Snap-Drift Self Organising Map

Dominic Palmer-Brown and Chrisina Draganova

London Metropolitan University, Faculty of Computing,
166-220 Holloway Road, London N7 8DB, UK
d.palmer-brown@londonmet.ac.uk, c.draganova@londonmet.ac.uk

Abstract. A novel self-organising map (SOM) algorithm based on the snap-drift neural network (SDSOM) is proposed. The modal learning algorithm deploys a combination of the snap-drift modes; fuzzy AND (or Min) learning (snap), and Learning Vector Quantisation (drift). The performance of the algorithm is tested on several well known data sets and compared with the traditional Kohonen SOM algorithm. It is found that the snap mode makes the learning in SDSOM faster than the Kohonen SOM, and that it leads to the formation of more compact maps. When using the maps for classification, SDSOM gives better performance, based on labelled winning nodes, than Kohonen SOM on a variety of data sets.

Keywords: Neural networks, Self-organising Map, Snap-drift, modal learning, unsupervised Learning.

1 Introduction

The standard snap drift neural network (SDNN) algorithm [1-5] has proved invaluable for continuous learning in many diverse applications. It is essentially a simple modal learning method, which swaps periodically between the two learning modes (snap and drift). The unsupervised snap drift algorithm has previously been successfully applied in several domains including the analysis and interpretation of data representing interactions between trainee computer network managers and a simulated network management system [2], where it helped to identify patterns of the user behaviour. It has also been used in feature discovery and clustering of speech waveforms recorded from non-stammering and stammering speakers [3]. Phonetically meaningful properties of non-stammering and stammering speech were discovered, and rapid automatic classification into stammering and non stammering speech was found to be possible. Most recently, snap-drift has been successfully applied to categorising student responses to multiple choice questions in a virtual learning context [5]. In this work, SDNN is deployed in a self-organising map, to ascertain whether the advantages of snap-drift over LVQ alone (drift, without snap) transfer into the formation of topological maps. We are interested in processing speed, classification performance and data visualisation (the shape of the resultant maps).

The self-organising feature map algorithm developed by Kohonen[6] has been used widely in clustering analysis and visualization of high-dimensional data [7]. The SOMs can be also used for pattern classification by applying fine tuning of the map

K. Diamantaras, W. Duch, L.S. Iliadis (Eds.): ICANN 2010, Part II, LNCS 6353, pp. 402 – 409, 2010.

with LVQ learning algorithms [8, 9, 10]. The Kohonen feature map was inspired by the idea that self-organising maps resemble the topologically organised maps found in the cortices of the brain [11]. The Kohonen SOM algorithm is based on unsupervised learning realised by finding the best matching node (the winner) on the map to the input vector and adapting the weights of the winner and the topological neighbourhood nodes. After the training finishes each node on the map identifies a particular input vector and the organisation of the map reflects the original organisation of the input data.

2 Snap-Drift Self-Organising Map

2.1 Snap-Drift Algorithm

Snap-drift learning uses a combination of fuzzy AND (or MIN) learning (*snap*), and Learning Vector Quantisation *(drift)* [10]. Abstractly speaking, the Snap-Drift algorithm can be expressed as:

$$Snap\text{-}Drift = \alpha(Snap) + (1\text{-}\alpha)(drift) \tag{1}$$

The weights are updated using the following:

$$w_{ji}^{(new)} = \alpha(I \cap w_{ji}^{(old)}) + (1\text{-}\alpha)(w_{ji}^{(old)} + \beta(I - w_{ji}^{(old)})) \tag{2}$$

where w_{ji} = weights vectors; I = binary input vectors, and β = the drift speed constant. When $\alpha = 1$, fast, minimalist (snap) learning is invoked:

$$w_{ji}^{(new)} = I \cap w_{ji}^{(old)} \tag{3}$$

This works for binary data, otherwise equation (3) becomes the fuzzy AND of weight with data, Min(I ,$w_{ji}^{(old)}$). Consequently, Snap encodes, within the weights, the common elements of all patterns that activate the node (neuron) for learning.

In contrast, when $\alpha = 0$, (2) simplifies to:

$$w_{ji}^{(new)} = w_{ji}^{(old)} + \beta(I - w_{ji}^{(old)}) \tag{4}$$

which implements a simple form of clustering (drift) or LVQ, at a speed determined by β. Finally, in the case of either snap or drift, weights are normalized:

$$w_{ji}^{(new)} = w_{ji}^{(new)} / | w_{ji}^{(new)}| \tag{5}$$

The snap and drift modes provide complementary features. Snap capturing the common elements of the group of patterns as represented by the minimum values on each input dimension, whereas drift captures the average values of the group of patterns. Snap also has the effect of contribution to rapid convergence.

2.2 SDSOM

We present a new version of Self-organising Map, which combines SOM with the snap-drift algorithm. The SDSOM has the same architecture (Fig.1) as a standard SOM, with a layer of input nodes connecting to the self organising map layer. A shrinking neighbourhood is used during training, as in SOM, with the weight vector of each neighbour of the winning node being adapted according to the input pattern. The key difference in SDSOM is the weight update, which consists of either snap (min of input and weight) or drift (LVQ, as in SOM). The following steps illustrate the SDSOM algorithm:

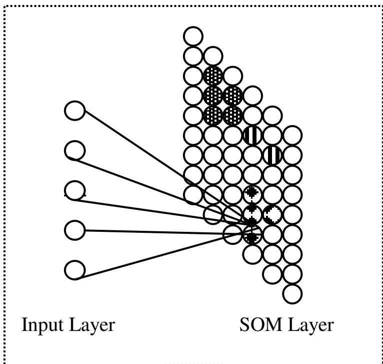

Fig. 1. SOM architecture

Step 1: Initialize parameters: $\alpha = 1$ (equation (1))
Set size of the SOM layer map.
Initialize neighborhood size.
Initialize weights between input and SOM layer with the values of randomly selected input patterns.
Normalize weights.
Initialize learning rate β for drift mode
Initialize maximum number of epochs
Step 2: For each epoch (t)
Swap the value of α to 1 or 0
Step 2.1: For each input pattern
Step 2.1.1: Find the winning node in SOM with the largest net input
Step 2.1.2: Update weights of the wining node and its neighbour nodes according to the current learning mode (equation (2))
Step 2.1.3 Normalize weights (equation (5))
Step 2.2: Decrease the neighborhood size with 1
Step 3: Terminate when maximum number of epochs is reached
Step 4: Evaluate the results by labelling SOM layer nodes

The shaded nodes in Fig. 1 represent different classes or labels. Nodes receive the class label of the majority of the patterns for which they win. There is generally a tendency for neighbouring nodes to have the same class, given the nature of SOMs, but this is not forced by the labelling algorithm.

3 Experiments and Results

3.1 Data Sets

A range of data sets are chosen, presenting a variety of learning challenges. They vary in terms of the number of input variables, the number of classes, and the level of separability of the classes. Since they are all known and freely available they provide useful benchmark comparators, not only with SOM, but with a number of other neural computing and other machine learning techniques.

The **Animal** data set is artificial and consists of 16 animals described by 13 attributes such as size, number of legs etc. [7]. The 16 animals are grouped into three classes (the first one represents bird, the second represents carnivore and the third represents herbivore).

The **Iris** data set has three classes setosa, virsicolor and virginica [12, 13]. The iris data has 150 patterns, each with 4 attributes. The class distribution is 33.3% for each of 3 classes. One of the classes is linearly separable from the other two, and the two are linearly inseparable from each other.

The **Wine** data set is the result of a chemical analysis of wines grown in the same region in Italy but derived from three different cultivars [14]. The analysis determines the quantities of 13 constituents (input variables) found in each of the three types of wines. There are 178 patterns with the following distribution: class 1 - 59, class 2 – 71, class 3 – 48.

The **Ecoli** data set contains 336 patterns with 7 attributes and 8 classes, which are the 'localization sites', distributed as follows [15]:

cp (cytoplasm)	143
im (inner membrane without signal sequence)	77
pp (perisplasm)	52
imU (inner membrane, uncleavable signal sequence)	35
om (outer membrane)	20
omL (outer membrane lipoprotein)	5
imL (inner membrane lipoprotein)	2
imS (inner membrane, cleavable signal sequence)	2

3.2 Results

In the experiments, 20% selections of the patterns of each data set are allocated for testing and the remaining 80% form the training set. For each run the training and testing patterns are selected at random from the entire data set . SOM is trained for 500 epochs and SDSOM for 200 epochs. This is long enough for the maps to be stable in all cases.

In order to perform a labelling of nodes for the purposes of classification the number of patterns for which the node wins is accumulated for each class and for each node. The majority class, with the highest number of patterns, becomes the class label of that node. The training classification score is the percentage of patterns categorised by nodes of the correct class. The training class labels are retained for use in testing.

Table 1. Mean % correct classification for train and test sets based on 5 runs. Standard deviation given in the brackets

Method/Data set		Animals	Iris	Wine	Ecoli
SDSOM	train	100 (0)	100 (0)	100 (0)	96.2 (0.8)
	test	**100 (0)**	**99.4 (1.3)**	**92.6 (1.3)**	**84.6 (2.5)**
Kohonen SOM	train	100 (0)	100 (0)	100 (0)	100 (0)
	test	100 (0)	95 (3.1)	86 (6.3)	81 (2.1)

The percentage of correct classifications is the percentage of patterns for which the winning node has the same class label as the class of the pattern. Nodes in the map that by the end of training have not associated patterns for which they win are not labelled. During testing, if a winning node is unlabelled (which is rare) then the most active labelled node provides the class (correct or incorrect). Each test consists of 5 repeat trainings of the data, with the training set being randomly selected from the whole data set in each case. The maps formed are very similar for each of the repeat trainings, and the classification results presented are the averages for the 5 results. The standard deviations across the 5 tests are also presented in table 1 and for SDSOM it is only about 1%.

The **Animal** data (the maps in Fig. 2) presents a relatively easy classification task because each pattern differs quite significantly, therefore it is a simple challenge for any method to separate or classify them individually without the need for generalised rules. Both SOM and SDSOM perform well. There is however an important qualitative difference between the two results, as Fig. 2a and Fig. 2b clearly shows. SDSOM has projected the classes onto the map in a linearly separable fashion; two straight lines can separate the three animal classes on the map. This is not possible in the SOM, which mixes the herbivores and carnivores to a greater extent. The snap mode finds some common elements that are specific to herbivores that are not based on the overall similarity of herbivores across all dimensions, which is the limitation of LVQ, or any method that calculates overall similarity.

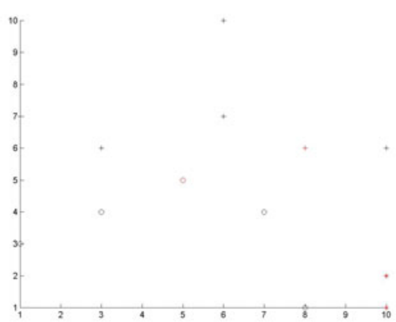

Fig. 2a. SDSOM 10x10 applied to Animal Data set + (bird) o (carnivore) * (herbivore)

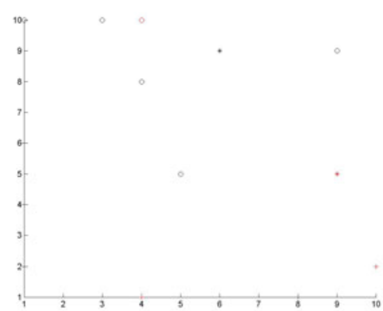

Fig. 2b. SOM 10x10 applied to Animal Data set + (bird) o (carnivore) * (herbivore)

The **Iris** maps differ substantially between SOM and SDSOM. The SOM map presents a widely dispersed set of points. They are nonetheless in clear regions associated with the three classes. However, the lines between classes in the map are curved with several changes of direction and there is no margin between the classes, even in the case of the linearly separable classes. In the SDSOM map, the margin between setosa and the other two classes is significant, and the linearly inseparable verginica is more tightly grouped than in SOM. These factors give a classification advantage to SDSOM of 99% as opposed to 95%, and the t test (t=2.92, p=0.05) indicates a 95% probability of this 4% difference being statistically significant.

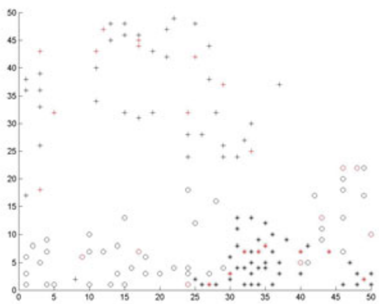

Fig. 3a. SDSOM 50x50 applied to Iris Data set o (verisicolor) + (setosa) * (virginica)

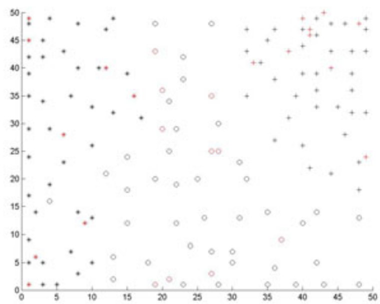

Fig. 3b. SOM 50x50 applied to Iris Data set + (setosa) o (verisicolor) * (virginica)

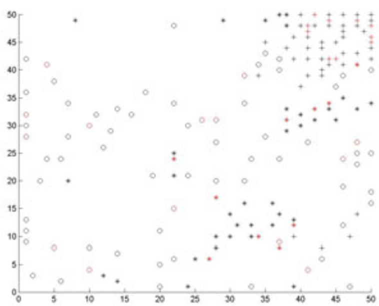

Fig. 4a. SDSOM 50x50 applied to Wine Data set + (class 1) o (class 2) * (class 3).

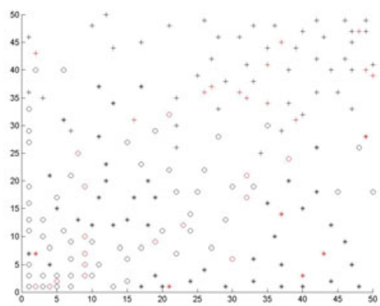

Fig. 4b. SOM 50x50 applied to Wine Data set + (class 1) o (class 2) * (class 3).

The average separation on the **Wine** data map of the classes is larger in SDSOM, and the classification is 92.6% as opposed to 86% with SOM, a result that is 90% likely to be statistically significant (t=2.27, p=0.1).

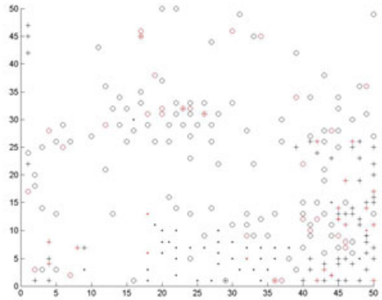

Fig. 5a. SDSOM 50x50 applied to Ecoli Data set + (class 1) o (class 2) * (class 3).

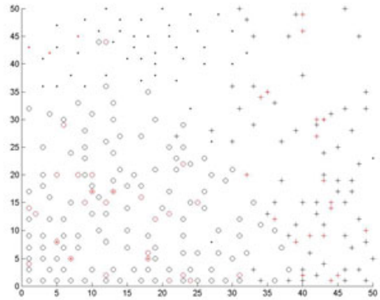

Fig. 5b. SOM 50x50 applied to Ecoli Data set + (class 1) o (class 2) * (class 3).

The **Ecoli** data set has 8 classes but only 3 are given on the pictures above. The E.coli data is clearly a challenging problem, with a range of methods achieving 81% correct classification, for example using an ad hoc structured probability model, a binary decision tree, or a Bayesian classifier [15]. Similarly, SOM based classification yields 81%. SDSOM achieves 84.6%, with a t test indicating 95% confidence (t=2.45, p=0.05) of the improvement over SOM being statistically significant.

4 Conclusion

In this work a method for using the snap-drift principle in the training of a Self Organinzing Map is considered. The resulting algorithm is called snap-drift SOM (SDSOM). SOM is useful for data visualisation and to some extent for classification. SDSOM explores whether some advantages of the modal combination of LVQ and min learning that has been effective elsewhere with snap-drift can be successfully transferred into SOMs. SDSOM requires fewer epochs than the original SOM because snap is a rapidly convergent form of adaptation, but we also see that there are differences both in terms of the maps that are formed and the classification results obtained from the maps. Snap-drift creates tighter groupings within the map and typically wider margins between groupings, and where those groupings correspond to classes this supports more effective classification. Because snap modifies drift by minimising or removing dimensions in weight space, fewer input variables are represented in the weight vectors than in LVQ and the feature gradient across the SD Self-organising Map is steeper than in SOM. Consequently, the map class-regions are more compact in SDSOM. When compactness is measured as the average proportion of the map space covered by the outline of the class groupings, the SDSOMs are about 30% more compact than the SOMs. The groupings are also more widely separated. The average distance between the centroids of groupings in the SDSOMs is 35% of the maximum possible separation, compared to 20% in SOMs.

References

1. Lee, S.W., Palmer-Brown, D., Tepper, J., Roadknight, C.M.: Snap Drift: Realtime Performance guided Learning. In: International Joint Conference on Neural Networks IJCNN, Portland, Oregon, vol. 2, pp. 1412–1416 (2003)
2. Lee, S.W., Palmer-Brown, D., Roadknight, C.M.: Performance guided Neural Network for Rapidly Self Organising Active Network Management. Neurocomputing 61C, 5–20 (2004)
3. Lee, S.W., Palmer-Brown, D.: Phonetic Feature Discovery in Speech using Snap-Drift. In: Kollias, S.D., Stafylopatis, A., Duch, W., Oja, E. (eds.) ICANN 2006, Part II. LNCS, vol. 4132, pp. 952–962. Springer, Heidelberg (2006)
4. Ekpenyong, F., Palmer-Brown, D., Brimicombe, A.: Extracting road information from recorded GPS data using snap-drift neural network. Neurocomputing 73, 24–36 (2009)
5. Palmer-Brown, D., Draganova, C., Lee, S.W.: Snap-Drift Neural Network for Selecting Student Feedback. In: International Joint Conference on Neural Networks IJCNN, Atlanta, pp. 391–398 (2009)
6. Kohonen, T.: Self-Organisation and Asssociative Memory, 3rd edn. Springer, Heilderberg (1989)
7. Ritter, H., Kohonen, T.: Self-Organizing Semantic Maps. Biological Cybernetics 61, 241–254 (1989)
8. Kohonen, T.: Learning Vector Quantisation. Helsinki University of Technology, Laboratory of Computer and Information Science, Report TKK-F-A-601 (1986)
9. Kohonen, T.: Learning Vector Quantisation. Neural Networks 1, 303 (1988)
10. Kohonen, T.: Improved Versions of Learning Vector Quantization. In: IJCNN, vol. 1, pp. 545–550 (1990)
11. Kohonen, T.: Self-organised formation of topologically correct feature maps. In: Biological Cybernetics, vol. 43. Springer, Heilderberg (1982)
12. Fisher, R.A.: The use of multiple measurements in taxonomic problems. Annual Eugenics 7(Part II), 179–188 (1936); also in Contributions to Mathematical Statistics. John Wiley, NY (1950)
13. Duda, R.O., Hart, P.E.: Pattern Classification and Scene Analysis, p. 218. John Wiley & Sons, Chichester (1973)
14. Forina, M.: An Extendible Package for Data Exploration. In: Classification and Correlation. Institute of Pharmaceutical and Food Analysis and Technologies, Genoa, Italy
15. Horton, P., Nakai, K.: A Probablistic Classification System for Predicting the Cellular Localization Sites of Proteins. In: Intelligent Systems in Molecular Biology, pp. 109–115 (1996)