An Investigation of the Negative Selection Algorithm for Fault Detection in Refrigeration Systems

Dan W Taylor[1,2] and David W Corne[1]

[1] Department of Computer Science, University of Reading, Reading, UK
d.w.corne@reading.ac.uk

[2] JTL Systems Ltd, 41 Kingfisher Court, Hambridge Road, Newbury, UK
dan.taylor@jtl.co.uk

Abstract. Supermarkets lose millions of pounds every year through lost trading and stock wastage caused by the failure of refrigerated cabinets. Therefore, a huge commercial market exists for artificially intelligent systems which are able to detect the early symptoms of faults. Previous work in this vein, using real-world data and now in the throes of being deployed commercially, has employed evolved neural networks to predict volumes of temperature and other alarms emerging from refrigeration system controllers, and also to predict likely refrigerant gas loss from such alarm patterns. In this work we turn to the use of in-cabinet temperature data which has recently become available, and the aim is to predict refrigeration system faults from the pattern of in-cabinet temperature over time. We argue that artificial immune system inspired technologies are particularly appropriate for this task. The negative selection algorithm is therefore investigated as a tool to detect anomalous patterns in temperature data. Using a simple AIS system in preliminary experiments to assess feasibility, we compare the performance of a simple matching rule based on Euclidean distance and one using traditional *r contiguous bits* matching. We also investigate a 'differential' encoding scheme which reduces the size of pattern space while retaining what we feel to be the essential elements of patterns in this application. We find that feasibility for AIS in this application is proven, with particularly good self-detection rates, and a fault-detection rate adequate as a springboard for further development. The best AIS configuration of those examined here seems to be that which uses the novel differential encoding in conjunction with the *r*-bits matching rule. The differential encoding scheme is simple, yet seems powerfully suited to such time series pattern analysis tasks.

1 Introduction

The role of a supermarket refrigerator is to keep food within a constant temperature range and thus ensure that customers' purchases remain in a good condition. The reputation of the store and, more importantly, the health of the consumer depends upon the refrigerator's ability to perform this core task. There are, however, a huge number of other factors which affect the design of refrigeration systems. Power usage must be kept to a minimum for both financial and environmental reasons. Maintenance must also be kept to a minimum to ensure that a refrigerated cabinet can remain in constant operation. Temperatures must be recorded on a regular basis in order to comply with public health legislation, and cabinets must be able to cope with continual disturbances caused by customers selecting their purchases.

For these reasons, supermarket refrigeration systems are far more complex in their design and operation than household refrigerators and many larger commercial refrigeration systems. In almost all supermarkets, refrigerant is pumped around the store via a network of piping. The evaporation of refrigerant removes heat from cabinets on the shop floor and the resultant gas is then compressed and pumped to condensers outside the building, where heat is expelled. Evaporators, the component in which refrigerant is allowed to evaporate to absorb heat, must be regularly defrosted to prevent a build-up of frost and ice caused by moisture in the atmosphere. Defrost often involves the application of heat to the evaporator from an electric heating element.

Embedded control systems within every refrigerator are used to control the evaporator and thus the temperature of the cold-space and to manage defrosts. Similar controllers in plant machinery govern the operation of compressors and condensers as load on the refrigeration system changes. These are linked to a central data management unit which co-ordinates the timings of defrosts and provides a mechanism for logging data. A PC is linked directly to the central data management unit to provide additional data logging, reporting and management functionality for maintenance engineers and store staff.

Primarily owing to the fact that when this goes wrong (as it very commonly does), the cost adds up to several millions of pounds per year for supermarket chains, this highly complex system provides a rich source of data for experimentation with data mining, prediction and classification techniques for the prevention of faults. There exists a great demand for such prediction systems within the supermarket industry.

JTL Systems Ltd are a refrigeration controls manufacturer, based in the UK. JTL operate an Alarm Monitoring Centre, which monitors alarms raised by systems in supermarkets throughout the UK. Monitoring centre staff manage the dispatch of engineers and warn store staff of faults, based on the alarms they receive. The alarm monitoring centre currently monitors around 40,000 controllers in 500 supermarkets distributed across the UK. Around 1.5 million alarms

are received each year. Previous work by the authors has involved the development of systems to predict daily alarm volumes [1] and to detect the early symptoms of refrigerant leaks based on alarm data received at the monitoring centre [2]. Recent advances in connectivity within the JTL network have made the collection of temperature data from cabinets much quicker. It is now feasible to begin targeting faults on a cabinet by cabinet basis. One of the most common cabinet level faults is *icing-up*.

Icing up occurs when defrosts are not sufficiently effective and a thick layer of ice builds up on the evaporator. The ice layer acts as an insulator around the evaporator, drastically lowering its efficiency until it is no longer capable of cooling the air in the cabinet. At this point the attention of an engineer will be required and the cabinet will need to be emptied and shut down for many hours. It is also possible that products within the cabinet will have reached an unsafe temperature and will need to be destroyed. A great financial loss will be incurred due to engineering expenses, stock losses and the hidden costs of lost trading from the cabinet.

There are thought to be characteristic patterns in temperature log data which are symptomatic of icing-up. These patterns can be recognised by human experts and used to detect problems before they become too serious. However, due to the sheer quantity of temperature log data it is impractical to monitor the system in this way. There is a definite need, therefore, for an automated system which is able to recognise the symptoms of icing up and raise an alarm to prevent stock being endangered and to allow timely preventative maintenance.

The task of recognising the symptoms of icing up in noisy real-world data is an ideal application for an AI based classification system, such as a neural network, rule based classifier or suchlike. The main problem with the development of such a system is the lack of data representing fault conditions. Though in excess of 4 billion temperature log data items are available to us, there is no feedback system in place to match temperature data to the occurrence of a fault. It is, therefore, extraordinarily hard to find a useful number of data patterns which are symptomatic of an iced up cabinet. Despite this, an abundance of data is available which is indicative of normal operation. Most AI techniques, such as neural networks or rule based classifiers, need both positive and negative examples in order to learn correctly. This makes them close to unusable in this problem domain.

Artificial Immune Systems (AIS) are a relatively new paradigm in AI and have the distinct advantage of requiring only positive examples for training. This makes AIS an ideal technology to investigate in this application. The remainder of this paper details work carried out to investigate the use of AIS to detect the early symptoms of icing up in supermarket refrigerators. For completeness, Section 2 gives some brief background into the biological inspiration for AIS and also includes a brief review of relevant research in the field to date (readers familiar with AIS can skip this section, although section 2.2 does provide the introductory focus for the issue of matching, the strategy for which is an experimental variable later on in section 4). A more detailed exploration of the refrigeration data can be found in section 3. The results of experiments carried out to investigate the usefulness of AIS in this domain and to determine appropriate AIS configurations are included in section 4 and some conclusions and suggestions for further work can be found in section 5.

2 Immune Systems, Artificial and Otherwise

Artificial Immune Systems (AIS) have become an increasingly popular area for study by computer scientists in recent years. The term AIS refers to any computer system which is inspired by the natural immune system. This is by no means limited to computer models of the immune system which are used by biologists to gain a better understanding of the subject. Detailed reviews of the various AIS architectures which exist can be found in [3] and [4]. This section briefly outlines the operation of the human immune system, before detailing the nature of the *negative selection algorithm*, which forms the basis of experiments carried out later in the paper.

2.1 The Human Immune System

The immune system is a natural, distributed system which protects us from attack by foreign bodies known as *pathogens*. The job of the immune system is to monitor the body and make classifications as to whether the items it encounters are *self* or *non-self*. Everything which the immune system categorises as non-self, such as bacteria, cancer or tumour cells, must be destroyed. Everything which is a natural and healthy part of the body should be categorised as self and should remain untouched by the immune system. The term *antigen* is used to denote anything which the immune system is able to recognise. Antigens are essentially features which occur in both the body's cells and pathogens such as bacteria. The self/non-self categorisation is based on the antigenic patterns on the surface of cells.

The human immune system operates on two main levels. Innate immunity is the first and most simple line of defence. The innate immune system consists of basic, passive defences such as the skin, mucus membranes and other structures which help to keep foreign bodies out. When the innate immune system fails to keep a pathogen out of the body it is up to the acquired (or adaptive) immune system to destroy it [5]. Acquired immunity is based on a population

of lymphocytes [6]. Lymphocytes are cells which circulate around the tissues of the body in the blood stream. They are capable of recognising and responding to pathogens, initiating an immune response which destroys them.

Two types of lymphocyte exist, B-cells and T-cells. Both of these are able to respond to antigenic patterns, presented on the surface of pathogens. Receptors on the surface of lymphocytes chemically *match* antigenic material. T-cells stimulate B-cells to proliferate and destroy the pathogen. The magnitude of this immune response is proportional to the strength of the match between T-cell and pathogen.

On detection of a pathogen, B-cells undergo a process of *Clonal Expansion* and *Affinity Maturation*. During this process B-cells rapidly reproduce with high mutation rates (known as *hypermutation*). Those B-cells which more closely match the antigen are encouraged to reproduce more quickly, while those which do not match will reproduce more slowly. Thus, the population of B-cells which can correctly recognise and react to the pathogen in question is quickly increased in size. Clonal expansion refers to the rapid, single parent reproduction which takes place to produce a large population of targeted B-cells. Affinity maturation is the Darwinian process which increases, through mutation and the reward of closely matching lymphocytes, the strength of the match between pathogen and B-cells within the immune repertoire.

T-cells are produced in the bone marrow, but undergo a *Maturation* process in the thymus. Maturation involves the presentation of *Self* proteins to naïve T-cells. Any T-cell which matches a self protein is destroyed. This prevents the immune system attacking the body itself. Auto-immune diseases are caused by a breakdown in the maturation process which allows T-cells which match self to be allowed out of the thymus. The censoring process which destroys T-cells which match self patterns is known as *Negative Selection*.

2.2 Artificial Immune Systems and Negative Selection

The negative selection algorithm was first used by Forrest et al [7] as a means of detecting unknown or illegal strings for virus detection in computer systems. A population of detectors is created to perform the job of T-cells. These detectors are simple, fixed length binary strings. A simple rule is used to compare bits in two such strings and decide whether a match has occurred. Such a match is equivalent to a match between lymphocyte and antigen.

Every randomly generated candidate detector is compared to every pattern in the self set. The self set is analogous to the self proteins stored in the thymus in that it contains examples of self, against which detectors are tested. Any detector which matches any pattern in the self set is not included in the detector set. New patterns can then be gathered from the system which is being monitored. These patterns are translated into the appropriate binary form and compared to the detector set. If the tested pattern matches any detector in the detector set then it can be guaranteed that that pattern is non-self and action can be taken accordingly.

If an exact match were required by the matching rule, the detector set would need to contain detectors for every possible illegal string which could occur. This would lead to a huge computational overhead and an impractical algorithm. Instead, detection is probabilistic. Only *r* contiguous bits are required to be identical for a match to occur. The value of *r* is known as the matching threshold. Figure 1 shows two 8 bit binary strings which match when the matching threshold is less than 5.

0	1	1	**1**	**1**	**0**	**1**	0
0	1	0	**1**	**1**	**0**	**1**	1

Fig. 1. Two binary strings which match when r ≤ 4 but do not match if r > 4. Matching bits are shown in bold

Much other work has concentrated on the use of artificial immune systems and the negative selection algorithm for virus detection and computer security [8] [9]. Work has also been done to use self/non-self discrimination to detect anomalies in time series data [10] [11] [12] [13]. This is of particular interest and relevance to us. Time series data, in order to be used by negative selection based algorithms, must be transformed into a series of short binary strings. This is done by sequentially sampling the data in blocks of a given size. For example, for a time series $T=\{t_0, t_1, \ldots t_n\}$ and a pattern length of 4, the first pattern would contain $P_0=\{t_0, t_1, t_2, t_3\}$, the second would contain $P_1=\{t_4, t_5, t_6, t_7\}$ and so on. Each value is converted and stored sequentially in a binary string which forms the final representation of the pattern.

The self set is constructed using data patterns which represent the normal operation of the system. Patterns should be long enough to capture any important system behaviours. The detector set is then generated and negative selection is used to ensure that no detector matches any self pattern. New data from the system can then be matched against these detectors to find anomalies. Work in this area has, to date, generally focussed on the use of simulated datasets such as Mackey-Glass time series and simulated cutting tool data.

In the anomaly detection systems created by Dasgupta, Forrest et al [10] [11] [12] the binary encoding detailed above is used. However, by encoding self and detector sets as binary strings we run the risk of destroying the semantic value of relationships between data items [13], as the r contiguous bits required to match can lie across word boundaries. It has been shown, however, that increasing the number of symbols used to represent patterns (i.e. using a decimal rather than binary encoding) greatly increases the required size of the detector set and, thus, increases the computational complexity of the algorithm [13].

Another, more simple, matching rule can be applied in situations where a greater number of symbols is required, for whatever reason. This matching rule is based on the Euclidean distance between the patterns in p dimensional space, where p is the number of data items in each pattern. A threshold, D, is chosen for the matching rule and patterns are said to match if the distance between them is less than this threshold value. Patterns then *cover* an area of problem space with radius D. We explore this and other matching scenarios later on in section 4.

3 Refrigeration Temperature Data

3.1 Data Visualisation using "aiVIS"

Extensive use was made of the aiVIS data visualisation technique [14] which is essentially a visualisation technique for artificial immune networks. aiVIS also provides an invaluable tool for visualising the topologies of complex data sets and has been shown to give better results than other techniques such as principle component analysis for this task.

Data patterns are represented as dots in a two dimensional display space. The position of each dot is initially chosen at random and bears no relation to the data within the pattern itself. A mutual repulsion between all dots exists, which is proportional to proximity and is limited to a given radius. A measure of similarity is used to compare data in the patterns which the dots represent, similar dots are attracted to a degree proportional to the similarity of the patterns they represent. The Euclidean distance between data patterns is normally used as the measure of similarity. When iteration begins, similar dots will move toward each other, while other dots will move apart. After a given number of iterations the forces of repulsion and attraction will balance and movement will decrease. Full details are available in [14].

The result of this simple algorithm is that data will automatically cluster itself spatially. The effect is similar to that seen in a self organising map [15]. Figure 2 shows the aiVIS display on initialisation and after a series of iterations. The data set being viewed is a 'healthy' refrigeration data set, as detailed below in section 3.2, which contains further discussion of this figure.

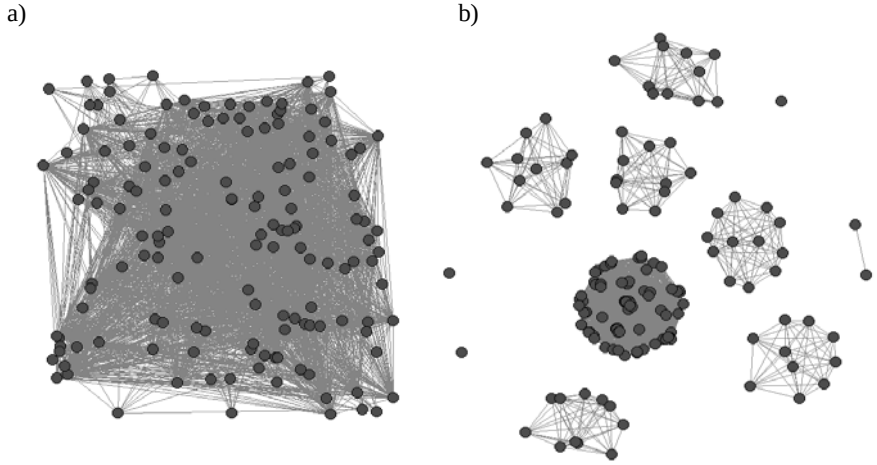

Fig. 2. aiVIS display of a) the initial random distribution of data patterns and b) the final clustered data set. See text in section 3.2.

3.2 Data Gathered from Sites

The operation of refrigerated cabinets in supermarkets is managed by an embedded controller. This controller uses temperatures, measured from various points around the cabinet, to control the flow of refrigerant through the evaporator and thus maintain a constant temperature. Each cabinet controller is linked, via a wired network, to the central data management unit, which it in turn linked to a PC. UK law demands that, in the interests of public health, the temperature of refrigerated food be logged at regular intervals throughout the day. The site PC maintains this logging database. Four

temperatures from each cabinet are logged at 15 minute intervals, 96 times a day. Most stores will contain around 100 refrigerated cabinets.

Figure 3 shows a sample plot of temperature data from a freezer cabinet over two days. Note the large peaks in the data. These correspond to defrosts of the evaporator which normally take place every 6 to 8 hours.

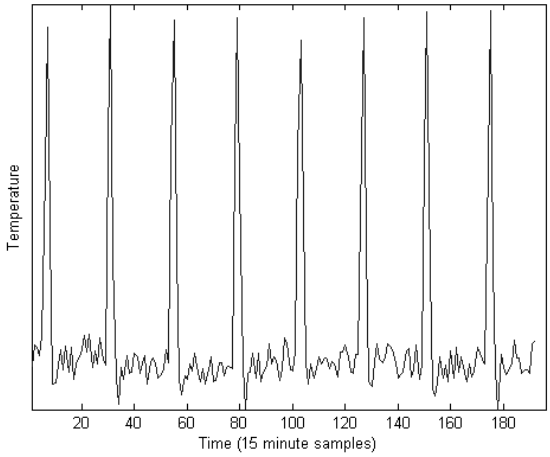

Fig. 3 Sample refrigeration temperature data showing transient defrost peaks.

Conversion of the temperature data to a form which is appropriate for AIS involved the translation of each real valued data item to a one of a finite set of symbols. The symbols chosen for use here are positive integers {0 .. n}. Here we use a set of 10 symbols, represented by the integers {0 .. 9}. The range of the real valued data is divided into ten equally sized compartments. Data items are translated to the index of the compartment in which they fall. This is similar to [12] but in this case we do not perform the final step, converting the data to its binary representation.

Two data sets, A and B, are used for validation in the experiments presented in the next section. These contain appropriately encoded data from two ice cream cabinets at a particular supermarket.

Figure 2 shows an aiVIS visualisation of data set A. The left figure (2(a)) shows the initial situation before running the aiVIS clustering algorithm, while the right figure (2(b)) shows the result. The large central cluster corresponds to patterns with no defrost features (generally consisting of a series of very small values {0, 1}). The satellite clusters each correspond to a group of patterns with defrost peaks at particular points in the cycle.

3.3 Simulated Fault Data

As mentioned towards the end of section 1, the determination of temperature data which is known to correspond to an occurrence of icing-up is very difficult. For this reason, it was appropriate to create a simulated fault data set for testing purposes. One of the main symptoms of an iced-up cabinet is a slow or incomplete *defrost recovery*. Defrost recovery is the decrease in temperature after the defrost is over. This decrease should be as fast as possible to protect stock. If the evaporator is encased in ice it is less efficient, so more time will be taken to bring the temperature of the cabinet down. This can be easily simulated for a temperature value X at time t by summing a proportion of previous temperature values into the new value.

$$X_t = X_t + (0.5\ X_{t-1}) + (0.25\ X_{t-2})\ . \qquad [1]$$

This will cause an unusually long defrost recovery which should be detectable by our AIS. A set, F, of faulty data was created using this technique on real refrigeration data. Not all of the patterns in F should be flagged as non-self by our AIS as they correspond to periods between defrosts where no fault is apparent. However, at least 50% of the patterns in F do exhibit signs of icing up.

4 Experiments

This section gives details of the experiments performed to investigate suitable encoding schemes and matching rules for detecting icing up in refrigeration temperature data sets. The data set which is being examined for faults will henceforth be known as the *checking set*. The two healthy data sets, A and B, and the simulated faulty data set, F, detailed in the previous section are used throughout as checking sets. A variety of data sets are used to form self sets of various sizes and types. In all cases, candidate detectors are generated at random before negative selection and insertion into the detector set as appropriate.

4.1 Experiments with Euclidean Matching Rule

The first experiments use a detector set of 10,000 detectors. The Euclidean matching rule is used for matching during negative selection and during examination of the checking set. Self, detector and checking set patterns are all of length 10. The self set contains data from four frozen food cabinets. Table 1 summarised the results.

Table 1. Results of experiment 1. 10,000 detectors, Euclidean matching rule used with D = 5 and D = 10. Columns show the data set used and the percentages of patterns categorised as self and non-self

Data	D = 5		D = 10	
	% self	% non-self	% self	% non-self
A	75%	25%	65%	35%
B	91%	9%	74%	26%
F	57%	43%	47%	53%

The results in table 1 show that the artificial immune system is having limited success when the Euclidean matching rule is used. Data sets A and B should be categorised in their entirety as self. The majority of the faulty data set should be categorised as non-self.

Note that lowering the threshold for the Euclidean matching rule from 10 to 5 gives us an improvement, simply by lowering the coverage of our detector set. Though performance is thereby improved on healthy data, it is degraded on the fault data. The lower rate of false non-self predictions can simply be seen as the result of the lower overall probability of non-self predictions.

An examination of the checking set with aiVIS revealed that the incorrectly categorised self patterns (i.e. healthy data patterns which have been flagged as non-self) are clustered closely together. Hence it was reasonable to posit that the detector set is matching an entire class of healthy data. Further examination showed that these wrongly classified patterns contained defrost peaks at a particular position. It turns out that the self set did not, in this case, contain defrost peaks in the same position. We can conclude from this that the high rate of misclassification in this first experiment was due to insufficient sampling. However, the same detector set correctly classifies many of the oversized defrost peaks in F. Though, given the insufficient sampling, we cannot be sure why this is, there seem to be two possibilities: either the oversized defrost peaks happen to fall at a position which the AIS has *not* learned is "self", or the detector set has correctly (rather than accidentally) categorised the oversized peaks as non-self.

In the next experiment we attempted to address the problems encountered in the preliminary experiment above by increasing the size of the self set and ensuring that it contains examples of patterns with the defrost peak in as many different positions as possible. The experimental setup was exactly as above, with the exception of the self set, which is extended to contain data from the original four and six additional frozen food cabinets from the same site.

Table 2. Results of experiment 2. 10,000 detectors, D = 5 and D = 10

Data	D = 5		D = 10	
	% self	% non-self	% self	% non-self
A	91%	9%	86%	14%
B	97%	3%	96%	4%
F	66%	34%	59%	41%

Table 2 reveals that the improved self-set leads towards a marked improvement over previous experiments in terms of false non-self predictions on healthy data sets A and B. A notable but less marked improvement is seen on the faulty data set, F. Note also that the results for detection with a threshold of D = 5 agree with our observations above. Lower matching thresholds cause a lower rate of non-self classifications on both healthy and faulty data.

Examination of checking sets A and B with aiVIS shows that many wrongly classified patterns do not fall in clusters and are unique, often with no lines of similarity to other patterns. These are obviously outliers in terms of the problem space and are, therefore, much harder to learn.

Next we attempt to enhance the performance of the AIS further using a still larger self set. Results can be found in Table 3. The experimental setup was again exactly the same as above, but with data from 13 additional cabinets added to the self set.

Table 3. Results of experiment 3. 10,000 detectors, D = 10

Data	% Self	% Non-self
A	99%	1%
B	100%	0%
F	65%	35%

Again, we find further improvement in the recognition of self, which now appears near perfect, but a robust level of fault detection still challenges the technique. Overall, from the experiments performed with an AIS using the Euclidean matching rule we can conclude that, as might be expected, a larger self set allows for the generation of a much better detector set. We can also conclude that, at least for the problem at hand, lower matching thresholds lead to the generation of a detector set with a worse coverage of the problem space.

4.2 Comparison of Euclidean and R-Bits Matching Rules

This section details experiments to compare the Euclidean matching rule, detailed above, with the R-bits matching rule as applied to decimally encoded data. Detector sets are generated using both the Euclidean and R-bits matching rules. A small self set is used initially, to give an idea of how each rule copes with incomplete data. Data from 8 frozen food cabinets is used to generate the self set. Both detector sets contain 10,000 randomly generated detectors. A threshold of D = 10 is used for the Euclidean matching rule and a threshold of R = 5 is used for the R-bits matching rule. In all cases data patterns are of length 10.

Table 4. Results of experiment 4. 10,000 detectors, D = 10, R = 5

Data	Euclidean		R-bits	
	% self	% non-self	% self	% non-self
A	69%	31%	81%	19%
B	80%	20%	84%	16%
F	51%	49%	73%	27%

From these results we can see that, on a small self set with incompleteness in terms of examples of every class of data, the R-bits rule is less likely to make false non-self predictions on healthy data. However, the R-bits rule fails to correctly categorise as many of the faulty patterns in F. This, again, is probably due to poor coverage of problem space by the detectors generated with the R-bits matching rule.

Table 5 shows results from experiments using the R-bits matching rule with matching thresholds of R = 3 and R = 7. Other than the changes in matching threshold, the experimental setup here is the same as above.

Table 5. Results of experiment 5. 10,000 detectors, R-bits matching rule used with R = 3 and R = 7

Data	R = 3		R = 7	
	% self	% non-self	% self	% non-self
A	53%	47%	99.9%	0.1%
B	55%	45%	99.9%	0.1%
F	39%	61%	100%	0%

The results in table 5 show that decreasing R makes detectors more sensitive, but does not make them more accurate. Though performance on F is improved, it is greatly reduced on A and B. On the other hand, increasing R made the detector set so insensitive that it failed to detect any of the fault data in F. We can conclude from this experiment that the

best value for R is around R = 5. Next we compare the performance of Euclidean and R-bits matching rules with a larger self set, using data from 13 frozen food cabinets.

Table 6. Results of experiment 6. 10,000 detectors, R = 5, D = 10

Data	Euclidean		R-Bits	
	% self	% non-self	% self	% non-self
A	99%	1%	95%	5%
B	100%	0%	92%	8%
F	68%	32%	85%	15%

Note that the results here are the best presented so far. Both matching algorithms perform comparably on A and B, but the Euclidean rule gives a better non-self classification rate on the faulty data set F. The fact that the Euclidean matching rule performs better here is probably due to the fact that the decimal encoding makes the number of detectors required to *cover* the problem space much higher. That is to say, when using a decimal encoding and the R-bits matching rule the probability of any randomly chosen pattern being categorised as non-self is low.

4.3 The R-Bits Matching Rule with Differentially Encoded Data

In attempt to decrease the size of the pattern space in a sensible way, and thus potentially increase the effectiveness of the R-bits matching rule, we experimented with a 'differential encoding' scheme. In conjunction with experts in the application area, our view is that the key elements of a faulty defrost temperature curve are not the precise pattern of real-valued temperatures, but the local 'ruggedness' of the temperature curve. The scheme we detail next is designed therefore to capture this structural element of the temperature pattern, while eschewing information which might otherwise lead either rthe Euclidean or *R*-bit matching rules astray. In close connection with this, the differential encoding allows us to use fewer symbols in our encoding scheme, decreasing the pattern space from 10^{10} for a decimal encoding to 10^3. The details are as follows. To construct a pattern in this encoding from a straightforward string of temperature values, we compare the temperature value at time *t* and at the previous sample point, t–1. We then use three symbols {0, 1, 2} to represent an upward slope, no change, and a downward slope respectively. The symbol we use for any given data value X at time t is given by:

```
if (Xt < Xt-1) then
  Symbol = 0
else if (Xt > Xt-1) then
  Symbol = 2
else
  Symbol = 1
```

An important point of note regarding this encoding scheme is that the comparison of data values at time t and t–1 is performed after the data has been encoded into a decimal form as in the above experiments. If the comparison is made based on real valued data we find that much of the semantic value is lost due to very minor differences between X_t and X_{t-1}. Using this technique we encode changes in successive temperatures, rather than the temperatures themselves. The following experiments examine the effectiveness of this encoding scheme when used in conjunction with the R-bits matching rule.

Table 7. Results of experiment 7. 10,000 detectors, R = 5, R = 7 and R = 8

Data	R = 5		R = 7		R = 8	
	% self	% non-self	% self	% non-self	% self	% non-self
A	99%	1%	96%	4%	96%	4%
B	91%	9%	86%	14%	88%	12%
F	80%	20%	59%	41%	64%	36%

In the next experiment, number 7, we attempt to asses the performance of the differential encoding scheme. Experimental setup is identical to that of experiment 6, with the use of the differential encoding scheme being the only difference. Table 7 shows the self/non-self classification rates for data sets A, B and F with various matching thresholds.

Although the system with R = 5 is the best performer on the healthy data sets (A and B) it does not provide a suitable rate of non-self categorisations on the faulty data set, F. When R = 7 we find that, although performance on A and B is slightly worse, the rate of non-self detection in F is much better. If the matching threshold is increased further, to R = 8, the performance on A and B is again enhanced, though we observe fewer non-self categorisations on faulty data.

4.4 Comparison of the Three Techniques

As a final comparison between the three encoding scheme/matching rule combinations studied here, table 8 shows the best results from the Euclidean matching rule and the R-bits rule with both decimally and differentially encoded data. A larger self set was used for these experiments to give the best results.

Table 8. Comparison between the three techniques presented here. 10,000 detectors are used with D = 10 for Euclidean matching rule, R = 5 for R-bits matching as applied to decimally encoded data and R = 7 for R-bits matching as applied to differentially encoded data

Data	Euclidean		R-bits (decimal encoding)		R-bits (differential encoding)	
	% self	% non-self	% self	% non-self	% self	% non-self
A	99%	1%	93%	7%	96%	4%
B	100%	0%	94%	6%	86%	14%
F	65%	35%	89%	11%	59%	41%

These results show that the best technique in terms of avoiding false non-self predictions in healthy data sets is the Euclidean matching rule with a decimal data encoding. A better all round performance, however, is given by the R-bits rule with a differential encoding for data. Though the rate of false non-self predictions is higher for this technique, the generated detector set is far more sensitive to non-self in the fault data.

5 Conclusions

In this paper we have reported on the some extensive but early-stage investigations of the use of AIS technology in spotting fault patterns in supermarket refrigeration systems. Specifically, the focus here is on time series temperature data from supermarket freezer cabinets, and the aim is to arrive at a fault detection system capable enough to be deployed on in-store PCs and/or at the JTL monitoring centre. Related work (in terms of the application) has examined the use of evolved neural networks to predict both future alarm totals [1] and refrigerant gas loss from controller alarms data [2], and this has recently been deployed commercially. Such commercial deployment is also a goal of this work, thereby exploiting the almost unique and continually maintained data sources available to JTL, and thereby being able to combat a major source of financial and customer concern to supermarkets.

AIS technology was therefore investigated here, owing to its promise and applicability in scenarios where ‘healty’ data are plentiful but faulty data are very hard to come by. The key task is to detect the early symptoms of icing up in refrigerated cabinets in supermarkets. Experiments with a number of different data encoding schemes and matching rules have been done, and this has allowed us to pinpoint a good design for seeding further development of an AIS system for this task. In particular, we have observed that the R-contiguous bits matching rule, in conjunction with a specialised differential encoding of data (which to our knowledge has not been employed before in published work on time-series data using AIS), gives the best trade-off between a low rate of false positives on healthy data and an accurate classification rate on *faulty* data. Along the way we have observed that the use of the Euclidean matching rule in this application gives a particularly low rate of false non-self predictions, but the cost in terms of failure to spot faults seems too high to recommend it as the focus of our further investigations.

The main result of the work reported here is that we regard the feasibility of AIS on this application as proven, taking into account the following points, and the potential offered by any of a number of extensions which are now under way in continuing work. We noted predictable and marked improvements in performance as more data were made available to construct the self set. In terms of the JTL application, this augurs well, since there are orders of magnitude more real data available to us than has been used in the experiments reported here. With an appropriate encoding – in particular the differential encoding – and the R-bits matching rule, we know that we can produce a system which spots towards half of the faults at the cost of around 10% false positive rate. This is well beyond what can be achieved with, for example, artificial neural networks at present since we do not have negative data. Nevertheless, the fact that we can to some (unknown) extent construct suitable negative data leads to the possibility of using such in the training of neural networks and other supervised learning systems for this task. This, along with further investigation of encodings based on the differential one, are prioritised for future work, along with the employment of more advanced architectures for the AIS itself towards achieving better fault prediction rates without compromising on false positives. For example, we plan

to investigate an addition to the AIS which others have looked at based on Immune Network theory, which models Clonal Expansion and Affinity Maturation by replacing poor detectors with new ones which are created from successful existing detectors using an evolutionary algorithm.

References

1. Dan Taylor, David Corne, David Taylor and Jack Harkness "Predicting Alarms in Supermarket Refrigeration Systems Using Evolved Neural Networks and Evolved Rulesets", *Proceedings of the World Congress on Computational Intelligence (WCCI-2002)*, IEEE Press *(2002)*
2. Dan Taylor and David Corne "Refrigerant Leak Prediction in Supermarkets Using Evolved Neural Networks", *Proceedings of the 4th Asia Pacific Conference on Simulated Evolution and Learning (SEAL), (2002)*
3. Leandro N de Castro and Jon Timmis "Artificial Immune Systems: A Novel Paradigm to Pattern Recognition", *Artificial Neural Networks in Pattern Recognition*, University of Paisley *(2002)*
4. Dipankar Dasgupta and Nii Attoh-Okine "Immunity-Based Systems: A Survey", *Proceedings of the IEEE International Conference on Systems, Man and Cybernetics*, IEEE Press *(1997)*
5. C A Janeway "How the Immune System Recognizes Invaders", *Scientific American: 41-47, (1993)*
6. Niels K Jerne "The Immune System", *Scientific American (229): 52-60, (1973)*
7. Stephanie Forrest, Alan S Perelson, Lawrence Allen and Rajesh Cherukuri "Self-Nonself Discrimination in a Computer", *Proceedings of the IEEE Symposium on Research in Security and Privacy*, IEEE Press *(1994)*
8. Stephanie Forrest, S A Hofmeyr, A Somayaji and T A Longstaff "A Sense of Self for Unix Processes", *Proceedings of the IEEE Symposium on Research in Security and Privacy*, IEEE Press *(1996)*
9. Jeffery O Kephart "A Biologically Inspired Immune System for Computers", *Proceedings of the Fourth International Workshop on the Synthesis and Simulation of Living Systems, (1994)*
10. Dipankar Dasgupta "Using Immunological Principles in Anomaly Detection", *Proceedings of Artificial Neural Networks in Engineering (ANNIE), (1996)*
11. Dipankar Dasgupta and Stephanie Forrest "Tool Breakage Detection in Milling Operations using a Negative-Selection Algorithm", *Technical Report CS95-5*, Computer Science, University of New Mexico *(1995)*
12. Dipankar Dasgupta and Stephanie Forrest "Novelty Detection in Time Series Data Using Ideas from Immunology", *Proceedings of The 5th International Conference on Intelligent Systems, (1996)*
13. Shantanu Singh "Anomaly Detection Using Negative Selection Based on the R-Contiguous Matching Rule", *Proceedings of ICARIS 2002*, University of Kent at Canterbury Printing Unit *(2002)*
14. Jon Timmis "aiVIS - Artificial Immune Network Visualisation", *EuroGraphics UK 2001 Conference Proceedings, (2001)*
15. T Kohonen “Self Organising Maps Second Edition” Springer *(1997)*