A Dynamic Adaptive Self-Organising Hybrid Model for Text Clustering

Chihli Hung and Stefan Wermter
Centre for Hybrid Intelligent Systems
School of Computing and Technology, University of Sunderland, UK
[chihli.hung;stefan.wermter]@sunderland.ac.uk

Abstract

Clustering by document concepts is a powerful way of retrieving information from a large number of documents. This task in general does not make any assumption on the data distribution. In this paper, for this task we propose a new competitive Self-Organising (SOM) model, namely the Dynamic Adaptive Self-Organising Hybrid model (DASH). The features of DASH are a dynamic structure, hierarchical clustering, non-stationary data learning and parameter self-adjustment. All features are data-oriented: DASH adjusts its behaviour not only by modifying its parameters but also by an adaptive structure. The hierarchical growing architecture is a useful facility for such a competitive neural model which is designed for text clustering. In this paper, we have presented a new type of self-organising dynamic growing neural network which can deal with the non-uniform data distribution and the non-stationary data sets and represent the inner data structure by a hierarchical view.

1. Introduction

Clustering by document concepts is useful to reduce the search space for linking a query to relevant information. One well-known project is WebSOM [1], which employs a Self-Organising Map (SOM) for clustering documents and presents them on a 2-dimensional map. Documents with a similar concept are grouped into the same cluster and clusters with similar concepts are located nearby on a map. This is the main difference between neural clustering and traditional statistical cluster analysis, which only assigns objects to clusters but ignores the relationship between clusters. The Self-Organising Map (SOM) combines non-linear projection, vector quantization (VQ), and data clustering functions [2]. However, in terms of the clustering algorithm, a SOM suffers from the following problems:

- The network structure including the topology and the number of units has to be set before training. Different architectures lead to different results. The fixed architecture is not ideal for non-uniform distributions. This constraint causes an unnecessarily large vector quantization error.
- Using a large single map with a huge data set is not ideal and so a hierarchical approach may be needed.
- A SOM is not ideal for non-stationary information environments. Real world knowledge, for example, news stories, is changing over time. An algorithm which handles non-stationary data sets will offer more flexibility for a web-based project, such as WebSOM.

In this paper, we focus on a text clustering task and propose an alternative model, the Dynamic Adaptive Self-organising Hybrid model (DASH), to address the above deficiencies. We use the new Reuters news Corpus, RCV1[1], as our main test-bed and evaluate DASH based on classification accuracy and average quantization error.

The remainder of this paper is organised as follows. In Section 2, we give a general review of current related competitive models. In section 3, we introduce the DASH approach. In section 4, we test the features of the DASH using two small data sets. Section 5 contains three experiments using the new Reuters Corpus given under different scenarios. Then a conclusion is presented in section 6.

2. Related Unsupervised Competitive Models

Several related unsupervised neural learning models have been proposed to enhance the practicability of the SOM. Different modifications of the SOM suggest different enhancements from different viewpoints. These models can be divided into four groups, which are static models, dynamic models, hierarchical models and non-stationary learning models.

Static models, such as the pure competitive learning (CL) [2,3] and Neural Gas (NG) [4], relax the constraint of a fixed topological structure, i.e. a grid, of SOM. Dynamic models, such as the Growing Grid (GG) [5] and Growing SOM (GSOM) [6], try to define a model with no need of prior knowledge for the number of output units by an incremental growing architecture. Hierarchical models,

[1]The new version of the Reuters news corpus can be found at http://about.reuters.com/researchandstandards/corpus/

such as the TreeGCS [7], Multilayered Self-Organising Feature Maps (M-SOM) [8] and Growing Hierarchical Self-Organizing Map (GHSOM) [9], offer a detailed view for a complicated clustering task. Non-stationary learning models, such as the Growing Cell Structure (GCS) [10], Growing Neural Gas (GNG) [11], Incremental Grid Growing (IGG) [12], Growing Neural Gas with Utility criterion (GNG-U) [13], Plastic Self Organising Map (PSOM) [14] and Grow When Required (GWR) [15], contain unit-growing and unit-pruning functions which are analogous to biological functions of remembering and forgetting under a dynamic environment.

We focus on models which are able to handle a data set with the nature of hierarchical relationships, non-uniform distributions, or non-stationary varieties. For a neural model to offer automatic hierarchical clustering, it may need a function to further prune the map by removing unsuitable units to form several partitions on a map. Hodge and Austin [7] use this technique to produce synonym clusters as an automatic thesaurus. However, this unit-pruning function seriously depends on a pre-defined constant threshold. Based on the unknown data distribution, this threshold is very difficult to determine. Second, the partition is formed because of the nature of the input data. One should not foresee that a hierarchy must be built by a competitive model with the unit-pruning function. A proper policy may build such a hierarchy by further developing a whole map from a unit with many input data mapped to this unit or with higher error information, e.g. [8] and [9]. We take advantage of both concepts to build DASH as an automatic hierarchical clustering model.

For the non-stationary data set, a trained unit or training unit should be updated by a unit which is trained with new input samples. This is performed by the unit-pruning or connection-trimming function. A model with the connection-trimming function should be based on a global aged consideration. The reason is that a local age variable of a connection does not grow when units of this connection is not activated. That is, the aged connection may be kept forever so that the capability of self-adjustment for a model to new stimuli is diminished. Thus, a model, such as the GNG and GWR, using the connection-trimming function based on a local aged consideration can be treated as an incomplete non-stationary model only.

On the other hand, the stop criterion of models should not be a time-dependent threshold, such as iteration or epoch. However, this stop criterion is used for the models in our survey. Moreover, an unsuitable constant unit-pruning or a connection-trimming threshold may make the model train forever but learn nothing. This constant value can be very small or very large, which is totally dependent on trial-and-error. Therefore, it is not a good idea to use such a constant threshold for a big data set. Unfortunately, the GCS, GNG, IGG, GNG-U, PSOM and GWR apply a constant threshold for detection of unsuitable units. We argue that a unit-pruning or a connection-trimming threshold should be automatically adjusted to suit different data sets during training.

3. Dynamic Adaptive Self-organising Hybrid (DASH) Model

3.1. The Need for DASH

We start by inspecting the features of text clustering for a real-world task. First, the quantity of text information is continuously growing so the information is not static. Therefore, a text clustering model should allow the learning of growing knowledge. It implies that a clustering model which contains a time-based decaying learning function is not suitable for such a task. Second, text information usually has some relationship with time, for instance, news. Some specific events often occur during a specific period. Thus, in this period, several news articles with similar topics are presented repeatedly and the recent information is more important. Therefore, a clustering model should be able to handle dynamic knowledge acquisition during this period. Third, clustering should use a hierarchical concept to complement searching. A hierarchical structure for a large set of data is not only necessary to keep the elasticity of the query response but also to analyse the contents easily. Due to this reason, a text clustering model should explicitly offer hierarchical learning for a large and complicated text set. However, none of the existing models meet all the needs of the features required for a text clustering task. This leads to the development of the DASH.

3.2. Features of DASH

From the viewpoint of concept, the DASH is an integrated model of the GNG and GHSOM, and contains several unique features. The DASH is a growing self-organising model which has characteristics of a dynamic structure, hierarchical training, non-stationary data learning and parameter self-adaptation. Three main percentage-like parameters, which influence the style of the DASH architecture, have to be defined. The first one is τ, which has an impact on how well DASH represents the current data set and a link to the size of a map. The second one is S_{min}, a minimum number of input samples which a map represents. This parameter affects the depth of a DASH hierarchy. The third one is a connection-trimming variable, β, which functions as a unit-pruning threshold. However, the β variable is self-adjusted when the current model does not grow continuously to meet the requirement of a map quality, i.e. the AQE, which is defined as the average distance between every input vector and its Best Matching Unit (BMU) [16]. An example of the DASH structure is given in (Figure 1).

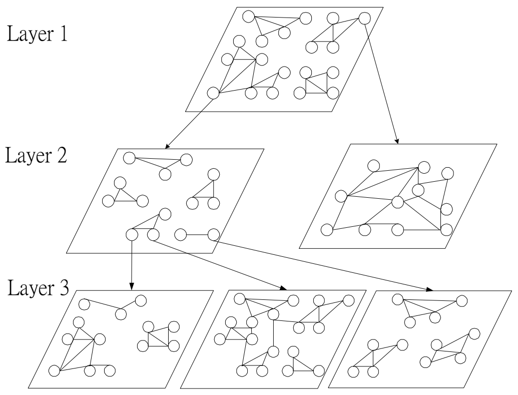

Figure 1. The hierarchical structure of DASH

The flowchart of DASH is shown in (Figure 2), which involves two main iterations and seven processes. The inner iteration is a GNG-like learning procedure for each map in a hierarchy. However, unlike the GNG, the DASH applies a global connection-trimming function instead of a local one to remove aged relationships between units and therefore, the isolated units are pruned globally. The GNG grows in every pre-defined cycle, which is determined by trial-and-error. In contrast, this cycle is a part of DASH, which is mutually decided by τ , S_{min} and the number of input samples in the current map (see Eq. 5 in Appendix). Furthermore, the connection-trimming and growing period are constants for GNG but they are self-adjusted variables for DASH.

The outer iteration is a GHSOM-like recursive training cycle. The GHSOM applies two constants, i.e. τ_1 and τ_2 , to define the size and the depth of GHSOM architecture, respectively. The DASH uses τ , which is percentage-like parameter, to decide the size of map but use S_{min} to decide how detailed the data samples are represented by the DASH. For training kernel, the GHSOM directly employs the SOM training algorithm for each unit-growing procedure. For example, a GHSOM with 3x3 units in a map needs to employ 3 traditional SOM training cycles, i.e. one for a 2x2 structure, one for 2x3 (or 3x2) structure and one for 3x3 structure, using the same number of input data. However, only the training for the final map, i.e. the 3x3 map, is necessary. This behaviour may not be suitable for a real-world clustering task. The DASH applies a modification of GNG training procedure. For training a child map, the GHSOM applies a threshold based on the AQE in top level. However this threshold is dependent on the AQE in its direct parent level for DASH. This feature makes the DASH enforce a distributed learning which trains a whole input set by training several smaller input sub-sets separately. Moreover, the GHSOM does not contain a unit-pruning function. Once units grow, they have no chance to be removed. Conversely, the DASH contains unit-pruning and unit-growing functions, which can deal with the non-stationary and non-uniform data set. The detailed DASH algorithm is shown in the Appendix.

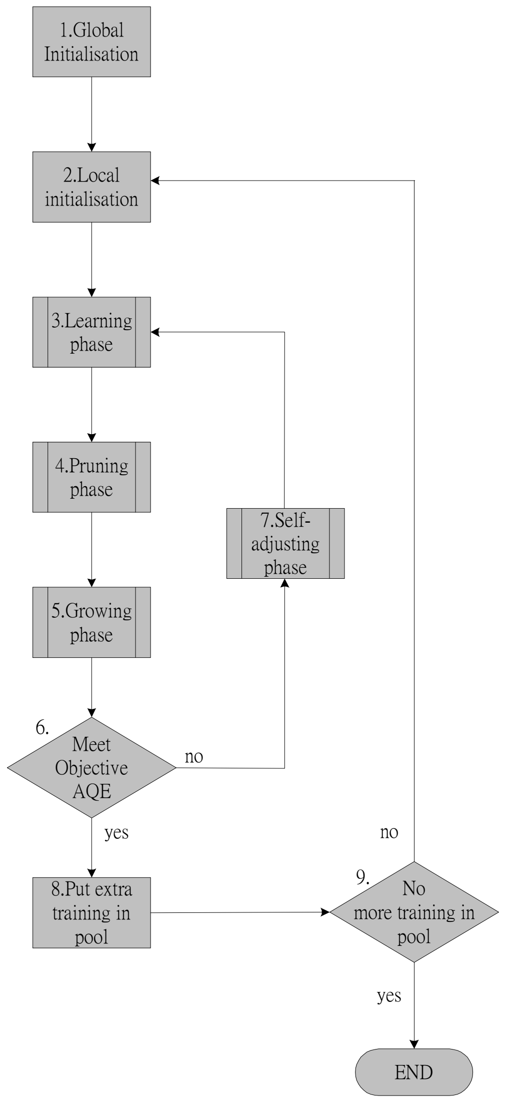

Figure 2. The flow chart of the DASH algorithm

4. Initial Experiments

We design two small experiments to present how DASH works. In the first experiment, we use a pre-arranged four-corner data set to test how the model deals with a non-uniform distribution. In the second experiment, we use a jumping-corner data set to show the capability of handling a non-stationary data set for the model.

4.1. Four Corners

A 4-corner data set is applied to test the ability of competitive models for learning a non-uniform data distribution. Each corner contains 30x30 2-D input vectors. The results of the SOM and DASH are presented and both models can faithfully represent the associated input vectors on 4 corners. However while the SOM contains many "dead units" and cannot represent data well at the borders of corners (Figure 3), DASH successfully removes the unsuitable connections to form 4 clusters during learning (Figure 4).

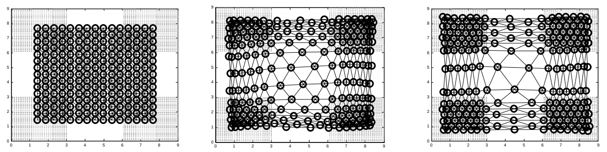
Figure 3. The convergence of SOM training at initial, 1,000 and 10,000 iterations

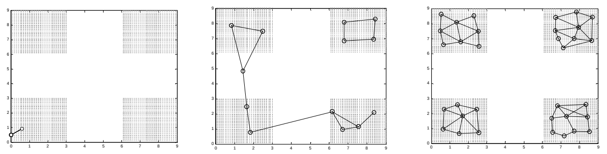
Figure 4. The convergence of DASH training at initial, 1,000 and 10,000 iterations

4.2. A Jumping Corner

A jumping-corner data set is used to mimic the non-stationary input data. A data set distributed in the bottom left corner at the beginning moves to the top right corner at iteration 5001. It can be seen that a new data set in the top right corner substitutes for the existing data set in the bottom left corner. The SOM learns well in the beginning because the data set is a uniform distribution (Figure 5). However, when the existing data set is replaced by the new data set, some units of the SOM cannot be re-trained since the learning rate is decayed. Thus, "dead units" are inevitable for a SOM to train a non-stationary data set. Conversely, DASH removes unsuitable existing units when new input stimuli happen (Figure 6).

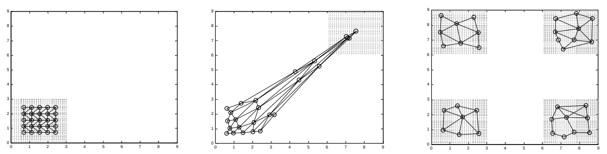
Figure 5. The convergence of SOM training at initial, 6,000 and 10,000 iterations

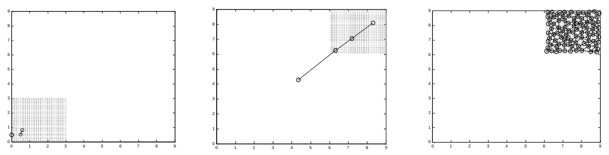
Figure 6. The convergence of DASH training at initial, 6,000 and 10,000 iterations

5. Experiments

5.1. Description of New Reuters Corpus and Data Set

We evaluate our model by the classification accuracy and AQE, which have also been used in the work of Kohonen et al. [17]. Based on the classification accuracy criterion, a pre-labelled corpus is necessary. We work with the new version of the Reuters corpus, RCV1, since this news corpus is a representative test for text classification, a common benchmark and a fairly recent comprehensive data source [18, 19]. This corpus is made up of 984 Mbytes of news. The number of total news articles is 806,791 which contain about two hundred million word occurrences. One hundred and twenty six topics are defined in this new corpus. Each news article is pre-classified to 3.17 topics on average but 10,186 of them are assigned to no topic.

In this paper, the 8 most dominant topics in the corpus are considered initially. Because a news article can be pre-classified as more than one topic, the multi-topic combination is treated as a new topic in this research. Thus, the 8 dominant topics are expanded into 40 combined topics for the first 10,000 news articles. Each full-text document is represented using vector space model (VSM) [20]. Due to the massive dimensionality of vectors, we remove the stop words, confine to words shown in WordNet [21] and lemmatise each word to its base form. WordNet is a net of words which only contains open-classed words, i.e. nouns, verbs, adjectives and adverbs. After this pre-processing, there are 16,122 different words in the master word list. The 1,000 most frequent words are picked from the master word list since this method is as good as most dimensionality reduction techniques [22].

5.2. Static Data Set

The first 10,000 full-text documents without title information is used as our test-bed. The pre-processing procedure mentioned in the previous section is used. We apply a normalised TFxIDF as the vector representation approach [23]. In this experiment, three different τ, i.e. 95%, 90% and 85% of DASH are applied. For convenience, they are termed DASH95, DASH90 and

DASH85 in this paper. The S_{min} is 1% and the initial β is 95%. Please note β is an adjustable parameter during training. These three parameters are the same for following experiments. We compare the results with six other models, i.e. the CL, SOM, NG, GG, GCS and GNG. We use 15x15 = 225 units for each model, but this number is only an estimate for dynamic models, i.e. the GG, GCS and GNG. All learning rates of models are initialised to 0.1. Such a learning rate is certainly decayed in some models, such as the CL, SOM and NG. SOM fine-tuning training starts with a 0.001 learning rate. Other models, such as the GG, GCS, GNG and DASH, also use an extra learning rate which is 0.001 for training runner-up units of BMU. Except the DASH, we train all models using 10,000 iterations. The DASH stop criterion is defined by objective AQE. According to the results in (Table 1), we notice that models with a sticker structures have higher AQEs. For example, the SOM and GG contain a grid-style topographic structure which may be difference from the relationships between data. Thus, their AQEs are higher than other models in this paper. The performance of DASH85 is worse than that of DASH90, which contains a smaller AQE and a higher accuracy. The reason is that a lower τ , i.e. 85%, introduces a bigger size of map which consists of 262 units in the first layer. That is, the average number of units mapped to a unit is only 38.17 ($10{,}000 \div 262 \cong 38.17$) which is much smaller than the stop criterion of S_{min} ($10{,}000 \times 1\% = 100$). Thus, only 3 maps are in the hierarchy of DASH85 but there are 21 maps in the DASH90. However, the total training time of the DASH85 is longer than that of the DASH90 training.

Table 1. A comparison of neural models for a static data set

	CL	SOM	NG	GG	GCS
AQE	0.836	0.930	0.837	0.881	0.820
accuracy	65.49%	69.16%	69.54%	68.82%	68.18%

	GNG	DASH95	DASH90	DASH85
AQE	0.823	0.818	0.790	0.802
Accuracy	68.60%	69.60%	70.42%	68.37%

5.3. Knowledge Acquisition

This section is to test the ability of models to handle the non-stationary data set. The pre-processing and vector representation approaches are the same as those in the previous section. We treat the new data set as new knowledge that complements the existing knowledge. The first 5,000 full-text documents are applied as an existing data set and the second 5,000 full-text documents as a new data set. There are three scenarios, which evaluates the relationships of models and non-stationary data sets. The existing data set is used for all experiments in the beginning. The new data set is introduced in scenario 1 at iteration 10,000, scenario 2 at iteration 30,000 and scenario 3 at iteration 50,000. SOM rough-training is stopped at iteration 30,000 and its fine-tuning training is stopped at 50,000 iterations. The stop criterion of DASH is finding the objective AQE. According to our experiments, the SOM does not suffer from the new data set seriously, if the distribution of the new data set is similar to that of the existing data set (Table 2). In this case, the performance of the SOM is comparable to the non-stationary model, i.e., the DASH.

Table 2. A comparison of SOM and DASH for knowledge acquisition scenario

		10000iter	30000iter	50000iter
SOM	AQE	0.937	0.938	0.940
	accuracy	69.06%	68.44%	69.08%
DASH	AQE	0.771	0.780	0.802
	accuracy	72.23%	71.42%	70.39%

5.4. Knowledge Update

This section is to compare the performance of models under a non-stationary environment where the existing data set is treated as out-of-date knowledge and should be updated by the new knowledge, i.e. the new data set. The same pre-processing procedure mentioned above is used. To mimic the non-stationary data set, the first 10,000 full-text documents are transformed by using the normalized TFxIDF vector representation as our new data set but by using the non-normalized TFxIDF as the old data set. The averaged weights of the existing set are much higher than those of the new data set. Thus, the AQEs are also much higher when models deal with the existing data set. We also introduce the new data set at iteration 10,000 for the scenario 1, iteration 30,000 for the scenario 2 and iteration 50,000 for the scenario 3.

Some non-stationary competitive models such as GNG-U and GWR have been tried in these experiments. However, it is not possible to use their unit-pruning and connection-trimming constant thresholds for both data sets. When a proper threshold is set for the existing data set, this threshold is always too large for the new data set. Thus, models do not grow. Conversely, if a threshold is suitable for the new data set, this threshold is always too small for the existing data set. However, we should not set such a threshold by presuming the distribution of the new data set. Therefore, we only test our model and SOM in these experiments.

According to our experiments, the SOM clearly suffers from the decayed learning rate (Table 3). The new data samples are not learnt completely, so the accuracy drops while the AQE increases. On the other hand, DASH removes all unsuitable trained units very fast and adjusts its new objective AQE automatically. Thus, there is no large difference between the performance at each point when a new data set is introduced during training for DASH.

Table 3. A comparison of SOM and DASH for knowledge update scenario

		10000iter	30000iter	50000iter
SOM	AQE	0.948	1.352	2.513
	accuracy	64.37%	21.52%	25.75%
DASH	AQE	0.784	0.776	0.793
	accuracy	71.40%	72.25%	69.30%

6. Conclusion

Due to the features of a real-world text clustering task, a neural model which can handle both static and non-stationary datasets and represent the inner data structure by a hierarchical view is necessary. This paper has presented such a new type of self-organising dynamic growing neural network, i.e. DASH. In terms of concept, the DASH is a hybrid model of GHSOM and GNG but the DASH contains several unique features, such as the parameter self-adjustment, hierarchical training and continuous learning. Based on these features, a real-world document clustering task has been demonstrated in this paper. Those existing models which are designed for the non-stationary data sets may not be suitable for a real-world clustering task. The main reason is the difficulty of determining unit-pruning and connection-trimming parameters. Furthermore, those non-stationary models should not use a time-dependent stop criterion. For more complex data sets, such as a document collection, a hierarchical structure is preferable. The DASH is a hierarchical neural approach which functions as a non-stationary distributing learning facility.

7. Reference

[1] T. Honkela, S. Kaski, K. Lagus, and T. Kohonen, "Newsgroup exploration with WEBSOM method and browsing interface", *Report A32*, Helsinki University of Technology, 1996.

[2] T. Kohonen, *Self-organization and associative memory*, Springer-Verlag, Berlin, 1984.

[3] S. Grossberg, "Adaptive pattern classification and universal recoding: I. Parallel development and coding of neural feature detectors", *Biological Cybernetics*, vol. 23, 1976, pp. 121-131.

[4] T. Martinetz and K. Schulten, "A 'Neural-Gas' network learns topologies", *Artificial Neural Network*, vol. I, 1991, pp. 397-402.

[5] B. Fritzke, "Growing grid-a self-organizing network with constant neighborhood range and adaptation strength", *Neural Processing Letters*, vol. 2 no. 5, 1995, pp. 9-13.

[6] D. Alahakoon, S.K. Halgamuge and B. Srinivasan, "Dynamic self-organizing maps with controlled growth for knowledge discovery", *IEEE Tractions on Neural Networks*, vol. 11, no. 3, 2000, pp. 601-614.

[7] V. Hodge and J. Austin, "Hierarchical growing cell structures: TreeGCS", *Proceedings of the Fourth International Conference on Knowledge-Based Intelligent Engineering Systems*, 2000.

[8] H. Chen, C. Schuffels and R. Orwig, "Internet categorization and search: a self-organizing approach", *Journal of Visual Communication and Image Representation*, vol. 7, no. 1, March, 1996, pp. 88-102.

[9] A. Rauber, D. Merkl and M. Dittenbach, "The growing hierarchical self-organizing maps: exploratory analysis of high-dimensional data", *IEEE Transactions on Neural Networks*, vol. 13, no. 6, 2002, pp.1331-1341.

[10] B. Fritzke, "Growing cell structures – a self-organizing network for unsupervised and supervised learning", *Neural Networks*, vol. 7, no. 9, 1994, pp.1441-1460.

[11] B. Fritzke, "A growing neural gas network learns topologies", *Advances in Neural Information Processing Systems 7*, G. Tesauro, D.S. Touretzky, and T.K. Leen, eds., MIT Press, Cambridge MA, 1995, pp. 625-632.

[12] J. Blackmore and R. Miikkulainen, "Incremental grid growing: encoding high-dimensional structure into a two-dimensional feature map", *Proceedings of the IEEE International Conference on Neural Networks (ICNN'93)*, San Francisco, CA, USA, 1993.

[13] B. Fritzke, "A self-organizing network that can follow non-stationary distributions", *Proceedings of ICANN'97, International Conference on Artificial Neural Networks*, Springer, 1997, pp. 613-618.

[14] R. Lang and K. Warwick, "The plastic self organising map", *IEEE World Congress on Computational Intelligence*, 2002.

[15] S. Marsland, J. Shapiro and U. Nehmzow, "A self-organising network that grows when required", *Neural Networks*, vol. 15, 2002, pp. 1041-1058.

[16] T. Kohonen, *Self-organizing maps*, Springer-Verlag, 2001.

[17] T. Kohonen, S. Kaski, K. Lagus, J. Salojärvi, J. Honkela, V. Paatero and A. Saarela, "Self organization of a massive document collection", *IEEE Transactions on Neural Networks*, vol. 11, no. 3, 2000, pp. 574-585.

[18] S. Wermter and C. Hung, "Selforganising Classification on the Reuters News Corpus", *The 19th International Conference on Computational Linguistics (COLING2002),* Taipei, Taiwan, 2002, pp.1086-1092.

[19] T.G. Rose, M. Stevenson and M. Whitehead, "The Reuters Corpus Volume 1 - from Yesterday's News to Tomorrow's Language Resources", *Proceedings of the Third International Conference on Language Resources and Evaluation*, Las Palmas de Gran Canaria, 2002, pp. 29-31.

[20] G. Salton, *Automatic Text Processing: the Transformation, Analysis, and Retrieval of Information by Computer*, Addison-Wesley, USA, 1989.

[21] G.A. Miller, "WordNet: a dictionary browser", *Proceedings of the First International Conference on Information in Data*, University of Waterloo, Waterloo, 1985.

[22] S. Chakrabarti, "Data mining for hypertext: a tutorial survey", *ACM SIGKDD Explorations*, vol. 1, no. 2, 2000, pp. 1-11.

[23] G. Salton, and C. Buckley, "Term-weighting approaches in automatic text retrieval", *Information Processing & Management*, vol. 24, no. 5, 1988, pp. 513-523.

8. Appendix – DASH Algorithm

For convenience, we describe the main structure of the model as follows. Let $A=\{L_1,L_2,\ldots L_l\}$, where A is the set of sub-maps. Let $L=\{U_1,U_2,\ldots U_u\}$, where U_i is the unit i in the map L. Each U_i has an error variable, err_i. Let C_{ij} be the binary connection between U_i and U_j. Each C_{ij} has a variable, age_{ij}, to store the connection age. Let the input distribution be $p(X)$ for the input set X. Let $X=\{x_1,x_2,\ldots x_n\}$, where x_i is the input sample i in the input set X. We define the weight vectors for an input sample and for a unit as x_i and w_i respectively. Then the precise processes of the DASH algorithm are as below.

1) Global network initialisation
   1.1) Define a map quality index, τ, where $0<\tau\le 1$. τ decides the objective AQE for a child map. It controls the extent of the size for a single map and is also the stop criterion for the child map training. A smaller τ builds a bigger map, which means that a map contains a larger number of units. An assumption is made before training that there is a virtual map L_0 above the first map L_1. L_0 contains only one unit whose weight vector, w_0, is the mean value of the untrained input data set, X, which contains N input samples.

$$w_0 = \frac{1}{N}\sum_{i=1}^{N} x_i \qquad \text{(Eq. 1)}$$

   Thus, the AQE of L_0 is:

$$AQE_0 = \frac{1}{N}\sum_{i=1}^{N} \|x_i - w_0\| \qquad \text{(Eq. 2)}$$

   1.2) Define learning rates α_b and α_s for the Best Matching Unit b, and its neighbours s, respectively, where $0<\alpha_s<\alpha_b<1$.
   1.3) Define an age threshold, β, for a connection C_{ij}, where $0<\beta\le 1$. The β cooperates with the current highest age of connection to decide whether a connection is too old. A β adjusting parameter, J_β, is defined as well, which is used to modulate β based on the current data samples.
   1.4) Define S_{min}, a minimum number of input samples which a map represents. The default value is two because the minimum number of units is two in a map. S_{min} can also be set as a proportion of the size of input data, where $0<S_{min}<1$. In this case, S_{min} will be found by $S_{min}\times N$. S_{min} influences the depth of a DASH hierarchy. A smaller S_{min} makes the DASH expand deeper down a hierarchy.
   1.5) Let O_l be a temporal maximum number of units in a map for the layer l. It is defined by (Eq. 3). The value of 3 is used as the minimum of O_l because a sub-map of DASH starts with 2 units, which allows the model with one spare unit to grow. The value of 100 is applied in (Eq. 3) for two reasons. The first is that the model is better if it can achieve the quality requirement using a smaller map. The model is forced to train properly rather than adding units to pursue a smaller AQE. The second reason is that a very large map is not preferred because it is hard to analyse or visualise. Besides the parameter, O_l, an γ adjusting parameter, J_γ, is also defined to modify O_l, where $0<J_r\le 1$. O_l will be modified in the self-adjusting phase, if a map contains O_l units but does not meet the map quality.

$$\begin{cases} O_l = \max(3, \min(100, \frac{S_{\min}}{2})), \text{ where } l=1 \\ O_l = \max(3, \min(O_{l-1}\times\frac{\tau}{2}, \frac{S_{\min}}{2})), \text{ where } l>1 \end{cases} \qquad \text{(Eq. 3)}$$

2) Local network initialisation
   2.1) Determine an objective AQE based on the AQE in the direct parent map.

$$\begin{cases} objective\ AQE_l = AQE_{l-1}\times\tau, \text{ where } l=1 \\ objective\ AQE_l = AQE_{l-1}\times\tau^2, \text{ where } l>1 \end{cases} \qquad \text{(Eq. 4)}$$

   2.2) Based on O_l, define how often the unit grows as follows:

IterGrow= $\frac{N_l}{O_l - 1}$, (Eq. 5)
where N_l is the number of current input data.

2.3) Create two units and initialise weights randomly from $p(X)$.

2.4) Re-order the current data set randomly.

3) Learning Phase

3.1) Generate a data sample x_i for the model.

3.2) Calculate the Euclidean distance of each unit to x_i and decide the Best Matching Unit, *b*, and the Second Best Matching Unit, *s*, by

$$b = \arg\min_{n \in L} \|x_i - w_n\| \text{ and} \quad \text{(Eq. 6)}$$

$$s = \arg\min_{n \in l/\{b\}} \|x_i - w_n\| \quad \text{(Eq. 7)}$$

and connect them as C_{bs}, if it does not exist.

3.3) Update the weights to the BMU *b*, and other units *n*, with a connection from *b*:

$$w_b(t+1) = w_b(t) + \alpha_b \cdot (x_i - w_b) \text{ and} \quad \text{(Eq. 8)}$$

$$w_n(t+1) = w_n(t) + \alpha_s \cdot (x_i - w_n) \quad \text{(Eq. 9)}$$

3.4) Add 1 to the age variables for all connection *C*, but zero to C_{bs}.

$$\begin{cases} age = age + 1 \\ age_{bs} = 0 \end{cases} \quad \text{(Eq. 10)}$$

3.5) Increase the error to the BMU error variable, err_b:

$$err_b(t+1) = err_b(t) + \|x_i - w_b\| \quad \text{(Eq. 11)}$$

4) Pruning Phase: at each *n IterGrow* iteration, where *n* is ≥ 1.

4.1) Find the maximum age of connections currently.

age_{max}=arg max(age) (Eq. 12)

4.2) Remove any connection whose age is larger than a portion of the maximum age of connections currently.

Remove C_{ij}, if $\frac{age_{ij}}{age_{\max}} > \beta$ (Eq. 13)

This will be carried out if the number of units is more than two.

4.3) Prune any unit without any connection but still keep the minimum number of units, 2.

5) Growing Phase: at each *IterGrow* iteration, insert a new unit as follows:

5.1) Find the unit *q* with maximum accumulated error:

$$q = \arg\max_{u \in L}(err) \quad \text{(Eq. 14)}$$

5.2) Find the unit *f*, the unit with the highest accumulated error amongst the neighbours of *q*.

$$f = \arg\max_{u \in Neighbours_q}(err) \quad \text{(Eq. 15)}$$

5.3) Insert a new unit *r* to a map and initialise its weight by interpolating weight vectors q and *f*.

$$w_r = \frac{w_q + w_f}{2} \quad \text{(Eq. 16)}$$

5.4) Set up the *err* variables for units *q*, *f* and *r*.

$$err_q = \frac{err_q}{2} \quad \text{(Eq. 17)}$$

$$err_f = \frac{err_f}{2} \quad \text{(Eq. 18)}$$

$$err_r = \frac{err_q + err_f}{2} \quad \text{(Eq. 19)}$$

5.5) Connect unit *r* to unit *q* and *f*. Set up the age variable for these two connections, i.e. C_{rq} and C_{rf}.

$$age_{rq} = age_{rf} = age_{qf} \quad \text{(Eq. 20)}$$

5.6) Remove the connection between unit *q* and unit *f*.

6) Check the condition whether the map AQE meets the objective AQE at each *IterGrow* iteration.

6.1) Evaluate the AQE for a map *l*:

$$AQE_l = \frac{1}{N_l} \sum_{i=1}^{N_l} \|x_i - w_{b(i)}\|, \quad \text{(Eq. 21)}$$

where $w_{b(i)}$ is the weight vector of BMU for input sample *i*.

6.2) If $AQE_l \leq objective\ AQE_l$, then stop training for this map, or go to the self-adjusting phase.

7) Self-adjusting Phase: at some *IterGrow* iteration, modify some parameters to suit the object AQE_l.

7.1) Increase the age threshold, β by the adjusting parameter, J_β, if units are not growing.

$$\beta(t+1) = \beta(t) \times (2 - J_\beta), \quad \text{(Eq. 22)}$$

where $0.5 < J_\beta \leq 1$.

7.2) Decrease the age threshold, if the number of units reach O_l, which is the reference number of units in a map:

$$\beta(t+1) = \beta(t) \times J_\beta, \quad \text{(Eq. 23)}$$

7.3) Increase the reference number of units in a map, if the number is reached.

$$O_l(t+1) = O_l(t) \times (2 - J_r), \quad \text{(Eq. 24)}$$

where $0.5 < J_r < 1$.

8) Put all units whose AQEs are greater than the objective AQE in the same layer into the training pool if the number of their associated input vectors is greater than Smin.

9) Continue the hierarchical training until there are no training requirements in the training pool.