Teaching Computers to Think Like Engineers

Embedding Artificial Intelligence into Diagnostic Software

Both manning and maintenance costs contribute significantly to the overall ship life cycle cost equation. Ship designs supporting minimum crew sizes and minimum maintenance requirements are consistent with a strategy that attempts to achieve reliability requirements while reducing Total Ownership Costs. To implement this strategy, advanced computer technology, such as Artificial Intelligence (AI), can be exploited for monitoring, control, and condition assessment of critical shipboard systems. Predictive maintenance (prognostic) systems, which can isolate imminent failures before they occur, are essential for the successful implementation of such well-proven cost-reduction strategies, such as Reliability-Centered Maintenance (RCM) and Condition-Based Maintenance (CBM).

Within this context, the marine industry is increasingly adopting CBM as cost-effective strategy, fostering the approach of performing maintenance only when objective evidence of need exists. Extending CBM into predictive analytics, prognostic systems that can identify the optimum time to perform preventive or corrective maintenance will save ship operators lots of money, while ensuring high equipment availability, optimized maintenance scheduling, and cost-effective maintenance resource allocation. However, because of the special skills and time required to implement CBM, it is desirable to design future ship systems such that artificial intelligence is imparted into the equipment itself, through a combination of hardware and software. The equipment should be capable of assessing its own health and alerting the crews when performance degradations or changes are detected or predicted. Some key attributes of future AI-based condition assessment systems include:

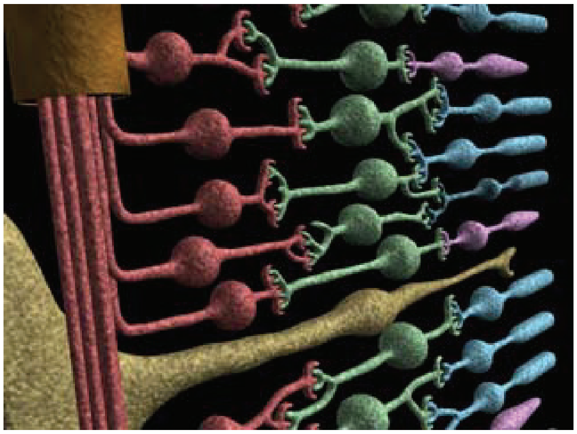
- Robust diagnostic and prognostic inferencing technology,
- Ability to function properly with missing, noisy, or corrupted measurement data,
- Ability to compute and assess uncertainty measures following valid statistical techniques,
- Ability to infer measurements that are either too costly or too difficult to acquire, and
- Ability to be rapidly deployed using existing experiential and empirical knowledge.

The robustness of the diagnostic/prognostic system is related to its ability to correctly detect specific faults, given process measurements and symptoms (alarm conditions), in much the same way a human engineer analyzes equipment health. Short of measuring every process variable, which is prohibitively expensive, the design goal of the AI system is to infer as much as possible from all available information.

Diagnostic robustness is directly related to the AI inferencing techniques used. Early expert systems employed rule-based or logic-based reasoning. Although relatively easy to build, the strict logic-based reasoning makes these types of systems vulnerable to bad input data, often the result of sensor problems. As a result of the "brittleness" of rule-based systems, they often tend to miss diagnostic calls when one of the rule antecedent conditions is false, as will occur when a sensor problem develops. *(See our other white papers discussing the prevalence of sensor problems on ships).*

AI Diagnostic Technology … that works like the human brain

Another type of AI reasoning technique particularly well suited to diagnostic applications is artificial neural networks. Neural networks are modeled after biological systems and are known to be good pattern recognition devices.



The human brain is the most powerful and complex computing device known to mankind. It is comprised of billions of nerve cells or neurons. Each neuron functions as a simple computer by processing its inputs and producing outputs. Each neuron sends its outputs to other neurons by way of nerve connections. There are billions of such connections, with a single neuron averaging ten thousand connections to other neurons in our brain. This collection of neurons and their nerve connections forms a complex network of signal pathways, called a neural network. Being one of nature's most amazing creations, the brain is able to perceive, remember, and think by virtue of patterns of electrical energy that propagate across our neural networks. This energy flow defines our intelligence.

Our computer scientists have developed software models of neural networks that can learn and perform brain-like functions. These models, often referred to as artificial neural networks, are able to learn from examples and are particularly useful for certain tasks, such as pattern recognition.

Artificial neural networks (ANNs) have several attractive features, including:

- Able to learn from training examples,
- Capable of real-time pattern recognition,
- Capable of classifying novel input patterns not included in training data, and
- Tolerant of noisy or incomplete input patterns.

ANNs have the ability to learn input/output associations for pattern recognition problems; for example, the associations of symptoms and faults that are typical of diagnostic and prognostic applications. Learning occurs in the network through numerical adjustment of internal ANN weight parameters. Given a particular set of training data, comprised of symptoms and related faults, the ANN learns to associate faults with given symptom patterns by adjusting its internal weights accordingly. Associative memory is encoded internally in the network structure (connectivity between neurons) by virtue of the network's weight parameters.

Because ANNs are tolerant of noisy or incomplete input patterns, they can be used to implement much more robust diagnostic/prognostic systems that those following a rule-based approach. Even if one or more symptom values are missing, perhaps due to a failed sensor, the network is still able to make the closest fault association based on the training examples that it has learned.

The critical aspect of deploying ANNs is having good training data. One strategy for deploying ANNs for diagnostic applications is to train the network from detailed measurement signatures captured in coincident with specific machinery failure events (e.g. vibration traces). If the network can learn multi-dimensional signatures of machinery behavior either leading up to or subsequent to a failure event, then the network can serve as a useful prognostic or diagnostic aid. It can issue user alerts when it senses similar signatures in the future. This approach represents an ideal scenario, but as a practical strategy for rapidly deploying diagnostic/prognostic systems, it has several shortcomings:

- Good maintenance practices tend to prevent failures from occurring. As such, actual failure data is extremely scarce and very expensive to collect and/or create,
- The fault coverage of actual failure data is typically very narrow and it may require many years of data collection to obtain an adequate data set for neural network training,
- Unless collected under controlled or known conditions, historical failure data may be incomplete or include unreliable measurement values, which, if used for network training, may adversely impact the network's fault classification performance, and
- Typical monitoring systems do not store data at adequate sampling rates to ensure that data are recorded during the actual failure event and that sufficient data are recorded to accurately classify the failure event.

It is certainly desirable to use detailed failure data for neural network training, however, as a practical matter, the data will be insufficient to provide coverage for all possible machinery faults which might occur. This fact severely limits the scope of diagnostic/prognostic coverage of a system developed solely from this strategy.

Practical Approach to Building Diagnostic Knowledgebases

A practical alternative to developing ANN training data is to rely on the experiential and engineering knowledge of domain experts to construct a diagnostic knowledgebase. This can be accomplished by conducting a comprehensive Failure Mode and Effects Analysis (FMEA) on the machinery plant equipment under diagnosis. In so doing, a more complete scope of coverage can be developed over a much shorter time interval, and typically at less cost than by following the aforementioned strategy of failure event signature analysis. The system can be fielded sooner, and through suitable feedback monitoring and maturation processes, its performance can be improved on the basis of new experiential information. In addition, the effect of any inaccuracies in the training knowledgebase can be attenuated by incorporating probabilistic techniques, such as the Probabilistic Neural Network (PNN) used in DEXTER.

Failure Mode and Effects Analysis (FMEA)

The objective of the FMEA is to precisely define the scope of diagnostic coverage by providing detailed definitions of individual machinery diagnostics. Each machinery system is broken down into its major components. As illustrated in figure 1, subject matter experts, including engineering crews, can be interviewed to obtain their insights into common machinery problems, maintenance practices, and potential problems based on established practice. Based on information gathered during this process, probable failure modes of the various machinery system components can be enumerated. The causes and effects (i.e. measureable symptoms with available sensors) are then traced out through the impacted systems, relying on piping and instrumentation drawings as a tool, as well as physically surveying the machinery spaces.

Component failure modes can be assessed independently of each other, regardless of the fact that one failure mode may produce exactly the same effects as several others. Measurable effects are constrained, as always, by available sensor instrumentation in the machinery plant. The results of the FMEA will define the diagnostic coverage the system and will:

- Enumerate the probable equipment failure modes,
- Identify salient sensor measurements, and
- Identify the fault/symptom relationships.

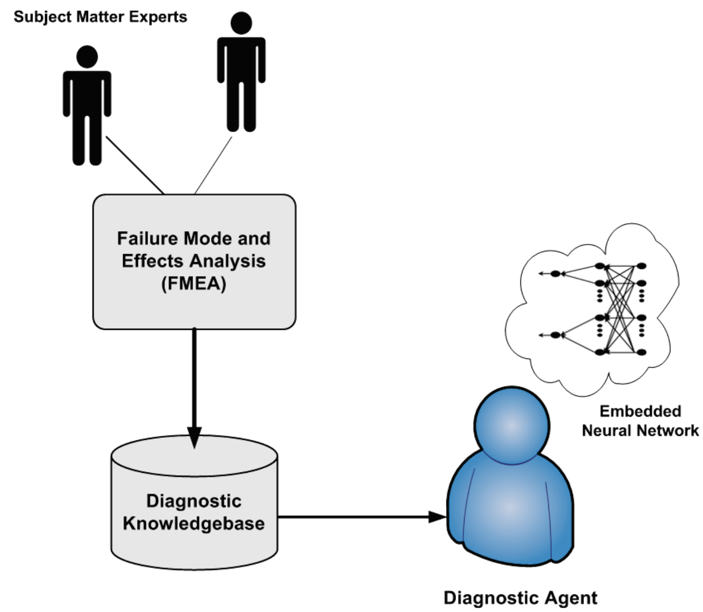


Figure 1 – Diagnostic Knowledgebase Creation and Management

For each fault enumerated during the FMEA, a corresponding list of related symptoms should be identified. Figure 2 illustrates a typical diagnostic specification for a lube oil temperature regulator valve fault. Jointly, all similar diagnostics are organized into a knowledgebase that can be represented as a large fault-symptom matrix. This FMEA knowledgebase is then used to train the Probabilistic Neural Network diagnostic reasoner discussed in the following section.

Component: **LO Temp Regulator Valve**

Fault: **LO temp regulator valve stuck open to cooler**

Symptoms: LO inlet temp - LOW
Duplex strainer differential pressure - HIGH
LO pressure to engine - HIGH
Average bearing temp - LOW
LO sump tank temp - LOW
Full flow filter differential pressure - HIGH
LO attached pump discharge pressure - HIGH

Figure 2 - Typical Fault-Symptom Specification

Probabilistic Neural Networks (PNN)

The primary attractive features of the PNN method as a practical means of rapidly deploying automated diagnostic/prognostic systems are:

- PNN training is effectively instantaneous, as opposed to the slow error convergence training of other ANN techniques, such as backpropagation, which can require hundreds of thousands of computational iterations. Besides the reduced effort for system commissioning, this feature is extremely attractive for allowing training data set modifications and PNN retraining in the field by end-users.

- Classification probabilities are directly output from the PNN, generated by its nonlinear decision surfaces, which approach the Bayes optimal as the number of training samples increases. This is a clear advantage over rule-based approaches incorporating subjective probability estimates or confidence intervals without statistical basis.

- PNN is capable of handling situations in which one or more input variables are missing or are corrupted. This makes the method attractive for real-world applications where sensor failures occur on a regular basis, such as in a shipboard environment.

A PNN can be used to classify symptom patterns according to the faults that may have generated the alarm conditions. The PNN is pre-trained to learn the associations between a large number of faults and their corresponding symptom patterns. Once trained, the PNN can be connected to the machinery plant automation system to perform real-time diagnostics *(this is how DEXTER works)*.

Most ANNs perform some sort of statistical computation on patterns contained in a training data set. These internal statistics are then used to classify new patterns presented as inputs to the trained network. The classification problem can be posed as an example of Bayesian classification, in which it is desired to categorize a set of inputs (symptom pattern). In the context of diagnostic applications, the categories represent the different machinery faults.

The PNN estimates class conditional probability density functions (PDFs) according to the following equation:

$$f_A(\mathbf{X}) = \frac{1}{(2\pi)^{p/2}\sigma^p}\frac{1}{m}\sum_{i=1}^{m}\exp\left[-\frac{(\mathbf{X}-\mathbf{X}_{Ai})^T(\mathbf{X}-\mathbf{X}_{Ai})}{2\sigma^2}\right] \tag{1}$$

where:

i	=	symptom pattern number
m	=	total number of training symptom patterns
$\mathbf{X}$	=	input symptom pattern
$\mathbf{X}_{Ai}$	=	i^{th} training pattern from category A
σ	=	"smoothing parameter"
p	=	dimensionality of measurement space.

Equation (1) defines the PDF for each fault as the sum of several multivariate Gaussian distributions centered at each training sample for a given class. This equation is applied to each fault and directly outputs the probability of the fault given an input symptom pattern. Conceptually, the input symptom vector is compared to the training symptom vector for the fault class. The closer the match between the two, the higher probability of the fault classification. An important characteristic to note is that the fault probability can still be obtained even if one or more components of the input symptom vector are unavailable or mismatched. In these cases, the resulting fault probabilities may be lower; however, when compared to a rule-based reasoning system, the PNN will still yield a diagnostic, whereas **the strict logic of the rule-based system will fail if only a single input component is false.**

Equation (1) is implemented in the pattern units of the PNN, as depicted in figure 3. The input units simply feed the input values to the pattern units. Each input unit has a connection with every pattern unit. The pattern units form the dot product of the input pattern vector, $\mathbf{X}$, with a weight vector, $\mathbf{W_i}$. There is one pattern unit for each training pattern and the weights associated with the pattern unit are equal to the values of the corresponding training vector. The dot product calculated in each pattern unit undergoes a nonlinear transformation in the PNN using an activation function similar specified by the Gaussian PDF given in equation (1).

Network training is accomplished by setting the weight vector of each pattern unit equal to the values of one of the training vectors. In this way, each training vector uniquely defines the weights of one pattern unit. As previously mentioned, a separate pattern unit is required for every training pattern; however, this is easy to implement using the fault-symptom matrix resulting from the machinery plant FMEA.

The input vector, $\mathbf{X}$, is comprised of the symptom pattern representing either current alarm conditions or predicted alarm conditions, depending on whether diagnostics or prognostics are being performed. Alarm conditions can be quantized to varying levels of resolution. In the current version of DEXTER, alarms are represented by a three-way classification as HIGH, LOW, or NORMAL states numerically encoded into the input vector. These classifications are performed by statistical thresh-holding of model-based deviation values, rather than on raw sensor values used in most existing alarm monitoring systems. This allows early fault detection and identification of evolving machinery problems.

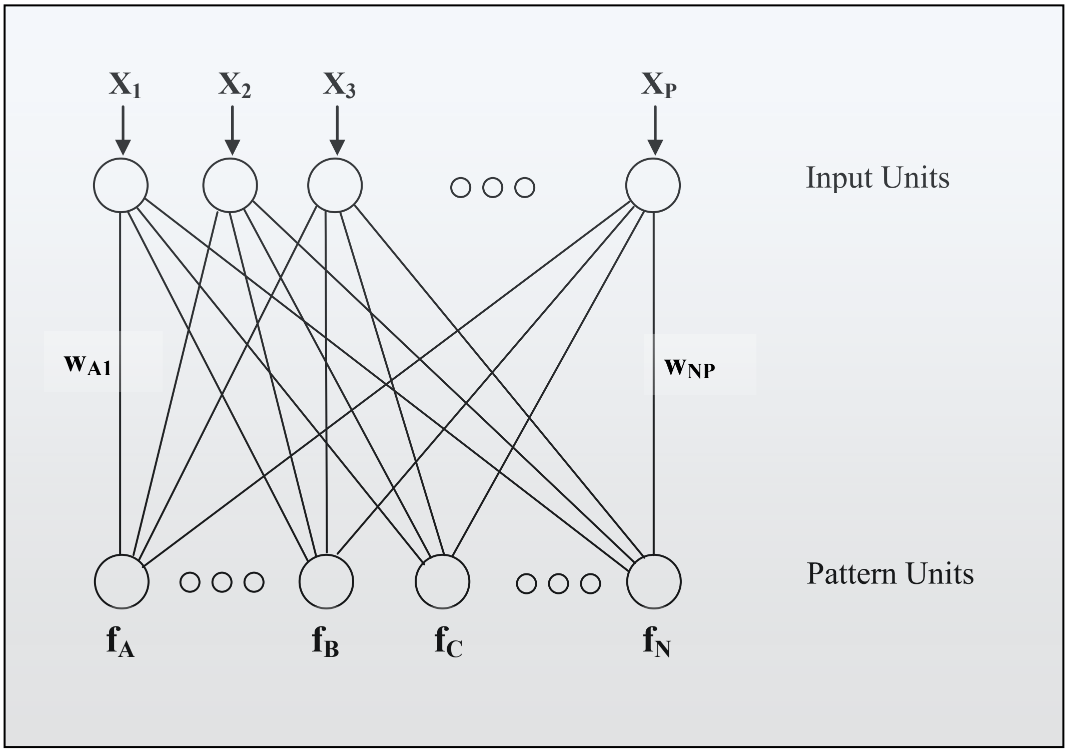


Figure 3 – PNN for Computing Fault Probabilities

The only parameter to be adjusted in the PNN is the "smoothing" parameter, **σ**, which is related to the variance of the underlying PDF. This parameter controls the ability of the PNN to generalize when the input vectors do not exactly match the training vectors.

A Diagnostic Example

One of the most difficult tasks in fielding successful diagnostic systems is dealing with the issue of uncertainty. Uncertainty issues arise in both sensor reliability and sensor measurement classification (i.e. symptom generation). Sensor failure, which includes out-of-calibration condition, is a common occurrence aboard ships, yet all high level automations functions, including monitoring, control, performance analysis, and diagnostics/prognostics, critically depend on accurate sensor measurement inputs. As such, a sensor failure can have a significant influence on the reliable performance of these systems.

To illustrate how rule-based diagnostic systems fall apart when sensors fail, consider the following example. Certain rule antecedent conditions are defined as particular sensor states, such as in the following diagnostic:

Related Component: JW Temp Regulator Valve

Fault: JW temp regulator valve failed open to JW cooler

Symptoms: JW temp to engine - LOW
JW temp from engine - LOW
LO temp to engine - LOW
JW temp from LO cooler - LOW
Avg. engine exhaust temp - LOW
JW cooling system effectiveness - HIGH

Figure 4 – Example Diagnostic for JW Temp Regulator Valve

The above diagnostic can readily be converted to a rule as follows:

IF:
JW temp to engine LOW
AND
JW temp from engine LOW
AND
LO temp to engine LOW
AND
JW temp from LO cooler LOW
AND
Avg. engine exhaust temp LOW
AND
JW cooling system effectiveness HIGH
THEN:
JW temp regulator valve failed open to JW cooler

This rule is strictly a logic-based statement that evaluates to either True or False. The absence on one of the rule's antecedents, such as JW temp to the engine, due to a failed temperature sensor, will effectively disable this diagnostic rule, since without this measurement input, the rule will always logically evaluate to False. This problem is commonly referred to as "brittleness" in discussions of rule-based diagnostic systems, as the basic functionality of the diagnostic system is brittle and breaks down in the face of sensor failures.

Uncertainty also arises in the classification of the sensor inputs into normal and abnormal states. Most alarm monitoring systems use thresholds to define boundaries between normal and abnormal (high or low) machinery conditions. This classification or alarm generation process is entirely dependent on the selection of suitable alarm thresholds. There is a certain degree of uncertainty related to setting these thresholds, as they can be based on anything from manufacturer specified limits to statistically derived limits to settings for annoyance alarm prevention. Referring back to the previous diagnostic rule example, it is clear that the misclassification of the JW temperature LOW state will have the same brittle effect on the diagnostic system as if the sensor had failed.

Uncertainty related to sensor failures and alarm generation can result in erroneous and/or unreliable performance of diagnostic systems. Missed diagnostic calls and false positives translate directly into added maintenance costs, either from unexpected machinery failures or unnecessary maintenance activities. The robustness of the diagnostic system can have a direct relationship to maintenance expenditures, as well as to equipment reliability. The treatment of uncertainties, such as those described above, is therefore a serious concern for diagnostic/prognostic systems employed as maintenance decision aids.

The PNN used in DEXTER is capable of handling missing sensor inputs in its diagnostic reasoning. Figure 5 below shows the fault probabilities computed by a PNN trained to recognize the previous JW temperature regulator valve fault. The PNN probabilities are plotted against the number of symptoms present as inputs to the PNN. For example, with the first symptom (JW temp to engine LOW) as the only input, the PNN computes the probability of the fault as 7%. The addition of the second symptom (JW temp from engine LOW) causes the PNN's probability to increase to 11%. Each additional symptom input increases the probability of the fault, up to the maximum of 100% when all symptoms are input to the PNN. The brittleness associated with a rule-based approach is avoided.

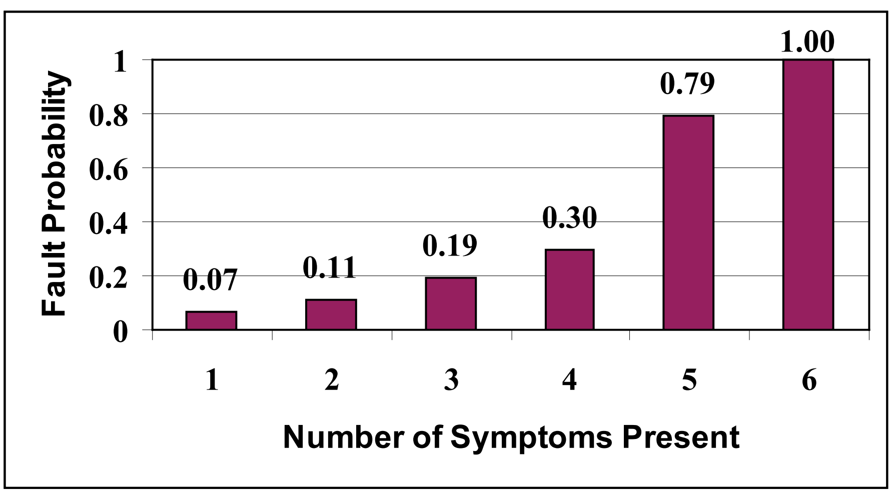


Figure 5 – PNN-computed Probabilities for JW Temp Regulator Valve Fault

Summary

A practical approach to developing diagnostic and prognostic systems is to incorporate experiential and engineering knowledge of domain experts into a diagnostic knowledgebase. This can be accomplished by conducting a FMEA on the machinery plant under diagnosis. The resulting knowledgebase can then be used as training data for neural network pattern recognition techniques, such as the PNN. In so doing, a more complete scope of coverage can be developed over a much shorter time interval, and typically at minimal cost.

The PNN method is attractive as a practical means of rapidly deploying automated diagnostic and prognostic systems because:

- PNN training is effectively instantaneous,
- Fault classification probabilities are directly output from the PNN,
- PNN's generalization capabilities can handle situations in which one or more input variables are missing or are corrupted. This makes the method attractive for real-world applications where sensor failures occur on a regular basis, such as in a shipboard environment, and
- PNN's can be rapidly deployed using existing experiential and empirical knowledge and can be remotely updated as new knowledge is acquired.

Extensions to Prognostics

The same PNN that encapsulates expert knowledge can be used for both diagnostics and prognostics. Automatic trending analysis and prognostics work together to predict future machinery faults and to determine when maintenance should be carried out. By predicting machinery problems before they occur, unexpected breakdowns can be avoided. In the absence of significant trends, equipment overhaul periods may be rationally extended, thereby eliminating unnecessary maintenance work. The ability to predict future maintenance requirements leads to improved maintenance planning and cost management. Maintenance and repair decisions can be tied to actual plant operating conditions, based on the severity of degrading trends and predicted plant problems. The next paper in our series will specifically address how DEXTER implements prognostics using the PNN described in this paper.

Our DEXTER software uses probabilistic neural networks for diagnostic and prognostic reasoning about machinery faults. DEXTER's neural networks learn to associate patterns of alarm conditions (symptoms) with the machinery faults. This allows DEXTER to monitor the heartbeat of a machinery plant just like a doctor monitors your blood pressure, the main benefit of both being early detection of health problems.

DEXTER is a mature software product that has been operating on some ships for over a decade. There are currently 44 US Navy ships with DEXTER installed.