Novelty Detection in Time Series Data using Ideas from Immunology

Dipankar Dasgupta
Dept. of Maths & Computer Science
University of Missouri-St. Louis
St. Louis, MO 63121

Stephanie Forrest
Dept. of Computer Science
University of New Mexico
Albuquerque, NM 87131

Abstract

Detecting anomalies in time series data is a problem of great practical interest in many manufacturing and signal processing applications. This paper presents a novelty detection algorithm inspired by the negative-selection mechanism of the immune system, which discriminates between *self* and *other*. Here self is defined to be *normal data patterns* and non-self is any deviation exceeding an allowable variation. Experiments with this novelty detection algorithm are reported for two data sets - simulated cutting dynamics in a milling operation and a synthetic signal. The results of the experiments exhibiting the performance of the algorithm in detecting novel patterns are reported.

1 Introduction

The normal behavior of a system is often characterized by a series of observations over time. The problem of detecting novelties or anomalies can be viewed as finding deviations of a characteristic property in the system of interest. Novelty detection is an important task in many diagnostic and monitoring systems. In safety-critical applications, it is essential to detect the occurrence of abnormal events as quickly as possible before significant performance degradation results. This can be achieved by continuous monitoring of the system for deviations from the normal behavior patterns. For example, drilling and high speed milling processes require continuous monitoring to assure quality production; machines such as jet engines require continuous monitoring to assure safe operation.

There have been a number of techniques suggested in the literature for detecting novelties, anomalies and faults in monitored systems. These include control charts, model-based methods, knowledge-based expert systems, pattern recognition and cluster analysis, hidden markov models, and neural networks. Most existing methods require either prior knowledge about various novelty conditions [11] or precise theoretical models [10] of the monitored system. A robust method, however, should detect any unacceptable (unseen) change rather than looking for specific (known) abnormal activity patterns. Recently, neural network-based methods have been used for novelty detection including Multi-Layer Perceptrons (MLP)[1, 13], which require prior knowledge of different novelty classes, and Adaptive Resonance Theory (ART)[2, 9], which do not.

This paper proposes a novelty detection method, which is based on ideas from the immune system. It is a probabilistic method that notices changes in normal behavior without requiring prior knowledge of the changes for which it is looking. In this way it resembles the approach to novelty detection taken by ART. Both neural networks and our immune system algorithm are biologically inspired techniques that have the capability of identifying patterns of interest. However, they use different mechanisms for recognition and learning.

2 Negative-Selection Algorithm

Our approach is inspired by the information-processing properties of natural immune systems [4, 7]. Natural immune systems are capable of distinguishing virtually any foreign cell or molecule from the body's own cells: this is known as self-nonself discrimination. This discrimination is achieved in part by T cells, which have receptors on their surface that can detect foreign proteins (antigens). T-cell receptors are made by a pseudo-random genetic rearrangement process, making it likely that some receptors will bind to self. Such self-reactive T-cells are censored in the thymus, with the result that only those cells that fail to bind to self-proteins are allowed to leave the thymus and become part of the body's immune system. This process is called negative selection.

We have defined an algorithm based on the principle of negative-selection algorithm [7]. Its basic steps are as follows:

- Define *self* as a multiset S of strings of length l over a finite alphabet that we wish to protect or monitor. For example, S may be a segmented file or a normal pattern of activity of some system/process.
- Generate a set R of *detectors*, such that each detector fails to match the strings in S. We use a partial matching rule, in which two strings match if and only if they are identical at at least r contiguous positions, where r is a suitably chosen parameter.
- Monitor S for changes by continually matching the detectors against S. If any detector ever matches, a change (or deviation) must have occurred.

In the original description of the algorithm [7], candidate detectors are generated *randomly* and then eliminated if they match self. In this paper, we generate detector sets using a more efficient algorithm [4] which runs in linear time with respect to the size of self.

3 Novelty Detection

We reduce the novelty detection problem to the problem of detecting whether or not a string has changed, where a change (or match) implies a shift in the normal behavior pattern.

We first map real-valued data into a discrete form. An analog value is normalized with respect to a defined range and discretized into bins (or intervals). Each datum is assigned the integer corresponding to the bin in which it falls. The integer is then encoded in binary form. However, if an observed value falls outside the specified range, it is mapped to all 0's or all 1's depending on which side of the range it crossed. The size of the bins is thus determined by the number of bits used in the discretization. If each datum is encoded by m bits (which may be chosen according to the desired precision), then there would be $2^m - 2$ different bins between the maximum (MAX) and minimum (MIN) ranges of data.

3.1 Generation of Detectors

In our implementation, data are sampled from a moving time window and mapped to binary. Each window, therefore, is the concatenation of a fixed number (called Win_size) of data points. We collect the bit strings from a succession of windows, sliding along the time series in discrete steps (Win_shift) until the collection is sufficient to capture regularities of the normal system behavior.[1] As long as the time series data pattern maintains coherent behavior, these collected strings are sufficient to define normal behavior of the system. This collection of strings for windows is our self (S). We then generate strings that do not match any of the strings in S to be members of the detector set using the negative-selection algorithm discussed above. Our current detector generation was designed to be used off-line, although on-line methods may be possible. This is similar to the case of neural network training for novelty detection [1] or developing rules for anomaly-detecting expert-systems [8]. Our approach can be summarized as follows:

1. Collect time series (sensor) data that sufficiently exhibit the normal behavior of a system (these may be raw data at each time step, or average values over a longer time interval).
2. Examine the data series to determine the range of variation (MAX, MIN values) of data and choose the data encoding parameter (m) according to the desired precision.
3. Encode each value in binary form.
4. Select a suitable window size which captures regularities of interest.
5. Slide the window along the time series (in nonoverlapping steps) and store the encoded string for each window as *self*, from which detectors will be generated.
6. Generate a set of detectors that do not match any of the self strings.
7. Once a unique set of detectors is generated from the normal database of patterns, it can probabilistically detect any change (or abnormality) in patterns of unseen time series data.
8. When monitoring the system, we use the preprocessing parameters in step 3 to encode new data patterns. If a detector is ever activated (matched with current pattern), a change in behavior pattern is known to have occurred and an alarm signal is generated regarding the abnormality. We

[1] When Win_size = Win_shift the windows are nonoverlapping. The results reported in this paper used nonoverlapping windows.

use the same matching rule (for monitoring the system) as was used in generating detectors.

4 Example Applications

We have tested the feasibility of this novelty detection algorithm on a number of data sets, including the Mackey Glass series [2], simulated cutting tool dynamics in a milling process, and some sensory data. In this paper, we report data for the two of these examples (we achieved similar results on the other data set).

In milling industries, on-line monitoring of tool conditions is very important to achieve automated machining operation. To prevent possible damage to the workpiece and the machine tool (or to avoid production of defective parts and possible overloading of tools), a reliable and effective tool breakage detection technique is required for providing a rapid response to an unexpected tool failure. Usually, methods for monitoring a milling process use measurements of cutting parameters correlated with tool breakage [6]. These cutting parameters include temperature, cutting force, torque, vibration, acoustic emission, motor current, etc. Of these parameters, cutting forces are widely used for tool breakage detection for several reasons: (1) cutting force signals are much less dependent on the structure of the workpiece; (2) cutting force signals can be simulated easily and more accurately than acceleration and acoustic emission signal; (3) the cutting force is a very good indicator of the vibration between the tool and workpiece because of its higher sensitivity and more rapid response to the changes in cutting state.

The cutting force variation characteristics of normal and broken tools are different. Under normal (stable) cutting conditions, the cutting force periodically varies with the tooth frequency which depending on the spindle speed. If the tool is broken, the force changes as long as the broken tooth stays in the workpiece, since it can not remove the same amount of material as the other teeth. The number of tooth periods deviates from the stable cutting pattern depends on the number of teeth that are actively involved in the cutting zone.

We prepared simulated data for cutting operations using the vibratory model described in [5, 17]. This model has been used by many other investigators for tool breakage detection [14, 16]. In our experiments, a four-tooth cutter with uniform pitch, performing an end-milling is considered and one tooth is engaged in the cut at an angle ϕ, where the cutting angle varies from 0 to $\pi/2$ for every tooth engagement. The cutting force profiles were simulated using fourth order Runge-Kutta method for every time step ($dt = 0.0001$ sec), where displacements at step, $t+1$ are calculated from the cutting force data at step t. References [5, 17] give details of the vibratory model and calculation of the cutting force and vibration. The parameter values used in our simulation are as follows: [5, 14]:

Mass, $m_x = m_y = 10$ kg;
Damping coeff., $c_x = c_y = 471.9$ kg/s;
Spring constant, $k_x = k_y = 8.1 * 10^6$ N/m;
Feed rate/tooth, $f_t = 0.2$ mm;
Cutting coefficient, $K_c = 6.67 * 10^6$ N/m;
Depth of cut, $b = 0.508$ mm;
Spindle speed, $N_s = 600$ rpm;
Spindle diameter, $D = 40$ mm.

4.1 Experimental Results

In the first set of experiments, we simulated instantaneous cutting force patterns with and without tool breakage, shown in Figure 1. In this simulation, the tool was in normal (stable) cutting operation for 1500 time steps and then one tooth was broken, causing changes in cutting force signals at the corresponding tooth periods. In our experiments, we used the first 1000 data points as the self set, S, for generating detectors and the rest of the data series were used for testing. Results of the experiments are shown in table 1 and in figure 2. Table 1 shows the various parameters used for preprocessing data and for generating detectors. We tried several different parameter values and found the reported values most suitable (see [3] for details). In these experiments, we set $m = 6$

Encoding parameters	Matching threshold	No. of detectors	Tool Breakage Detection	
			Mean(Std. dev)	Detection
Win_size = 5 Win_shift= 5 l = 30, S = 200	10	40	12.20(5.52)	50.83%
	9	30	16.16(4.78)	67.33%
	8	20	21.88(2.53)	81.16%
Win_size = 7 Win_shift= 7 l = 42, S = 142	12	40	10.36(3.36)	62.78%
	10	30	20.38(5.57)	75.56%
	9	20	30.75(7.91)	93.28%

Table 1: Novelty detection results on tool breakage problem. Results are averaged over 50 runs. Column 4 shows the mean number of detections (number of times detectors activated). The standard deviations are shown in parentheses. The average detection rate is shown in column 5. This is the ratio of the average detection to the number of actual novel patterns in the data.

for binary encoding of data and two different window sizes are considered. Detection results (columns 4 and

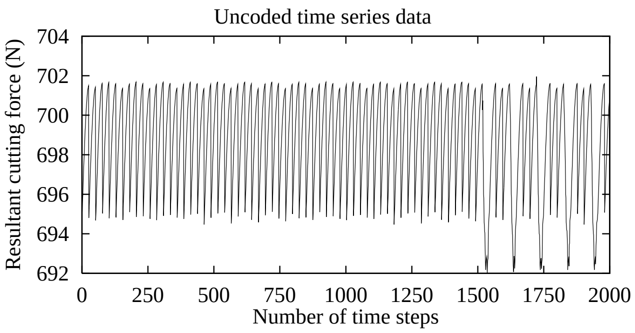

Figure 1: Simulated cutting force signals of normal behavior and with tool breakage in a milling operation. Here one tooth of the cutter is broken after 1500 time steps.

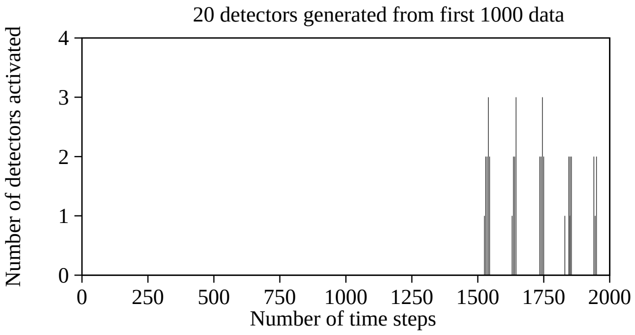

Figure 2: The height of vertical lines in the graph corresponds to the number of detectors activated when novel patterns are found.

5) show the mean number of times detectors were activated and the average detection rate in each case.

In all the test runs, the generated detectors could detect the tooth periods in which the changes in the force pattern occurred. Figure 2 shows a typical run and the number of activated detectors (novel patterns encountered) at different time steps. In this example, a maximum of three detectors is activated (out of 20) when there are significant changes. Note that the detectors remain inactive during the normal operation period, in particular, between 1000 and 1500 time steps where the data exhibit a normal pattern, thus avoiding false positives.

We next considered the second data series - a typical example found in many signal processing applications (figure 3), where signals of varying amplitude are observed under normal behavior. The data patterns, however, show regularities over a period of time. Detecting unknown changes (noise) in this signal pattern is a very difficult task, although monitoring such

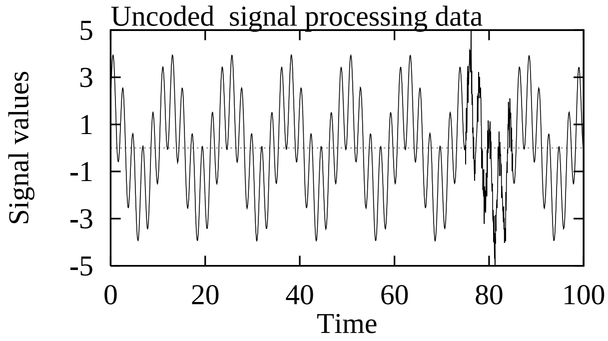

Figure 3: An example of signal processing data series. Here the signals are noisy during the time period 75 and 85.

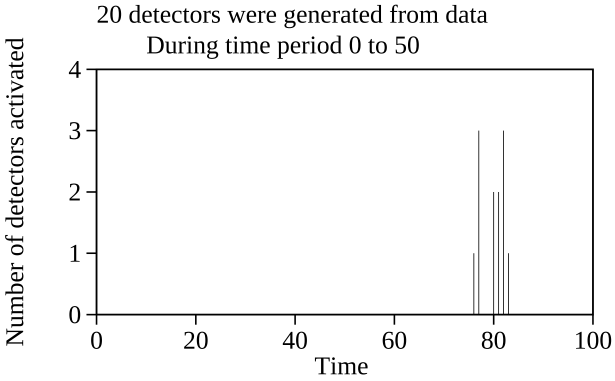

Figure 4: The vertical lines indicate the activation of detectors indicating the changes in signal pattern. The detectors are only activated during the time period when signals get distorted as in figure 3.

changes is essential in some applications. However, most existing threshold-based methods fail to detect small changes because of varying boundary conditions. We conducted experiments with similar parameter settings as in the previous example. Figure 4 shows a typical result demonstrating that the proposed algorithm can easily detect noisy signals by monitoring with a small set of detectors. In particular, we generated 20 detectors from initial data (data during 0 to 50 time steps are assumed as the normal pattern of signals). The detector set was then used to monitor future signal patterns (test signals), and it could detect changes during the time period 75 and 85. This suggests that the detection of gradual change can be monitored with a suitable detector set.

5 Observations

We observed that the performance of the algorithm varies with the choice of the matching threshold (r) for a defined string length (l) and encoding. With larger r, the generated detectors become sensitive to any novelty in the data patterns, so more detectors are necessary to achieve a desired level of overall reliability. On the other hand, if r is too small, it may not be possible to generate a reasonable size detector set from the available self, since there may not exist any unmatched strings (non-self) at that value of r. The choice of a suitable value is desirable for optimal performance of the algorithm.[2] This suggests that the value of r can be used to tune the reliability of detection against the risk of false positives.

Recently, neural network methods, in particular, ART networks have been used on the problem of detecting tool breakage in milling operations [12, 15]. Our results agree qualitatively with those of ART. However, there are some important differences between the two approaches:

- ART and other neural networks recognize/ classify input patterns in the space defined by the training data set (actual encoded space of data or signals), while our detection algorithm is based on negative selection, and recognizes patterns in the complement space as novel. Some circumstances will favor the positive selection approach while others will favor negative selection (See [4] for some explanation).

- Recognizing an input pattern is a global decision over the ART network, whereas the recognition of a novel pattern by detectors in our algorithm is a decentralized local decision. This property results in generating a quick response to any changes, and may be very useful in monitoring safety-critical systems.

6 Conclusion

In this paper we have proposed and demonstrated a method for novelty detection based on our earlier work in computer virus detection [7]. The objective of this work is to develop an efficient detection algorithm that can be used to alert an operator to any changes in steady-state characteristics of a monitored system. Our approach relies on a large enough sample of normal data to generate a diverse set of detectors that probabilistically notice any deviation from the normal. The detection system may be updated by generating a new set of detectors as the normal system behavior shifts due to aging, system modifications, change in operating environments, etc.

In this paper we demonstrated that the proposed algorithm successfully detects the tool breakage, and also can detect noise in signals. In these examples, the detection of a spurious change in data is not as important as the change in the data pattern over a period of time, and our probabilistic detection algorithm appears to be a feasible approach to such problems. There are a number of parameters which are tunable in both the preprocessing and the detector generation stage. In the preprocessing stage, the desired precision can be achieved by grouping similar analog data in the same bin, and the window size may be suitably chosen to capture the semantics of the data patterns. Note that the system can be monitored using different time-scales simultaneously. Also instead of directly encoding the time series data, it may be necessary to transform data (e.g. by Fourier transform) depending on the properties of sensor data. In cases of multivariate data series, the system can be monitored by a single set of detectors, constructing self strings by concatenation of patterns of data for each variable. A desired level of reliability can be achieved by changing the window size, matching threshold, and the number of detectors. Other encoding techniques, matching rules and generation algorithms are currently under investigation.

There exist many potential application areas in which this method may be useful. They include fault detection, anomaly detection, machine monitoring, signature verification, noise detection, patient's condition monitoring and so forth. The remarkable detection abilities of animal immune systems suggests negative-selection algorithms such as ours are well worth exploring.

Acknowledgments

The authors acknowledge many suggestions and comments from Patrik D'haeseleer, Ron Hightower and Derek Smith. We also express our appreciation to David Ackley, Tom Caudell, Bill Fulkerson and Roy Maxion for helpful critical comments. The second author received grants from National Science Foundation (grant IRI-9157644), Interval Research Corpora-

[2] In an ART network, a similar effect is also noticed when choosing vigilance threshold (ρ). A ρ value near zero gives a network with low discrimination, and a value near one gives a network with high discrimination (forming many clusters).

tion, General Electric Corporate Research and Development, and the Digital Equipment Corporation.

References

[1] C. M. Bishop. Novelty detection and neural network validation. *IEE Proceedings - Vision, Image and Signal processing*, 141(4):217–222, August 1994.

[2] Thomas P. Caudell and David S. Newman. An Adaptive Resonance Architecture to Define Normality and Detect Novelties in Time Series and Databases. In *IEEE World Congress on Neural Networks*, pages IV166–176, Portland, Oregon, 3-7 July 1993.

[3] Dipankar Dasgupta and Stephanie Forrest. Tool breakage detection in milling operations using a negative-selection algorithm. Technical Report Technical Report No. CS95-5, Department of Computer Science, University of New Mexico, 1995.

[4] P. D'haeseleer, S. Forrest, and P. Helman. An immunological approach to change detection: algorithms, analysis, and implications. In *Proceedings of IEEE Symposium on Research in Security and Privacy*, Oakland, CA, 7-9 May, 1996.

[5] M A Elbestawi, F Ismail, R Du, and B C Ullagaddi. Modelling machining dynamics including damping in the tool-workpiece interface. *Journal of Engineering for Industry*, 116:435–439, November 1994.

[6] M A Elbestawi, T A Papazafiriou, and R X Du. In-process monitoring of tool wear in milling using cutting force signature. *International Journal of Machine Tools & Manufacturing*, 31(1):55–73, 1991.

[7] S. Forrest, A. S. Perelson, L. Allen, and R. Cherukuri. Self-Nonself Discrimination in a Computer. In *Proceedings of IEEE Symposium on Research in Security and Privacy*, pages 202–212, Oakland, CA, 16-18 May 1994.

[8] Paul M. Frank. Fault Diagnosis in Dynamic Systems using Analytical and Knowledge-based Redundancy - A survey and some new results. *Automatica*, 26(3):459–474, 1990.

[9] H. B. Hwarng and C. W. Chong. A Fast-Learning identification system for SPC: An Adaptive Resonance Theory approach. *Intelligent Engineering Systems Through Artificial Neural Networks*, 4:1097–1102, 13-16 November 1994.

[10] Rolf Isermann. Process Fault Detection based on Modeling and Estimation Method - A survey. *Automatica*, 20:387–404, 1984.

[11] R. Kozma, M. Kitamura, M. Sakuma, and Y. Yokoyama. Anomaly Detection by neural network models and statistical time series analysis. In *Proceedings of IEEE International Conference on Neural Networks*, Orlando, Florida, June 27-29 1994.

[12] S Rangwala and D Dornfeld. Sensor integration using neural networks for intelligent tool condition monitoring. *Journal of Engineering for Industry*, 112:219–228, August 1990.

[13] S. Roberts and L. Tarassenko. A Probabilistic Resource Allocating Network for Novelty Detection. *Neural Computation*, 6:270–284, 1994.

[14] I N Tansel and C McLaughlin. Detection of tool breakage in milling operations-I. The time series analysis approach. *International Journal of Machine Tools & Manufacturing*, 33(4):531–544, 1993.

[15] I N Tansel and C McLaughlin. Detection of tool breakage in milling operations-II. The neural network approach. *International Journal of Machine Tools & Manufacturing*, 33(4):545–558, 1993.

[16] Y S Tarng and B Y Lee. Use of model-based cutting simulation system for tool breakage monitoring in milling. *International Journal of Machine Tools & Manufacturing*, 32(5):641–649, 1992.

[17] J Tlusty and F Ismail. Special aspects of chatter in milling. *Trans. ASME; Journal of Vibration, Acoustics, Stress, and Reliability in Design*, 105:24–31, January 1983.