possible here because the cluster mapping is always 1 to 1 and clusters “8” and “9” will always be mapped into the same cluster in the small SOM.

So the clustering algorithm correctly finds all (super-)clusters and the cluster assignment scheme correctly identifies clusters where it is possible. Together with the results from the comparison visualisations about the clusters’ variance, this reinforces the findings of the data shifts visualisations.

	0	1	2	3	4	5	6	7	8
Lowest	68%	52%	81%	59%	70%	95%	97%	94%	100%
Medium	88%	78%	86%	82%	84%	97%	97%	100%	100%
Highest	94%	99%	99%	100%	97%	99%	100%	100%	100%
Average	83%	76%	89%	80%	84%	97%	98%	98%	100%

Table 5.1: Cluster assignment confidences for SOMs trained on equidistant Gaussian cluster datasets “0” through “8”.

5.2 Cluster Shifts Visualisation on the Equidistant Gaussian Cluster Datasets

The cluster shifts visualisation, particularly the cluster assignment method, was tested using the “equidistant Gaussian cluster datasets” described in Section 4.1. They contain samples from three equidistant 2-dimensional Gaussian distributions; the distance between the Gaussian centres (means) increases from 0 in dataset “0” to 10 in dataset “10”.

For each of the datasets “0” through “8”, two SOMs were trained with exactly the same parameters: a map size of 10×10, an initial learning rate $\alpha_0 = 0.75$, an initial neighbourhood size $\sigma_0 = 10$ and 10,000 iterations. Only the random seed for the initialisation of the prototype vectors differed between both SOMs, being 42 for one and 421 for the other.

The resulting two SOMs for each dataset were then compared with each other. They were clustered into three clusters (because the data contains three classes) with hierarchical clustering, using Ward’s linkage function, and the clusters in both SOMs were assigned to each other. This yielded three cluster assignment confidences per dataset (one for each cluster pair). As described in Section 3.2.1, the “confidence” is the percentage of data vectors that moved from a cluster in the first SOM to a cluster in the second SOM. The highest, lowest and average cluster assignment confidence for datasets “0” through “8” are detailed in Table 5.1 and plotted in Figure 5.8.

Naturally, the confidences increase the farther apart the Gaussian centres get.

That the confidences get very close to 100% around dataset “5” and “6” is also not surprising, given the structure of the data. In dataset “6”, the samples of different classes are (almost) completely separated because the 3σ radiuses of the Gaussians don’t overlap (see Section 4.1 for a more detailed explanation). Thus the clustering algorithm always finds clusters almost perfectly corresponding to the classes and to each other.

A subset of the produced cluster shifts visualisations is shown in Figure 5.9. It contains the visualisations for datasets (top to bottom) “0”, “2”, “4”, and “6” – see Figure 4.1 for plots of the original data.

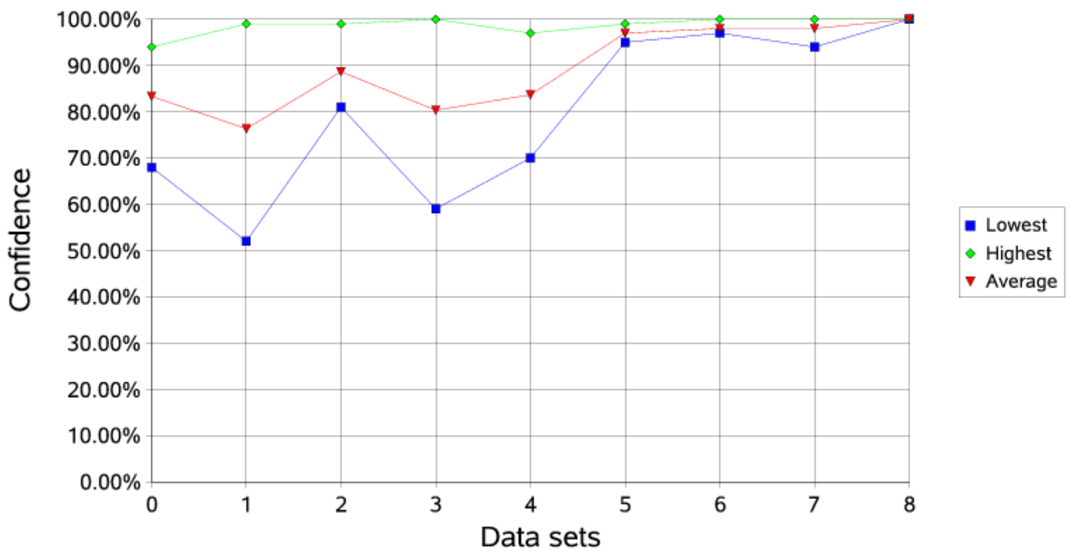


Figure 5.8: Cluster assignment confidences for SOMs trained on equidistant Gaussian cluster datasets "0" through "8".

In the topmost figure, the one for dataset "0", it is obvious that the different classes are mixed up randomly on both SOMs, just as is to be expected from the data. The clusters are, of course, also built rather randomly, so the cluster assignment confidences are relatively low, averaging to 83% (compare Figure 5.8 and Table 5.8). The confidences are as high as 83% (which isn't bad for other datasets which are not built artificially) because the clustering algorithm detects similar areas in data space as clusters in both SOMs – only the right SOM is turned upside down in comparison to the left SOM. As the clusters are so random and don't correspond to a single class each, it can't be verified if the clusters that correspond to a class are mapped onto each other.
In the next figure, based on dataset "2", the classes start to separate, moving into different corners of the SOM. The clusters aren't so random any more – one can make out a yellow, a green, and a red cluster (although there are errors, of course). The yellow, green, and red clusters are correctly mapped onto each other, the average confidence increases to almost 90%.
The third figure (for dataset "4") shows quite well-separated classes, but the clustering algorithm doesn't find the clusters corresponding to the classes perfectly. There are errors along the class borders, the clustering is unprecise in separating the classes, thus the the average cluster assignment confidence decreases to 84%.
The last figure (for dataset "6") shows clearly separated classes, no data vectors from different classes are mapped onto the same unit. The clustering is still a bit unprecise, however, the average cluster assignment confidence rises to almost 100%. (The cluster assignment confidence finally reaches 100% in dataset "8".)

Apparently, the cluster assignment method works well. Where there are clusters

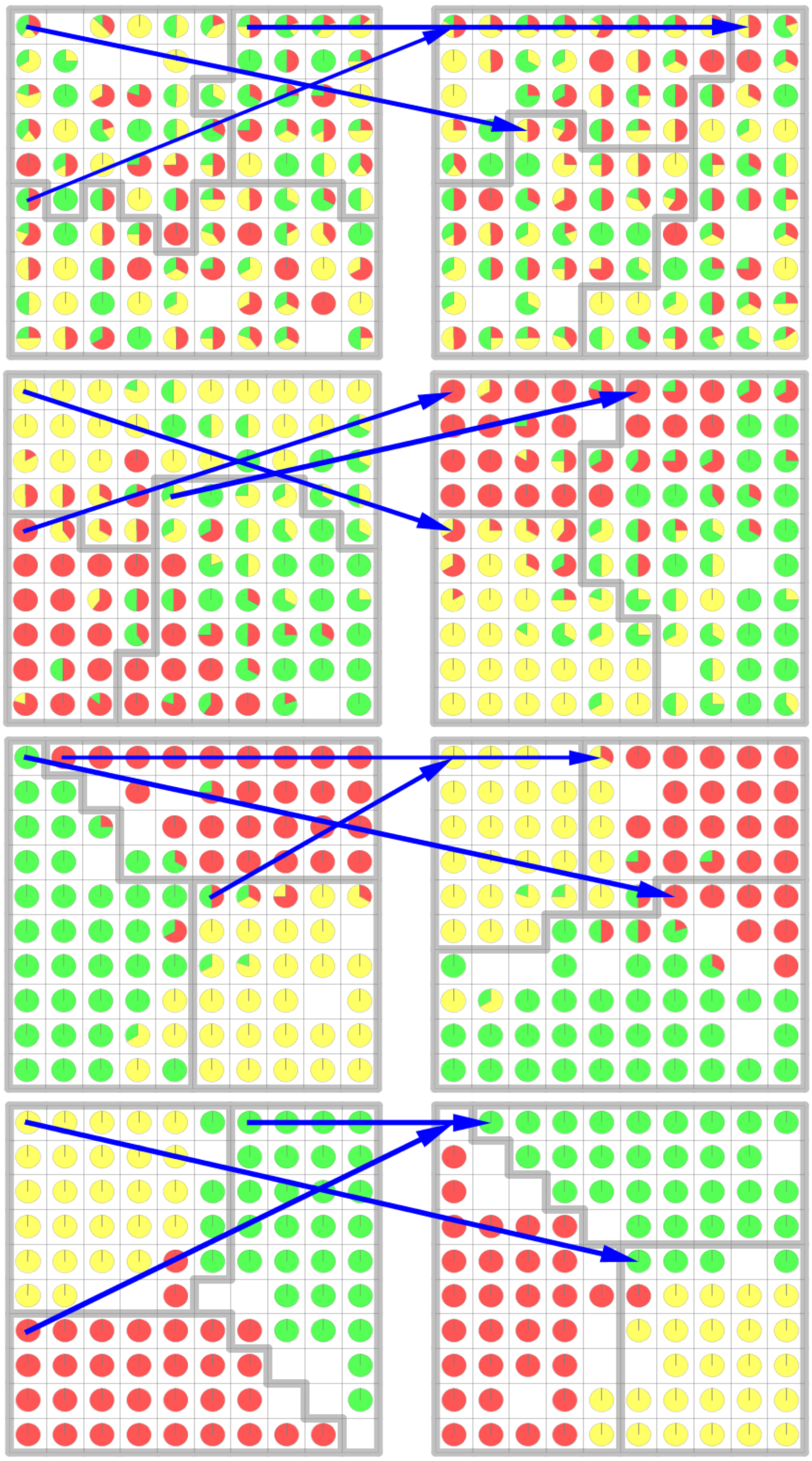


Figure 5.9: Cluster shift comparisons between pairs of SOMs trained on the same datasets. The datasets are the equidistant Gaussian cluster sets (top to bottom) “0”, “2”, “4”, and “6” (see Figure 4.1).

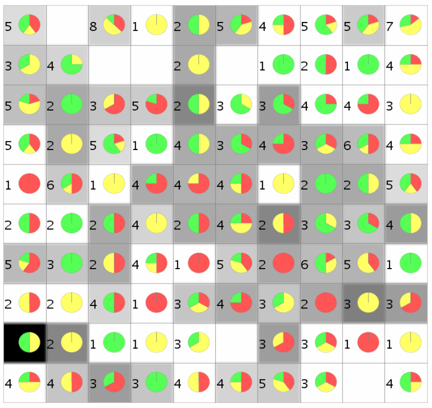


Figure 5.10: "Comparison – Mean" visualisation of the two SOMs trained on equidistant Gaussian cluster dataset "0".

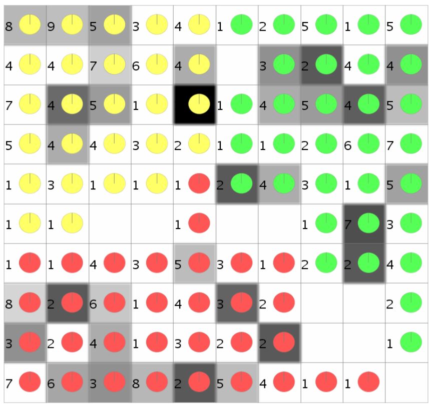


Figure 5.11: "Comparison – Mean" visualisation of the two SOMs trained on equidistant Gaussian cluster dataset "6".

corresponding to the classes, the right clusters are identified with each other, so that the "green" cluster in the left SOM is always mapped onto the "green" cluster in the right SOM, and so on. The confidence value roughly corresponds to the grade of separation of the clusters and thus to the "cleanness" of the clusters.

5.2.1 Checking Results with Data Shifts Visualisation and Comparison Visualisation

The equidistant Gaussian clusters datasets were also tested with the other visualisations to check whether their results would support the findings from the previous section.

Figure 5.10 and 5.11 show two "Comparison – Mean" visualisations for two pairs of SOMs trained on the equidistant Gaussian cluster datasets "0" and "6" respectively. The maps are the ones used for Figure 5.9 in the previous section.

When comparing the two comparison visualisations there are two interesting differences:

Firstly, while in Figure 5.10 there are a lot of grey units (with a low mean pairwise distance) in the centre of the map, Figure 5.11 has very few units with a mean pairwise distance greater than zero in its centre. This may seem trivial, as in Figure 5.11 the clusters are already well-separated and the area in the centre is therefore sparse of data vectors. There is a Y-shaped rift between the clusters where the units have very few data vectors or none at all. So the grade of cluster separation is visible in these two comparison visualisations, too.

Secondly, there are a lot more units with a mean pairwise distance of zero in Figure 5.11 than in Figure 5.10. This probably results from the fact that dataset "6" has a clear inner structure (three distinct clusters) while dataset "0" consists only of one big Gaussian cluster. As dataset "0" has little structure, the fit of SOMs to

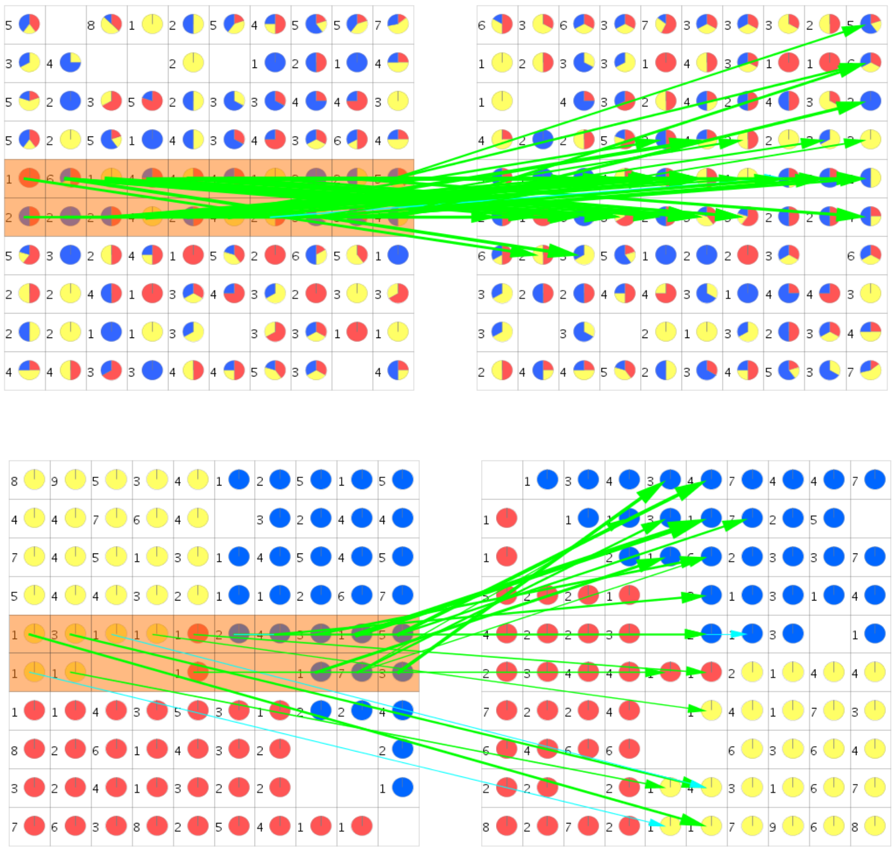
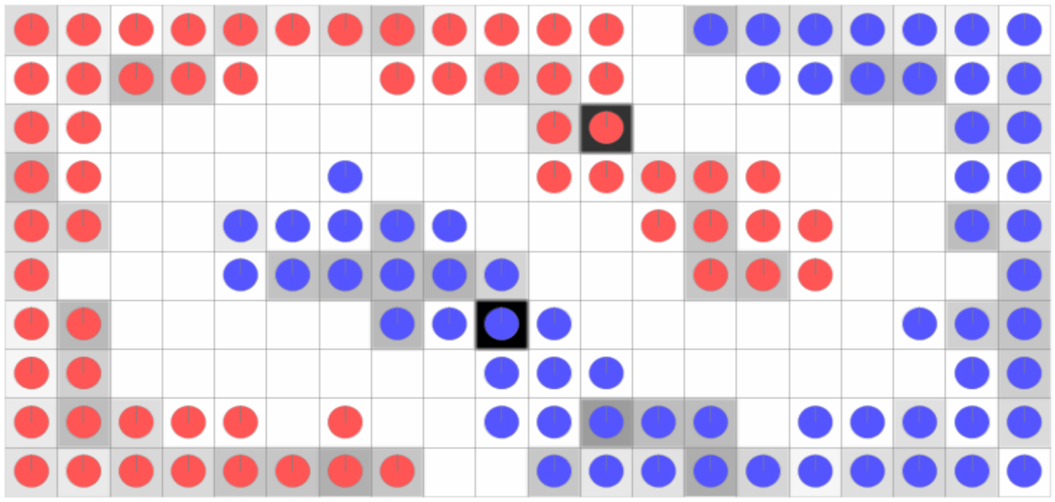


Figure 5.12: Data shift comparisons between pairs of SOMs trained on the equidistant Gaussian cluster datasets “0” (top) and “6” (bottom).

the input data depends a lot on their random initialisation, so the resulting SOMs differ. On the other hand, dataset “6” has a clear structure to which each SOM can fit, making the differences between SOMs with different random initialisations much smaller.

The structure of the input data also becomes visible in parts of the data shifts visualisations of the same SOMs. Figure 5.12 shows these visualisations where the two middle rows of units in each SOM are selected. The data vectors from the selections behave differently in both maps. For the maps trained on dataset “0”, the data vectors scatter widely on the right SOM. For the maps trained on dataset “6”, however, the data vectors are mapped into the two middle columns of the right map almost perfectly. Taking into account the mirrored orientation of the two SOMs, this means that the same data vectors get mapped to almost the same positions in those two maps.

So the data shifts and comparison visualisations back up each other and confirm the findings from the cluster shifts visualisation.



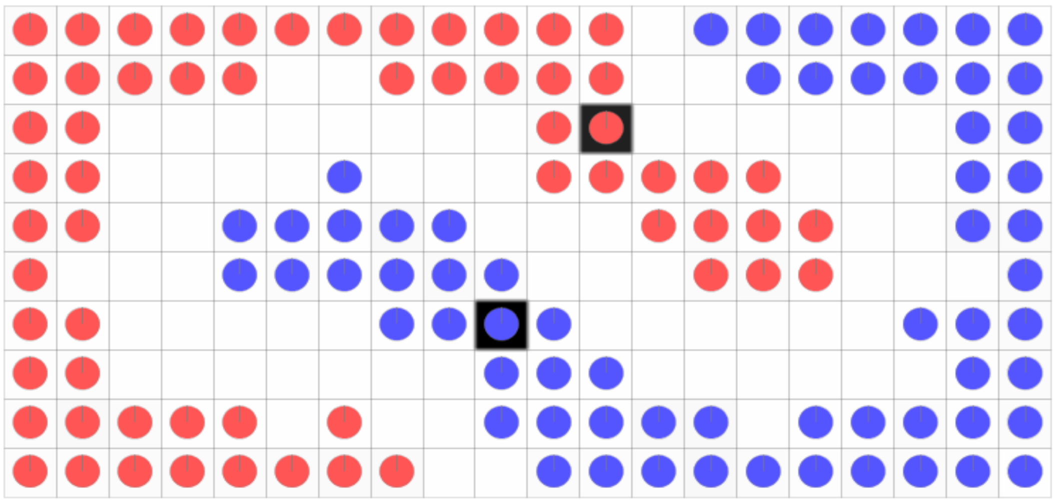


Figure 5.13: “Comparison – Mean” (top) and “Comparison – Variance” (bottom) visualisations for two SOMs trained on the intertwined rings dataset.

5.3 Comparison Visualisation on the Intertwined Rings Dataset

A useful application for the comparison visualisation is the detection of breaches in topology, which it can show by a high mean distance on a unit or area while the surrounding units have a low mean distance. In order to test this, the “intertwined rings dataset” was used (see Section 4.2).

Eight SOMs were trained on it with a size of 20×10 units, initial neighbourhood size $\sigma_0 = 10$, initial learning rate $\alpha_0 = 0.75$, and 10,000 iterations. The random seed differed in order to produce different projections, being 42, 21, 3515, 58, 486, 789, 8, and 999 respectively for SOMs “1”, “2”, “3”, “4”, “5”, “6”, “7”, and “8”.

The analysis was started by comparing SOM “1” with the SOM “3” using the “Comparison – Mean” and the “Comparison – Variance” visualisation. The results,

showing SOM “1” with a class information visualisation put on top, can be seen in Figure 5.13.
Obviously, the two rings were broken during the projection. However, there is no unit with samples from both classes mapped onto it, rather the two classes are mapped into well-separated connected areas that can be seen as broken circles.
The “Comparison – Mean” visualisation mostly shows low to medium values for the mean distance, but two units in the middle of the SOM catch the eye immediately because of their very high mean distance value (which makes them black). When looking a the “Comparison - Variance” visualisation, it becomes clear that these two units are the only ones to have significant variance. It can be suspected that the breach in topology that results from breaking the rings occurs at these locations in the data space – in the *other* SOM, SOM “3”.
In order to minimise the influence of small pairwise distances of vectors (resulting from vectors being mapped onto neighbouring units) the threshold in the “Comparison – Mean” visualisation was set to 3. This means that the pairwise euclidean distance between two vectors in SOM “3” (which lie on the same unit in SOM “1”) must be more than 3 to count towards the mean distance. The result should be that only vectors that moved very far away from each other show up in the visualisation. Sure enough, only the two noticeable units have a mean distance greater than zero with this setting.
To see what happens with the data vectors from these units, the data shifts visualisation was employed and put on top of the “Comparison – Variance” visualisation. (The parameters for the data shifts visualisation are not very important here but are given nonetheless: source threshold: 1, target threshold: 1, cumulative “on”, absolute count thresholds: stable: 5, outlier: 1.)
Figure 5.14 shows two snapshots of the combined visualisation. The upper snapshot shows the data shifts for the area around the upper unit of interest (highlighted in orange); SOM “3” is displayed at the top and SOM “1” below it. As was suspected, this unit marks the location in the data space where the “red” ring is broken in SOM “3”. The same is true for the lower unit of interest and the “blue” ring, as can be seen in the lower snapshot.

To further test the capabilities of the comparison visualisation, SOM “1” was then compared not only with SOM “3” but with all other seven SOMs, the threshold again set to 3. The resulting “Comparison – Variance” visualisation (combined with the class information) can be seen in Figure 5.15.
This visualisation now shows not only two areas of interest but five: the two in the middle already seen in the previous visualisations and additionally two at the ends of the blue conglomerate and one at the lower end of the red conglomerate.
It appears that these points of interest mark the breakage points of the rings which are most likely to be generated by SOMs. There are two breakage points on each

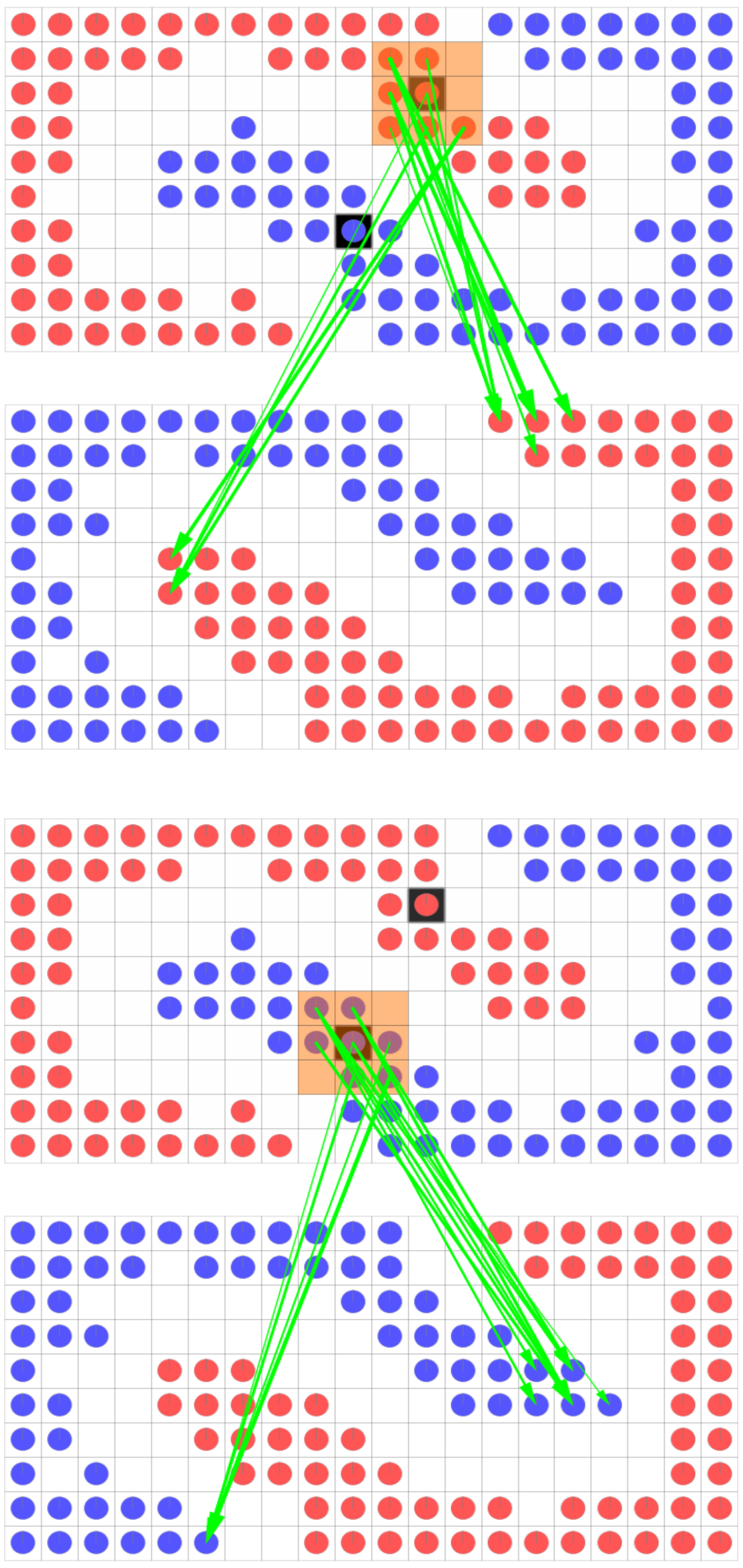


Figure 5.14: Combined data shifts and comparison visualisations for the intertwined rings dataset.

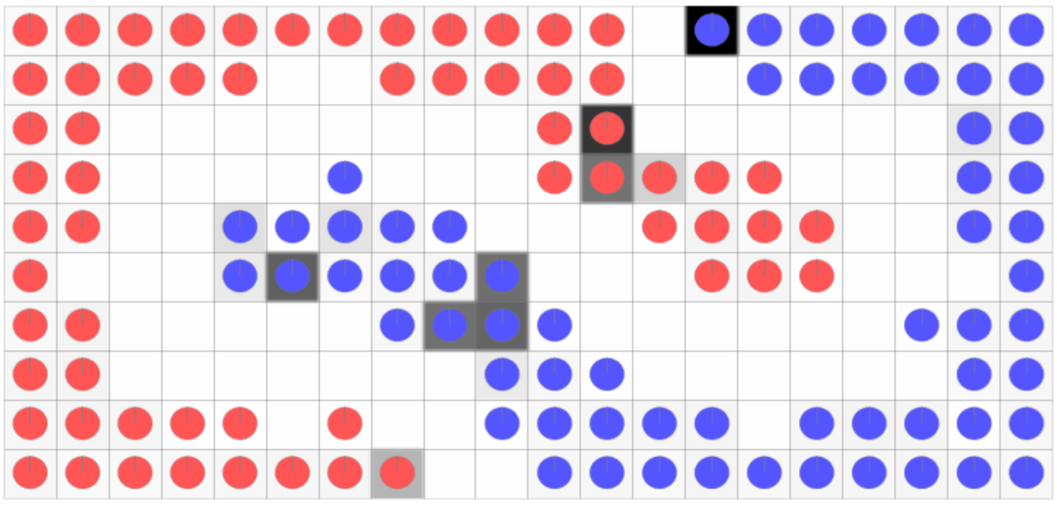


Figure 5.15: “Comparison – Variance” visualisation for eight SOMs trained on the intertwined rings dataset.

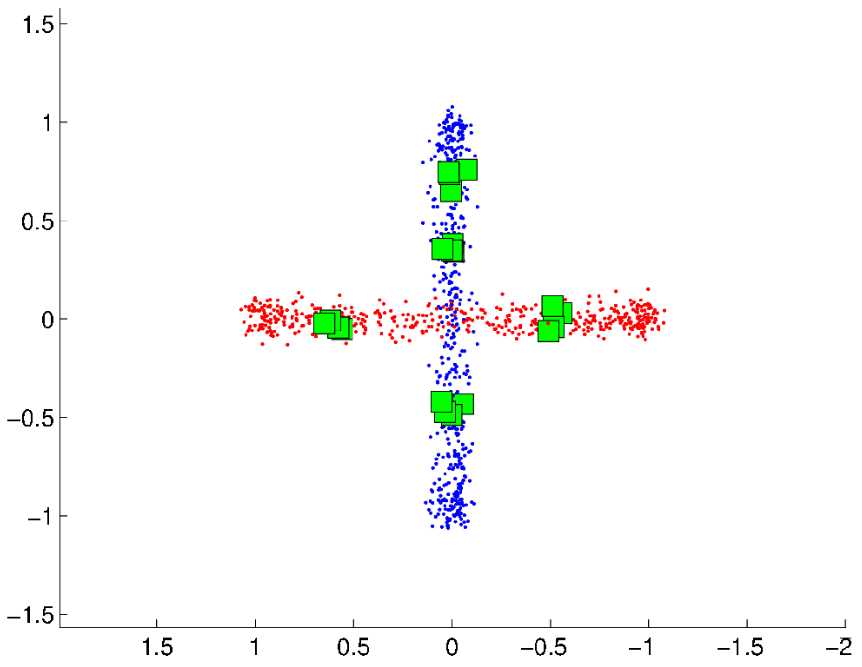


Figure 5.16: Plot of the input space for the intertwined rings dataset with the points of interest marked in green.

ring and they are symmetrical.
To check this hypothesis, the vectors on the units of interest can be marked in the plot of the original data, which results in Figure 5.16 (markers in green). And indeed, they show up as symmetrical points, one on each half circle, although the two ends of the blue conglomerate (which end up on the "upper" half of the blue ring in the plot) do not overlap as nicely as one might expect.
When looking back to Figure 5.14 it becomes clear that SOM "1" produced just the mirror images of both breakage points in SOM "3".

5.3.1 Checking Results with Data Shifts Visualisation and Cluster Shifts Visualisation

The intertwined rings dataset was checked also with the data shifts and the cluster shifts visualisations. Interesting findings of the data shifts visualisation were already given above in Figure 5.14: The breaches in topology become visible in the data shifts visualisation.
Apart from the breaking points of the rings, the topology is preserved well by the SOMs' projections, as can be seen from the comparison visualisation with low mean pairwise distances (see Figure 5.13). This is supported by another data shifts visualisation, see Figure 5.17, which compares SOM "1" (above) with SOM "3" (below) (and has the same parameters as Figure 5.14). The data vectors from a continuous connected area in one SOM move to a similar area in the second SOM. The rings are rotated in comparison to each other, but the data vectors remain within their small neighbourhoods in both SOMs. For clarity, only a part of the data shifts is shown in the figure, but the other shifts look similar.

A cluster shifts visualisation on the same two maps also yields interesting results, see Figure 5.18. When clustering the map into two clusters, each cluster holds the majority of data vectors from one ring and a small part of the other ring. This is to be expected because the intertwined rings dataset is not linearly separable. When separating the 3-dimensional rings by a plane, a small portion of each ring will always be cut off and assigned to the cluster of the other ring. In SOM "1", the cluster borders run through the points where the breaches in topology occur, the points where the rings are broken up in SOM "3". The breakage area of the blue ring is selected (marked in orange) in Figure 5.18. The breakage area for the red ring isn't shown but looks analogue.

Apparently, the data shifts and cluster shifts visualisations back up the results from the comparison visualisation about the breaches in the rings' topology.

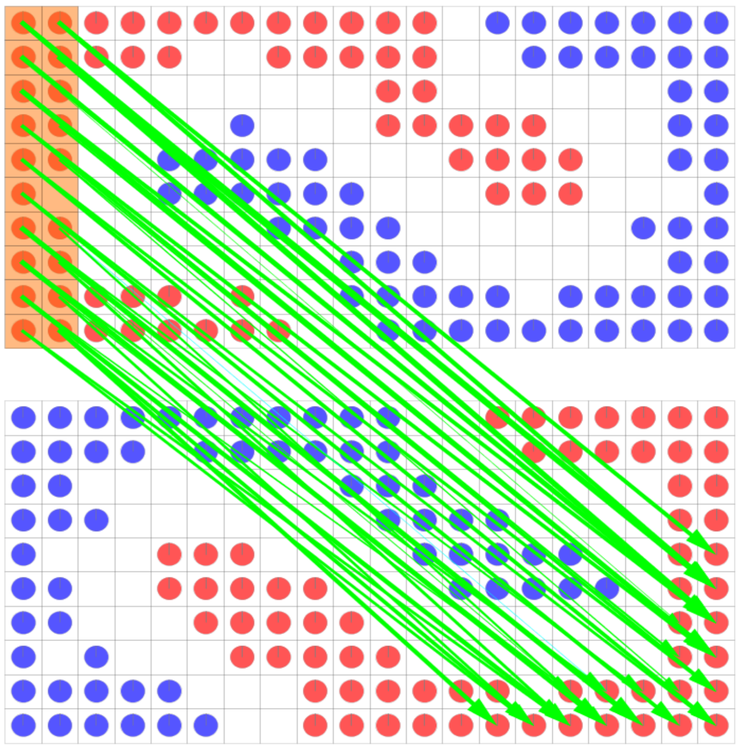


Figure 5.17: Data shifts visualisation of two SOMs trained on the intertwined rings dataset.

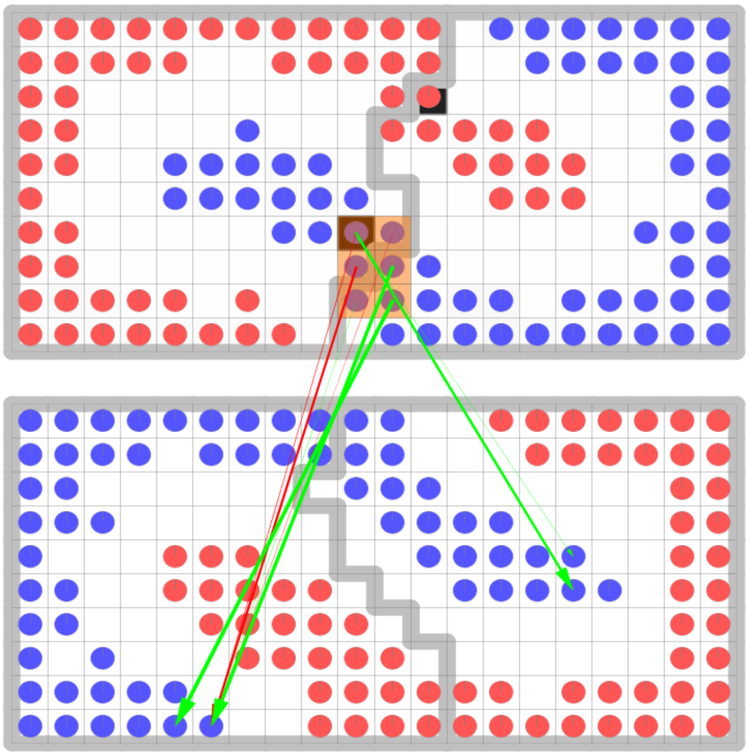


Figure 5.18: Cluster shifts visualisation of two SOMs trained on the intertwined rings dataset.

5.4 Assessing SOM Quality on Equidistant Gaussian Cluster Dataset “5”

To test how the proposed visualisations display differences in SOM quality, eight “well-trained” and two “badly-trained” SOMs were built on the equidistant Gaussian cluster dataset “5”, which contains 3 equidistant Gaussian clusters with distance 5 (see Section 4.1).

The parameters for the badly-trained maps were tweaked to impede a perfect fit of the map to the data, while the parameters of the well-trained maps were set to allow for a good fit to the data. The parameters that were the same for all maps were: a map size of 6×6 units, an initial learning rate of $\alpha_0 = 0.75$, and 500 training steps. The maps differed in their random seeds (to guarantee their independence) and their initial neighbourhood size (to tweak their fit to the data): The random seeds were 1807, 4209, 289, 2431, 4571, 2079, 3506, 4659, 1185, and 4088 respectively. The initial neighbourhood size σ_0 was 6 for SOMs “1” to “8” (well-trained) and 2 and 1 respectively for SOMs “9” and “10” (badly-trained). The small neighbourhood sizes were chosen to enforce errors in the SOMs’ preservation of topology.

The maps were then analysed and compared with the help of the three visualisations.

5.4.1 Data Shifts Visualisation

Of the many possible combinations of maps for data shifts visualisations, two were selected: A badly-trained map and a well-trained map were both compared with another well-trained map. Figure 5.19 shows the resulting two data shifts visualisations: At the top, SOM “10” (badly-trained) is compared with SOM “5” (well-trained); at the bottom, SOM “1” (well-trained) is also compared with SOM “5”. The visualisations’ parameter settings were: Cumulative mode on, source threshold: 1, target threshold 1, absolute stable count threshold: 5, absolute outlier count threshold: 1.

The first snapshot shows that the data vectors from the two middle rows of SOM “10” spread over almost the whole map in SOM “5”. Apparently, the two SOMs don’t represent the input data in a similar way, which is a hint that at least one of the maps doesn’t fit the data optimally.

The second snapshot, however, reveals that the vectors from the two middle rows from SOM “1” are mapped (almost perfectly) to the two middle columns of SOM “5”. This indicates that the SOMs’ orientations are mirrored in comparison to each other and that they both represent the data’s inner structure in a similar way.

So two SOMs (“1” and “5”) represent the input data in a similar way while the third SOM differs from them. This suggests that SOMs “1” and “5” fit better to the input data than SOM “10” – if they aren’t somehow similar by accident, which may be ruled out by comparing more SOMs with varying parameters with them.

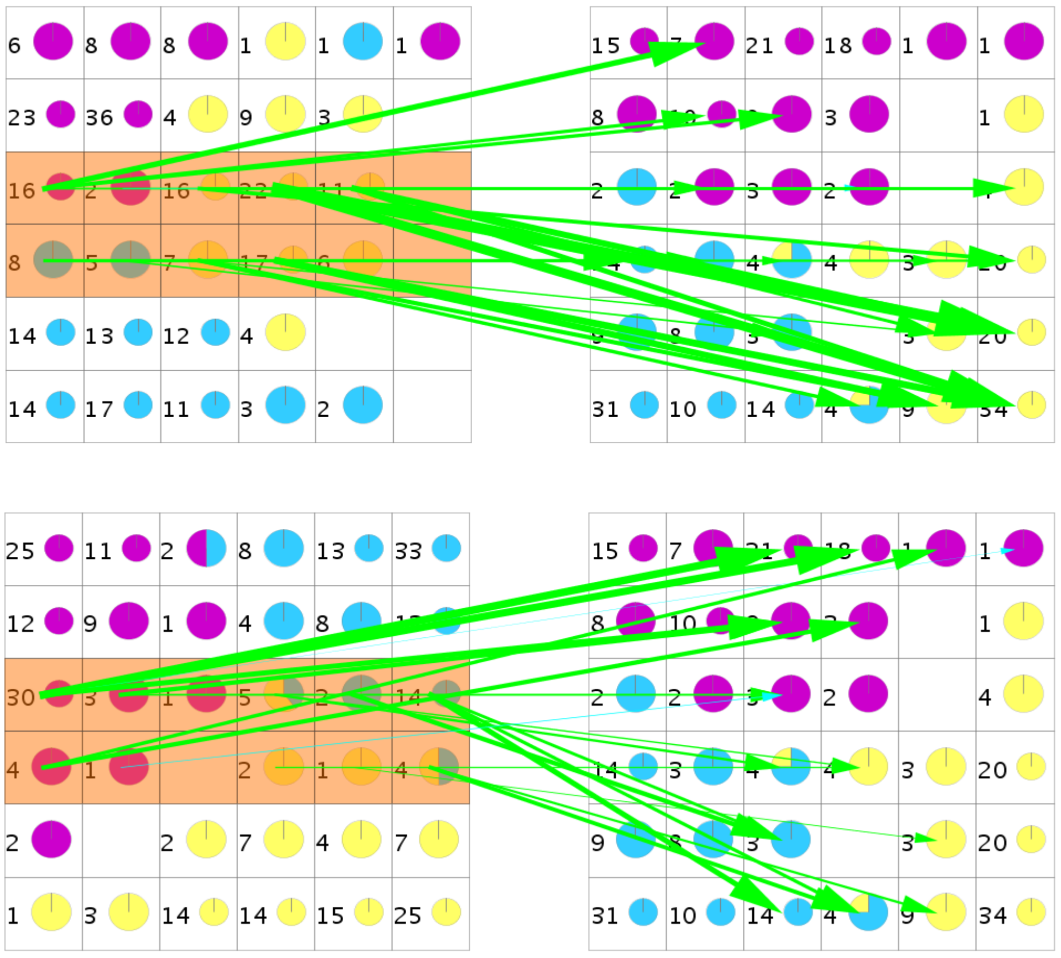


Figure 5.19: Data shifts visualisation of SOMs trained on equidistant Gaussian cluster dataset “5”, comparing SOMs “10” and “5” (top) and “1” and “5” (bottom).

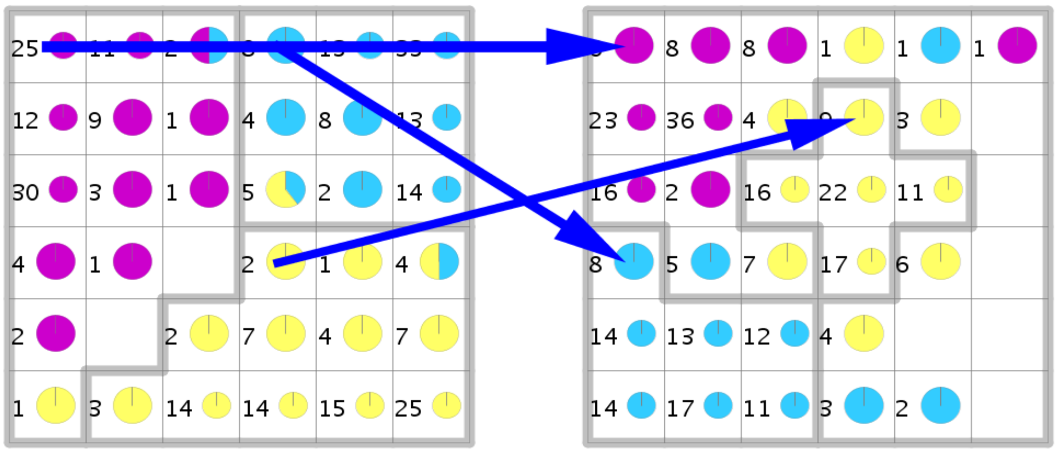


Figure 5.20: Cluster shifts visualisation of SOMs trained on equidistant Gaussian cluster dataset "5", comparing SOMs "1" and "10".

Thus, it gives an indication for the SOMs' quality.

5.4.2 Cluster Shifts Visualisation

A cluster shifts visualisation with the number of clusters set to three was done next. It compares SOMs "1" (well-trained) and "10" (badly-trained), see Figure 5.20. The different shape of the clusters in the two SOMs gives a strong indication that they don't represent the input data in a similar way. In SOM "1", three about equally sized and shaped clusters lie in three parts of the map. In SOM "10" there are two small and one large cluster, all with a different shape, and the large cluster clasps around one of the small clusters. The question is, which of the maps represents the data's structure better. It may be the case that SOM "1" oversimplifies the input data's inner structure into three similarly shaped clusters, but it seems more likely that the obscure shapes of the clusters in SOM "10" are the result of a badly trained map, especially as the units on the right-hand side border of SOM "10" have almost no data vectors mapped to them.
So, comparing the shapes of corresponding clusters in different SOMs may indicate the SOMs' quality as well.

5.4.3 Comparison Visualisation

Finally, four "Comparison – Mean" visualisations were generated, which compare the SOMs "1", "5", "9", and "10" respectively with all other SOMs. The visualisations are shown in Figure 5.21. It becomes apparent that the well-trained maps ("1" and "5") have very few units with high mean pairwise distances and a lot of units with low and approximately equal mean pairwise distances. (The units with no or only one data vector on them of course have a mean pairwise distance of 0 and should be disregarded here.) The badly-trained maps ("9" and "10") have more units with

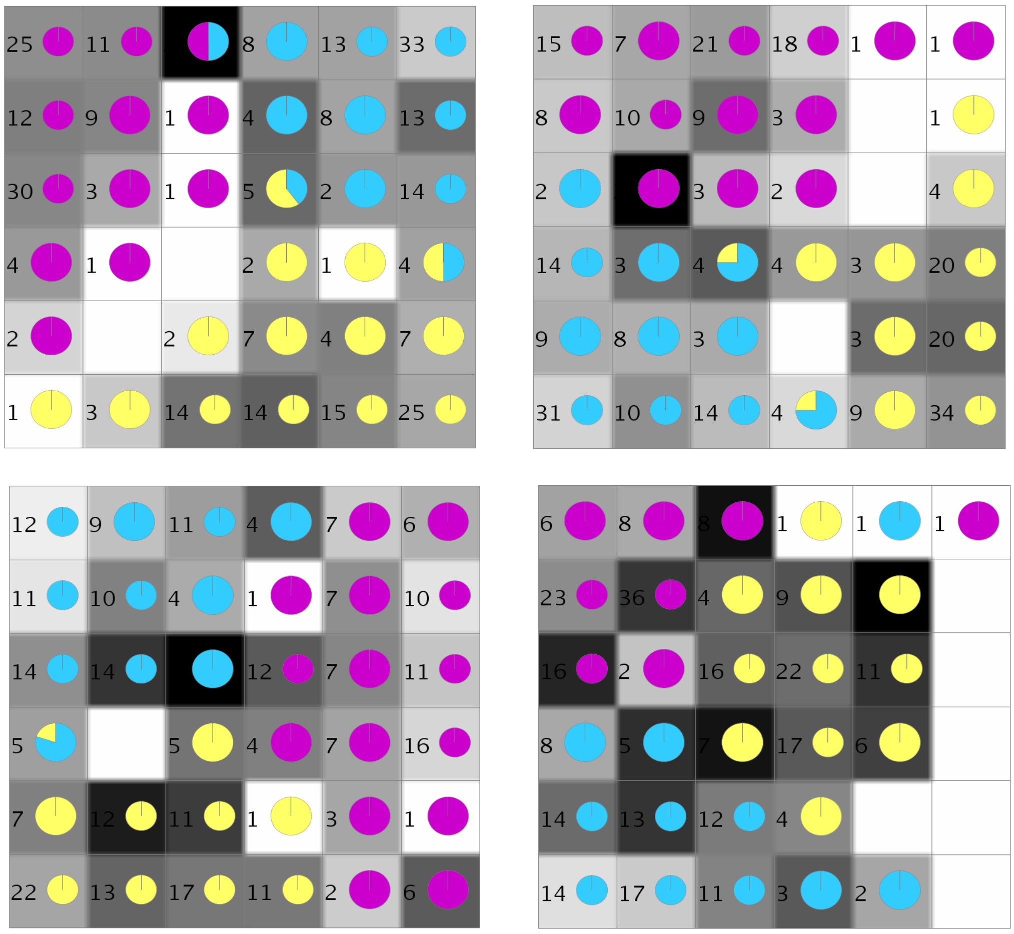


Figure 5.21: "Comparison – Mean" visualisations on ten SOMs trained on equidistant Gaussian cluster dataset "5". Top left: SOM "1", top right: SOM "5", bottom left: SOM "9", bottom right: SOM "10".

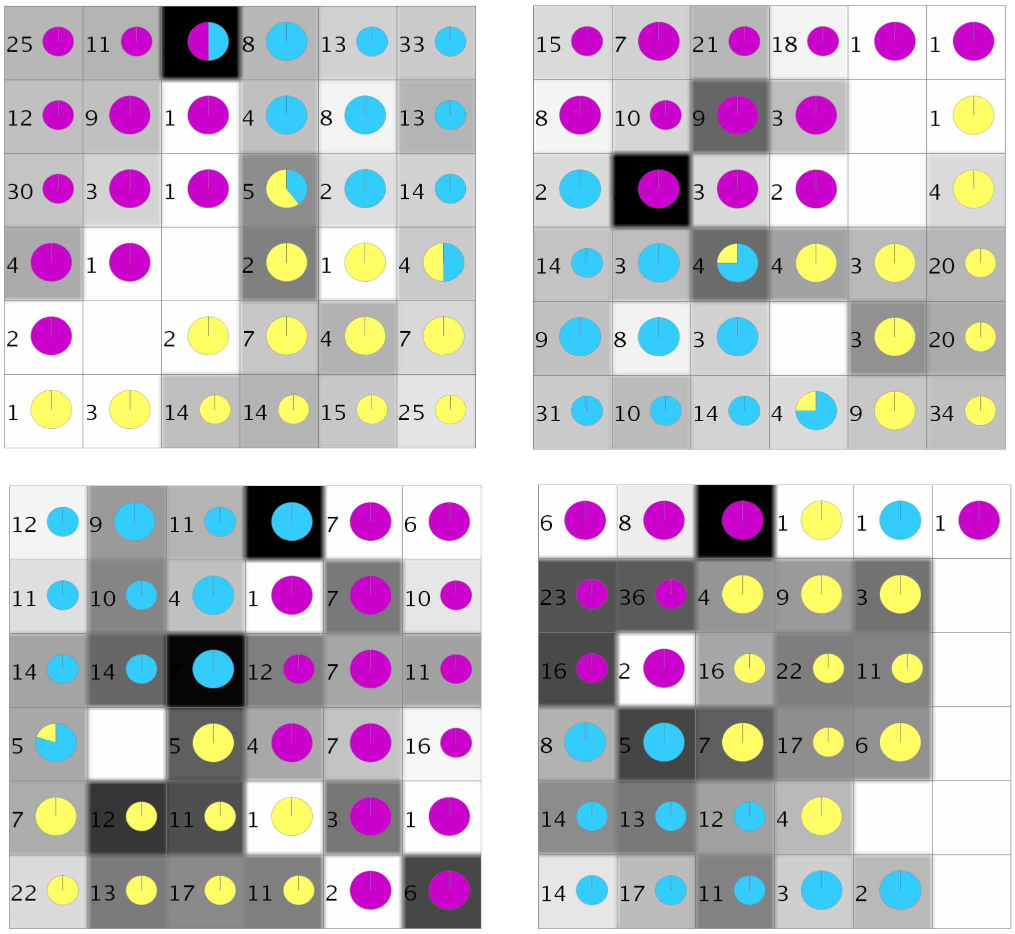


Figure 5.22: ‘
‘Comparison – Mean” visualisations on four SOMs trained on equidistant Gaussian cluster dataset “5”.]“Comparison – Mean” visualisations on four SOMs trained on equidistant Gaussian cluster dataset “5”. Top left: SOM “1”, top right: SOM “5”, bottom left: SOM “9”, bottom right: SOM “10”..

high mean distances and more variation of values between the units. Apparently, the variation between the units of the same map is lower for the good maps and higher for the bad maps. This seems logical because good maps should fit well to the input data and thus be very similar, whereas in the bad maps the influence of random factors and parameter settings is higher, which produces inhomogenous maps. (Remember that the colours can’t be compared directly between the visualisations, but that the variation within a map can, of course, be read from each visualisation.)
So it is possible here to discern good and bad maps by looking at the comparison visualisations and finding a high variation between the units there. This is also true if the proportion of good maps in the comparison is smaller. In the above case, the set of compared SOMs contained eight good maps and two bad maps. However, the results are similar if it contains only two good maps and two bad maps, as can be seen in Figure 5.22. But if there is only one good map in three bad maps, for

example, the distinction can't be made anymore just from looking at the comparison visualisation, so this method has its limits.

All in all, it seems that the three visualisations can be used to asses SOM quality, to indicate which of a set of compared maps are better and which are worse. However, all three visualisations compare SOMs with each other and don't compare the map in the output space with the input data. So all indications of quality are relative to the compared maps and not absolute. To get an absolute measurement of a SOM's fit to its input data, quality measures like those mentioned in Section 2.4 can be used.

5.5 Test on Real-life Data: Iris Dataset

A test on real-life data was done for all three visualisations with the Iris dataset (see Section 4.4). Two maps were trained on the data, both with an initial learning rate $\alpha_0 = 0.75$, 10,000 iterations and a random seed for prototype vector initialisation of 42, though with different sizes and initial neighbourhood sizes: one with 5×5 units and $\sigma_0 = 5$ and the other with 10×10 units and $\sigma_0 = 10$.
Then both maps were compared with all three visualisations to find out how the data vectors moved, and what the differences and similarities of the maps are.

5.5.1 Data Shifts Visualisation

First, a data shifts visualisation was done with the following parameters:

- A source threshold of 0 and a target threshold of 1, to account for the far bigger size of the second map: Of course, the data vectors will have more units to spread over in the second SOM, so the neighbourhood for the second SOM must naturally be larger.
- The cumulative option switched on, to make full use of the target neighbourhood radius (target threshold) in the second SOM.
- An absolute count threshold of 4 for stable shifts and 1 for outlier shifts, because the minimum number of data vectors on a unit in the smaller SOM is 3, and 4 is just a bit above it.

Information about the data vectors' classes was added to the visualisations, revealing the real inner structure of the data. The setosa class is marked in yellow, the versicolor class in dark blue, and the virginica class in light blue. This shows that the position of the data on the second map is mirrored in comparison to the first map.

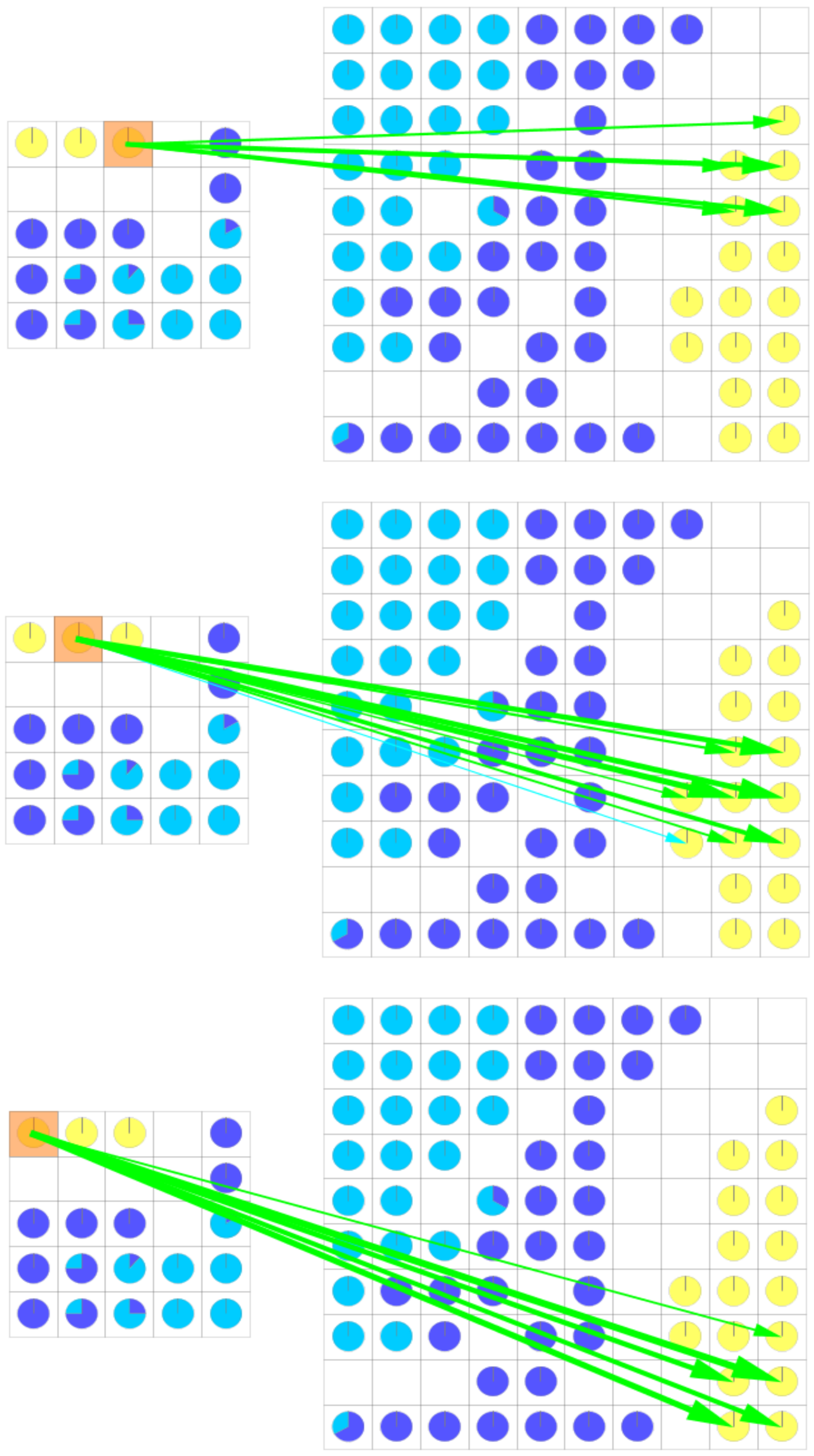


Figure 5.23: Data shifts visualisation: data vectors from the setosa class spread over more units in a larger SOM.

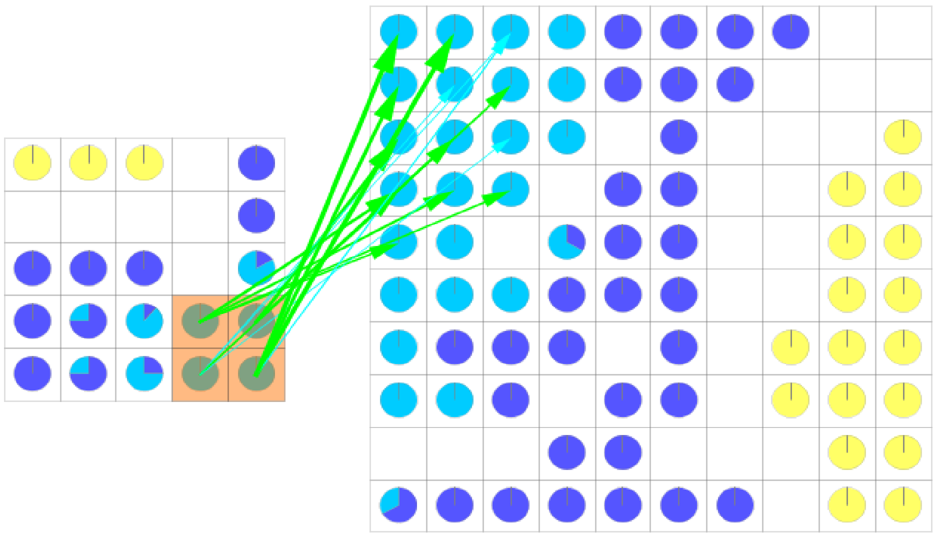


Figure 5.24: Data shifts visualisation: Data vectors from the core of the virginica class.

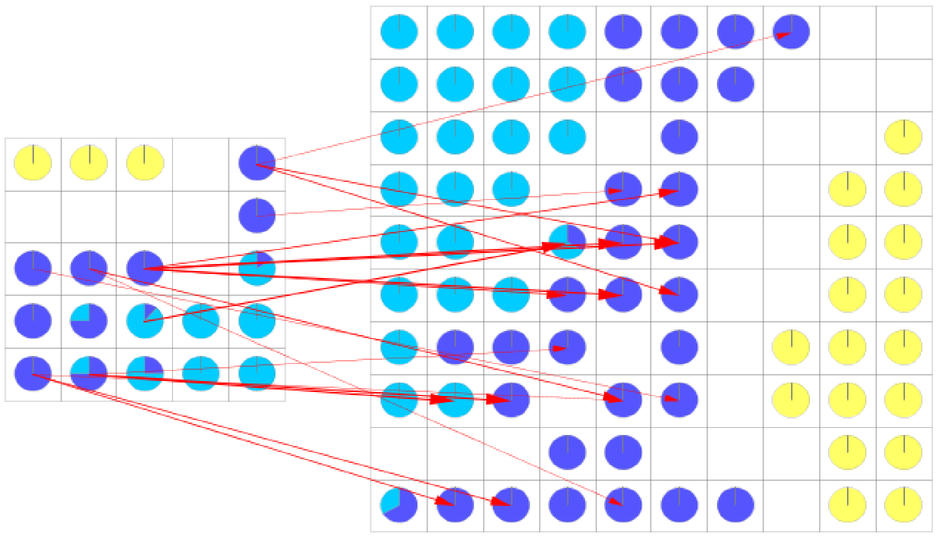


Figure 5.25: Data shifts visualisation: All outlier shifts in the data shifts visualisation.

Interesting results can be seen when looking at the setosa class (see Figure 5.23). Its 50 vectors all amass on 3 units in the top left corner of the small map, with a gap of empty units (onto which no data vectors were mapped) around them. Similarly, on the large map, the vectors come to lie in a clearly separated, elongated area on the right-hand border. The gap shows that the setosa samples separate themselves clearly from the other two classes (this is confirmed by looking at the statistical analysis of the Iris dataset [4]). The data shifts visualisation also shows how nicely the topology of the data is preserved in both SOMs: The vectors from the rightmost setosa unit in the small map is mapped onto the top of the elongated setosa area in the large map; the vectors from the middle unit in the small map are mapped onto the middle of the elongated area in the large map; and the vectors on the leftmost unit are mapped onto the bottom of the elongated area. This fits perfectly to the mirrored orientation of the SOMs.
The cohesive core of the virginica area is also clearly preserved in both SOMs, though its topological structure isn't repeated in both SOMs as strongly as with the setosa class. See Figure 5.24 for the mapping between the two SOMs.
The borders of the virginica area and the versicolor area, however, are not as cohesive and spread over a wider area than the other two classes. Figure 5.25 shows only the outlier shifts for the data shifts visualisation of the two SOMs. Most of the outlier shifts emerge from units in the versicolor area or the border of the virginica area.
The conclusion that can be drawn is that the setosa class and the core of the virginica class seem to be projected onto the SOMs' grids in a way that preserves the topology well, and the distinction between the setosa class and the other two classes is very clear. The borders of the virginica class and the versicolor class, however, overlap and are rendered differently on both maps.

5.5.2 Cluster Shifts Visualisation

To further investigate the properties of the dataset, the cluster shifts visualisation was used. Different numbers of clusters were tried out on the two SOMs: Figure 5.26 shows three different cluster visualisations with two, three, and four clusters respectively (from top to bottom).
If the number of clusters is two, the clustering algorithm puts all setosa samples into one cluster and all versicolor and virginica samples into the other cluster. The cluster assignment confidence here is 100% for both mappings. This, again, shows that the setosa class is clearly distinct from the other two, while versicolor and virginica are overlapping.
For the next case, if there are three clusters, all possible mappings and their confidences are given in Table 5.2. The setosa cluster is still the same as before and its mapping has 100% confidence. The other two clusters each represent one of the other two classes in the small map, but in the large map the virginica cluster gets

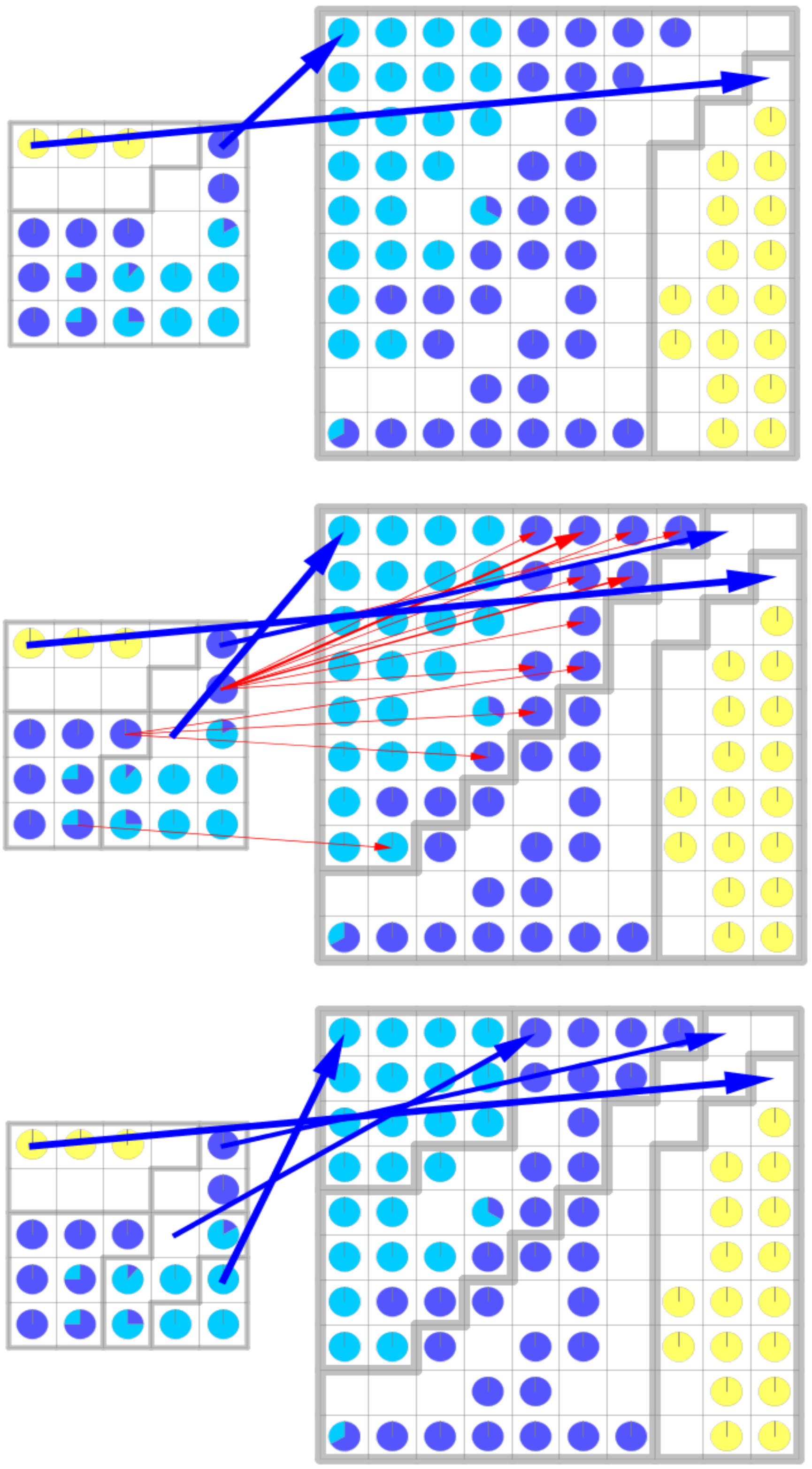


Figure 5.26: Cluster shifts visualisations of the Iris dataset with 2, 3, and 4 clusters (top to bottom).

Cluster in 1st SOM	**Cluster in 2nd SOM**	**Confidence**
setosa	setosa	100%
virginica	virginica	100%
versicolor	versicolor	69%
versicolor	virginica	31%
all others		0%

Table 5.2: Cluster assignment confidences for two maps trained on the Iris dataset, clustered into three clusters.

Cluster in 1st SOM	**Cluster in 2nd SOM**	**Confidence**
setosa	setosa	100%
virginica core	virginica core	100%
versicolor/virginica	versicolor/virginica	74%
versicolor	versicolor	69%
versicolor	versicolor/virginica	31%
versicolor/virginica	virginica core	26%
all others		0%

Table 5.3: Cluster assignment confidences for two maps trained on the Iris dataset, clustered into four clusters.

assigned quite a few versicolor samples as well. These show up as the outlier shifts drawn in in red. The virginica clusters' assignment confidence is 100%, the versicolor clusters' confidence is only 69%.

This points to a weakness in the confidence measure which only considers the direction from the first (the small) to the second (the large) map, and not both ways. The problem may be alleviated by changing the confidence measures as proposed in Chapter 6.

The last snapshot shows the cluster shifts visualisation with four clusters (with no stable or outlier shifts drawn in). All possible mappings and their confidences are given in Table 5.3. There is one cluster for the setosa class (assignment confidence 100%), one cluster for the core of the virginica class (assignment confidence 100%), one cluster for the versicolor class (assignment confidence 69%), and one cluster for the border between versicolor and virginica (assignment confidence 74%). This shows again that at their borders the versicolor class and virginica class are hard to distinguish.

5.5.3 Comparison Visualisation

Finally, a comparison visualisation ("Comparison – Mean") was used to find the units in the smaller SOM where the projection onto the two-dimensional SOM-grid is unstable – see Figure 5.27. The minimum pairwise distance threshold was set to 2.5, to reduce the impact of the bigger size of the larger SOM: The data vectors

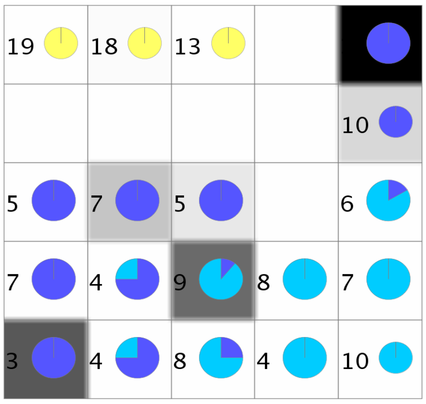


Figure 5.27: “Comparison – Mean” visualisation of the Iris dataset.

spread over more units in the larger SOM, thus the vectors that lie on one unit in the small SOM will spread over a couple of neighbouring units in the large SOM. This will distort the conclusions we wish to draw from the visualisation, so the threshold is used to compensate for the spread.

The units with the high mean pairwise distances (marked in shades of grey) all are either on the border between the versicolor and virginica classes or within the versicolor class. This points to the relative instability of the projection of the versicolor class onto the SOM-grid: data vectors from the versicolor class are projected differently in both SOMs. Yet again, these results suggest that the setosa class and to some extend the core of the virginica class are well-defined and distinct, while the border between virginica and versicolor and versicolor class itself are a relatively unstable area in a SOM projection.

So the results from the three visualisations support and reinforce each other – and thus may allow to draw useful conclusions for other, not so well-known datasets as well.

5.6 Summary

This chapter described the experiments conducted on the visualisations introduced in Chapter 3 on the datasets from Chapter 4.

It was found that they can be used to detect differences between the projections of two or more different SOMs, breaches in topology, rotations and reflections of SOMs relative to each other, clusters and sub-clusters, and clusters corresponding to each other. They give indications on the compared SOMs’ relative quality, seem to support and reinforce each other’s findings and may be useful to explore other datasets.

Chapter 6

Future Work

The visualisations described in this work may prove useful but surely can be improved upon to work faster, better, or give the user more options to fine-tune the visualisations' results.
This chapter gives a few ideas and suggestions on possible improvements that could be useful.

- The data shifts visualisation could be extended by making the neighbourhoods more user-adjustable. Currently, the neighbourhoods can differ in size/radius, but will always be circular. Other shapes of neighbourhoods could be added. Also, instead of defining the neighbourhood through shape and radius, one could just take the k nearest neighbours and use them as a neighbourhood.

- To improve the cluster shifts visualisation, different methods for establishing the confidence of a the cluster mapping could be tried out. Instead of computing the percentage of data vectors which moved from a cluster in the first SOM to a cluster in the second SOM, one could do it the other way round: compute the percentage of data vectors which moved from a cluster in the second SOM to a cluster in the first SOM. Or one could combine these two measures, possibly weighing them differently. This may result in a confidence measure more closely reflecting the similarity of the clusters but may increase the computational cost.

- Another way to improve the cluster mapping could be to change the method by which the entries from the table of confidences are picked. Instead of always choosing the highest available confidence, one could use an optimisation algorithm to maximise the overall sum of confidences. This could achieve better cluster mappings.

- The comparison visualisation could be extended by adding other distance functions to compute pairwise distances between vectors. For the cluster distance measure, other linkage functions such as complete linkage or average linkage could be tried out.

Of course this list is by no means exhaustive or claims to tackle all problems that may arise with the visualisations. It is merely a starting point for further development.

Chapter 7

Conclusion

This work introduced three new Self-organising Map visualisations which enable the user to compare two or more SOMs trained on the same data with different parameters: The data shifts visualisation, the cluster shifts visualisation and the comparison visualisation. Datasets were constructed to test the new visualisations. Experiments were conducted and analysed, and it was found that the visualisations can be helpful in determining the influence of parameter choice on the SOMs; they can be employed for finding breaches in topology or areas in the SOMs were the projection onto two dimensions is unstable and ambiguous. Likewise, particularly stable mappings which preserve the topology in the original data space especially well can be discovered.
During the experiments, possibilities for enhancement of the visualisations became apparent and solutions were proposed.
All in all, the proposed visualisations can be used in data exploration to gain knowledge about datasets and especially about SOMs trained on that data.

List of Tables

List of Figures

Bibliography

[1] Hans-Ulrich Bauer and Klaus R. Pawelzik. Quantifying the neighborhood preservation of self-organizing feature maps. *IEEE Transactions on Neural Networks*, 3(4):570–579, July 1992.

[2] Christopher M. Bishop. *Neural Networks for Pattern Recognition.* Oxford University Press, 1995.

[3] Michael Dittenbach, Robert Neumayer, and Andreas Rauber. Playsom: An alternative approach to track selection and playlist generation in large music collections. In *Proceedings of the First International Workshop of the EU Network of Excellence DELOS on Audio-Visual Content and Information Visualization in Digital Libraries (AVIVDiLib 2005)*, pages 226–235, Cortona, Italy, May 4-6 2005.

[4] Ronald A. Fisher. The use of multiple measurements in taxonomic problems. In *Annual Eugenics, 7, Part II*, pages 179–188, 1936.

[5] Teuvo Kohonen. Self-organized formation of topologically correct feature maps. *Biological Cybernetics*, 43:59–69, 1982.

[6] Teuvo Kohonen. *Self-Organizing Maps.* Springer, Berlin, Heidelberg, 1995.

[7] Joseph B. Kruskal and Myron Wish. Multidimensional scaling. *Sage University Paper Series on Quantitative Application in the Social Sciences*, 1978.

[8] Elias Pampalk. Aligned self-organizing maps. In *Proceedings of the Workshop on Self-Organizing Maps (WSOM'03)*, pages 185–190, Hibikino, Kitakyushu, Japan, September 11-14 2003. Kyushu Institute of Technology.

[9] Elias Pampalk, Werner Goebl, and Gerhard Widmer. Visualizing changes in the structure of data for exploratory feature selection. In *Proceedings of the Ninth ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, pages 157–166, Washington DC, August 24-27 2003. ACM.

[10] Elias Pampalk, Andreas Rauber, and Dieter Merkl. Using smoothed data histograms for cluster visualization in self-organizing maps. In *Proceedings of the*

Intl Conf on Artificial Neural Networks (ICANN 2002), pages 871–876, Madrid, Spain, August 27-30 2002.

[11] Georg Pölzlbauer. Survey and comparison of quality measures for self-organizing maps. In Ján Paralič, Georg Pölzlbauer, and Andreas Rauber, editors, *Proceedings of the Fifth Workshop on Data Analysis (WDA'04)*, pages 67–82, Sliezsky dom, Vysoké Tatry, Slovakia, June 24–27 2004. Elfa Academic Press.

[12] Angela Roiger. Analyzing, labeling and interacting with soms for knowledge management. Master's thesis, Department of Software Technology and Interactive Systems, Vienna University of Technology, Vienna, March 2007.

[13] Pang-Ning Tan, Michael Steinbach, and Vipin Kumar. *Introduction to Data Mining*. Addison-Wesley, 2006.

[14] Alfred Ultsch and H. Peter Siemon. Kohonen's self-organizing feature maps for exploratory data analysis. In *Proceedings of the International Neural Network Conference (INNC'90)*, pages 305–308, Dordrecht, Netherlands, 1990. Kluwer.

[15] Alfred Ultsch and C. Vetter. Selforganizing feature maps versus statistical clustering: A benchmark. Technical Report 9, Dept. of Mathematics and Computer Science, University of Marburg, Germany, 1994.

[16] Jarkko Venna and Samuel Kaski. Neighborhood preservation in nonlinear projection methods: An experimental study. In *Artificial Neural Networks—ICANN 2001*, pages 485–491. Springer, Berlin, 2001.

[17] Juha Vesanto. Som-based data visualization methods. *Intelligent Data Analysis*, 3(2):111–126, 1999.

[18] Juha Vesanto and Esa Alhoniemi. Clustering of the self-organizing map. *IEEE Transactions on Neural Networks*, 11(3):586–600, May 2000.

[19] Juha Vesanto, Mika Sulkava, and Jaakko Hollmén. On the decomposition of the self-organizing map distortion measure. In *Proceedings of the Workshop on Self-Organizing Maps (WSOM'03)*, pages 11–16, Hibikino, Kitakyushu, Japan, September 2003. Kyushu Institute of Technology.

[20] Joe H. Ward. Hierarchical grouping to optimize an objective function. *Journal of the American Statistical Association*, 58(301):236–244, March 1963.