Health-State Estimation and Prognostics in Machining Processes

Fatih Camci and Ratna B. Chinnam

Abstract—**Failure mechanisms of electro-mechanical systems usually involve several degraded health-states. Tracking and forecasting the evolution of health-states and impending failures, in the form of remaining-useful-life (RUL), is a critical challenge and regarded as the Achilles' heel of condition-based-maintenance (CBM). This paper demonstrates how this difficult problem can be addressed through Hidden Markov models (HMMs) that are able to estimate unobservable health-states using observable sensor signals. In particular, implementation of HMM based models as dynamic Bayesian networks (DBNs) facilitates compact representation as well as additional flexibility with regard to model structure. Both regular HMM pools and hierarchical HMMs are employed here to estimate on-line the health-state of drill-bits as they deteriorate with use on a CNC drilling machine. Hierarchical HMM is composed of sub-HMMs in a pyramid structure, providing functionality beyond an HMM for modeling complex systems. In the case of regular HMMs, each HMM within the pool competes to represent a distinct health-state and adapts through competitive learning. In the case of hierarchical HMMs, health-states are represented as distinct nodes at the top of the hierarchy. Monte Carlo simulation, with state transition probabilities derived from a hierarchical HMM, is employed for RUL estimation. Detailed results on health-state and RUL estimation are very promising and are reported in this paper. Hierarchical HMMs seem to be particularly effective and efficient and outperform other HMM methods from literature.**

Note to Practitioners — **Today's high competitive environment forces industry to decrease operating & support cost, whose one of the most contributing factors is maintenance and repair cost. Thus, industry is interested not only in the identification of failures, but also in identification of failure states, their progression and forecasting. This paper presents health state estimation and remaining useful life prediction in machining processes with a case study on drilling processes.**

Index Terms **— Condition-based-maintenance, diagnostics, health-state estimation, prognostics, remaining-useful-life, dynamic Bayesian networks, hidden Markov models**

I. Introduction

Condition-Based-Maintenance (CBM) is a maintenance technology that employs such tasks as monitoring, classification, and forecasting to increase system readiness and safety while reducing costs attributed to reduced maintenance and inventory, increased capacity, and enhanced logistics and supply chain performance [1]. Unlike time-based *preventive maintenance* and *corrective maintenance* practices, CBM aims to avoid both unnecessary maintenance actions as well as machine failures. The Center for Intelligent Maintenance Systems estimates that $35 billion per year would be saved in the United States alone if CBM technology were widely employed [2].

The failure mechanisms of electro-mechanical systems usually involve several degraded health-states. For example, a tiny change in a bearing's position could cause a small nick in the bearing, which could cause scratches in the bearing race in time, which then could cause additional nicks, which could then lead to complete bearing failure [3]. Tracking and forecasting the health-state of a machine is very critical for detecting, identifying, and localizing the failure as well as carrying out proper maintenance. Hence, employing effective diagnostic and prognostic algorithms/methods is an important prerequisite for widespread deployment of CBM [4].

Diagnostics is the process of identifying and localizing the machine failure, and determining its primary cause and severity, whereas *prognostics* is the process of estimating the *remaining-useful-life* (RUL) [5]. Diagnostics is, in essence, a classification problem, and there are many methods proposed and implemented in the literature that attempt to resolve this problem; this is in much contrast to prognostics, which is essentially a forecasting problem. However, most diagnostic algorithms have limited potential in that they cannot detect failure modes in a timely manner. See [1] for a thorough review of popular diagnostic algorithms and methods.

Diagnosing effectively the earliest stages of a failure, even if the machine is serving its intended function, is not only important but is a prerequisite for prognostics. It is logical to diagnose these health-states through their effects on observed sensor signals since we are unable to 'observe' real health-states. The primary challenge then within diagnostics is to achieve high classification accuracy in identifying health-states given sensory signals such as vibration, current, temperature, etc.

Prognostics is a dynamic process that evolves in time from the moment the machine is first used until it fails. RUL should not be confused with *expected life expectancy*, which is the '*mean-time-to-failure*' of an average machine/component [6]. Expected life expectancy is the average life of similar components/machines or a family of machines, while RUL is the time-to-failure of a specific machine, which is being

Re-revised manuscript received in August 2009. This work was supported in part by NSF DMI Grant 0300132 and TUBITAK (The Scientific and Technological Research Council of Turkey) under project number 108M275.

F. Camci, Assistant Professor of Computer Engineering Department at Fatih University, Istanbul Turkey 34500 (e-mail: fcamci@fatih.edu.tr).

R. B. Chinnam, Associate Professor of Industrial and Manufacturing Engineering Department at Wayne State University, Detroit, MI 48202 USA (corresponding author, phone: (313) 577.4846, fax: (313) 578.5902, e-mail: r_chinnam@wayne.edu).

monitored. Prognostics is far more difficult than diagnostics, and is currently regarded the Achilles' heel of CBM [1].

Despite considerable advances in sensing hardware, communications, information technologies, and software algorithms, equipment *health monitoring* and *diagnostics* are still largely reserved for only the most critical system components and have not found their place in the mainstream [7]. Worse, there exist no robust *prognostics* methods (for predicting remaining-useful-life of existing assets), a vital enabler of *condition-based maintenance*, for even the most critical system components. The aim of the models presented in this article is to squarely address this issue in the domain of machining processes (one of the most common family of manufacturing processes), in particular, for efficient and economical replacement of cutting tools.

This paper employs *Hidden Markov Models* (HMMs), which characterize doubly embedded stochastic processes with an underlying stochastic process that can be observed through another stochastic process and have been successful in tackling such difficult tasks as automatic speech recognition (ASR) [8] ,[9]. The tasks of both ASR and equipment diagnostics have many commonalities. Speech signals are quasi-stationary, and so are the sensory signals such as machine vibration [10]. Quasi-stationary signals in some sense terminate in an absorbing state and show stationary behavior in any reasonable time scale [11]. In addition, words should be recognized in automatic speech recognition, although they are spoken by different speakers, whereas health-states should be recognized in diagnostics although machine behavior can be quite different due to such factors as manufacturing and assembly variation, operating/maintenance history, and aging. Beyond the aforementioned commonalities, implementation of HMMs for diagnostics is more difficult than their implementation in speech recognition. For example, the number of phonemes is a relatively small finite set in ASR (resulting in sound and word libraries), a notion that is neither observed nor justified in machine diagnostics. In addition, S/N ratios tend to be far better in speech signals unlike machine sensor signals. Nevertheless, speech signal only remains stationary over intervals of approximately 10ms. In comparison, machine vibration signals remain stationary on time scales of many seconds and even minutes [10].

Regular HMMs were implemented for health-state estimation in the literature [3], [10] ,[12]-[16]. However, regular HMMs tend to be limited in their ability to represent complex systems. More importantly, in the absence of 'labeled' health-state sensor signal examples, the unsupervised learning process for diagnostic applications is computationally tedious, for it involves such methods as competitive learning [15]. In addition, regular HMMs do not have intrinsic transition probabilities between health-states since each HMM represents a distinct 'health-state'. Hence, they require additional methods to calculate health-state transition probabilities to be utilized in RUL estimation. Thus, effective prognostics could not be carried out using regular HMMs. While some have indicated the potential for using log-likelihoods of health-state HMMs for calculating RUL [12], [14], they rely on mostly empirical regression models. The aim of this paper is to be able to obtain effective diagnostics and prognostics results by overcoming aforementioned difficulties (inability to represent complex systems, computational difficulty, and lack of health-state transition probability for RUL estimation).

In this article, we present the implementation of hierarchical HMMs as dynamic Bayesian networks for health-state and remaining-useful-life estimation. The main contribution of this paper is to be able to obtain effective diagnostics and prognostics results for machining processes (e.g., drilling process) using Hierarchical Hidden Markov Model (HHMM). Hierarchical HMM, a variant of a HMM that is composed of several sub-HMMs in a pyramid structure, strengthens the ability of an HMM to jointly represent multiple health-states along with their state transition properties, facilitating estimation of RUL.

The paper is organized as follows: Section II gives the problem description, while section III discusses hidden Markov Models (HMM) and dynamic Bayesian networks (DBN) and their implementations for health-state estimation (diagnostics) and RUL estimation (prognostics). Section IV presents results from a drill-bit application of regular HMMs and HHMMs for health-state estimation and RUL estimation using HHMM and Monte-Carlo simulation. Finally, Section V offers some concluding remarks.

II. Problem Description

A. Drilling Process

Drilling process is one of the most commonly used machining processes in industry [17], [18]. Up to 50% of all machining operations in US involve drilling [19]. In addition, around 40% of metal removal operations in the aerospace industry involve the drilling process [20]. For example, production of a typical small jet fighter requires about 245,000 holes to be drilled [21].

Tool breakage and/or excessive wear may cause fatal defects in the product. Quality of drilled holes is crucial for some 60% of rejected parts are often attributable to poor hole quality [18]. Drill-bit, as a cutting tool, is one of the main components that affects the hole quality. Thus, early detection of drill-bit breakage and/or excessive wear is important and has been studied extensively [17]-[21]. While there is a large body of literature targeting monitoring and diagnostics of drilling processes, there is very little work related to drill-bit prognostics.

It is generally agreed that the most appropriate sensor signals for drill-bit monitoring and diagnostics are longitudinal thrust-force and torque signals [17],[18]. For example, a careful review of diagnostics methods that employ 11 different sensor signals for monitoring drill-bits reveals that the most commonly used and valuable signals are indeed thrust-force and torque signals [22]. For these reasons, we too employ these same sensor signals for monitoring the drilling process.

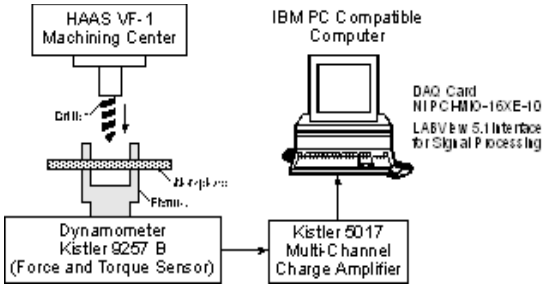


Figure 1: Experimental setup for capturing thrust-force and torque degradation signals during drilling process.

Fig. 1 illustrates the drilling process along with a data acquisition system. The signals collected during the actual drilling process (from the time the drill-bit enters the work piece until it exits the work piece from the other side), labeled 'hole signal vector(s)' will be used for monitoring the drill-bit health. Thus, the life of the drill-bit can be discretely modeled in terms of number of holes successfully drilled by the bit rather than actual drilling time.

B. Problem Formulation

When used, drill-bits undergo deterioration, and in the process, go through several 'health-states' (say 'brand new' state to the 'failure' state as do humans from infancy to death). The fundamental problem this paper targets is two-fold: identification of current drill-bit state (*diagnostics* – health-state estimation) and remaining useful life (*prognostics* – number of additional holes the drill-bit can successfully drill from the 'current' state to 'failure' state).

1) Diagnostics – Health State Estimation

Health-states of the drill-bit cannot be clearly observed due to the nature of the process. In most industrial settings, it is very difficult if not impossible to stop and physically observe/assess the drill-bit state after every hole. However, the effects of the health-states can be observed indirectly through the signals under observation (i.e., thrust-force and torque). Since the identification of the health-states cannot be deterministic due to the variations within the process, material and other external factors, they will be represented with probabilities. The diagnostics problem can be formulated as follows:

$$HS = \arg\max_{S_i} \left(P\left(X_t = S_i \mid O_{1:t} \right) \right) \ \forall i, \ \ i = 1...N \quad (1)$$

where X_t is health-state variable at time t, $O_{1:t}$ is the observed signal vector(s) from time 1 to t, S_i is the health-state i, N is the number of possible distinct health-states, and HS is the identified health-state with highest probability.

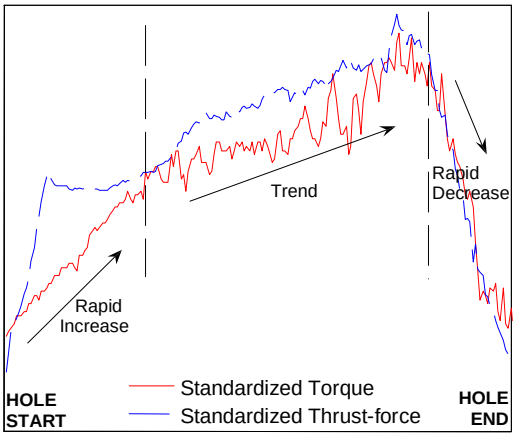


Figure 2: Sensor signals from a typical hole drilling cycle revealing different sub-state signatures.

Eq. (1) would be adequate for diagnostics if the observations within a hole given the health-state were stationary (having static statistical properties). Unfortunately, thrust-force and torque signals are non-stationary during the hole generation. When the drill-bit enters the material, the thrust-force and torque signals increase rapidly. Later, they become relatively smooth with increasing trend, partially due to increase in friction between the side wall of the partially completed hole and the rotating drill-bit. Once the drill-bit produces the hole (goes through the other side of the work piece), the thrust-force and torque signals decrease rapidly in amplitude. In other words, there exist other states, labeled 'sub-states', within the hole that affect the observations (their amplitudes and signatures) besides overall drill-bit 'health-states'. Fig. 2, a sample plot of sensor signals from a single hole, clearly reveals these sub-states. In addition, health-states not only affect observations, but also affect the sub-states and their transitions. Hence, the monitoring algorithms have to explicitly account for transitions between both the sub-states within a hole as well as overall health-states for effective diagnostics.

The diagnostics formulation can be re-written to handle the sub-states as follows:

$$HS = \arg\max_{S_i} \left(P\left(X_t^1 = S_i \mid O_{1:t}, X_{1:t}^2 \right) \right) \ \forall i, \ \ i = 1...N \quad (2)$$

$X_{1:t}^2$: Sub-states from time 1 to t

X_t^1: Health-state at time t

Note that these formulations help us characterize the problem, but do not present a solution. These formulations may not be solvable directly and may need several other formulations to represent the relationships. For example, the sub-states $X_{1:t}^2$ are not observable and have to be estimated using inference algorithms.

2) Prognostics – Estimation of Remaining Useful Life

Here, the goal is to estimate the number of additional holes the drill-bit can successfully drill from the current state to failure state. This can be formulated as follows:

$$Find \ n: \ X_t^1 = i, X_{t+1}^1 = j, ..., X_{t+n-1}^1 = l, X_{t+n}^1 = f \quad (3)$$

where $t, t+1, t+2, \ldots, t+n$ represent the distinct hole sequence (t denotes current hole and $t+n$ the last successfully drilled hole prior to failure), $i, j, \ldots, l, f$ are the health-state transitions, f the failure state ($f > l \geq \ldots \geq j \geq i$), and n is a random number representing the number of remaining drilling cycles or holes.

Next section discusses the models to be employed for addressing the formulations from Eqns. (2) and (3).

III. Modeling Background

Two different modeling approaches that employ hidden Markov models (HMMs) will be presented here to solve the diagnostic and prognostic formulations discussed above (i.e., Eqns. (2) and (3)). First approach employs a 'set' of HMMs whereas the second approach employs a single Hierarchical-HMM (HHMM).

A. Hidden Markov Models (HMM)

This section discusses hidden Markov models and their application to drill-bit health-state estimation using 'competitive learning' [23]. Readers familiar with HMMs can skip the first sub-section and move to the second sub-section, which gives the application details.

1) HMM Description

In state-space modeling, a stochastic system can be described as being in one of a finite number of states at any time. The system evolves through the states according to a set of probabilities associated with each state as demonstrated in Fig. 3. The model is called a hidden Markov model if states are not observable (hidden) and are assumed to be causing the observations. The system behavior depends on the current state and predecessor states. A special case, first-order HMM, assumes that only the current state is responsible for producing the observations. In the remainder of this paper, HMM implies a first-order HMM.

To better understand HMM, let us consider an urn and ball system with 3 urns and a different number of colored balls in each urn [24]. An urn is selected randomly; then, a ball is chosen from this urn, its color is recorded, and it is placed back into the urn it is chosen from. In the next step, a new urn is selected, and a ball is chosen from this urn and recorded. This process is repeated a finite number of times resulting in a finite observation sequence of ball colors. Now, assume that the system is in a different room and handled by someone else so that we don't see the selected urns. The only event observable to us is the colors of the selected balls. Obviously, the simplest HMM representation of this system corresponds to states being urns with different color probabilities for each urn.

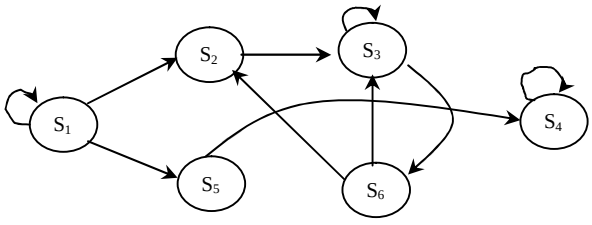


Figure 3: A Markov chain with 6 states and state transition probabilities (arrows represent non-zero state transition probabilities).

There are several elements to an HMM: number of states (N), observations, state transition probability distribution, observation probability distribution, and initial state distribution. X_t denotes the state at time t and O_t denotes observation at time t, which might either be a discrete symbol $O_t \in \{1, \ldots, L\}$ or a feature vector from L dimensional space, $O_t \in R^L$. State transition probability distribution models the probability of being in state i at time t, given that it is in state j in time $t-1$, and is denoted as $A = \{\alpha_{i,j}\} = P(X_t = i \mid X_{t-1} = j)$. Observation probability distribution defines the probability of observing k at time t given the state i, denoted as $B = \{b_i(k)\} = P(O_t = k \mid X_t = i)$. These distributions are either mass functions in the case of discrete observations or specified using a parametric model family -commonly Gaussian- in the case of continuous observations. Initial state distribution is the probability of being in state i at $t = 0$ and is denoted as $\pi(i) = P(X_1 = i)$. Generally, $\lambda = (A, B, \pi)$ is used to specify a HMM. The rest of this paper employs the Gaussian observation model.

There are three basic problems of interest to be solved given the above model specifications.

- How to compute the probability of obtaining the observation sequence $O = O_1 O_2 \ldots O_T$ given the model λ ? (i.e. $P(O_1 O_2 \ldots O_T \mid \lambda) = ?$) The forward-backward (FB) algorithm [25],[26] is commonly used for this since it is more efficient than the direct evaluation method.
- How to identify the most likely state sequence that might produce the observation sequence? The Baum-Welch algorithm, also called the Expectation Maximization (EM) algorithm, uses both the forward and backward procedures to solve this problem [27].
- How to adjust or learn the parameters of λ in order to maximize the likelihood of the given observation sequence? The EM algorithm solves this problem as well.

These three problems are tightly linked and studied extensively in the literature. The standard HMM solution to these problems requires an exponential number of parameters to specify the transition and observation models since it calculates the Cartesian product of the state-spaces of each example. This means requiring excessive amounts of data to learn the model (high sample complexity) and exponential time for inference (high computational complexity). For example, the FB algorithm cycle takes $O(Tk^{2N})$ operations. For more detailed information about HMMs, see [24].

2) HMMs for Drilling Process Diagnostics with Competitive Learning

As mentioned in section II.B.1, the thrust-force and torque signals are non-stationary within a hole. In other words, there exist several sub-states within a hole such as 'rapid increase', 'trend', and 'rapid decrease' states as displayed in Fig. 2. In standard HMM modeling, the state of the HMM (X_t)

represents these states in the hole. Given that HMM states are modeling the sub-states within a hole, a single HMM is incapable of modeling the different overall 'health-states' witnessed during the life of a typical drill-bit. Thus, we employ a set of HMMs, hoping that at the end of the (competitive learning based) training process, each HMM within the set will represent a distinct health-state.

Identification of current health-state given observations of a hole is performed as a competition among HMMs. Under competitive learning [23], HMMs (undergoing learning) compete to represent the health-state dynamics present within each training example, i.e., sensor signals from a hole, presented one at a time in random order to the HMM pool. The HMM with the highest fitness, i.e. highest log-likelihood value, is identified as a winner and is considered the representative of the current health-state. In general, only the HMM winner is allowed to further learn using the current training example (however, literature offers other learning schemes that might allow few nearest neighbors of the HMM winner to learn as well). This process is repeated for all holes of all drill-bits that are being used for training. Training continuous until predefined number of iterations or convergence is achieved (see [23] for more details). Fig. 4 further illustrates this competitive learning based training procedure for the HMM pool. Once trained, for actual on-line diagnostics, the health-state of any given hole sensor signal vector is defined by the trained HMM with the highest log-likelihood value as illustrated in Fig. 5.

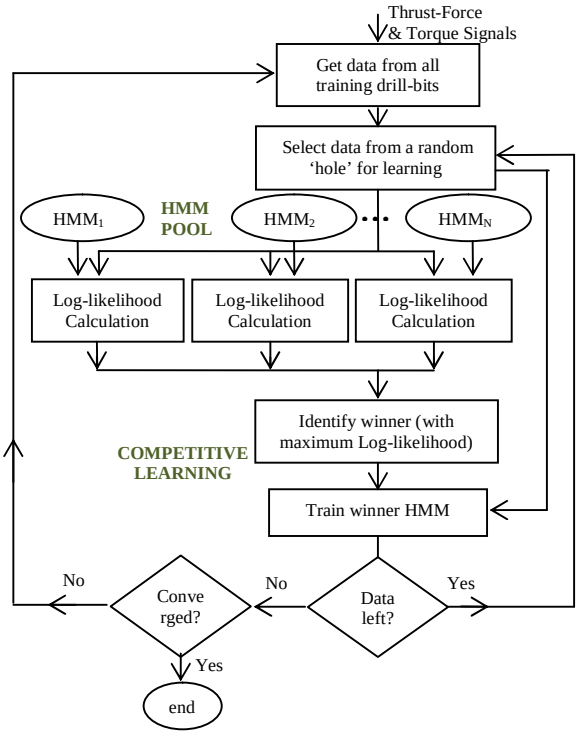


Figure 4: Procedure for training HMM pool through competitive learning for health-state diagnostics.

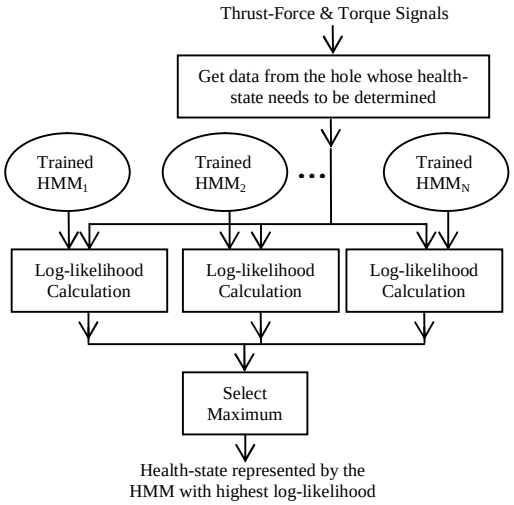


Figure 5: Health-state inference with regular HMMs.

B. Hierarchical Hidden Markov Models (HHMM)

Hierarchical HMM (HHMM), proposed by Fine *et al* [28], is an extension of an HMM that is designed to model hierarchical structures for sequential data. In the HHMM, each state is considered to be a self contained probabilistic model. More precisely, each state of the HHMM is itself an HMM (or even a HHMM). When a state in an HHMM is activated, it will activate its own probabilistic model, i.e. it will activate one of the states of the underlying HHMM, which in turn may activate its underlying HHMM and so on [29]. The process is repeated until a special state, called a 'production state', is activated. Only the production states emit observations in the usual HMM sense.

Fig. 6 provides an illustrative example of this hierarchical structure of hidden-states. In the figure, X_t^1 represents the t^{th} top-state and X_k^2 represents the k^{th} sub-state. The states that do not directly emit observations symbols are called hidden or 'internal states' (X_t^1 and X_k^2 are hidden states). The transition of a hidden state from one to another within the same level is called 'horizontal transition'. For example, under X_1^1, transition from X_1^2 to X_2^2 is an example of horizontal transition. After the last state is reached in sub-states, a 'vertical transition' is allowed. For example, after sub state transitions reach the last state X_3^2 indicating that the hole has ended, health-state transition from X_1^1 to X_2^1 may occur in the example from Fig. 6.

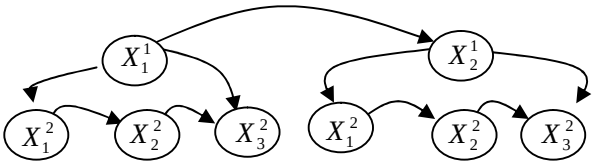


Figure 6: Hierarchical representation of states in HHMM: Top states (X_i^1) and Sub-states (X_j^2)

The methods for estimating the HHMM parameters and model structure are more complex than for the HMM, see [28] for more details. Dynamic Bayesian Network (DBN) can more efficiently represent HMMs with the added flexibility of implementing different variants of HMM such as HHMM. Readers who are familiar with DBNs may skip the next two sub-sections and move to the sub-section on HHMM for drilling process diagnostics.

1) Dynamic Bayesian Network (DBN)

Graph and probability theories are the basis of Bayesian networks, also known as Graphical Models, that combine the visual representation of variables (i.e., nodes) and conditional probabilities representing their cause-effect relationships. The conditional probability distribution of a node variable depends only on the parents of the node and is independent of its ancestors (i.e., parent's parents) given its parents. This is called the factorization property, and it dramatically reduces the number of model parameters.

While Bayesian networks are effective in their representation, they are limited to modeling 'static' relationships. However, sequential data arises in many areas of science and engineering, calling for modeling of dynamic processes. Dynamic Bayesian networks (DBNs) are state-space models that are effective for modeling these types of stochastic processes, and are more general and expressive than HMMs, where hidden variables are discrete, and Kalman Filter Models (KFMs), where the hidden variables are continuous [30], [31]. While state-space in HMMs consists of a single discrete random variable, DBN can represent the hidden-state in terms of a set of random variables. While KFM requires all the condition probability distributions to be linear-Gaussian, DBN allows arbitrary CPDs. In addition, HMMs and KFMs have a restricted topology, whereas, DBN allows much more general graph structures. Note that the term 'dynamic' in DBN means we are modeling a dynamic system, and does not mean that the Bayesian network graph structure changes over time. DBNs are thus designed to model probability distributions of hidden variables (states) that evolve in time by using sequenced observed variables that are generated by these hidden-states [32]. DBN structure consists of different levels such as *observation level* (O nodes - shaded) and *hidden-state level* (X nodes – not shaded), as illustrated in Fig. 7. Observation in time t (O_t) is generated by hidden-state X_t and previous observation (O_{t-1}). Once the structure is designed, conditional probability distributions (i.e., initial probability distribution $P(X_1 = i)$, state transition distribution $P(X_t \mid X_{t-1})$, and the observation distribution $P(O_t \mid X_t)$) of each node given its parents are learned during the 'training' process. It is also assumed here that transition and observation functions do not change over time. In this work, the observation distribution is assumed to be continuous and Gaussian as represented in (4).

$$P(O_t \mid X_t = i) \sim N(\mu_i, \sigma_i^2) \tag{4}$$

Even though a variable from different time instants could be potentially represented as distinct variables in the DBN network, the difficulty in representation and computation is obvious even from problems with limited time history (e.g., 10 distinct variables will be required to represent 1 variable in 10 time instants). Hence, variables in different time slices that have the same parental structure are represented as one variable in DBNs to obtain compact representation. For example, in Fig, 7, hidden-states in all time slices except the initial time slice (i.e., $t = 2,3,...n$ but not $t = 1$) have the same parental structure, in which each variable has a previous hidden-state as the sole parent, and are represented as one variable X_t. Similarly, observable variables in all time (i.e., $t = 1,2,...n$) are represented as O, since they have a hidden-state of the same time slice as the parent. Representation of HMM as a DBN thus requires just three variables (i.e., X_1, X_t, and O; one for the initial hidden variable, one for other hidden variables, and one for the observable, respectively). The goal of a DBN acting as a HMM is to infer the hidden-state given the observation sequence, which can be represented more precisely as $P(X_t = i \mid O_{1:t})$. Fig. 7 illustrates the implementation of HMM as a DBN with shaded nodes representing observable variables and blank nodes representing hidden variables.

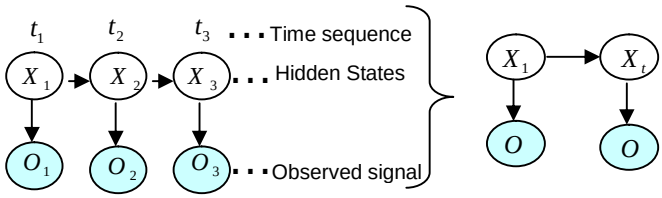


Figure 7: Representation of HMM as a dynamic Bayesian network (DBN).

2) Implementation of HHMM as DBN

In representing HHMM as a DBN, as illustrated in Fig. 8, variables in all levels are represented by a node in each time slice. Top-state (X_t^1) is replicated in Fig. 8 for each sub-state in Fig. 6 as well as the observation state, which represents the observed variable. As illustrated in Fig. 8, the hidden-states in top- and sub-state levels 'cause' the observation and top-level states also cause the sub-level state ('cause' is represented as an arrow). By replicating top-states for each sub-state, we encounter the danger of losing the hierarchical structure. In order to maintain the hierarchical structure, 'top level' state transitions are only allowed if the 'lower level' state reaches the last possible state. In Fig. 6 example, top-state cannot be X_2^1 unless sub-state reaches X_3^2. This can be achieved through a binary control variable *F*, allowing a top-level state change only if the control variable *F*=1. The control variable *F* is allowed to take a value of 1 only if the lower level reaches its last possible state. In other words, X_{t-1}^2 affects X_t^1 indirectly through F_{t-1}. X_t^1 can change only if $F_{t-1} = 1$. F_{t-1} becomes 1 only if X_{t-1}^2 reaches the last state. Hence, six variables (i.e., initial lower and top level hidden-states X_1^1, X_1^2; lower and top level hidden-states other than initial case X_t^1, X_t^2; control state F; and observed state O) are necessary to represent the this HHMM as a DBN.

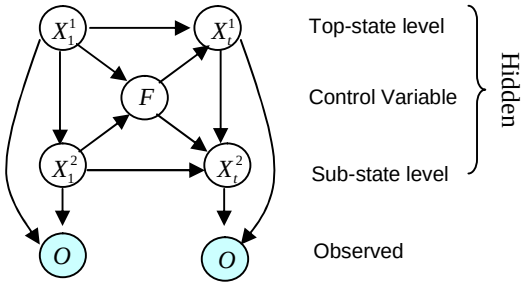


Figure 8: DBN representation of HHMM from Fig. 6.
X_t^d denotes node X in time t at level d

The conditional probabilities of variables in the first/initial time slice are initial distributions and written in (5) and (6):

$$P\left(X_1^1 = j\right) = \pi^1(j) \tag{5}$$

$$P\left(X_1^2 = i \mid X_1^1 = j\right) = \pi_j^2(i) \tag{6}$$

Here, $\pi^1(j)$ denotes the initial top-level state distribution and $\pi_j^2(i)$ the probability of sub-state being in state i, given that the upper state is in state j.

Given that variable F plays the key role of sustaining the hierarchical structure in the rest of the time slices, the conditional probabilities of top and lower hidden layers (other than initial ones) are based on F. If F is 'on' (i.e., F=1), it is a vertical transition (i.e., lets an upper level state change; e.g., from X_1^1 to X_2^1); otherwise, it is a horizontal transition (i.e., transition to a state in the sub-state level under the same top level state; from X_1^2 to X_2^2). In the top level, the conditional probability distribution is the top level state transition probability ($A^1(i,j)$) if the control variable F is 'on'; otherwise, the top level state is not allowed a transition as formulated in (7). In the sub-state level, conditional probability can be drawn from either the initial distribution or the transition distribution depending on the state of the binary control variable F as written in (8). The probability of turning control variable F 'on' is equal to the probability of transition to the last state as stated in (9). In addition, (10) gives the conditional distribution of the observed state.

$$P\left(X_t^1 = j \mid X_{t-1}^1 = i, F_{t-1} = f\right) = \begin{cases} 1 & \text{if } f = 0 \ \ \& \ \ i = j \\ 0 & \text{if } f = 0 \ \ \& \ \ i \neq j \\ A^1(i,j) & \text{if } f = 1 \end{cases} \tag{7}$$

$A^1(i,j)$ denotes transition probability from top state i to j in the top level of the hierarchy.

$$P\left(X_t^2 = j \mid X_{t-1}^2 = i, F_{t-1} = f, X_t^1 = k\right) = \begin{cases} A_k^2(i,j) & \text{if } f = 0 \\ \pi_k^2(j) & \text{if } f = 1 \end{cases} \tag{8}$$

$A_k^2(i,j)$ denotes transition probability from state i to j given that the high level state is in k and $\pi_k^2(j)$ the initial sub-state level distribution given that high level state is in k.

$$P\left(F_t = 1 \mid X_t^1 = k, X_t^2 = i\right) = A_k^2(i,l) \tag{9}$$

where l is the last state.

$$P\left(O_t \mid X_t^1 = i, X_t^2 = j\right) \sim N\left(\mu_{i,j}, \sigma_{i,j}^2\right) \tag{10}$$

The conditional probabilities are learned during the training process using the training dataset. See [33] for more detailed information about DBNs.

3) HHMM for Drilling Process Diagnostics

Realization of a DBN involves three phases: *Designing*, *Training*, and *Testing* as represented in Fig. 9. During the Design phase, the structure of the DBN (i.e., hidden and observable variables and their cause-effect relationships in the form of a directed acyclic graph) and the conditional probabilities are defined. Design phase includes the identification of number health-states and sub-states within a hole. In other words, number of different i and j values in Eqns. (5) and (6). These numbers are typically identified using subject domain expertise and/or trial-error and will be discussed further in section IV.

In the training phase, drill-bits are divided into two groups: one for training, other for testing. Given the observation sequence, parameters $\pi^1(j)$, $\pi_j^2(i)$, $A_k^2(i,j)$, $\mu_{i,j}$, and $\sigma_{i,j}^2$ (as displayed in Eqns. (5)-(10)) are learned using the training dataset. A hole is randomly selected from training dataset (any hole from any drill bit in the training dataset) and DBN (HHMM in our case) is trained with the data sequence of the selected hole. The techniques for learning the mentioned parameters in DBN given the data sequence are mostly straightforward extensions of the techniques for learning BNs, and often involve expectation maximization, such as the Baum-Welch (EM) method. This training process is repeated for all holes to be used for training, presenting them in random order. Training continues until convergence or predefined number of iterations are reached. Fig. 10.a illustrates this training of HHMM. Note that the toolbox developed by Kevin Murphy (Bayesian network toolbox *http://www.cs.ubc.ca/~murphyk/Software/BNT/bnt.html* [34]), employed for this work, has ready functions for training, which get the training data as input and performs the training and returns the trained HHMM with estimated conditional probability parameters.

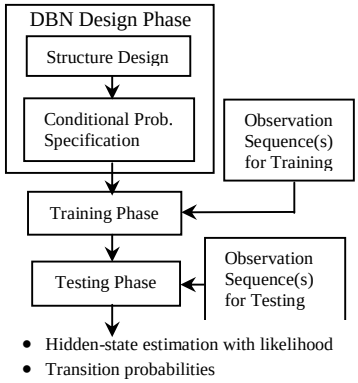


Figure 9: DBN application flowchart.

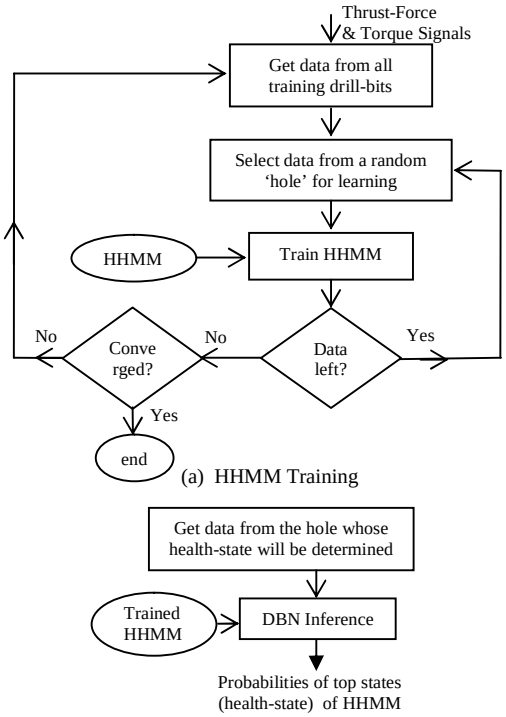


Figure 10: HHMM training and inference illustration.

Once training is complete, the testing phase (i.e., inference) involves estimation of the most likely hidden-state sequence given the observation sequence from all drill-bits in training and testing datasets, albeit separately. While exact inference of hidden-states in DBNs is possible through the forward-backward algorithm and others, it is NP-hard (computationally intractable). Fortunately, there are a variety of deterministic and stochastic approximate inference methods that are computationally efficient while offering reasonable performance. The BNT toolbox mentioned above also has functions for inference. See [28],[33] for more detailed information about methods used in training and testing phases of BNs and DBNs.

4) Prognostics - RUL Estimation

The primary task of prognostics is to estimate the remaining-useful-life (RUL) of the equipment. In the context of CBM, RUL can be defined as the operational hours or number of cycles to be completed before a system or component will require replacement or maintenance. Given the uncertainty of prognostics, it is also best to characterize RUL as a probability distribution.

The literature on prognostics is extremely sparse. In general, prognostics methods can be broadly grouped into two categories: *physics-based* and *empirical-based*. Even though physics-based prognostics models have been attempted for a variety of mechanical systems and sub-systems with some success [35], [36] and might give better results than empirical-based models, they are machine specific and much more expensive to implement. For these reasons, physics-based methods pose significant barriers to widespread deployment of CBM practices.

Empirical prognostic methods can be grouped into three categories. The first approach, *evolutionary prognostics*, involves the trending of key features combined with simplistic thresholds set from past experience and the analysis of the change rate from the current condition to the known failure in the feature space. In [37], failure is defined as specified level of degradation (i.e., threshold) and various degradation models are used to estimate RUL, the time to reach the threshold. Similarly, a Gamma process model is used for degradation modeling in [38]. However, many systems are not simple enough to set thresholds to the features for failure states. The second approach [39] is to utilize *statistical regression models* and/or *computational intelligence methods* such as neural networks to model known failure degradation paths in the feature space. However, these models are not very promising, especially for long-term forecasting, which is a necessity for estimation of RUL. The third approach, *future state estimation*, estimates a state vector that represents the equipment health condition from a brand new state to failure by employing subspace and non-linear dynamic methods [40]. These methods forecast the progression of health-states of the machine from the current state (estimated by diagnostic methods) to the failure state by employing transition probabilities between states and time spent in each state [35], [40], [41]. The proposed method for RUL estimation would qualify as a state-estimator-driven prognostic method. Kalman and Alpha-Beta-Gamma tracking filters are also examples of this approach [42]. In [43], health-states are predicted using a hidden degradation process identification method. However, this method is claimed to be in its infancy and applicable to non-complex conditions for now [43].

As state-estimator-driven prognostic method, HMM is used in [12], [13], [16]. In [16], the health-states for each hole are defined by the user and the estimated transition from one health-state to another is compared with the transition defined by the user. However, supervised learning concept may be misleading in cases where ground truth information is not available. In most health-state estimation problems, the true health-state is not known or impractical to collect. Thus, authors of [16] have presented an unsupervised learning concept in [12], in which HMM models are employed for RUL estimation. In particular, they employ a competitive learning method for health-state estimation. They also proposed three different RUL estimation methods. In the first method (method 1), state transition probability distributions

are estimated. In methods 2 and 3, health-state log-likelihood values are estimated using polynomial and quadratic regression methods, respectively, and RUL is calculated based on estimated log-likelihood values. In the HHMM method presented in this paper, transition probabilities learnt during the training process are used for RUL estimation. The results of the presented method will be compared with the results in [12].

In [13], hidden semi-Markov model is employed for a pump-system. Some of these same authors also proposed an auto-regressive hidden semi-Markov model [14] for diagnostics and prognostics of hydraulic pumps. Even though detailed information about diagnostics results is given in [13], the prognostics results are limited (only mean and covariance of one RUL estimation is reported).

Our proposed method is a future state estimator and employs Monte-Carlo simulation based on the health-state transition probabilities captured by the HHMM in order to characterize the RUL. Since we employed 'left-to-right' HHMM, only forward transition (from left to right) is allowed. In other words, health-state of a drill-bit may get worse or stay the same, but cannot get better as time progress. Health-state transition probability is calculated as follows:

$$P\left(X_t^1 = j \mid X_{t-1}^1 = i, F_{t-1}^1 = 1\right) = \begin{cases} A(i,j) & \text{if } i \le j \\ 0 & \text{otherwise} \end{cases} \qquad (11)$$

where $A(i,j)$ denotes transition probability from state i to state j.

As can be seen from Eqn. (11), the transition probability depends on the current state at (t-1) and possible future state at (t). The time spent in the current state does not affect the transition probability. Given that the proposed HHMM structure does not explicitly incorporate any state duration density, the assumption is that the state durations follow an exponential distribution. Future research will consider explicit state duration density modeling.

Given the current health-state and all possible future health-states and transition probabilities from a HHMM, RUL can be looked upon as the answer to the following question: *How many transitions need to be made to go from the current health-state to the failure state?* In the drill-bit monitoring application, number of transitions from the current state to the failure state corresponds to number of holes to be successfully drilled before the failure of the drill-bit, since a transition from a health-state to another one (or itself) occurs in each hole. Hierarchical structure of the HHMM does not allow multiple transitions in a hole (consider the transition of top-states and control variable F).

It is generally the case that any unit or system under consideration might endure multiple successful operational hours/cycles in the 'failure state' as identified by the HHMM. However, in the interest of not causing a real failure and allowing time for replacement/maintenance, we define as failure an 'entry' into the failure state. Given the learnt HHMM model and the intrinsic state-transition probability matrices, the proposed approach naturally characterizes RUL as a probability distribution, utilizing the frequency table generated by Monte-Carlo simulation. As is the case with any Monte-Carlo simulation-based estimation, large sample sizes are critical for accuracy.

The process is illustrated for a hypothetical system with five health-states and some non-zero state transition probabilities in Fig. 16. According to the figure, the system is currently in the second health-state with the fifth health-state defined as the failure state. RUL, given the current state, is the number of transitions necessary to reach the failure state (i.e., state #5) from the current state (i.e., state #2) by transitioning according to health-state transition probabilities. RUL distribution can be obtained by simulating the transition process several times, yielding, distinct RUL values.

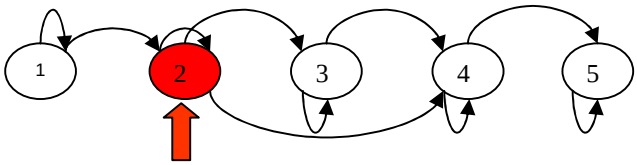


Figure 11: Illustration of equipment health-state transition paths (including self-transitions).
1: Brand new equipment, 5: Failure state, 2: Current health-state

IV. Implementation and Results

The proposed health-state and RUL estimation methods were applied to a drilling process, the most popular industrial machining process. The intent was to estimate on-line (in a non-intrusive way) the health-state of the drill-bit and RUL to facilitate timely replacement of the bit in order to avoid any failures within the work-piece and/or premature replacement. Drill-bits are normally subject to gradual wear along the cutting lips and the chisel edge, which leads to a series of transitions in health-states from a 'brand new' state to a 'failure state' [15]. The objective of on-line estimation of health-states and RUL was achieved through the processes disclosed below.

The experimental setup (see Fig. 1) consisted of a HAAS VF-1 CNC Machine, a workstation with LabVIEW software for signal processing, a Kistler 9257B piezo-dynamometer for measuring thrust-force and torque, and a NI PCI-MIO-16XE-10 card for data acquisition. Twelve drill-bits were used for the experiment, and each was operated until it reached a state of physical failure. Thrust-force and torque sensor signals were employed for health-state estimation given their strong correlation with the health condition of the drill-bit [17], [18]. Stainless steel bars with a thickness of 0.25 inches were used as specimens for tests. The drill-bits used consisted of high-speed twist drill-bits with two flutes and were operated under the following conditions without any coolant: feed-rate of 4.5 inches-per-minute (ipm) and spindle-speed of 800 revolutions-per-minute (rpm). The thrust-force and torque data were collected for each hole from the time instant the drill-bit penetrated the work piece through the time instant the drill-bit protruded from the other side of the work piece. The data was collected at 250 Hz, considered adequate to capture cutting tool dynamics in terms of thrust-force and torque. The number of data points collected for a hole changed between 380 and 460. Data from each hole was binned to 24 RMS (root mean square) values. For illustrative purposes, data collected from drill-bit #5 are depicted in Fig. 11. Given the substantial

difference in the amplitudes of the thrust and torque signals, normalization (i.e., shifting the means to zero and scaling the standard deviations to unity) seems to generally help speed up the DBN parameter estimation process during learning. It involves calculating the mean thrust force (a scalar) and standard deviation of the thrust force (another scalar) of data from all holes of all training dataset drill-bits. We then scale the thrust force signal values by subtracting the mean and dividing the difference by the standard deviation (essentially a shifting and scaling operation). The procedure is repeated for torque signals as well. The established mean and standard deviation parameters of thrust force and torque signals are then employed for scaling signals from the testing dataset as well.

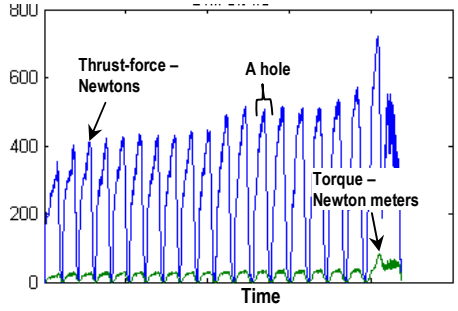


Figure 12: Trust-force and torque data from drill-bit #5.

In order to evaluate the effectiveness of the proposed methods, health-state classification 'accuracy' needs to be assessed. Reliable information is essential for calculation of ideal classification accuracy. For the given experiment, trustworthy information consists of the actual health-state of drill-bits, which might be identified as a function of wear and tear on the cutting lips or the chisel edge at the end of each drilling cycle, an extremely tedious proposition. However, this evaluation process between drilling cycles causes the drill-bit to cool down and lead to a weak representation of true degradation under actual operating conditions. In addition, many real world applications lack accurate information about the health-state of the equipment.

In the absence of accurate and reliable information, we have evaluated the proposed method in two ways: first evaluation involves health-state estimation performance and the second evaluation is based on RUL estimation performance. As for health-state estimation performance (i.e., diagnostics), three criteria are employed: number of reverse health-state jumps, uniformity, and health-state resolution. Reverse health-state jump denotes the case when the diagnostic model indicates that the drill-bit is revisiting a prior health-state after transitioning to a different health-state. For example, if the diagnostic model indicates that a particular drill-bit has reached a state of failure during on-line monitoring and then later indicates that the drill-bit is once again in an operational health-state, we have a reverse health-state jump. Uniformity measures the consistent presence of the different health-states in all the drill-bits used for the training process and during on-line monitoring. Each health-state needs to be visited by most, if not all, drill-bits for generalization of health-state estimation. This is a reasonable measure for this experiment and may not be appropriate for systems that fail due to multiple failure mechanisms. The last criterion, health-state resolution, ensures adequate resolution between a brand new drill-bit and a complete failure. This is measured here by counting the number of identified health-states. As for RUL estimation performance evaluation, we compare the estimated RUL, which is calculated using the estimated health-states, with true or witnessed RUL and will be discussed in prognostics section.

Both regular and hierarchical HMMs (as described in section III) were implemented for diagnostics and prognostics using data from nine drill-bits for training and three for testing and are discussed in the next sub-section. The application is implemented using MATLAB based on Kevin Murphy's Bayesian network toolbox, available at *http://www.cs.ubc.ca/~murphyk/Software/BNT/bnt.html* [34].

A. Health State Estimation with Competitive Learning of HMMs

As outlined in section III.A.2, in the competitive learning case, several HMMs (HMM_1, HMM_2, etc.), each of which is eventually expected to represent a distinct health-state, are created. A data sequence (i.e., 24 data points) for a random hole (among all training drill-bit data) was selected and the probability of obtaining this data sequence given the HMM model is calculated for all HMMs (1st basic problem mentioned in HMM section). Log-likelihood value is the log-probability of obtaining the data sequence given the HMM calculated using the Forward-Backward algorithm. In general, the resulting likelihood in HMM is represented as log-likelihood values, since the exact distribution of the likelihood ratio is very difficult to determine in testing nested hypotheses, as in [15], [24], [44]. The HMM with the highest log-likelihood value is labeled the 'winner', inferring that the drill-bit is in the health-state represented by the winner. The selected data sequence is used for further incremental training of the winning HMM alone. Standard training methods referred in section II were used. This unsupervised competitive learning method is expected to lead each health-state to be represented by a distinct HMM, since an HMM that learned similar data sequences will have higher likelihood values during the competition.

Initialization is an important issue for building HMMs for health-state estimation in terms of reducing training times and improving diagnostic performance. In this investigation, the first and last holes of all drill-bits were used for initialization of the 'first' and 'last' HMM models in the pool, respectively. Then, one hole from each drill-bit is selected in such a way that the selected holes are as far away from each other as possible to initialize the remaining HMMs. For example, with four HMMs in the competitive learning pool and data available from 22 holes from drill-bit #1, holes 1, 8, 15, and 22 were used to initialize HMM_1, HMM_2, HMM_3, and HMM_4, respectively. After initialization, HMMs were trained through pure competitive learning as explained in detail in section III.A.2. Data sequences from all training holes were presented in a random order to the HMM pool in each epoch of the competitive learning process. The learning process is terminated when reverse jumps (i.e., classification error) per epoch reach zero or when a pre-specified maximum number of epochs are reached.

Here is a note of caution regarding the selection of number of hidden-states for each HMM prior to competitive learning. In our experiments, we varied the number of hidden-states within each HMM by starting with two states and increasing it until the non-stationarity is captured adequately. As with most statistical models where measures of goodness-of-fit improve with model complexity, so is the case with HMMs. As we increase the number of hidden-states, so does the log-likelihood [44]. However, as is the case with any model, we prefer simpler models to complex models, justified by the principle of Occam's razor [45]. Thus, we need to evaluate the performance of HMMs by increasing the number of hidden-states and stop when the rate of improvement in the log-likelihood begins to diminish. If the number of hidden-states is increased too much, they can also overlap, compromising the distinction between the different sub-states, and in turn, the health-states. In our experiments, we did not see any significant improvement in performance beyond four hidden-states per HMM.

For the given datasets, best results based on the aforementioned evaluation criteria (i.e., *number of reverse health-state jumps*, *uniformity*, and *health-state resolution*) were obtained when the HMM pool was initialized with 4 HMMs. This selection process would of course vary from application to application and should be based on subject domain expertise and desired health-state resolution. In our current application of monitoring drill-bits, 4 distinct health-states are considered adequate, with the understanding that once trained, HMMs can be used for health-state diagnostics on-line to replace the drill-bit in a timely fashion (i.e., replacement at transition from the third health-state to the last health-state or failure state). After proper initialization, the competitive learning has been performed. As described in section III.A.1, each trained HMM can be fully characterized by $\lambda = (A, B, \pi)$, where, $A = \{\alpha_{i,j}\} \quad = P(X_t = i \mid X_{t-1} = j)$ denotes the state-transition matrix, $B = \{b_i(k)\} = P(O_t = k \mid X_t = i)$ the observation distribution given the state, and $\pi(i) = P(X_1 = i)$ the initial state distribution. All our studies adopted the standard Gaussian observation process assumption during modeling.

Given that each HMM is initialized with 4 hidden or sub-states, each trained HMM (supposedly representing a distinct health-state) yields 4 Gaussian distribution observation models (one for each hidden state). Fig. 12 plots the mean (denoted by '+' sign) and co-variance structure of the observation distributions given the hidden-states for all four of the health-state HMMs that underwent the competitive learning process, super imposed on data from the complete life history of drill-bit #1. The orientation of the co-variance structure ellipse is based on the covariance matrix, and is scaled to pass through the $\pm 1\sigma$ points on the two principal component axes. In each of the sub-plots, data sequences are also color coded based on their similarity to the distinct health-state HMMs (green for data sequences that closely match the particular health-state HMM and yellow for those with low similarity – meaning similar to one of the other three health-state HMMs).

As can be seen from Fig. 13, the thrust-force and torque signals in 2D space look similar to an ellipsoid. All sub-plots are showing data from all holes of drill-bit #1, however, with color coding. Holes represented (under competition) by each of the four distinct health-state HMMs are coded green in the sub-plots, with sup-plots (a), (b), (c), and (d) corresponding to health-states represented by HMM_1, HMM_4, HMM_3, and HMM_2 (from brand new state to failure) respectively. Each sub-plot also shows in red the centers as well as the constant density contours representing the covariance structures of the sub-health-states of the corresponding trained HMM. The location of the center of the ellipse corresponds to the mean vector of the observable bi-variate Gaussian density, and the major and minor axes of the constant density contours represent the eigen-vectors of the covariance matrix. For example, Fig. 13(a) shows the four centers and the covariance structure constant-density contours for HMM_1. As drill-bit wears, the blunt cutting edges increase the thrust-force and torque signal amplitudes. The green ellipsoid data pattern is small for HMM_1 (see Fig. 13a), whereas it moves out producing a bigger ellipsoid like pattern as the drill-bit gets closer to failure health-state. Tracking the location of the four sub-states of the four 4 distinct HMMs, there is evidence that the HMMs are attempting to represent different health-states or life stages of the drill-bit. The figure illustrates that the data moves from the interior to the exterior as the drill-bit is used, and sub-states of HMMs can represent this movement.

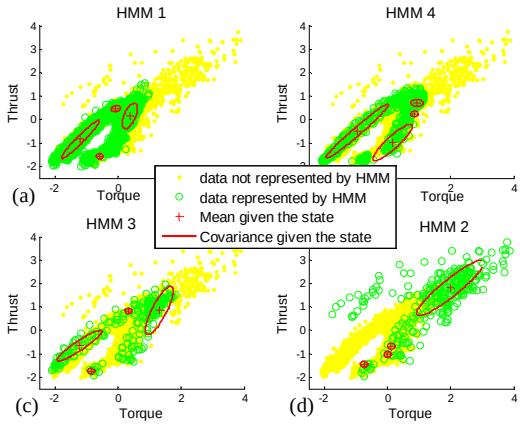


Figure 13: Mean and co-variance structure of sub-states of the four health-state HMMs superimposed on data from drill-bit #1.

After the HMMs were trained, health-states of drill-bits for all holes are identified hole by hole. The data for the corresponding hole is fed to all HMMs and the one with highest log-likelihood value is defined as the representative of the health-state, as illustrated in Fig. 5. The log-likelihood values of the four HMMs for drill-bit #1 are plotted in Fig. 14. The *x*-axis gives the life of the drill-bits in terms of holes and the *y*-axis gives the log-likelihood values for the four health-state HMMs. As can be seen from the figure, the log-likelihood values are highest for HMM_1, HMM_4, HMM_3, and HMM_2, as we go from brand new to failure states, respectively. Meaning, HMM_2 learned the 'failure state' during the competitive learning process, whereas HMM_1 learned the 'perfectly healthy state'. Note that we arbitrarily labeled the randomly initialized HMMs in order from 1 to 4 prior to the competitive learning process. This order, as indicated, is arbitrary and has no impact on the final solution.

What matters is the result of the competitive learning process. As indicated above, it is the log-likelihood plots from the training and testing cycles (past competitive learning) that reveal the true order of the health-states and the corresponding HMMs. It is not important if the failure state is represented by HMM_1 or HMM_2 as long as it is always represented by the same HMM both during training phase and the testing phase, indicating successful learning.

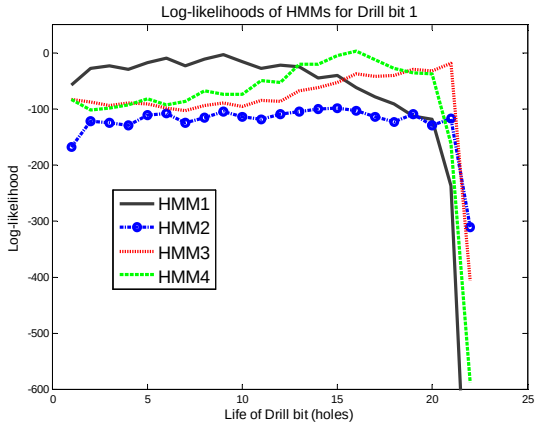


Figure 14: Log-likelihood trajectories of the four health-state HMMs past competitive learning for data from drill-bit #1.

The health-state estimation results from competitive learning are given in Table 1 for all twelve drill-bits. Each drill-bit is represented in a row (total of 12 rows for 12 drill-bits) with various numbers of successfully drilled holes represented in columns. An empty cell denotes that the corresponding drill-bit in the row failed before reaching the corresponding hole in the column. The number in each cell denotes the HMM number that represents the health of the drill-bit in the corresponding row and hole in the corresponding column. HMM_1 represents the 'good' health-state and HMM_4, HMM_3, and HMM_2 represents health-states closer to failure respectively, HMM_2 representing the health-state just before the failure-state. Different colors (gray levels in black-white print) are used for ease of visualization. As can be seen from the table, the trained HMMs were successful in representing the health-states without yielding any reverse jumps, and in addition, majority of drill-bits visited or underwent all the distinct health-states.

In general, the number of HMMs within the competitive learning pool and the number of hidden-states allowed within each HMM should be optimized. Too many or too few HMM hidden-states lead to data over-fitting (hence poor performance on the testing set) or poor performance (even within the training set), respectively [15]. For our drill-bit monitoring application, it was noticed that when less than four states are used, the reverse jumps increase (i.e., classification accuracy decreases), whereas when more than four HMMs are used, some of the HMMs either couldn't win any competition or tended to represent a part of a health-state. For example, in the case of 5 HMMs used for training, both HMM 3 and 5 represented the final failure state (i.e., the last hole of all drill-bits).

Table 1: Health-state estimation/labeling results for data from all holes of all drill-bits using HMM pool competitive learning.
(1: brand new, 4: used, 3::excessively used, 2: close to failure)

Holes		1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19	20	21	22	23	24
Drill-bits	1	1	1	1	1	1	1	1	1	1	1	1	1	4	4	4	4	4	4	3	3	3	2		
	2	1	1	1	1	1	1	1	1	1	1	1	1	4	4	4	4	3	2						
	3	1	1	1	1	1	1	1	1	1	1	1	1	4	3	2									
	4	1	1	1	1	1	2																		
	5	1	1	1	1	1	1	1	1	1	1	4	4	2											
	6	1	1	1	1	1	1	2																	
	7	1	1	1	1	1	1	1	1	1	1	4	4	3	2										
	8	1	1	1	1	1	1	1	1	1	1	1	1	1	4	4	4	4	4	4	4	3	3	2	
	9	1	1	1	1	1	1	1	1	1	1	1	1	1	4	3	2								
	10	1	1	1	1	1	1	1	1	2	2														
	11	1	1	1	1	1	1	1	4	2															
	12	1	1	1	1	1	1	1	1	4	4	4	4	4	4	3	3	2							

B. Hierarchical Hidden Markov Model for Health State Estimation

Hierarchical HMM (HHMM) is designed to handle complex systems. We implemented a two-level HHMM with top-level states that represent health-states with sub-states representing the non-stationarity within the hole. HHMM gives us the opportunity to model all health-states by using a single overall model (it is enough to train one HHMM instead of employing competitive learning for a pool of HMMs). In addition, the top-level state transition probabilities make the modeling of transitions between health-states possible, leading to RUL estimation, something not directly possible when employing a HMM pool. Health-states in HMM pool are represented by distinct HMMs, in which, transition from a health-state to another (a HMM to another one) cannot be modeled.

In the designing phase of DBN, we set a two-level hierarchical structure: one level for health-states, other for non-stationarity within the hole. The conditional probabilities are defined as in Eqns. (5)-(10). Note that the health-state of the drill-bit should go forward (i.e., deteriorate) not backwards (e.g., transition from 'torn out' state to 'brand new' state is not allowed). Thus, we implemented a 'left-to-right' HHMM, which allows only forward state transitions in order to represent health-state progression, with appropriate initial transition probabilities.

The mean and variance parameters $(\mu_{i,j}, \sigma^2_{i,j})$ of the observation distribution given the hidden states (see Eqn. (10)), representing thrust-force and torque signals, are learnt during training by passing training data as in Fig. 10.a. Number of mean and variance parameters to be learned for the different observation distributions is the product of number of health-states and number of sub-states under each health-state. For example, if 4 health-states are identified with 5 sub-states under each health-state, there will be 20 mean vectors and 20 covariance matrices. Fig. 14 illustrates the projection of mean and covariance matrices of observation state given 4 health-states and 5 sub-states within the HHMM. Each graph in the figure (total 4) represents a health-state. Five mean and contour plots are displayed in each graph representing the sub-states under given health-state. As can be seen from the figure, not unlike the HMM pool case, the data moves from interior to the exterior as drill-bit is used and sub-states of HHMM can effectively represent this movement.

In addition to mean and variance parameters of observed variables, the conditional probabilities (Eqns. (5)-(9)) are also learnt during training. Then, the top state probabilities given the data sequence of holes are obtained by inference as in Fig. 10.b. Most likely top state sequence is calculated as the health-states given the observation sequence (2nd basic problem mentioned in HMM section III.A.1). Fig. 15 displays the likelihood values for HHMMs with four states at the top-level for a drill-bit. As can be seen from the figures, using the one trained HHMM, the health-states can be identified as 'brand new', 'used', and 'excessively used' and 'close to failure' states. The probabilities are more distinguishable compared to regular HMMs with competitive learning displayed in Fig. 13.

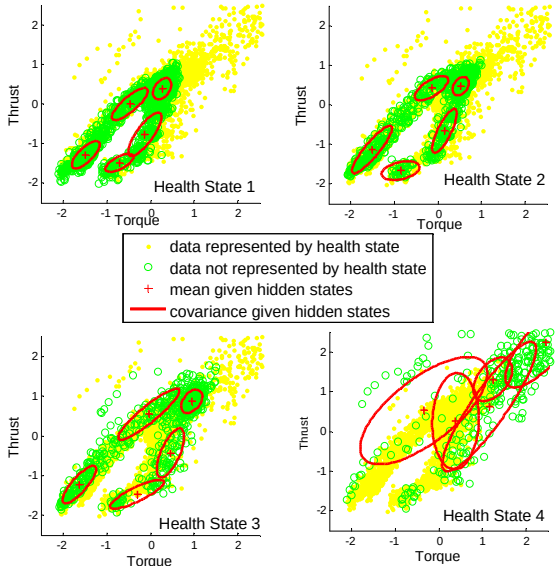


Figure 15: Mean and co-variance structure of sub-states of HHMM superimposed on data from drill-bit #1.

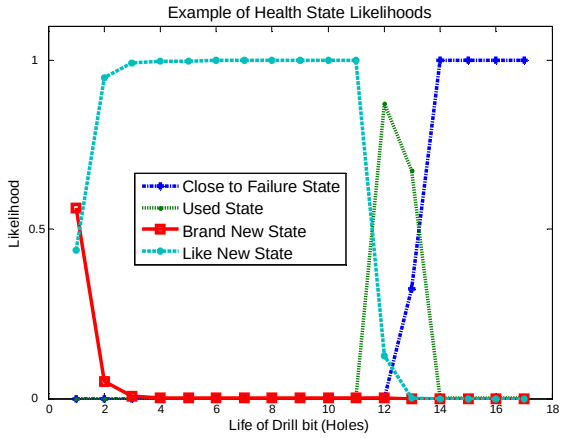


Figure 16: Log-likelihood trajectories of top-level (health) states #4 of HHMM for data from drill-bit #12.

The health-state of a drill-bit for a given hole data is identified as the top state of the HHMM with highest likelihood value. Table 2 gives the health-state estimation of all 12 drill-bits displayed in rows and various numbers of holes shown in columns for HHMM with five health-states. Number in each cell represents the top level state value (health-state) in the corresponding drill-bit and hole. Health-state #4 is represented only in three drill-bits. Thus, health-states #4 and #5 can be combined into one health-state. As can be seen from the table, health-state estimation results seem better with HHMM when compared to the case of employing a committee of HMMs, since almost all drill-bits visit all the health-states. There are once again no reverse jumps.

Table 2: Health-state estimation/labeling results for data from all holes of all drill-bits using a HHMM (1: brand new, 2: like new, 3: moderately used, 4: excessively used, 5: close to failure)

Drill-bits \ Holes	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19	20	21	22	23	24
1	1	1	1	1	1	1	1	1	1	1	2	2	2	3	3	3	3	3	3	3	4	5		
2	1	1	1	1	1	1	1	1	2	2	2	2	2	2	3	3	3	5						
3	1	1	1	1	1	1	1	1	2	2	2	2	3	3	5									
4	1	1	1	1	2	3	5																	
5	1	1	1	1	1	1	1	1	2	2	2	3	5											
6	1	1	1	1	1	2	3	5																
7	1	1	1	1	1	1	1	2	2	2	2	3	3	5										
8	1	1	1	1	1	1	2	2	2	2	2	2	2	2	2	2	3	3	3	3	3	3	4	5
9	1	1	1	1	1	1	1	1	1	1	2	2	2	2	3	5								
10	1	1	1	1	1	1	1	2	3	5														
11	1	1	1	1	1	1	2	3	5															
12	1	1	1	1	1	2	2	2	2	2	2	2	3	3	3	4	5							

In trying to optimize the structure of HHMM, the general principle of Occam's razor is once again relevant. We tried different numbers of health-states from three through six. While we prefer high health-state resolution, as the number of top-level states (health-states) increases, some of the states might not represent distinct health-states. Therefore, we will choose the highest number of top-level states (health-states), each of which will be represented consistently within the life of the different drill-bits used for training.

The second parameter to be optimized is the number of sub-states, which identifies the number of states within a hole. In our application, experimental evaluation suggests that five sub-states are adequate to represent the data from a hole. This is optimized with trial-error starting with two and increased one-by-one. The maximum number of states with non-overlapping mean and co-variance structure contours is considered as optimum.

The training computational time for HHMM varied between 174 and 255 seconds depending on the number of states used in top and lower levels. In our experiments, this was comparable to regular HMM committees, which takes around 212 seconds. Careful comparison of HMMs trained using competitive learning for health-state estimation versus using a single hierarchical HMM revealed the following advantages for using HHMM:

1.) *Better Classification*: HHMM gives better health-state estimation than regular HMM (compare Tables 1 and 2).
2.) *Training*: No need for competitive learning. It is enough to train just one HHMM. Initialization of sub-state HHMs is not required for HHMM.
3.) *Topological Ordering*: It naturally forces topological structure, which minimizes reverse jumps.
4.) *Transition Probability*: HHMM automatically calculates the transition probabilities between health-states (i.e. top level states), which regular HMM pool

lacks. Transition probabilities between health-states are important for RUL calculation (i.e., prognostics).

The next sub-section discusses a case study of RUL estimation using HHMMs.

C. RUL Estimation Using HHMM

The following section builds on the results previously reported in Section IV. Here we employ the proposed Monte-Carlo approach to estimate on-line the RUL of drill-bits used on the CNC drilling machine. Here we define RUL as the potential number of holes to be successfully drilled before failure. All results reported in this study are based on Monte-Carlo simulation sample size of 10,000 (adequate given the MTTF of a brand new drill-bit under the stated operating conditions is under 20 holes). During each of the 10,000 simulation runs, next health-state is estimated based on the transition probabilities (say 0.5 to state #2, 0.3 to state #3, and 0.2 to state #4) by generating a uniformly distributed random number between 0 and 1. The location of the random number (0-0.5, 0.5-0.8, and 0.8-1.0) identifies the next state (#2, #3, and #4). This process is repeated considering the calculated next state as the current state until the failure-state is reached. Then, the number of transitions is counted as the RUL value, since each transition represents a hole. This is repeated for all samples yielding 10,000 RUL values.

Probability of RUL being a value (say r) is calculated as the ratio of number of r's obtained in RUL calculation to the simulation sample size (10,000 in our case).

RUL probability distribution is calculated for each hole of all twelve drill-bits assuming the corresponding hole as the most recent hole drilled. For illustration purposes, RUL probability distributions of drill-bit #2, which failed in the 17th hole, for several holes (holes 1, 3, 6, 9, 12, 13, 14, 15, 16) are given in Fig. 17. The *x*-axis displays the overall life of the drill-bit (in holes), which is the sum of the RUL estimate and the total number of holes already drilled (i.e., expected total number of holes to be drilled), and the *y*-axis displays probability. The arrow within each graph shows the actual life of the drill-bit, 17 for the given example (i.e., drill-bit #2). As illustrated in the graphs, the variance of RUL probability distribution decreases and the accuracy of RUL estimation increases as the drill-bit approaches failure. In all drill-bits, the proposed method successfully estimates the RUL as zero with 100% accuracy in the hole just prior to failure.

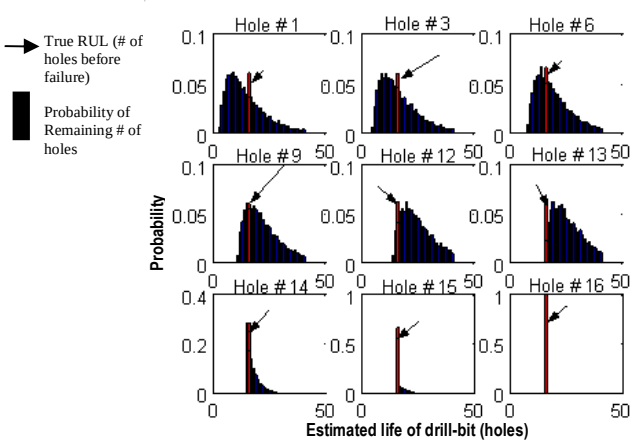
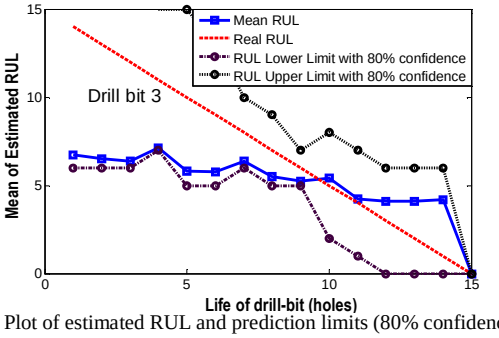
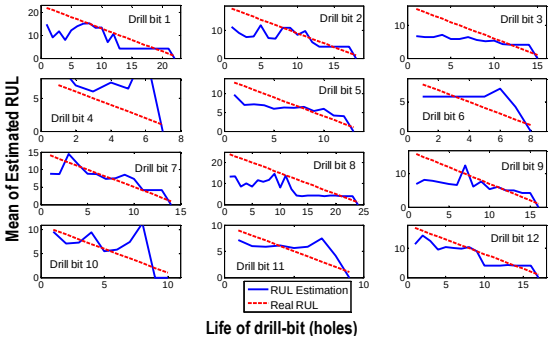


Figure 17: Estimated RUL probability mass function at different stages of the life of drill-bit #2.



(a) Plot of estimated RUL and prediction limits (80% confidence) along with true RUL for drill-bit #3



(b) Plots of estimated RUL along with true RUL for all 12 drill-bits

Figure 18: Estimated and true RUL for drill-bits.

Fig. 18.a reports the estimated RUL and prediction limits (80% confidence) along with true RUL for drill-bit #3. Fig. 18.b reports the estimated RUL and prediction limits (80% confidence) along with true RUL for all 12 drill-bits (each sub graph is for a drill-bit). The *x*-axis represents the time (hole) at which the RUL is estimated and the *y*-axis the estimated RUL. The dashed linear line is the true RUL value. The solid line is the mean of the estimated RUL values. As it is more effective to report the RUL as a range rather than a singular value, we also report the estimated prediction limits calculated with 80% confidence (outer dotted lines in Fig. 18.a). Prediction limit is the minimum range of RUL values that compose the confidence percentage of the whole sample.

The prediction limits are identified as shown in (12):

$$P(b \le r \le e) = \frac{\#(b \le r \le e)}{s} \ge cf \qquad (12)$$

where, r denotes the estimated RUL value, e and b denote the upper and lower prediction limits, respectively, cf the desired confidence, s the sample size (i.e., the total number of RUL estimations), and $\#(b \le r \le e)$ the number of RUL values between e and b. The width of the prediction limits is simply, $\ell = e - b$. Consider that there exist 10 RUL estimations (e.g., 7, 8, 6, 7, 7, 5, 6, 4, 6, 5) obtained from simulation. For 80% confidence, the width of the prediction limit is 7-5=2, since eight out of ten are within the prediction limit (7 and 5). To avoid clutter, the prediction limits for the 12 drill-bits are not displayed in Fig. 18.

Thus, the expected output from the proposed prognostic module includes prediction limits and confidence that leads to an expression such as "*RUL is between 4 and 6 holes with 95% confidence*". Given a certain confidence, the narrower the prediction limits (smaller its width) the higher the precision and the more useful the RUL estimation. Fig. 19 displays the width of prediction limit results of the proposed prognostic module. As expected, at the very early stages of drill-bit use, the precision is poor (i.e., width of the prediction limits is high). As the drill-bit approaches failure, the precision improves (i.e., width of the prediction limit decreases) and estimation becomes more valuable.

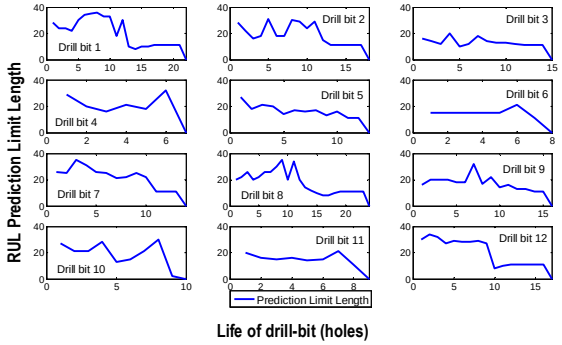


Figure 19: Width of RUL prediction limits for all twelve drill-bits based on 95% confidence.

In the proposed prognostic method, it is assumed that the health-state transition can only occur between holes. In other words, a drill-bit cannot be partially in one health-state and jump to the other health-state during the same hole. This is a reasonable assumption and necessary for RUL calculation, which leads to integer RUL values. In calculating width of the prediction limits, the target confidence is used as the minimum acceptable confidence threshold. For example, if the confidence values of RUL being between *'4 and 6 holes'* and *'4 and 7 holes'* are 93% and 97%, respectively, then the proposed method will choose the latter for calculating precision if the required confidence threshold is 95%.

While the above discussion pertains to the quality of RUL 'interval estimates' available from the Monte-Carlo procedure, one could also assess the quality of the 'point estimate' (i.e., the mean of the RUL frequency distribution). Accuracy, denoted acc, evaluates the quality of point estimate, and is calculated as the ratio of number of RUL estimations that exactly match the true RUL to the sample size of RUL estimations, as indicated in (10):

$$acc = P(r = actr) = \frac{\#(r = actr)}{s} \qquad (13)$$

where $actr$ denotes the true or actual RUL value, $\#(r = actr)$ denotes the number of RUL values that match $actr$, and $P(r = actr)$ the probability of r being equal to $actr$.

Consider that there exist 10 RUL estimations (e.g., 7, 8, 6, 7, 7, 5, 6, 4, 6, 5) obtained from simulation. If the real RUL value is 6, then the accuracy is 0.3, since there are 3 RUL estimates that match the true RUL over a set of 10 predictions.

Fig. 20 reports these accuracies for the drilling case study, which is basically the probability that the point estimate matches the actual RUL. It is evident from the plots that the estimation accuracy continuously improves as drill-bits approach failure. The two minor exceptions pertain to drill-bits #4 and #10, which only lasted 7 and 9 holes, respectively. Their short lives may be an indication of lack of consistent degradation prior to failure. However, note that the RUL is estimated as 0 with 100% confidence just prior to failure in all cases.

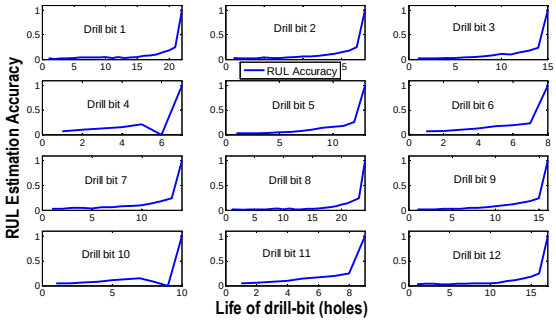


Figure 20: RUL estimation accuracy for all twelve drill-bits.

In order to compare the RUL estimation results with results from [12], circular cross validation procedure is employed. Twelve drill-bits are separated into training and testing data as illustrated in Table 3. Tables 4 and 5 display average R^2 and median RMSE (Root Mean Square Error) of the presented method. Table 6 compares the results with results in [12]. Note that columns in Tables 4 and 5 represent the drill-bit used as n^{th} training or testing drill-bit, not necessarily the n^{th} drill-bit in Table 3.

The results from employing methods 2 and 3 presented in [12] are reported as mean R-square and median RMSE in Table 6. The result of method 1 in [12] is reported as a figure that displays the transition from health-state good to medium and from medium to bad, not in the form of R-square and RMSE. Given that method 1 is seen to be inferior to methods 2 and 3 of [12], we limit our comparisons to methods 2 and 3.

Table 3: Circular cross-validation training and testing datasets

DB #	1	2	3	4	5	6	7	8	9	10	11	12	# Trn	# Tst	#Trn	#Tst
# Holes	22	18	15	7	13	8	14	24	7	10	9	17	DBs	DBs	Holes	Holes
Set 1	1	1	1	1	1	1	1	1	1	0	0	0	9	3	128	36
Set 2	0	1	1	1	1	1	1	1	1	1	0	0	9	3	116	48
Set 3	0	0	1	1	1	1	1	1	1	1	1	0	9	3	107	57
Set 4	0	0	0	1	1	1	1	1	1	1	1	1	9	3	109	55

Set 5	1	0	0	0	1	1	1	1	1	1	1	9	3	124	40
Set 6	1	1	0	0	0	1	1	1	1	1	1	9	3	129	35
Set 7	1	1	1	0	0	0	1	1	1	1	1	9	3	136	28
Set 8	1	1	1	1	0	0	0	1	1	1	1	9	3	129	35
Set 9	1	1	1	1	1	0	0	0	1	1	1	9	3	118	46
Set 10	1	1	1	1	1	1	0	0	0	1	1	9	3	119	45
Set 11	1	1	1	1	1	1	1	0	0	0	1	9	3	123	41
Set 12	1	1	1	1	1	1	1	1	0	0	0	9	3	138	26

Table 4: Average R^2 for training and testing sets

	Training Set									Testing Set		
DB #	1	2	3	4	5	6	7	8	9	1	2	3
Set 1	0.82	0.88	0.81	0.94	0.94	0.84	0.91	0.84	0.83	0.93	0.90	0.91
Set 2	0.88	0.91	0.88	0.92	0.94	0.91	0.95	0.92	0.83	0.94	0.90	0.90
Set 3	0.96	0.86	0.91	0.87	0.90	0.96	0.94	0.94	0.90	0.83	0.86	0.97
Set 4	0.96	0.92	0.92	0.89	0.84	0.96	0.93	0.92	0.95	0.95	0.77	0.92
Set 5	0.98	0.96	0.96	0.87	0.90	0.92	0.92	0.96	0.89	0.97	0.82	0.82
Set 6	0.87	0.97	0.93	0.90	0.88	0.84	0.89	0.89	0.97	0.90	0.89	0.91
Set 7	0.94	0.86	0.88	0.98	0.92	0.85	0.90	0.93	0.91	0.90	0.90	0.99
Set 8	0.94	0.89	0.83	0.83	0.95	0.95	0.89	0.88	0.87	0.91	0.91	0.92
Set 9	0.96	0.97	0.92	0.77	0.96	0.97	0.97	0.91	0.84	0.93	0.92	0.95
Set 10	0.95	0.90	0.97	0.93	0.83	0.98	0.98	0.89	0.91	0.88	0.86	0.91
Set 11	0.90	0.94	0.92	1.00	0.94	0.75	0.95	0.97	0.99	0.88	0.86	0.85
Set 12	0.89	0.86	0.94	0.89	0.93	0.97	0.73	0.96	0.97	0.90	0.91	0.86

Table 5: Median RMSE for training and testing sets

	Training Set									Testing Set		
DB #	1	2	3	4	5	6	7	8	9	1	2	3
Set 1	4.17	3.33	1.75	6.38	2.21	6.15	0.79	4.85	6.71	4.48	4.65	2.80
Set 2	2.64	4.28	3.17	1.72	6.79	2.09	6.41	1.29	5.61	6.75	4.37	4.95
Set 3	4.05	3.29	4.81	3.2	2.05	5.11	0.81	4.62	1.59	6.79	5.50	3.18
Set 4	2.79	3.53	3.30	5.36	3.48	2.45	4.97	1.37	4.06	1.70	6.32	4.98
Set 5	6.25	4.04	4.84	3.19	4.71	3.33	2.16	6.00	1.10	5.46	0.86	6.12
Set 6	5.63	6.94	4.70	5.70	2.93	4.34	3.01	1.10	6.69	2.05	5.81	1.09
Set 7	2.93	4.62	8.63	5.82	6.83	2.09	4.17	2.32	1.30	8.08	3.36	7.86
Set 8	6.18	1.25	5.79	7.00	4.60	5.86	2.48	4.37	3.41	1.24	6.89	2.38
Set 9	1.83	6.10	1.85	5.55	6.75	4.49	4.98	2.65	4.49	3.00	1.30	6.76
Set 10	5.05	1.42	4.36	1.55	6.81	5.05	3.06	3.59	2.73	5.92	3.15	2.52
Set 11	2.28	4.95	1.63	4.32	1.72	7.18	5.20	2.57	3.59	2.89	5.80	3.43
Set 12	3.35	1.93	5.68	0.70	4.99	1.79	5.78	5.76	4.01	3.53	2.96	5.04

Table 6: Comparison of presented method with methods from [12]: TR: Training, TST: Testing

		R Square			RMSE	
		average	worst	best	median	
HMM in [12]	Method1	N/A	N/A	N/A	N/A	
	Method2	0.44	0.24	0.73	6.28	
	Method3	0.53	0.3	0.79	5.74	
Presented method	HHMM	TR 0.91	TST 0.90	0.73	0.996	TR 3.78 TST 4.03

As can be seen from the tables above, the proposed method highly outperforms the HMM models presented in [12]. Even the worst R-square result from the presented method is better than the average R-Square result of methods from [12]. Note however that R-square value equal to 1 does not necessarily indicate 100% accuracy and 100% confidence (mentioned before in eq. 12 and 13), but just indicative of the linear correlation between two prediction data sets. This can be seen in Set 11 and training drill-bit 4 (11^{th} row and 4^{th} column in Table 4 and 5) as R-square value being close to 1 and median RMSE value being 4.3 holes.

Besides diagnostic performance, RUL estimation accuracies (both point as well as interval estimates) should also influence the optimization of the HHMM structure (i.e., number of health-states and sub-states) to be used in HHMM. For the current case study, best results are obtained with four health-states and four sub-states for each health-state. Overall, from the results, we can conclude that the proposed HHMM approach seems rather effective in supporting diagnostics and prognostics needs of the drilling process.

V. Conclusion and Future Research

We implemented regular and hierarchical HMMs as dynamic Bayesian networks (DBNs) for health-state and remaining-useful-life (RUL) estimation. Casting HMMs as DBNs offers compact representation as well as additional flexibility with respect to the structure of the model. Competitive learning is employed for training pools of regular HMMs, each of which represents a distinct health-state. Although regular HMMs seem to give reasonable diagnostic accuracy, if diagnostic accuracy is defined simply as a function of number of reverse jumps among the health-states, they pose implementation difficulties such as long training times and the inability to facilitate calculation of RUL. On the other hand, a single hierarchical HMM can naturally represent all health-states and offer several advantages over pools of standard HMMs, such as better diagnostic accuracy and ease of implementation and training. Additionally, hierarchical HMM also directly captures health-state transition probabilities, which are a prerequisite for health-state prognostics (i.e., RUL estimation). Also discussed was the method used for RUL estimation that employs Monte Carlo simulation with HHMM. The proposed methods are implemented for monitoring drill-bits on a CNC machine. The results are very promising for development of effective methods for diagnostics and prognostics (in particular RUL estimation), even in the absence of a history of labeled examples or cases. The proposed method also significantly outperforms other HMM based methods from the literature. Future research will attempt to explicitly introduce state-duration densities into the HHMM model to overcome the need for the assumption of exponential duration density. In addition, HHMM structures that are more flexible will be attempted to facilitate the monitoring of systems that exhibit multiple and sometimes interacting failure modes.

Acknowledgements

The authors thank the editors and referees for their careful review and feedback, which substantially improved this paper.

References

[1] F. Camci, Autonomous, Process Monitoring, Diagnostics, and Prognostics Using Support Vector Machines and Hidden Markov Models, *PHD Dissertation*, Wayne State University, 2005

[2] Harbor Research Pervasive Internet Report, Approaching Zero Downtime: The Center for Intelligent Maintenance Systems, April 2003

[3] C. Kwan, X. Zhang, R. Xu, L. Haynes, A novel approach to fault diagnostics and prognostics, *Proceedings of ICRA '03: IEEE International Conference on Robotics and Automation*. 1(3), September 2003, 604-609

[4] NIST-ATP CBM Workshop Report, *NIST-ATP Workshop on Condition-Based Maintenance*, Atlanta, GA, November 1998 (http://www.atp.nist.gov/www/cbm/cbm_wp1.htm)

[5] K. Maynard, C. S. Byington, G. W. Nickerson, and M. V. Dyke, Validation of Helicopter Nominal and Faulted Conditions Using Fleet Data sets, *Proceedings of the International Conference on Condition Monitoring*, UK, 1999 129 –141

[6] S. J. Engel, B. J. Gilmartin, K. Bongort, A. Hess, Prognostics, The Real Issues Involved With Predicting Life Remaining, *IEEE Aerospace Conference Proceedings*, 6, 2000, 457-469.

[7] M.J. Roemer and G.J. Kacprzynski, Advanced diagnostics and prognostics for gas turbine risk assessment, *Proceedings of the 2000 IEEE Aerospace Conference*, Big Sky, Montana, March 18-25, 2000.

[8] I. Sanches, Noise-compensated hidden Markov models, *IEEE Transactions on Speech and Audio Processing*, 8(5), Sep 2000, 533-540

[9] B.D. Womack, J.H.L. Hansen, N-channel hidden Markov models for combined stressed speech classification and recognition, *IEEE Transactions on Speech and Audio Processing*, 7(6), Nov 1999, 668-677

[10] C. Bunks, D. McCarthy, T. Al Ani, Condition-based maintenance of machines using hidden Markov model, *Mechanical Systems and Signal Processing*, 14(4), July 2000, 597-612

[11] R. Dickman, R. Vidigal, Quasi-stationary distributions for stochastic processes with an absorbing state, *Journal of Physics A: Mathematical and General*, (35), 2002, 1147-1166

[12] R. B. Chinnam, P. Baruah, Autonomous diagnostics and prognostics in machining processes through competitive learning-driven HMM-based clustering, *International Journal of Production Research*, 1-20, in press, iFirst, doi: 10.1080/00207540802232930

[13] M. Dong, D. He, A segmental hidden semi-Markov model (HSMM) -based diagnostics and prognostics framework and methodology, *Mechanical Systems and Signal Processing*, 21 (2007), 2248-2266

[14] M. Dong, A novel approach to equipment health management based on auto-regressive hidden semi-Markov model (AR-HSMM), *Science In China Series F-Information Sciences* , 51(9) (2008), pp. 1291-1304

[15] R. B. Chinnam, P. Baruah, Autonomous diagnostics and prognostics through competitive learning driven HMM-based clustering, *Proceedings of the International Joint Conference on Neural Networks*, July 2003, 2466- 2471

[16] P. Baruah, P and R.B. Chinnam, R.B., HMMs for diagnostics and prognostics in machining processes, *International Journal of Production Research*, 43(6), March 2005, 1275-1293

[17] L. Fu, Ling S., Neural Network Based Online Detection of Drill Breakage in Micro Drilling Process, *Proc. 9th International Conference on Neural Information Processing (ICONIP'02)*, 2002 pp. 2054-2058

[18] H. M. Ertunc, K. A. Loparo, E. Ozdemir, H. Ocak, Real Time Monitoring of Tool Wear Using Multiple Modeling Method, *Proc. IEEE International Conference Electric Machines and Drives*, 2001. IEMDC 2001. pp. 687- 691

[19] R. J. Furness, T. Tsao, J. S. Rankin, M. J. Muth, and K. W. Manes Torque control for a form tool drilling operation, *IEEE Transactions on Control Systems and Technology*, Vol. 7, 1999, pp. 22-30

[20] K. Subramanian and N. H. Cook, Sensing of drill wear and prediction of drill life, *ASME Journal of Engineering for Industry*, Vol. 99, pp. 295-301, 1977

[21] S. Y. Hong, Knowledge-based diagnosis of drill conditions, *Journal of Intelligent Manufacturig*, vol. 4, pp. 233-241 1993

[22] E. Jantunen, A summary of methods applied to tool condition monitoring in drilling, *International Journal of Machine Tools & Manufacture*, 42 (2002), pp. 997-1010

[23] S. Haykin, *Neural networks : A comprehensive foundation*, 2nd Ed., Upper Saddle River, New York: Prentice Hall, 1999.

[24] R. Lawrence, A tutorial on Hidden Markov Models and Selected Applications in Speech Recognition, *Proceedings of IEEE* 77(2), 1989, 257-286

[25] J. Blimes., Data-driven extensions to HMM statistical dependencies. *International Conference on Speech Language Processing*, 1998.

[26] J. Blimes., *What HMMs can do*, Technical Report: UWEETR-2002-03, 2002.

[27] A.B. Poritz., *Hidden Markov Models: A Guided Tour*, ICASSP, 1988

[28] S. Fine, Y. Singer, and N. Tishby, The hierarchical hidden Markov model: Analysis and applications, *Machine Learning* 32, 1998, 41-62.

[29] Wikipedia Contributors, Hierarchical hidden Markov model, *Wikipedia, The Free Encyclopedia*, 8 April 2008, 20:59 UTC, <http://en.wikipedia.org/w/index.php?title=Hierarchical_hidden_Markov_model&oldid=204300061> [accessed 8 April 2008]

[30] P. Smyth, Belief networks, hidden Markov models, and Markov random fields: a unifying view, *Pattern Recognition Letters* 18(11-13), 1998, 1261-1268

[31] S. Roweis & Z. Ghahramani, A Unifying Review of Linear Gaussian Models, *Neural Computation* 11(2), 1999, 305-345

[32] T. Dean and K. Kanazawa, A model for reasoning about persistence and causation. *Artificial Intelligence*, 93(1–2), 1–27, 1989

[33] K. Murphy, Dynamic Bayesian Network: Representation, Inference, and Learning, *PhD. Dissertation*, University of California, Berkeley, 2002.

[34] K. P. Murphy, The Bayes Net Toolbox for Matlab, *Computing Science and Statistics*, 33, 2001.

[35] C. Begg, T. Merdes, C.S. Byington, and K.P. Maynard, Mechanical System Modeling for Failure Diagnostics and Prognosis, *Maintainability and Reliability Conference (MARCON 99)*, Gatlinburg, May 1999

[36] G. J. Kacprzynski et al., Enhancement of Physics-of-Failure Prognostic Models with System Level Features, *Proceedings of the 2000 IEEE Aerospace Conference*, Big Sky, MT, March 2002.

[37] C. Joseph Lu and William Q. Meeker, Using Degradation Measures to Estimate a Time-to-Failure Distribution, *Technometrics*, 35(2), 1993, pp. 161-174

[38] J. Lawless and M. Crowder, Covariates and Random Effects in a Gamma Process Model with Application to Degradation and Failure, *Journal Lifetime Data Analysis*, 10(3), (2004), pp. 213-227

[39] C.S. Byington, M.J. Roemer, and T. Galie, Prognostic Enhancements to Diagnostic Systems for Improved Condition-Based Maintenance, *Proceedings of the 2000 IEEE Aerospace Conference*, Big Sky, MT, March 2002.

[40] C.S. Byington and A.K. Garga, Data Fusion for Developing Predictive Diagnostics for Electromechanical Systems, *Handbook of Multisensor Data Fusion*, D.L. Hall and J. Llinas eds., CRC Press, FL: Boca Raton, 2001.

[41] J.P. Cusumano, D. Chelidze, and N.K. Hecht, Using phase space reconstruction of tract parameter drift in a nonlinear system, *Proceedings of the ASME 16th Biennial Conference on Mechanical Vibrations and Noise, Symposium on Time-Varying Systems and Structures*, September 14-17, 1997.

[42] A. Ray, S. Tangirala, Stochastic Modeling of Fatigue Crack Dynamics for Online Failure Prognostics, *IEEE Transactions on Control Systems Technology*, 4(4), July 1996 443-451.

[43] Zhengguo Xu Yindong Ji Donghua Zhou, Real-time Reliability Prediction for a Dynamic System Based on the Hidden Degradation Process Identification, *IEEE Transactions on Reliability*, 57(2), 2008, pp. 230-242

[44] L.R. Rabiner, A tutorial on hidden Markov models and selected applications in speech recognition, *Proceedings of IEEE*, 1989, 77, 257-285.

[45] D.J.C. MacKay, *Information Theory, Inference and Learning Algorithms*, 2003 (Cambridge University Press).

Fatih Camci is an Assistant Professor in the Department of Computer Engineering at Fatih University, Istanbul Turkey. He worked as senior project engineer at Impact Technologies in Rochester, NY for two years with experience in the development of diagnostic/prognostic strategies for naval ship systems and military aircraft applications. Camci received his PhD in Industrial Engineering from Wayne State University. He received his B.S. degree in Computer Engineering from Istanbul University (Turkey) and M.S. degree in Computer Engineering from Fatih University (Turkey). He has been involved in development of optimum maintenance scheduling tools and diagnostics/prognostic methods for naval ship systems. His expertise includes computational intelligence methods such as neural networks, fuzzy logic, support vector machines, etc, and optimization methods such as linear, quadratic optimization, and genetic algorithms, etc.

Ratna Babu Chinnam is an Associate Professor in the Department of Industrial & Manufacturing Engineering at Wayne State University (U.S.A.). He received his B.S. degree in Mechanical Engineering from Mangalore University (India) in 1988 and the M.S. and Ph.D. degrees in Industrial Engineering from Texas Tech University (U.S.A.) in 1990 and 1994, respectively. He is the author of over 75 technical publications in the areas of Intelligent Quality Engineering,

Condition-Based Maintenance Systems, Operations Management, and Computational Intelligence. He is currently the associate editor for the *International Journal of Modeling and Simulation*. He regularly serves on several international conference planning committees such as IJCNN, ANNIE, and IASTED's NIC. His past research is mostly funded by such agencies as the National Science Foundation. He carried out extensive collaborative research with Ford Motor Company, DaimlerChrysler, Chrysler, General Dynamics, and consulted for such companies as Sirius, Energy Conversion Devices, and Tecton. He is a member of Alpha Pi Mu, INFORMS, and the North American Manufacturing Research Institute.