7. Conclusions

7.1 Further Work

This section deals with the areas of work that were outside the scope of this thesis but are still noteworthy to give the reader an insight into the direction this research might take.

There are three ideas presented here:

1. Further investigation into the PSOM family. Exploration into the effects of different types of non-stationary class (such as overlapping classes) on the networks.
2. Self Tuning Plastics (STP). The aim of STP is to allow the PSOM and TPSOM to set their own parameters given changes in the input space. This would be achieved by the user setting a more intuitive criterion (such as desired error), rather than requiring prior information regarding the spatial and temporal distributions of the data. Investigations could be carried out into the types of dynamic system that would work well with STP and those that were not suitable (such as rapidly changing systems).
3. Artificial Viruses and Immune Systems. These are methods of forcing novelty or stability into the network by alteration of the topology such that the network is perturbed. The benefits of these additional algorithms are the removal of the redundant neurons and the removal of stable states. This would be a benefit in systems where the (E)PSOM has represented the class as two stable neurons groups. Only by linking the groups or removing one group can this situation be rectified.

7.1.1 Further investigation into the PSOM family.

To improve understanding of the PSOM family, the following investigations are suggested.

1. *Overlapping Classes*. A new dataset could be created in which outliers from some of the classes overlap with patterns from other classes. It is hypothesised that the PSOM would create two different classes that would be joined together with long links. The class labels of the neurons on the edges of this structure will be likely to oscillate between the two class numbers. Given that the PSOM has a sharper neighbourhood cut-off, compared to the smooth EPSOM neighbourhood, it would be expected that the PSOM would break the classes apart quicker. This is not necessarily desirable as a faithful representation of the input space may not be wanted.
2. *Slow Moving Classes*. A slow moving class is a class that moves in the input space in such a manner that it is still recognisable as the same class. Many classes in the real world experience this drift, normally due to physical properties. One example of this drift is in mechanical sensors where gradual wear records data differently. In radar, a slow drift may be caused by a changing of the aspect of the object that is being detected (such as an aircraft rolling). An investigation into the highest speed of motion that the algorithms could track would be useful. For the PSOM family, this speed is likely to be some fraction of the Node Building Parameter (a_n). As a_n decreases, the quickest speed at which the PSOM family can track is also likely to decrease. Beyond this highest speed, the fast moving class is likely to be seen as a new class, instead of being tracked as a known class and hence the PSOM family would detect it as novel.

7.1.2 Self Tuning Plastics

Self Tuning Plastics (STP) is the name given to methods of allowing the plastic networks presented to set their own parameters automatically. In all of the tests described in Chapters 3 and 5, there was an assumption that there was enough knowledge about the input environment to set the network parameters accurately. This information must be both spatial (to set the node building parameter) and temporal (to set link ageing or learning rate parameters). However, one advantage of parameters like these is that they can work without complete information about the input environment. The (E)PSOM networks have been shown to be robust in the setting of their parameters (not sensitive to the value to which they are set) and thus it may be possible to derive a function whereby the parameters of the network are derived from some other *known* quantities. As shown in the tests, fluctuations of network parameters do not generally lead to catastrophic failure of the network but a graceful degradation of its operation. It is this degradation that would be measured to gauge the performance of the tuned parameters.

The parameter setting function could operate like a cost function, whereby the size and recognition error of the network would be measured and used to control the parameters. It can be seen from PSOM Tests 2 to 4 (Chapter 3), that there is some correlation between the output error and the values of the node building parameter (a_n) and the link ageing parameter (b_a). These relationships could be exploited to minimise both over time or to minimise each on a proportional basis. The latter would require the user to set the proportions by means of more parameters, but this is undesirable as the aim of STP is to *reduce* the number of parameters set by the user. The new minimisation parameters would be more intuitive to set – for example according to speed or memory requirements. These parameters would represent a compromise between accuracy and efficiency; they would also be easier to set prior to training from goals given by the user than by analysing the data set.

One potential difficulty with such a scheme is coping with system dynamics. If the system to be modelled is very fast moving, then the STP network may not have enough example patterns from a class to gather information for parameter tuning *and* to measure the effect of the changes before the system has changed.

Another possible difficulty with STP is the effect of noise. In noisy systems, it is difficult to measure the recognition error, as the difference between novelty (the arrival of a new pattern) and a noisy version of an existing pattern is less pronounced. How the novelty threshold is set may be a difficulty for the STP process.

The user-defined parameters of STP would be based upon parameters such as efficiency, which can be calculated by evaluating the size of the network with respect to the number of different classes held. It can be seen from the results that for efficiency to increase, the number of neurons and links needs to be reduced. When this occurs, the recognition error increases. Thus, the user will need to trade these off against one another.

7.1.3 Artificial Virus and Immune Systems

This section details the workings of Artificial Virus and Immune Systems (not to be confused with Forrest and Perelson 1991 where an immune system was modelled to investigate pattern recognition). These are algorithms that operate asynchronously with the learning algorithm and affect the topology of the network. An Artificial Virus is an algorithm that removes structure (neurons and links) from the network to perturb the network from any possible stable states. An Artificial Immune System is an algorithm that adds links to the network to forge new connections between disconnected groups of neurons representing the same class.

7.1.4 Artificial Virus System

An Artificial Virus System (AVS) is defined as an algorithm that perturbs the topology of the network by removing structure. Structure in this case is defined as neurons and links. The AVS does this asynchronously with the learning algorithm and thus does not interfere with the normal learning operation of the network. By removing structure, the network may be forced to relearn information that is lost in the removal or to adapt existing neurons such that patterns may still be classified correctly. This is forcing novelty into the network as the input set has not changed. Novelty occurs naturally when a pattern from a new class in the input set is presented

to the network, however if the network has lost information due to the AVS, then an old class may be flagged as a new one. Forcing novelty has two aims:

1. To pull the network out of stable states.
2. To remove redundant neurons that the learning algorithm is unable to remove.

An example of a stable state is given in the Temporal Plastic Self Organising Map chapter (Chapter 4, Figure 4.7), where two groups of neurons would not be removed by the algorithm due to their independence. The AVS would remove structure from the inactive group and eventually remove the group itself. This is desirable from an efficiency point of view, as efficiency is compromised by having more neurons than is necessary.

7.1.5 Artificial Immune Systems

An artificial immune system (AIS) seeks to mimic the operation of a real immune system by repairing damage to the Neural Network, caused either by age or artificial viral infection. The AIS works asynchronously with the operation of the network and does not alter the training algorithm. The AIS seeks to build new links between neurons that represent the same class, but have long since been split apart. The AIS also serves to forge new links between groups that might be similar, working in opposition to the AVS by adding stability and structure to the network. An example of this 'repairing' of two separated classes is shown below.

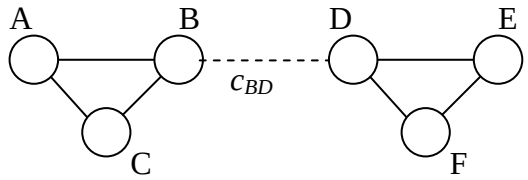

Figure 7.1 Two stable neuron groups joined by an immune system link

Error! Reference source not found. shows two stable neuron groups, ABC and DEF. These two groups both represent the same class but are not connected. Currently, there is no method to connect these two groups together. An AIS could be used to create a new link, c_{BD} between these two groups, correcting the topology of the network. The AIS is a stabilising method, and thus it might be seen as a desirable technique to operate continually on the network. This is not the case however, as the

addition of superfluous links will have a detrimental impact on the efficiency of the network.

The AIS has two functions:

1. Addition of links between neurons not already connected.
2. Strengthening of existing links between neurons.

The AIS can operate in two distinct manners:

1. *Global Immune System.* Measures the global properties of the network (such as a running average of the Recognition error) and then adds or strengthens neurons given this global information.
2. *Travelling Immune System.* The Immune System is local to an area of the network and 'inhabits' a neuron. For each iteration, the Travelling immune system could either create new links or strengthen existing ones, moving from neuron to neuron.

The main advantage of the Global Immune System is that it can operate across the whole network and thus make a better decision about where to strengthen or create links. However, many comparisons will be required and this will add a large additional computational overhead to the operation of the network. This increase in computation is likely to be of the order of the square of the number of neurons as the immune system would compare every neuron to every other, giving $n \times n$ operations, where n is the number of neurons.

The more favourable method is the Travelling Immune System. Once initialised (randomly, as a function of network parameters or as a function of time), the Travelling Immune system operates in a similar manner to the Travelling Virus, moving from neuron to neuron. Instead of removing structure, the Travelling Immune System adds or strengthens the structure. This immune system is not bounded by, but does use, the topology of the network, as it can cross to other parts (creating links as it moves).

7.1.6 Balancing Immune and Viral Systems

The Artificial Immune Systems and Artificial Viral Systems could be used concurrently on the network. Although both algorithms may not be applied at the same time, they could be used together to perturb the network. The virus would remove useless links and the immune system would prospectively add links, in search of a better structure. It is important to note that the damage and repair to the network might be such that the learning algorithm cannot operate consistently, however. Therefore, a balance between the two is required (if both are used). It would be worth investigating the effects of this balance on systems that require some viral aid to keep efficient or immune system additions to remain connected. Extensive testing under different conditions and parameter settings would be required to determined the usefulness of immune and viral systems, either on their own or concurrently.

7.2 Summary

This thesis has focused on Dynamic Neural Networks, Networks which can alter their topology to adapt to datasets that are infinite and non-stationary. This kind of network must be able to change its entire structure over time. This can be facilitated by the addition and/or the removal of neurons and links. Growing networks do add topology, but cannot operate on unbounded input sets that occur for a theoretical infinite time. Pruning networks remove topology, but only as a method of increasing efficiency. Classic Neural Networks ignore any temporal information about the ordering of patterns. This can be encoded with time information (as with temporal Neural Networks) but is not explicit within the structure of the network. Only one example of a truly dynamic Neural Network has been found – the GNG-U algorithm that had the ability to track non-stationary datasets by adding and removing neurons. It performs this function in a topology preserving manner but no time information was encoded in the structure of the network – at any one time the network showed the current state of the input environment but no information regarding its past.

Three novel algorithms were presented in this thesis. The Plastic Self Organising Map is an extension of Kohonen's Self Organising Map. Unlike the SOM, the PSOM's structure is plastic and the neighbourhood is specified by spatio-temporal links between the neurons. The PSOM also has the ability to add and remove neurons

in response to the dynamics of the input set. New classes can be classified and old ones forgotten.

The Extended Plastic Self Organising Map replaces the discontinuous neighbourhood function of the PSOM and replaces it with a continuous one. By doing so, the Clustering Threshold of the PSOM is removed. To facilitate a stronger neighbourhood function, the link lengths are also scaled. Apart from these two characteristics, the EPSOM functions in same way as the PSOM.

The major difference in the Temporal Plastic Self Organising Map with respect to the other two algorithms is that a link is unidirectional and represents a probability of the destination neuron representing the next class presented, given the origin of the link represents the current class. Thus, the class of the next input can be quantatively estimated given that the current input. This allows a Markov-chain map of the temporal properties of the input environment to be generated from the data directly. As the TPSOM is a dynamic network, it can also operate on Markov chains for which there is a switch between 2 state transition matrices.

The operation of the three networks was studied empirically. An example, artificial data set was created to study the operation of the networks. This data set was non-stationary, including a class that disappeared and one that appeared late in the network's operation. All algorithms were able to classify the different classes, and keep a compact structure by removing neurons and links that represented classes that were no longer part of the input set. The (E)PSOM algorithms showed a spatio-temporal representation of the input space that was difficult to interpret. The TPSOM topological graph was unambiguous. A noise data set was created using the artificial data set as the starting point. The (E)PSOM algorithms continued to classify well but the TPSOM network's topological output was difficult to interpret due to the large number of links.

The effects of the parameters were tested by using identical data sets and varying the parameter settings. It was found that the PSOM and (E)PSOM operate in a similar fashion, with the PSOM performing better in noise free environments and the EPSOM operating better in noisy environments.

To show a real-world problem being solved, the algorithms were evaluated using a radar dataset. In this data set a class is a radar source (such as an aeroplane) and each pattern is an instance of a radar pulse arriving at a receiver. In this case, the (E)PSOM correctly classified the patterns but had a tendency to split one of the input classes. This was because the class boundaries were of an irregular shape. Further investigation into the manner that the EPSOM propagates its class numbering may be beneficial in improving the performance under these conditions. The ability to set a sharp class boundary in the PSOM allowed it to outperform its extended counterpart. The TPSOM did not find any structure in the Radar data, but this was because of a lack of structure in the data, rather than ineffective operation of the network.

A comparison was made between the PSOM and its closest counterpart, GNG-U (Fritzke, 1997), using two discontinuous classes. Both networks found the two distinct classes. The final GNG-U network gave a topographical representation of the input space but did so in many more iterations than the PSOM. Although the PSOM operated more quickly, its final representation was a spatio-temporal one and thus more difficult to interpret. GNG-U requires the user to analyse the topography to identify the different classes, whereas the PSOM performs this task automatically.

It was found that the TPSOM could represent a Markov-like system with a state transition matrix similar to that of the input. The TPSOM was also shown to work on a Markov chain, for which the state transition matrix was changed halfway through the operation of the network. The network topology changed smoothly and adjusted to the new input system. Classic Markov chain structures are static, but this result indicates that the TPSOM may be used to identify systems that are *dynamic Markov structures*. An example would be a passage of music that follows a sequence during a verse but then changes to a different sequence for a chorus.

Vowel pairs from Charles Dickens' Great Expectations were used to show the TPSOM operating on a real dataset. In this case, it represented the main statistical properties of the text but not the subtleties. This was due to the manner in which new links are added and the effect of the network learning rate.

It was found that the (E)PSOM algorithms are less sensitive to their parameters than the TPSOM. This is not surprising result, as the TPSOM only uses one neuron to represent a class, rather than the multiple neurons used by the (E)PSOM.

The PSOM algorithm can be seen as a proof-of-concept algorithm, operating in a basic manner. Artificial virus and immune systems could be used to either strength the network or perturb it from stable states. Another area of interest is the notion of Self Tuning Plastics (STP) that aims to remove the requirement for *a priori* setting of network parameters and let the Neural Network set its own parameters based on network properties such as size.

In the search for a dynamic network that can operate on non-stationary systems, it has been shown that the PSOM, EPSOM and TPSOM networks have all demonstrated aspects of this ability. As long as the general characteristics (location, size and temporal persistence) of a class remain similar, the classes within the input domain can change entirely and the network will change to meet the alteration. The topologies of the networks grow and shrink such that they can represent the input space without a huge loss of efficiency. This growing action can also be used to detect novelty online. This is of benefit for mobile systems where it would not be feasible to have large data stores that a traditional neural network would require.

Three Dynamic Neural Networks have been proposed, whose topology changes in response to novelty in a non-stationary signal space. The PSOM and EPSOM, extensions on Kohonen's SOM can cluster and classify a multidimensional input space. In addition, the TPSOM can be used to build up Markov Chains. Further work might include perturbing the structure of these networks and finding further real world problems to solve.