Data Stream Mining with Limited Validation Opportunity: Towards Instrument Failure Prediction

Katie Atkinson[1], Frans Coenen[1], Phil Goddard[2], Terry Payne[1] and Luke Riley[1,2]

(1) Department of Computer Science, The University of Liverpool, Liverpool, L69 3BX, UK; (2) CSols Ltd., The Heath, Runcorn, Cheshire, WA7 4QX.

Abstract. A data stream mining mechanism for predicting instrument failure, founded on the concept of time series analysis, is presented. The objective is to build a model that can predict instrument failure so that some mitigation can be invoked so as to prevent the failure. The proposed mechanism therefore features the interesting characteristic that there is only a limited opportunity to validate the model. The mechanism is fully described and evaluated using single and multiple attribute scenarios.

Keywords: Data Stream Mining. Classification, Instrument Failure Prediction

1 Introduction

Instrument failure within scientific and analytic laboratories can lead to costly delays and compromise complex scientific workflows [14]. Many such failures can be predicted by learning a failure prediction model using some form of data stream mining. Data stream mining is concerned with the effective, real time, capture of useful information from data flows [7–9]. This necessitates the adoption of an incremental online mechanism to process the data as it becomes available. Data stream mining is often applied in domains where we wish to use data stream information for prediction purposes (for example in the case of email, the prediction of spam email versus non-spam email [3]). Thus where we wish to predict the value of some variable x in the context of a set of variables y that feature in the data stream. The principal challenges is that the data is potentially infinite and therefore only a fixed proportion can be stored.

A common application of data stream mining is the analysis of instrument (sensor) data with respect to some target objective [4, 5, 12]. This paper explores the idea of using data stream mining to predict the failure of the instruments (sensors) themselves. This has the novel feature that persistent validation of the data stream model is not possible, the idea is that on predicting instrument failure appropriate maintenance is scheduled and therefore any predicted failure cannot normally be confirmed. More specifically this paper presents a probabilistic time-series analysis technique applied to data stream subsequences to predict

instrument failure. Of note is the mechanism whereby significant attributes in the data stream are separated from noise attributes using a probabilistic learning approach.

2 Formalism

We assume a set of k instruments $\mathbf{Inst} = \{inst_1, inst_2, \ldots, inst_k\}$. Periodically each (operational) instrument $inst_i$ transmits a "data packet" D_i. A data packet D_i associated with an instrument $inst_i$ comprises n values $\{v_1, v_2, \ldots, v_n\}$ such that each value is correlated with a data attribute. The set of available attributes is indicated using the notation $A = \{a_1, a_2, \ldots, a_n\}$. There is a one-to-one correspondence between D_i and A ($D_i| = |A|$). All instruments in the set **Inst** are assumed to subscribe to the same set of attributes. Periodically an instrument fails as a consequence of some fault. The conjecture is that this failure can be predicted by analysis of the attribute values.

The data packets continuously received from the collection of k instruments form a data stream. The information in the data packets is processed to produce a continuous time series. Each instrument $inst_i$ has a collection of n time series (one per data attribute), denoted by $\mathbf{T_i}$, associated with it such that $\mathbf{T_i} = \{T_{i,1}, T_{i,}2, \ldots, T_{i,n}\}$. Each time series T_{ij} comprises m real-valued readings $T_{ij} = \{t_{ij_1}, t_{ij_2}, \ldots, t_{ij_m}\}$ ordered chronologically such that t_{ij_m} is the most recent. The instruments continuously generate and transmit data and thus the value for m, for each time series, increases with time. Note that at any specific time the values for m across the set of instruments **Inst** is not necessarily the same, although it will be constant with respect to the collection of time series $\mathbf{T_i}$ associated with a specific instrument $inst_i$. Given a time series $T_{ij} = \{t_{ij_1} t_{ij_2}, \ldots\}$ for attribute j associated with instrument i the subsequence $S_{ij} = \{s_{ij_1}, s_{ij_2}, \ldots\}$ is a subsequence of T_{ij} such that $|S_{ij}| < |T_{ij}|$). The *current subsequence* for a time seres T_{ij} is the sequence $\{t_{ij_{m-p}}, \ldots, t_{ij_m}\}$ where p is some predefined subsequence length. Thus a current subsequence is the most recently received set of p values for a particular attribute j associated with a particular instrument i, we indicate this using the notation S'_{ij}.

3 Failure Prediction

From the foregoing the idea is to predict instrument failure according to the nature of the current time series subsequences associated with a particular instrument $inst_i$. We can learn the nature (shape) of subsequences that are good predictors of failure from observing the subsequences associated with instruments that have failed immediately prior to their failure. This is conceptualised as a two class classification (prediction) problem, failure versus non-failure.

There are a variety of ways of conducting time series based classification but it is generally acknowledged that the most satisfactory is the K Nearest Neighbour algorithm (KNN) [1, 2] where K is the number of neighbours considered (in many cases $K = 1$ is used, hence 1NN). The main issue is the distance measure

to be used [6]. The simplest measure is Euclidean distance; in which case the distance between two equal length subsequences $M = \{m_1, m_2, \ldots, m_l\}$ and $N = \{n_1, n_2, \ldots, n_l\}$ is calculated using Equation 1. An alternative might have been to use something more sophisticated like Dynamic Time Warping (DTW) [13, 15]; however, for the short time series considered with respect to the work presented in the paper, this seems unnecessary. The Euclidean based approach was therefore adopted.

$$dist(M, N) = \sum_{i=1}^{i=l} (m_i - n_i)^2 \quad (1)$$

As noted above, a particular challenge associated with the form of data stream mining considered in this paper is that we do not know which attribute is the sentinel attribute. So that we can learn which is the relevant attribute we need to allow λ instruments to fail. If we start making predictions at too early a stage in the process, instrument failure will be predicted unnecessarily causing the affected instrument to be taken off line, so that mitigating maintenance can be undertaken, when this was not required. Thus the process begins with a "learning phase" where we allow a number of instruments to fail so that we can build an effective *KB* for the purpose of future instrument failure prediction. This learning phase is measured in terms of the parameter λ.

Algorithm 1 Instrument Failure Prediction Engine ($main()$)

```
1: Input D_i
2: if D_i = null then
3:    Instrument has failed addSubsequencesToKB(S'_i)
4:    pruneKB()
5: else
6:    for ∀v_ij ∈ D_i do
7:       Update time series S'_ij in DB by removing S'_{ij_{m-p}} and adding adding v_ij
8:    end for
9:    if Not in learning phase then
10:      prediction(inst_i)
11:   end if
12: end if
```

Algorithm 2 Failure Prediction ($prediction(inst_i)$)

```
1: for j = 1 to j = n do
2:    for ∀S_j ∈ KB do
3:       if distance(S'_ij ∈ DB, S) ≤ σ) then
4:          Predict failure for inst_i
5:          Break
6:       end if
7:    end for
8: end for
```

The top level instrument failure prediction process is presented in Algorithm 1. The algorithm uses two storage structures *KB* (Knowledge Base) and *DB* (Database). The first is a set of subsequences of length p that are indicative of failure. *KB* is empty on startup and built up as the data stream starts to be processed. With respect to *KB* we used the following notations: (i) S indicates the complete set of time series subsequence, each of length p, contained in *KB*; (ii) S_j indicates the set of subsequences belonging to attribute j ($S_j \subseteq S$), and (iii) S_{j_z} indicates a specific subsequence z associated with attribute j ($S_{j_z} \in S_j$). The second storage structure, *DB*, holds a set of subsequences of length p, one for each attribute j, associated with each instrument i. *DB* is also empty on startup and built up as the data stream starts to be processed. With respect to *DB* we use the following notation: (i) S'_i to indicate the set of current subsequences associated with instrument $ints_i$, and (ii) S'_{ij} to indicate the current subsequence associated with instrument $ints_i$ for attribute j. The input to Algorithm 1 is the most recent data packet D_i for instrument $inst_i$. If this is empty ($D_i = null$) then this indicates an instrument failure and *KB* is updated with the set of current subsequences S'_i associated with instrument $inst_i$ using the function $addSubsequenceToKB(S_{ij})$ (line 3). A *KB* pruning process is then applied (line 4); the significance of this will become clear later in this section. Otherwise we update the *DB* entry for $inst_i$ by, for each attribute j, removing the oldest value and appending the newly received value (line 7). Recall that subsequences are of length p. If we are no longer in the learning phase we then enter into the prediction phase (line 10).

The prediction process is described in Algorithm 2. During this process the set of current time series sub sequences S'_i in *DB*, associated with instrument $inst_i$, is compared with the corresponding subsequences contained in *KB* (using the Euclidean distance measured calculate using equation 1; however, any other similarity measure could equally well have been adopted). If the computed distance is less than or equal to a given similarity threshold σ (line 3) failure for instrument $inst_i$ is predicted (line 4) and as a result appropriate maintenance applied.

The *KB* update process, $addSubsequenceToKB(S'_i)$, is presented in Algorithm 3. As noted above the challenge here is that we do not know which attribute is the sentinel attribute. We assume at least one, but it may be all n attributes or some proportion between 1 and n. Each subsequence S in *KB* has a weighting w associated with it. This weighting is adjusted as further current time series subsequences are added to *KB*. Whenever a weighting associated with a subsequence falls below a given threshold γ the subsequence is removed; this is the pruning step included in Algorithm 1 (line 4). The weighting w_{j_z} for a particular *KB* subsequence S_{j_z} is calculated as presented in Equation 2 where: (i) $count_{j_z}$ is a count of the number of occasions that S_{j_z} has been recorded as a result of instrument failure, and (ii) $|KB|$ is the size of *KB* in terms of the number of subsequences held.

$$w_{j_z} = \frac{count_{j_z}}{|KB|} \tag{2}$$

Algorithm 3 Update KB ($addSubsequencesToKB(S'_i)$)

```
1: Input S'_i
2: for j = 1 to j = n do
3:    if S'_ij ∈ KB then
4:       count_j = count_j + 1
5:    else
6:       Add S'_ij to KB
7:       count_j = 1
8:    end if
9: end for
10: for j = 1 to j = n do
11:    for z = 1 to z = |S_ij| (S_ij ∈ KB) do
12:       weight_jz = count_jz / |KB| (Equation 2)
13:    end for
14: end for
```

Algorithm 4 KB pruning ($prune(S'_i)$)

```
1: for j = 1 to j = n do
2:    for z = 1 to j = |S_j| (S_j ∈ KB) do
3:       if weight_jz < γ then
4:          remove S_jz from KB
5:       end if
6:    end for
7: end for
8: if KB has changed then
9:    Recalculate weightings as per lines 10 to1 4 in Algorithm 3
10: end if
```

Returning to Algorithm 3 the input is a set of current time series subsequences S'_i, one subsequence per attribute in the system, associated with instrument $inst_i$. The set $S'_i = \{S'_{i1}, S'_{i2}, \ldots, S'_{in}\}$ is processed attribute by attribute. If S'_{ij} is already in KB its count is updated (line 4), otherwise S'_{ij} is added to KB (line 6) and its associated count set to 1 (line 7). Once the process of incorporating S'_i into KB is complete the weightings are (re)calculated for the entire contents of KB.

Algorithm 4 presents the KB pruning process whereby subsequences with associated weightings of less than γ are pruned from the KB on the grounds that they have only been infrequently associated with instrument failure and are therefore not good predictors of such failure. If such sequences were retained in KB this would cause erroneous instrument failure prediction causing the instrument to be taken off-line unnecessarily. Selection of the most appropriate value for γ is thus important and is discussed further in Section 4.

4 Evaluation

The proposed instrument failure prediction mechanism, as described in Section 3 above, was evaluated using a simulated environment comprising a collection of

virtual instruments. The nature of this environment is presented in Sub-section 4.1. The metric used to measure performance is introduced in Sub-section 4.2. Evaluation was then conducted by considering two categories of scenario. The first was a simplistic scenario that considered instruments that featured only one sentinel attribute. The second considered instruments that featured a collection of attributes one of which was the sentinel attribute. The results obtained are presented and discussed in Sub-sections 4.3 and 4.4 respectively.

4.1 Simulation Environment

The simulation environment used for evaluation purposes assumed k instruments that featured n attributes each. The simulation operated on a loop. On each iteration a proportion of the instruments performed some sample measuring activity Whether an instrument is active or not is decided in a probability driven random manner (a probability value of $p_{sample} = 0.4$ was used). On each iteration the attribute values associated with each instrument were updated. To this end two different types of attribute were utilised: (i) activity dependent attributes and (ii) activity independent attributes. The values associated with the first were incremented/decremented on each simulation iteration whenever an instrument was active. Activity independent attributes were incremented/decremented on each iteration regardless of whether an instrument was active or not. Activity dependent attributes were designated to cause instrument failure once a particular sentinel value was reached, we refer to such attributes as *sentinel attributes*. In any given simulation the number of designated attributes that can potentially cause failure could be between 1 and n (thus between one and all); however, with respect to the evaluation presented here, scenarios featuring only one activity dependent attribute were used, this was therefore designated as the sentinel attribute. Attributes not so designated simply acted as providers of "noise".

The simulation operated using a Multi-Agent Based Simulation (MABS) environment such that each agent (instrument) communicated with a central controller agent, the Instrument Failure Prediction Engine (IFPE), which would in turn form part of a Laboratory Instrument Management System (LIMS)[1]. The IFPE monitors the incoming instrument data and, using the algorithms presented in Section 3, predicts instrument failure. The interface between an instrument and the IFPE was enabled using a bespoke software interface called a *Dendrite*[2]. In the simulation, whenever instrument failure is detected, the affected instrument goes into a *maintenance state* for a simulation time t_{maint} after which it comes back "on stream" but with its attribute value set reset to the start values. Should the IFPE fail to detect an instrument failure in time the instrument will fail and, within the realm of the simulation, require replacement. The replacement time is given by a simulation time of $t_{replace}$ ($t_{replace} > t_{maint}$). For the evaluation reported here $t_{replace} = 20$ and $t_{maint} = 10$ was used.

[1] A LIMS is a software system designed to manage laboratory operations. LIMS are used throughout the laboratory analysis industry.

[2] The Dendrites software is available from CSols Ltd, `http://www.csols.com`.

4.2 Evaluation Metrics

Prediction/classification systems are typically evaluated using metrics such as precision and recall, true positive and false positive rate [11]. However in our case, because we wish to intervene prior to failure we have no information as to whether our predictions were correct or not. Instead we use an accumulated "gross profit" (GP) measure. On each iteration of the simulation, whenever an instrument conducts some sampling activity, a profit of g_{sample} is gained. Instrument maintenance incurs a cost of g_{maint} and instrument replacement a cost of $g_{replace}$ such that $g_{sample} < g_{maint} < g_{replace}$ (off course when an instrument is undergoing maintenance or is being replaced there is an additional cost because the instrument is also not earning anything).

p	Similarity threshold σ									
	0	1	2	3	4	5	6	7	8	9
2	**711**	706	687	657	627	593	561	528	504	480
3	691	**696**	692	683	666	650	623	601	583	557
4	650	683	**687**	677	672	663	654	639	622	605
5	575	657	**677**	675	667	660	654	645	640	627
6	470	615	659	**668**	667	661	650	644	636	633
7	351	558	634	657	**660**	658	651	644	636	632
8	248	479	595	635	650	**655**	653	646	639	632
9	167	389	540	606	635	646	**648**	645	641	634

Table 1. Comparison in terms of gross profit ($k = 20$).

4.3 Single Sentinel Attribute Evaluation

The aim of the single attribute evaluation was to determine the effect of different σ (similarity threshold), p (subsequence length) and k (number of instruments) settings on the GP measure and to provide some insight into the failure prediction mechanism. This was done by running the simulation using a range of σ values ($[0, 9]$) against a range of p values ($[2, 10]$) and using $k = 20$ and $k = 40$. Recall that the minimum size of a subsequence is 2 (otherwise it will not be a sequence). For each parameter combination 1000 simulation runs were conducted, with 200 iterations for each run, and average values recorded for gross profit, number of instrument failures, number of maintained instruments and the size of KB. Throughout a single incremental, activity dependent, attribute was used with the sentinel value set to 20.

Tables 1 and 2 show the GP values obtained (best results indicated in bold font) when $k = 20$ and $k = 40$. From the tables it can be seen that there is a clear correlation between σ and p in terms of GP, for best results larger p values require a larger associate σ value. In both cases ($k = 20$ and $k = 40$) the best result, highest GP, was obtained using a combination of $\sigma = 0$ (exact matching) and $p = 2$ (GP of 711 and 1445 respectively). The lowest GP was generated using $\sigma = 9$ and $p = 9$ because prediction was at its least precise using these parameters and consequently maintenance was frequently conducted prior to this being a necessity. For reference the average GP value over 1000

p	Similarity threshold σ									
	0	1	2	3	4	5	6	7	8	9
2	**1445**	1425	1381	1319	1260	1186	1126	1064	1014	967
3	**1425**	1409	1396	1374	1338	1300	1256	1210	1167	1120
4	1381	**1400**	1395	1369	1352	1333	1312	1277	1245	1211
5	1299	1374	**1387**	1368	1343	1330	1310	1299	1285	1259
6	1156	1326	**1369**	1364	1348	1331	1306	1291	1277	1268
7	949	1250	1338	**1353**	1347	1336	1316	1291	1276	1262
8	712	1136	1292	1332	**1340**	1334	1321	1303	1284	1265
9	501	986	1223	1299	1325	**1328**	1320	1308	1291	1275

Table 2. Comparison in terms of gross profit ($k = 40$).

repeated simulations and 200 iterations per simulation, when failure prediction and maintenance was not conducted, using $k = 20$ was 44, decreasing steadily as the simulation continued (88 when $k = 40$).

p	Similarity threshold σ									
	0	1	2	3	4	5	6	7	8	9
2	**2**	1	1	1	1	1	1	1	1	1
3	**3**	2	1	1	1	1	1	1	1	1
4	**7**	3	2	2	1	1	1	1	1	1
5	**12**	5	3	2	2	2	1	1	1	1
6	**20**	9	5	3	2	2	2	2	1	1
7	**29**	13	7	4	3	2	2	2	2	2
8	**37**	19	10	6	4	3	3	2	2	2
9	**44**	26	14	9	6	4	4	3	2	2

Table 3. Comparison in terms of number of failed machines ($k = 20$).

p	Similarity threshold σ									
	0	1	2	3	4	5	6	7	8	9
2	**2**	1	1	1	1	1	1	1	1	1
3	**4**	2	1	1	1	1	1	1	1	1
4	**7**	3	2	2	1	1	1	1	1	1
5	**13**	6	3	2	2	2	1	1	1	1
6	**24**	10	5	3	2	2	2	2	1	1
7	**39**	16	8	5	3	3	2	2	2	2
8	**57**	25	12	7	5	4	3	2	2	2
9	**74**	36	18	10	7	5	4	3	3	2

Table 4. Comparison in terms of number of failed machines ($k = 40$).

p	Similarity threshold σ									
	0	1	2	3	4	5	6	7	8	9
2	60	62	63	66	68	70	73	75	76	**78**
3	58	61	62	64	65	66	68	70	71	**73**
4	55	60	62	63	64	65	66	67	68	**69**
5	48	57	60	62	63	64	65	66	**67**	**67**
6	40	53	58	61	63	64	65	66	**67**	**67**
7	30	47	55	59	61	63	64	65	66	**67**
8	21	41	51	56	59	61	63	64	65	**66**
9	13	33	46	53	57	60	61	63	64	**65**

Table 5. Comparison in terms of number of maintained machines ($k = 20$).

p	Similarity threshold σ									
	0	1	2	3	4	5	6	7	8	9
2	122	125	129	133	138	143	147	151	155	**158**
3	120	125	127	129	132	135	138	141	144	**148**
4	116	123	126	129	131	132	134	136	139	**141**
5	109	119	124	127	130	131	133	134	136	**138**
6	97	115	121	125	128	131	133	135	136	**137**
7	80	107	118	123	126	129	131	133	135	**137**
8	60	97	112	119	124	127	130	132	134	**136**
9	42	84	106	115	121	125	128	130	132	**134**

Table 6. Comparison in terms of number of maintained machines ($k = 40$).

Tables 3 and 4 show comparisons of the average number of instruments that fail during the simulation runs. The number of instruments that fail decreased as the value for the σ threshold increased. This was because instruments were going into maintenance well before they would actually fail. Using $\sigma = 9$ similarity between a time series sequence in the *KB* and that currently associated with an instrument is measured in a fairly course manner hence more instruments go into maintenance. This is illustrated in Tables 5 and 6 which show the number

of instruments maintained. From these two tables it can be seen that as the similarity threshold σ increased, the number of machines maintained increased also, because the prediction becomes coarser. It can also be noted that the number of machines maintained increased as p decreased. Comparing the statistics for the number of instruments maintained given in the tables with those for the number of failed instruments, these are roughly inversely proportional. For completeness, when running the simulation without failure prediction (over 1000 simulation runs with 200 iterations per simulation) the average number of failed machines when $k = 20$ is 54, and when $k = 40$ it is 109.

p	Similarity threshold σ								
	0	1	2	3	4	5	6	7	8	9
2	**2**	1	1	1	1	1	1	1	1
3	**3**	2	1	1	1	1	1	1	1
4	**6**	3	2	2	1	1	1	1	1
5	**12**	5	3	2	2	2	1	1	1
6	**20**	9	4	3	2	2	2	2	1	1
7	**29**	13	7	4	3	2	2	2	2	2
8	**37**	19	10	6	4	3	3	2	2	2
9	**44**	26	14	9	6	4	4	3	2	2

Table 7. Comparison in terms of *KB* size ($k = 20$).

p	Similarity threshold σ								
	0	1	2	3	4	5	6	7	8	9
2	**2**	1	1	1	1	1	1	1	1	1
3	**3**	2	1	1	1	1	1	1	1	1
4	**6**	3	2	2	1	1	1	1	1	1
5	**13**	6	3	2	2	2	1	1	1	1
6	**23**	10	5	3	2	2	2	2	1	1
7	**39**	16	8	5	3	3	2	2	2	2
8	**57**	25	12	7	5	4	3	2	2	2
9	**74**	36	17	11	7	5	4	3	3	2

Table 8. Comparison in terms of *KB* size ($k = 40$).

# Atts. (n)	σ	p	ω	λ	# Fail. Inst.	# Main. Inst.	Final *KB* Size	# *KB* Values Pruned	GP
2	1	2	0.250	22	23	319	1	28	2884
3	1	2	0.225	25	26	323	1	60	2727
4	1	2	0.250	27	28	320	1	93	2701
5	1	2	0.175	31	32	325	1	137	2527
6	1	2	0.150	29	30	342	1	160	2333
7	1	2	0.125	30	31	352	1	194	2176
8	1	2	0.125	35	36	332	1	261	2308
9	1	2	0.100	36	37	352	1	300	2009
10	2	2	0.100	33	34	394	1	313	1502

Table 9. Best parameter settings for a range of attribute set sizes, and $k = 20$. Average results obtained from 500 simulation runs per parameter permutation, 1000 iterations per simulation

Tables 7 and 8 show the recorded sizes, in terms of number of time series subsequences, of *KB* for $k = 20$ and $k = 40$. As anticipated the number of subsequences in *KB* decreases as the σ value increases. As already noted, this is because as σ is increased the prediction becomes less precise so the *KB* requires fewer subsequences. The number of subsequences in the *KB* also decreases with p; this is because as the p value is reduced the number of possible value combinations making up a time series subsequence also decreases (there are therefore fewer possible subsequences that can be included in the *KB* when p is small).

4.4 Multiple Attribute Evaluation with a Single Sentinel Attribute

To evaluate the proposed instrument failure prediction mechanism in the context of multiple attributes a single sentinel attribute was used, selected at random from the set A. In the simulation the value for the remaining attributes were modelled using a sin curve with a fixed amplitude of 20 and a multiple of the attribute number as the frequency. For the "key attribute" (the attribute designated as the cause of failure) a sentinel value of 20 was again used. Experiments were conducted to investigate the effect of using a range of values for ω and λ. Recall that ω is the weighting threshold used to prune unwanted TSS values from KB; while λ is the learning window size, in other words the number of instruments that the mechanism allows to fail prior to commencing failure prediction.

ω	Learning window size (λ)							
	24	26	28	30	32	34	36	38
0.050	-14543	-14492	-13685	-13652	-13360	-13448	**-12686**	-12774
0.075	-6801	-6072	-5522	-5010	-5033	-4498	**-4460**	-4518
0.100	-1131	-1056	-529	-637	-173	-12	**22**	12
0.125	890	**1540**	1174	1377	1359	1306	1373	1245
0.150	1913	1788	2049	2097	2194	2061	**2245**	2125
0.175	2251	2291	2307	2355	**2401**	2377	2393	2285
0.200	2276	2406	2364	2292	2454	2428	**2481**	2448
0.225	-138	-140	-139	-138	-139	**-137**	-139	-138
0.250	-139	**-137**	-138	-137	-138	-138	-139	-139

Table 10. Learning window size (λ) versus Weighting threshold (ω), comparison in terms of gross profit ($k = 20$, $n = 5$, $\sigma = 1$ and $p = 2$)

Table 9 shows the mean best parameter settings for a range of numbers of attributes ($n = [1, 10]$) that serve to maximise the GP value. For each parameter combination (σ, p, ω and λ) the simulation was run 1000 times for 1000 iterations and mean values recorded. In addition to the best parameter settings, the number of failed and maintained instruments, the size of KB at the end of each run and the number of pruned KB values was also recorded as well as the final GP value. Without prediction and consequent maintenance the average number of failed machines was 281, and the average GP was -140. Note that as n was increased the number of noise attributes increased and it became harder to predict instrument failure hence the value of λ increases with n. Consequently erroneous instrument failure prediction causes larger numbers of instruments to, perhaps unnecessarily, go into maintenance. The increase in the number of instruments going into maintenance as n increases is reflected by the decreasing recorded GP values. The prediction becomes harder because the number of potential KB values increases, hence the amount of KB pruning that is undertaken also increases with n. From the table it is interesting to note that $\sigma = 1$ and $p = 2$ consistently produce the best result. It is also interesting to note that $|KB| = 1$ is consistently recorded; this is because when using $p = 2$ there are only two possible time series subsequences that can be included in KB and if $\sigma = 1$ is used one of these will then be superfluous.

Tables 10 and 11 shows the effect on GP using a range of values for ω ({0.050, 0.075, 0.100, 0.125, 0.150, 0.175, 0.200, 0.225, 0.250}) and a range of values for λ ({24, 26, 28, 30, 32, 34, 36, 38}) when $n = 5$ and $n = 10$ respectively. For the experiment $\sigma = 1$ and $p = 2$ was used because, as demonstrated above, these settings produced good results. For each parameter combination the simulation was again run 1000 times for 1000 iterations and the average GP values recorded. From the tables it can be seen that the choice of the most appropriate value for ω is important, either side of the optimum value, GP quickly starts to fall. This is particularly so with larger values of n as can be seen by comparing the two tables.

ω	Learning window size (λ)							
	24	26	28	30	32	34	36	38
0.050	-11099	-10851	-10638	-10668	-11150	-10689	**-10414**	-10499
0.075	-1838	-1735	-1058	-995	-888	-1411	**-786**	-932
0.100	1005	1191	1057	1098	1011	**1561**	1338	1314
0.125	**-137**	-139	-138	-138	-139	-139	-138	-139
0.150	-138	-139	-138	-138	**-137**	-138	-138	-139
0.175	**-138**	-139	-139	-139	-140	-138	-138	-138
0.200	-140	-140	-140	**-138**	-139	-139	-138	-139
0.225	**-138**	-139	-139	-140	-138	-140	-138	-140
0.250	-139	-138	-139	**-137**	-140	-140	-140	-139

Table 11. Learning window size (λ) versus Weighting threshold (ω), comparison in terms of gross profit ($k = 20$, $n = 10$, $\sigma = 1$ and $p = 2$)

5 Summary and Conclusions

A mechanism, founded on time series analysis, for predicting instrument failure using data stream mining has been proposed. The mechanism has been fully described. The novel feature of the mechanism is that, unlike in the case of more standard data stream mining prediction applications, there is only limited opportunity for validation. Evaluation was conducted using a simulated multi-agent based environment comprising k virtual machines communicating, via a bespoke interface called a Dendrite, with a central Instrument Failure Prediction Engine (IFPE). The mechanism uses four parameters: (i) a time series subsequence similarity threshold σ, (ii) a time series subsequence length value p, (iii) a weighting threshold ω used when pruning KB and (iv) a learning window measured in terms of the number of instruments allowed to fail λ. The presented evaluation indicated that best results are obtained when $\sigma = 1$ (almost exact matching between current time series subsequences associated with individual instruments and subsequences in KB) and $p = 2$ (the size of a subsequence). The optimum value for λ increases with n. Finding the optimum value for ω is challenging. The sensitivity associated with the ω parameter is thus a subject for future work. The intention is also to investigate the operation of scenarios where we have several sentinel attributes and alternative prediction mechanisms using (for example) dynamic classification association rule or decision tree based techniques. The advantage of these last two techniques is that they take into account

the negatives as well as the positives; the mechanism that has been presented in this paper, although operating well, only predicts failure, not non-failure. Incorporating both might provide for a better predictor.

References

1. Bagnall, A. and Lines, J. (2014). An Experimental Evaluation of Nearest Neighbour Time Series Classification. Technical report 1406.4757, Cornell University Library.
2. Batista, G., Wang, X and Keogh, E (2011). A complexity-Invarient Distance Measure for Time Series. Proc 2011 SIAM International Conference on Data Mining, pp699-710.
3. Carmona-Cejudo, J.M., Manuel Baena-Garcá, N. del Campo-Ávila, J., Morales-Bueno, R., Gama, J. and Bifet, A. (2011). Using GNUsmail to Compare Data Stream Mining Methods for On-line Email Classification. Proc. 2nd Workshop on Applications of Pattern Analysis, JMLR (Journal of Machine Learning Research) Workshop and Conference Proceedings, Vol. 17, pp1218.
4. Cohen, I., Goldszmidt, M., Kelly, T., Symons, J. and Chase, J.S. (2004). Correlating instrumentation data to system states: A building block for automated diagnosis and control. Proc 6th Symposium on Operating Systems Design and Implementation (OSDI'04), pp231-244.
5. Cohen, L., Avrahami-Bakish, G., Last, M., Kandel, A., and Kipersztok, O. (2008) Real time data mining-based intrusion detection. Information Fusion (Special Issue on Distributed Sensor Networks), 9(**3**), pp344-354.
6. Ding, H., Trajcevski, G., Scheuermann, P., Wang, X. and Keogh,E. (2008). Querying and Mining of Time Series Daya: Experimental Comparison of Representations and Distance Measures. Proc. VLDB'08, pp1542-1552.
7. Gaber, M.M., Zaslavsky, A. and Krishnaswamy S. (2005) Mining Data Streams: A Review. ACM SIGMOD Record, 34(**2**), pp18 - 26.
8. Gaber, M.M., Gama, J., Krishnaswamy, S., Gomes, J.B. and Stahl, F. (2014). Article: Data stream mining in ubiquitous environments: state-of-the-art and current directions Wiley Interdisciplinary Reviews: Data Mining and Knowledge Discovery; 4(**2**), pp116-138.
9. Gama, J (2010). Knowledge Discovery from Data Streams. Chapman and Hall.
10. Gama, J., Zliobaite, I., Bifet, A., Pechenizkiy, M. and Bouchachia, A (2014). A survey on concept-drift adaptation. ACM Computing Surveys, 46(**4**), 2014.
11. Hand, D.J. (2009). Measuring classifier performance: a coherent alternative to the area under the ROC curve. Machine Learning, 77(**1**), pp103123.
12. Kargupta, H., Bhargava, R., Lou, K., Powers, M., Blair, P., Bushra, S. and Dull, J. (2004). VEDAS: A Mobile and Distributed Data Stream Mining System for Real-Time Vehicle Monitoring Proc. 2004 SIAM International Conference on Data Mining, pp300-311.
13. Sakoe, H. and Chiba, S. (1978). Dynamic programming algorithm optimization for spoken word recognition. IEEE Transactions on Acoustics, Speech and Signal Processing, 26(**1**) pp43-49.
14. Stein, S., Payne, T.R. and Jennings, N.R. (2008). Flexible QoS-Based Service Selection and Provisioning in Large-Scale Grids. UK e-Science 2008 All Hands Meeting (AHM), HPC Grids of Continental Scope.
15. Vintsyuk, T.K. (1968). Speech discrimination by dynamic programming. Kibernetika, Vol. 4, pp8188.