UNIVERSITY OF CALIFORNIA,
IRVINE

Neuroscience Inspired Algorithms for the Predictive Maintenance of Manufacturing Systems

THESIS

submitted in partial satisfaction of the requirements
for the degree of

MASTER OF SCIENCE

in Computer Engineering

by

Arnav Vaibhav Malawade

2021

DEDICATION

To my parents, Vaibhav and Mathangi, for their unwavering love and support. To my sister, Krupa, for always being there for me. To my friends for their encouragement and inspiration. To Tessa, for her love, patience, and belief in me.

TABLE OF CONTENTS

LIST OF FIGURES

LIST OF TABLES

ACKNOWLEDGMENTS

I would like to thank Professor Al Faruque, Professor Khargonekar, and Professor Shoukry for their feedback, mentorship, and support.

I would also like to thank the National Science Foundation for their funding assistance via awards CMMI-1739503 and ECCS-1839429 as well as Graduate Assistance in Areas of National Need (GAANN) for their funding assistance via award P200A180052.

The text of this thesis/dissertation is a reprint of the material as it appears in “Neuroscience-Inspired Algorithms for the Predictive Maintenance of Manufacturing Systems”, in IEEE Transactions on Industrial Informatics 2021 [2]. The co-authors listed in this publication directed and supervised research which forms the basis for the thesis.

ABSTRACT OF THE THESIS

Neuroscience Inspired Algorithms for the Predictive
Maintenance of Manufacturing Systems

By

Arnav Vaibhav Malawade

Master of Science in Computer Engineering

University of California, Irvine, 2021

Professor Mohammad Al Faruque, Chair

If machine failures can be detected preemptively, then maintenance and repairs can be performed more efficiently, reducing production costs. Many machine learning techniques for performing early failure detection using vibration data have been proposed; however, these methods are often power and data-hungry, susceptible to noise, and require large amounts of data preprocessing. Also, training is usually only performed once before inference, so they do not learn and adapt as the machine ages. Thus, we propose a method of performing online, real-time anomaly detection for predictive maintenance using Hierarchical Temporal Memory (HTM). Inspired by the human neocortex, HTMs learn and adapt continuously and are robust to noise. Using the Numenta Anomaly Benchmark, we empirically demonstrate that our approach outperforms state-of-the-art algorithms at preemptively detecting real-world cases of bearing failures and simulated 3D printer failures. Our approach achieves an average score of 64.71, surpassing state-of-the-art deep-learning (49.38) and statistical (61.06) methods.

Chapter 1

Introduction

Predictive Maintenance (PM) is an emerging new paradigm in manufacturing where symptoms of machine degradation are detected before failures occur. It is a major part of the Industry 4.0 and smart manufacturing vision. Using sensor readings, process parameters, and other operational characteristics, PM can help maximize tool life by reducing the number of unnecessary repairs performed while also reducing the likelihood of unexpected failures [3]. In the United States alone, improper maintenance and the resulting outages cost more than 60 billion dollars per year [4]. Thus, smart data-driven paradigms such as PM have the potential to reduce industrial production costs significantly.

Recently, many statistical, machine learning (ML), and deep learning (DL) techniques for PM have been proposed. However, these methods are not without their shortcomings: statistical methods require extensive domain knowledge and often do not generalize well to more complex use cases, while DL and ML techniques often require large amounts of training data and are susceptible to increased error as machines age over time. Furthermore, ML and DL algorithms are highly susceptible to noise, making them insufficiently robust for industrial settings without data preprocessing. Due to the high noise level and diversity among industrial systems, PM models that do not require significant preprocessing or domain knowledge are considered more practical [5].

To overcome these issues, we propose the use of a learning algorithm inspired by neuroscience called **Hierarchical Temporal Memory (HTM)**, pioneered by Hawkins and Blakeslee [6]. Using binary *sparse distributed representations* (SDRs) to represent data and an architecture incorporating feed-forward, lateral, and feedback connections, HTMs emulate the interactions between pyramidal neurons in the neocortex. HTMs are *online learning* algorithms that require less application-specific tuning, are robust to noise, and adapt to variations in the data as they continuously learn. In practice, this means HTMs can efficiently learn from a single training pass over small training datasets with little to no hyperparameter tuning. These characteristics also enable HTMs to learn in near real-time. For these reasons, they are suitable for practical applications such as detecting early symptoms of failure in manufacturing equipment. In this work, we demonstrate the effectiveness of an HTM-based anomaly detection methodology at detecting these symptoms in roller-element bearings and 3D printers.

1.1 Related Work

We focus on the specific task of PM on roller-element bearings due to their broad application and utility in manufacturing. We also evaluate Additive Manufacturing (AM) as it is a modern technique that presents unique challenges due to the dynamics of 3D printers. Here, we briefly discuss works related to PM for roller bearings and additive manufacturing.

Many PM methods use statistical models due to their simplicity and explainability. These approaches rely on extracted time and frequency domain features. For example, the energy entropy mean and root mean squared (RMS) values of wavelets were used to diagnose ball bearing faults in [7]. In another example, the spectral kurtosis (SK) of vibration and current signals was used to detect and classify the surface roughness of ball bearings in [8]. Using a particle filter method, Zhang et al. performed fault detection on bearings similar to those found in helicopter oil cooler fans [9].

In addition to statistical methods, ML techniques have been applied to a wide array of industrial prognosis tasks. One such method: AutoRegressive Integrated Moving Average (ARIMA), is one of the most popular techniques for time-series forecasting and was used to predict failures and identify quality defects in a slitting machine in [10]. In another approach, Tobon-Mejia et al. used Mixture of Gaussians HMMs and Wavelet Packet Decomposition to estimate the Remaining Useful Life (RUL) of roller-element bearings [11].

DL methods such as Long Short-Term Memory (LSTM) Networks and Convolutional Neural Networks (CNNs) have also been used extensively for PM. In one example, Feng et al. used an LSTM for detecting anomalies in industrial control systems [12]. Additionally, an RNN-LSTM was used to perform PM on an air booster compressor motor used in oil and gas equipment in [13].

Due to the increased complexity and relatively late adoption of AM systems, PM techniques for AM have not been studied in great detail. Proposed approaches often draw from research in related applications, such as PM for bearings. For example, Yoon et al. evaluated the feasibility of AM equipment fault diagnosis using a piezoelectric strain sensor and an acoustic sensor. In this work, features such as RMS value, kurtosis, skewness, and crest factor were used to detect faults [14]. Deep learning has also been used for AM anomaly detection, such as in [15] where a neural network was used to classify faults in 3D printer vibration data.

Despite the proliferation of statistical, ML, and DL approaches to PM for manufacturing, to the best of our knowledge, no HTM-based solutions have been proposed. However, the structural and temporal properties of HTM algorithms allow them to excel at cross-domain tasks that apply to manufacturing, such as anomaly detection [16]. Since the core objective of PM in manufacturing is detecting early symptoms of part failure, HTMs are a natural candidate for this task. HTMs were shown to match or surpass neural networks at detecting and classifying foreign materials on a conveyor belt in a cigarette manufacturing plant [17]. HTMs have also proven effective at detecting anomalies in crowd movements [18], traffic patterns [19], human vital signs [20], electrical grids [21], and computer hardware [22].

1.2 Research Challenges

Overall, PM for manufacturing presents the following key research challenges:

1. Identifying time-series anomalies in near real-time despite ambient noise.
2. Learning efficiently from small training datasets to improve applicability to practical use cases.
3. Developing a solution that can be generalized to many heterogeneous manufacturing systems without requiring extensive domain-specific tuning.
4. Adapting to changes in data statistics (i.e., machine aging).

Despite the successes achieved by existing methods in the aforementioned applications, industrial manufacturing systems are diverse and complex, making it difficult to find solutions that generalize across applications. Consequently, PM systems require specialization, which necessitates specialized knowledge and cross-domain skills. This is especially true in the case of bearing-failure prognosis, as bearing design and lifetime management lies squarely in the mechanical and materials engineering domains.

It is difficult for any single technique to address all these research challenges effectively. For example, statistical methods such as thresholding based on kurtosis or spectral analysis are highly efficient and real-time capable but require explicitly defined health indicators and thresholds, which are machine- and application-specific. Also, stationary methods including RMS, kurtosis, and crest factor are only effective for stationary signals (signals with time-invariant statistical properties), but bearing vibration signals are generally cyclostationary (statistical properties vary cyclically) or non-stationary (statistical properties change depending on speed and load conditions)[23]. Spectral kurtosis is applicable to non-stationary and non-periodic signals but is sensitive to noise and outliers [24].

Classical ML algorithms such as AR Models, Support Vector Machines, Hidden Markov Models (HMM), Random Forests, and k-Nearest Neighbors have been demonstrated for PM in existing work, but require the extraction of explicit health indicators (features) from data [25]. These algorithms also require application-specific hyperparameter tuning, data preprocessing as they have poor noise robustness [5], and regular updates of model settings as they do not adapt to account for machine aging [25]. Moreover, both HMM and AR methods are ineffective on non-stationary signals [23].

In DL algorithms such as neural networks and LSTMs, health indicators can be learned implicitly by the network. However, a network trained for one machine cannot generalize to a new machine without retraining with a large amount of data for hundreds or thousands of epochs. Larger models may be able to generalize better, but the complexity of training and optimizing these models increases drastically with size [25]. This domain-specific training and tuning process can be expensive, time-consuming, and impractical for real-world use cases. Like the ML methods, DL algorithms also have poor noise robustness [26] and require high-quality data, or else performance can suffer significantly [5]. To address this, significant preprocessing steps are often needed to generate clean data for these models [5].

As stated in Section 1.1, HTM-based anomaly detection methods have demonstrated success in several distinct fields. However, to the best of our knowledge, no prior work has comprehensively explored HTM's ability to model vibration data or demonstrated its practical value for PM. Overall, all of these existing methods fall short of addressing one or more research challenges.

1.3 Our Novel Contributions

To address these key research challenges and improve on the PM performance demonstrated by previous works, our paper presents the following contributions:

1. We demonstrate the ability of HTM-based anomaly detectors to detect early symptoms of bearing failure in several months' worth of real-world vibration data. We show that HTM's can efficiently learn with only a single training pass.

2. We demonstrate the ability of HTMs to generalize across applications without much fine-tuning and their ability to continuously learn and adapt by evaluating their anomaly detection performance on a second, highly dynamic application: 3D printer vibration data. These characteristics of HTMs make them more practical for real-world use cases.

3. We compare the performance of HTM anomaly detection methods against state-of-the-art anomaly detection techniques and traditional machine prognosis methods such as condition-based maintenance. Specifically, we evaluate each algorithm's anomaly detection accuracy and robustness to noise.

4. We demonstrate the efficiency and real-time capability of HTM-based prognosis by comparing its execution time with that of the other techniques.

Chapter 2

Background

2.1 Hierarchical Temporal Memory

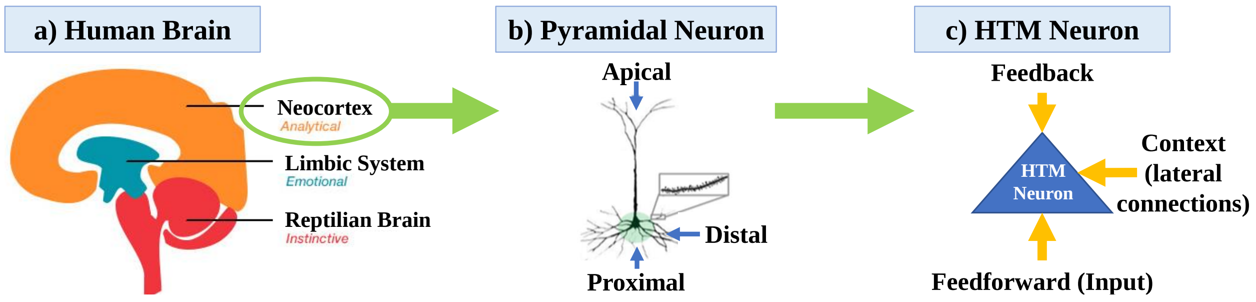

Figure 2.1: How Neocortical structures are modeled by Hierarchical Temporal Memory. The neocortex is composed of a large number of interconnected pyramidal neurons, each with proximal (feed-forward), apical (feedback), and distal (lateral) dendrites to connect to other neurons. These relations are modeled in HTM neurons as feed-forward, feedback, and lateral connections.

Hierarchical temporal memory is a sequence learning framework modeled after the structure of the neocortex in the human brain [6]. The basic unit of HTM is a neuron modeled after those present in the neocortex (Fig. 2.1(b)). These neurons are stacked on top of one another to form a column like the 'cortical column' of the neocortex. The final HTM is a composition of many such columns. A single HTM neuron (Fig. 2.1(c)), is connected to two types of segments: (i) proximal segments (aggregation of feed-forward connections from the input) and (ii) distal segments (aggregation of lateral connections from neurons of the other columns). Each HTM neuron can be in three states: (i) inactive (the default state), (ii) predictive, and (iii) active. The predictive state of a neuron is determined by the activity of the distal segments, which in turn is determined by the activation state of the other neurons. A neuron becomes active at any time only if it was in the predictive state at the previous instant, with an exception that will be described in Section 3.1. When the sequences of activations are viewed temporally, it is easy to see that the distal segments provide the temporal context for activation and thus capture the temporal relations. The column structure augments this capability of HTM by enabling them to store multiple such overlapping temporal sequences. Further details on the HTM-based anomaly detection methodology are discussed in Section 3.1.

2.2 PM of Roller-Element Bearings

Roller-element bearings perform the critical task of reducing friction between rotating parts in machinery. Generally, catastrophic bearing failures present warning signs such as anomalous vibrations and/or noise. These anomalies can occur due to environmental factors (moisture or debris entering the bearing) as well as installation errors (misalignment, excessive loads, or poor/improper lubrication) [27]. Recently, sensor-based techniques that leverage vibration and temperature data to monitor bearing health have been proposed. For example, the NASA Bearing Dataset and the Pronostia Bearing Dataset contain vibration and temperature data for several bearings which were run until failure [1, 28]. In both datasets, anomalies in the vibration and temperature signals increase in size and frequency as the bearings approach failure, showing a strong correlation between the sensors' readings and system state.

2.3 PM of 3D Printers

3D printing is a manufacturing process where a physical object is constructed from layers of material in an iterative process. Fused Deposition Modeling (FDM) is a standard technique where melted thermoplastic is extruded through a moving print head nozzle to build each layer. To ensure precision, stepper motors control the extrusion rate of the nozzle as well as the X, Y, and Z-axis movement of the print head. Since the motors, bearings, and belts are moving parts, they are prone to wear and must be regularly maintained to prevent component failures. As shown in [29], these components leak vibration information that can be used by PM systems. However, this leaked information is non-stationary since 3D printers move on multiple axes and change direction and speed often, presenting a challenge for conventional PM methods.

Chapter 3

Methodology

3.1 Anomaly Detection using Hierarchical Temporal Memory

The end-to-end framework for the HTM-based detector is shown in Figure 3.1. Our methodology for anomaly detection consists of the following steps. First, the time-series vibration data $X(t)$ is taken as input and encoded into a Sparse Distributed Representation (SDR). Next, the SDR is passed through the spatial pooler. The spatial pooler's output is fed into the temporal pooler, which then outputs a prediction for the next activation $\Pi(t_{n+1})$. Simultaneously, the prediction from the previous time step $\Pi(t_n)$ is compared with the column activations in the current time step $A(t_n)$ to give a prediction error value: a high error value indicates that this activation was not expected and may be anomalous. Finally, the anomaly detector uses the historical distribution of anomaly scores to calculate the anomaly likelihood $L(t_n)$ for the current data point based on the prediction error value; if $L(t_n)$ exceeds a set threshold, then $X(t_n)$ is flagged as an anomaly. In the following paragraphs, we describe each of these components in detail.

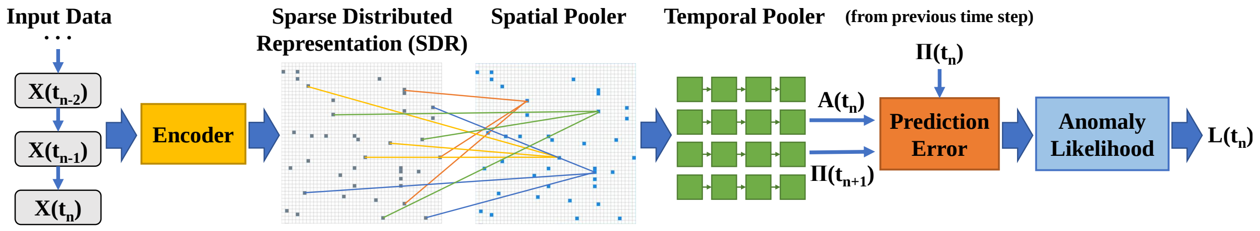

Figure 3.1: HTM Anomaly Detection Framework. The time-series input $X(t)$ is encoded into a Sparse Distributed Representation (SDR). This information is passed through a spatial pooler and a temporal pooler before outputting a prediction $\Pi(t_{n+1})$ for the next set of column activations. The prediction error between $\Pi(t_n)$ and $A(t_n)$ and the historical distribution of anomaly scores are used to determine the anomaly likelihood $L(t_n)$.

3.1.1 Encoder

The first stage in processing the input data $X(t)$ is the *encoder*. The encoder converts the incoming data point $X(t)$ into a *sparse distributed representation* (SDR). This representation is a vector of binary values, and it is sparse because only 2% of the bits are activated for any input. This contrasts with deep learning methods that store and learn a dense, distributed representation. Later, we shall describe the advantages of using a sparse representation. We denote the output of the encoder by x, a $1 \times n$ vector.

3.1.2 Spatial Pooling

The second stage is *spatial pooling*. The spatial pooler identifies spatial relations between different regions of the encoder's output through the proximal connections. Spatial poolers can also be stacked to identify more complex relations. The *proximal segment* of each neuron in a column is initialized such that each neuron, where the neurons of the same column share the same proximal segment, is connected to a large fraction of the inputs (50%). The output of this stage is also an SDR representing the columns of the HTM that will be activated in the final output. We denote the *spatial pooling* operation mathematically by $I_k(.)$, where the input is the list of columns ordered in decreasing order of their proximal segment values, and k indicates the number of columns to be picked for activation from the top of this list. The number k is typically the top 2%, so the output representation is sparse. Let y_c denote the activation of the columns and P denote the proximal connections where P is a binary matrix of size $n \times N$. Then

$$y_c = I_k(xP) \tag{3.1}$$

3.1.3 Prediction

The next stage is *prediction.* The prediction for the next time step is the predictive state of the HTM at the end of the current time step. Let the weights of the lateral connections of the dth distal segment of the ith neuron of jth column be $D^d_{i,j}$. We note that only those weights of connections which are above a certain threshold are considered to be *established* and the rest are set to zero. A neuron (i, j) enters the predictive state provided the sum of activations of at least one of the distal segments exceeds a certain threshold, θ_d. Denote the predictive state of a neuron at time t_n by $\pi_{i,j}(t_n)$. We denote the current *activation state* of all neurons at time t_n by $A(t_n)$. We denote the total predictive state by the matrix $\Pi(t)$, whose elements are therefore $\pi_{i,j}(t_n)$. Mathematically, $\pi_{i,j}(t_n)$ is given by,

$$\pi_{i,j}(t_n) = \begin{cases} 1; & \text{if } \exists\ d \text{ s.t. } ||D^d_{i,j} \odot A(t_n)||_1 > \theta_d, \\ 0; & \text{otherwise.} \end{cases} \tag{3.2}$$

where $\odot$ denotes the element-wise multiplication operation.

3.1.4 Temporal Pooling

The final stage is *temporal pooling.* Temporal pooling computes the activation state $A(t_n)$ (an $M \times N$ matrix where M is the number of neurons per mini-column and N is the number of mini-columns in the layer) of the HTM, which is also the output of HTM based on a temporal context. A neuron i is activated provided its column is activated, i.e., $y_c(j) = 1$ and provided it is in the predictive state, i.e., $\pi_{i,j}(t_{n-1}) = 1$. The other neurons in this column are inhibited. If none of the neurons in a column that is active are in the predictive state, then all the neurons of this column are activated. Here, the predictive state $\pi_{i,j}(t_{n-1})$ from the previous time step is the temporal context. This temporal context is updated at the end of this time step as described in the prediction step above. Let $a_{i,j}(t)$ be the i, jth element of $A(t_n)$ denoting the activation state of neuron i in column j. Then, the temporal pooling operation can be mathematically described as:

$$a_{i,j}(t) = \begin{cases} 1; & \text{if } y_c(j) = 1 \text{ and } \pi_{i,j}(t_{n-1}) = 1, \\ 1; & y_c(j) = 1 \text{ and } \sum_i \pi_{i,j}(t_{n-1}) = 0, \\ 0; & \text{otherwise.} \end{cases} \tag{3.3}$$

Figure 3.1 shows the different stages of HTM processing in the context of anomaly detection. After activation, the prediction error between the prediction from the previous time step $\Pi(t_n)$ and the current activation state $A(t_n)$ is computed and passed to the anomaly likelihood block, which uses the historical distribution of anomaly scores to determine if $X(t_n)$ is a true anomaly.

3.1.5 Learning

HTMs use a Hebbian-type learning algorithm that reinforces the connection weights of the segments that correctly predict the activation at the next time-step. Each time step, the weights are re-evaluated as follows. The connection weights of an activated neuron's segments that originated from previously active neurons are increased. The connection weights from neurons that were not active in the previous time-step are decreased. Additionally, weights of connections that are wrongly predicted are also decreased but at a lesser rate, i.e., forgetting happens at a slower rate than updating. It is this type of learning that allows HTMs to *learn continuously and adapt to changes over a long term.* The learning algorithm is discussed in much greater detail in [30].

3.1.6 On Capacity, Robustness, and Efficiency

Here, we illustrate why HTMs are efficient and robust to noise. Let us consider an HTM with a large n, where n denotes the size of the encoder's output, x, a binary vector. Denote by w the maximum number of bits that can be one. Typically, w is small relative to n. Given this, lets define: $\alpha := w/n$. Here, α is a measure of sparsity and denotes the fraction of the bits that can be active in the SDR of size n. An example would be, $n = 2048$ and $w = 4$ and so $\alpha \approx 0.002$.

The number of possible unique encodings, N_e that can be stored in vector x, given n and w, is given by,

$$N_e = \begin{pmatrix} n \\ w \end{pmatrix} = \frac{n!}{w!(n-w)!} \tag{3.4}$$

For example, if $n = 2048$ and $w = 20$ then $N_e = 10^{47}$. Given N_e, the probability that one

SDR x will match another SDR y, which is randomly picked, is trivially computable:

$$\mathrm{P}(x = y) = 1/N_e \tag{3.5}$$

Thus, the probability of a false match is, for all practical purposes, zero. This shows that SDRs can store and recall reliably an astronomically large number of vectors. Consequently, it follows that HTMs can *store and recall reliably an astronomically large number of sequences.*

We can now relax the requirement and say that two SDRs are equivalent if $\theta(< w)$ or more bits match. In this case, the matching is allowed an error of up to $w - \theta$ bits. Denote by $\Omega_x(b)$ the set of sparse vectors (of size n and sparsity α) that have an overlap of b bits with x. Then, the probability that a false match will be generated, P_{fm}, is given by,

$$\mathrm{P}_{fm} = \frac{\sum_{b \geq \theta} \Omega_x(b)}{N_e}, \text{ where } \Omega_x(b) = \begin{pmatrix} w \\ b \end{pmatrix} \times \begin{pmatrix} n - w \\ w - b \end{pmatrix} \tag{3.6}$$

Clearly, the probability of a false match has increased by allowing an error of up to $w - \theta$. In the same example as above, if $\theta = 10$, then $w - \theta = 10$, that is an error up to 50% is allowed. We find that the probability of a false match is still $1/10^{13}$, which for all practical purposes is zero. This is what gives SDRs and thereby HTMs *robustness to noise.*

The sparsity of x allows for sparse computation, which makes computations with SDRs very efficient. For a representation x of size n and sparsity α, one does not need to store information on all the bits. Instead, one can just store the address of the locations of bits of value one. Then, for an operation like matching, one just needs to check the value of the bits of the vector y at its corresponding locations; this is doable almost in constant time. We can trivially extend this argument to show that the spatial pooling, prediction, and temporal pooling operations described above can also be performed very efficiently in HTMs, thus giving HTMs their *computational efficiency*. Next, we discuss our experimental setup for demonstrating the performance of the HTM-based anomaly detector.

3.2 Experimental Setup

We evaluate our proposed methodology on real-world bearing failure and simulated 3D printer failure datasets. Here, we discuss details about these datasets and the scoring system used for evaluation.

3.2.1 Bearing Dataset

We used the NASA Bearing Dataset and the Pronostia Bearing Dataset [1, 28]. The NASA Bearing Dataset contains three tests of bearings run to failure. The Pronostia Bearing Dataset contains vibration snapshots recorded with three different radial load and RPM settings. The accelerometer data for Test 2 of the NASA Dataset is shown in Figure 3.2. In total, our testing set consists of 40 vibration data files and 191 labeled anomalies.

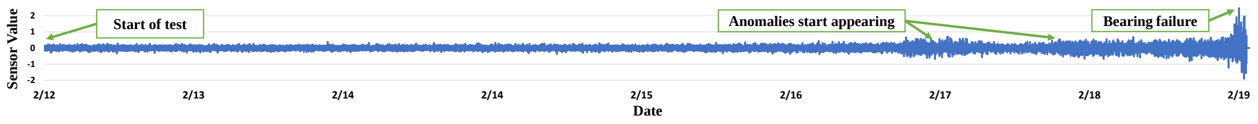

Figure 3.2: Accelerometer Data from Test 2 of the NASA Dataset [1]. Symptoms of bearing failure can be seen on 2/17 and 2/18 before the bearing's outer race failed on 2/19.

3.2.2 3D Printer Dataset

Our experimental testbed for collecting vibration data from a 3D printer is shown in Figure 3.3. The 3D printer uses one stepper motor to control each movement axis (X, Y, and Z). We

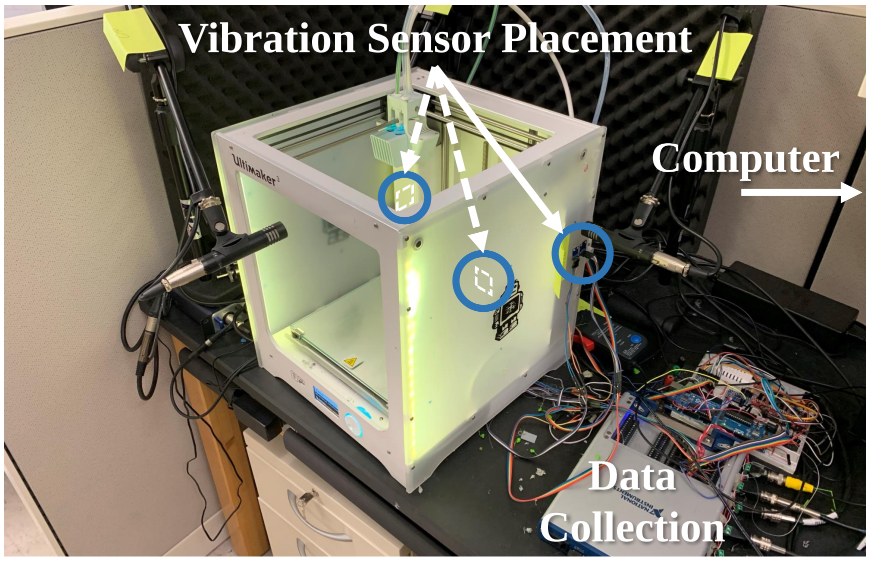

Figure 3.3: Experimental Testbed used to collect vibration data from our 3D printer. Three accelerometers were placed on the printer in total; one sensor was placed directly behind each of the printer's three stepper motors.

placed one accelerometer directly behind each stepper motor to capture vibration data from prints of various 3D objects. To the best of our knowledge, no publicly available 3D printer component-failure datasets exist, and generating real-world failures would risk damaging our equipment. Thus, we instead opted to generate synthetic anomalies in the 3D printer vibration data.

3D printer vibration signals are inherently non-stationary, meaning that their statistical properties vary with time. However, since printers contain bearings and rotating components with similar dynamics, they share the same time-series and frequency domain features as those correlated with bearing health, such as power spectral density (PSD) [23, 24]. For example, in Figure 3.2 it is clear that the overall power of the vibration signal increases as the bearing nears failure. Intuitively, this same phenomenon will occur in a 3D printer as components wear out. Thus, we synthesized anomalies in the 3D printer vibration data by mapping the PSD from our bearing failure data to the 3D printer data. This composition enabled us to simulate the magnitude changes characteristic of bearing and component failures in the 3D printer while preserving the frequency components unique to the 3D printer.

Our PSD mapping algorithm, shown in Algorithm 1, operates on a sliding window over one bearing vibration file and one 3D printer vibration file. For each window t, the following steps are performed: First, the Fast Fourier Transform (FFT) $X_b[t]$ of the bearing time-series data $b[n]$ is calculated for a pre-set frequency bin-size. Next, the power in each frequency bin is calculated. Then, we calculate the ratio C between the previous window's power value and the current power value in each bin. This ratio is used to scale the corresponding frequency bin in the FFT of the 3D printer data $FFT(p[t])$, yielding an FFT with synthesized anomalies $X_s[t]$. Finally, the Inverse FFT (IFFT) of $X_s[t]$ is taken and added to the output at location $s[t]$.

The result after all iterations is a 3D printer vibration signal with synthesized anomalies $s[n]$. Using this mapping algorithm, we produced a simulated 3D printer failure dataset containing 15 test cases and 57 hand-labeled anomalies.

Algorithm 1: PSD Mapping Algorithm

```
Result: 3D printer data with anomalies: $s[n]$
Initialize bearing and 3D printer data: $b[n]$, $p[n]$;
Initialize output signal: $s[n] \leftarrow [0, ..., 0]$;
$t \leftarrow 1$;
while $t < length(p)$ do
    $X_b[t] \leftarrow FFT(b[t])$;
    $P_b[t] \leftarrow |X_b[t]|^2$;
    $C \leftarrow \sqrt{P_b[t]/P_b[t-1]}$;
    $X_s[t] \leftarrow FFT(p[t]) \odot C$;
    $s[t] \leftarrow s[t] + IFFT(X_s[t])$;
    $t \leftarrow t + 1$;
end
```

3.2.3 Anomaly Detectors

To evaluate the performance of HTMs at PM, we use the following two HTM-based anomaly detectors in our approach with slightly different temporal memory implementations, which we denote as HTM [16] and TM-HTM [31]. To explore the effectiveness of anomaly likelihood for HTM-based detectors, we evaluated HTM and TM-HTM with three different anomaly likelihood configurations:

1. no anomaly likelihood: the prediction error of the HTM was directly used as the anomaly score.
2. historical distribution (HD): the implementation described in Section 3.1.
3. LSTM-based predictor (LP): The HD anomaly likelihood block was replaced with a 2-layer LSTM predictor trained to predict normal HTM prediction error values in order to filter out false positives/noise. The prediction error of the LSTM was used as the final anomaly score.

We also evaluated baseline and state-of-the-art anomaly detectors including an RNN-based detector configured to use LSTM cells (denoted as LSTM) [32] (similar to [12, 13]), Windowed Gaussian (based on the tail probability of the distribution over a sliding window), a threshold-based detector (similar to condition-based maintenance and [7]), EXPoSE [33], Contextual Anomaly Detector (CAD-OSE) [34], Relative Entropy [35], Etsy Skyline [36], KNN Conformal Anomaly Detector (KNN-CAD) [37], Bayesian Changepoint (BC) [38], Random (random anomaly score), and Null (constant anomaly score). All of the listed algorithms except LSTM were exposed to the training data once before testing and updated their models as they were exposed to unseen test data. LSTM was trained for over 1000 epochs on the training data and was tested with the model settings that resulted in the lowest validation loss. LSTM was tested offline, meaning that it did not update its model weights during testing. The LP anomaly likelihood configuration was also trained in this manner but used the HTM output as its input data instead.

3.2.4 Scoring

To score each algorithm fairly, we rely on the Numenta Anomaly Benchmark (NAB) [16]. NAB was designed to fairly benchmark anomaly detection algorithms against one another. It contains a built-in anomaly scoring algorithm, normalization, and three threshold optimization settings: standard, low false positives (Low FP), and low false negatives (Low FN). NAB takes in datasets with labeled anomalies and produces *anomaly windows*. These are used to score anomaly detectors on how precisely they can pinpoint anomalies; early/on-time detections are rewarded, and very early/late detections are penalized.

The NAB scoring function is as follows: given an application profile $A = [A_{TP}, A_{FP}, A_{TN}, A_{FN}]$ specifying the weights for each kind of detection, and the position y of the detection relative to the anomaly window, the scoring function for each detection is:

$$\sigma^A(y) = (A_{TP} - A_{FP}) \left(\frac{1}{(1 + e^{5y})} - 1 \right) \tag{3.7}$$

These scores are summed up for all the detections in a file; the following weighted penalty is deducted for every missed detection (f_d): $A_{FN} f_d$. The summed score is then normalized to a 0-100 scale where 0 represents equivalent (or worse) performance to the Null detector, and 100 represents a perfect anomaly detector. An example of the scoring functionality is shown in Figure 3.4. To provide ground-truth values of anomaly locations in the dataset, we followed the NAB official anomaly labeling guide and manually labeled anomalies in each dataset. The first 15% of each vibration data file was used for training with the remaining 85% used for testing and scoring.

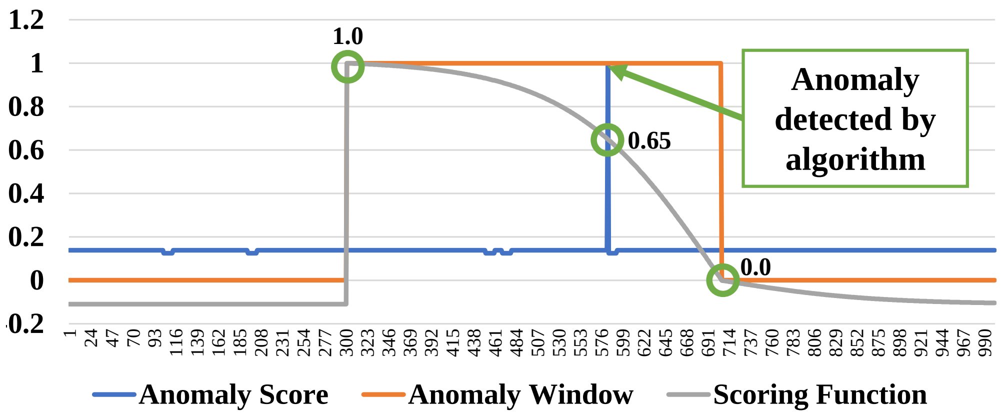

Figure 3.4: NAB Scoring Functionality: Detection scores are assigned according to the scoring function. The anomaly detected in this example is given a score of 0.65.

Chapter 4

Results

4.1 Roller Bearing Anomaly Detection

Table 4.1 shows the NAB results for the selected algorithms on the labeled bearing failure dataset as well as the total running time of each algorithm. The runtime was recorded over the complete dataset using a PC with an Intel Core i7-7700k processor. As shown in Table 4.1, TM-HTM+HD achieved the highest anomaly detection score for the Standard and Low FN profiles while HTM+LP achieved the highest score for the Low FP profile. TM-HTM+HD scored **67.05**, **73.33**, and **56.57** for the Standard, Low FN, and Low FP profiles, respectively. The approach that scored closest to HTM was Windowed Gaussian, which achieved scores of **64.70**, **70.50**, and **57.35** for the same profiles, respectively. HTM and HTM+LP performed better than TM-HTM TM-HTM+LP, indicating that TM-HTM's implementation only works well with the HD anomaly likelihood block.

As expected, the statistical methods (Windowed Gaussian, Threshold-Based, Relative Entropy) processed the dataset faster than the DL, ML, and HTM-based methods, albeit with lower performance. The HTMs using HD were 1.41x slower than the HTMs with no anomaly likelihood and 3.76x faster than the HTMs using LP on average. TM-HTM+HD processed the dataset **8.3x faster** than LSTM.

Anomaly Detector	Scoring Profile			Runtime (s)
	Standard	Low FN	Low FP	
TM-HTM+HD (Ours)	**67.05**	**73.33**	56.57	4728
HTM+HD (Ours)	66.38	71.93	55.33	5792
Windowed Gaussian	64.70	70.50	57.35	336
HTM+LP (Ours)	64.03	69.12	**57.47**	21084
HTM (Ours)	59.75	66.24	47.63	4277
TM-HTM+LP (Ours)	54.12	61.53	43.03	18508
TM-HTM (Ours)	54.39	63.33	32.47	3156
Etsy Skyline [36]	47.53	51.51	43.75	742632
CAD-OSE [34]	46.88	52.81	40.96	3589
EXPoSE [33]	41.75	44.80	36.96	5575
Threshold-Based	37.75	43.75	25.21	125
Relative Entropy [35]	34.97	37.05	32.94	806
LSTM [32]	33.99	38.13	28.38	43698
KNN-CAD [37]	32.31	43.06	4.69	4393
Random	3.06	9.16	0.00	233
BC [38]	0.00	0.00	0.00	10270
Null	0.00	0.00	0.00	235

Table 4.1: Normalized NAB scores for anomaly detection on the bearing failure dataset.

To evaluate the qualitative performance of each anomaly detector, we plotted the anomaly scores over time for each detector for Test 1 of the Pronostia Bearing dataset and compared them to the labeled ground truth anomaly windows in Figure 4.1.

4.2 3D-Printer Anomaly Detection

Table 4.2 shows our experimental results for the 3D printer dataset. HTM+HD achieved the highest score on the Low FN profile while LSTM achieved the highest score on the Standard and Low FP profiles. HTM+HD achieved scores of **63.03**, **73.18**, and **42.23** for the Standard, Low FN, and Low FP scoring profiles, respectively. LSTM scored **64.76**, **71.43**, and **51.34** at the same profiles, respectively. On both applications the HTM, TM-

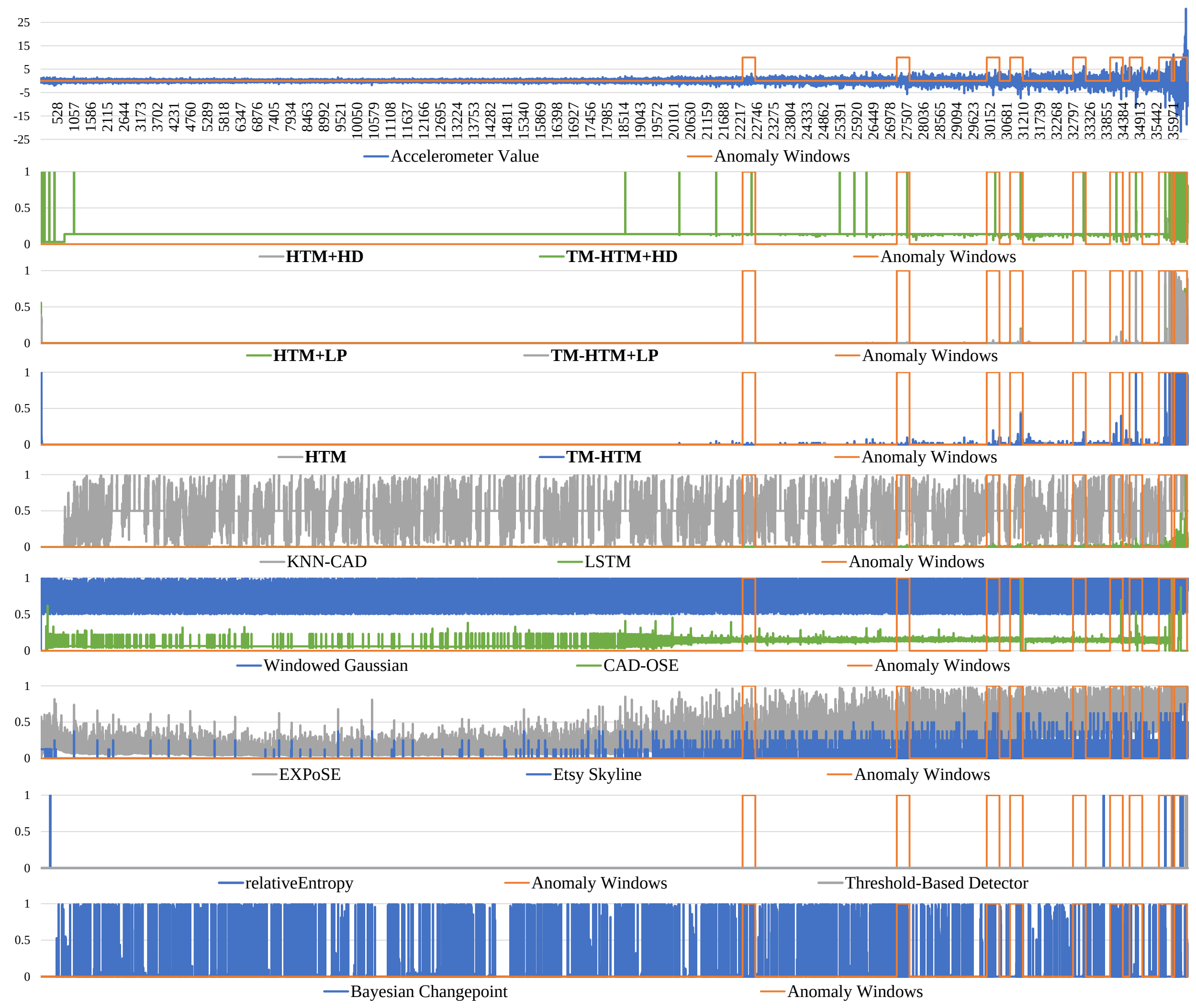

Figure 4.1: Anomaly scores for each detector in comparison to the ground truth anomaly windows for Test 1 of the Pronostia Bearing Dataset.

HTM, and TM-HTM+LP detectors performed worse than the HTM+HD, HTM+LP, and TM-HTM+HD detectors. Overall, the use of HD anomaly likelihood yielded the best HTM performance across applications. Each algorithm's execution time is consistent with the results shown in Table 4.1.

Anomaly Detector	Scoring Profile			Runtime (s)
	Standard	Low FN	Low FP	
HTM+HD (Ours)	63.03	**73.18**	42.23	5813
LSTM [32]	**64.76**	71.43	**51.34**	16414
TM-HTM+HD (Ours)	58.01	68.13	44.14	3364
Windowed Gaussian	57.42	67.99	49.31	189
HTM+LP (Ours)	54.56	65.45	36.76	10062
CAD-OSE [34]	54.69	63.27	34.72	1573
KNN-CAD [37]	53.76	65.55	23.48	3375
HTM (Ours)	49.21	62.25	31.16	2713
TM-HTM+LP (Ours)	45.31	57.90	21.93	7456
TM-HTM (Ours)	41.43	54.43	13.86	1724
Relative Entropy [35]	41.78	52.49	15.96	668
Etsy Skyline [36]	39.16	47.84	19.91	146165
BC [38]	21.33	23.64	17.55	1946
Random	11.34	24.75	3.74	49
EXPoSE [33]	6.04	6.20	6.04	7136
Threshold-Based	0.00	0.00	0.00	54
Null	0.00	0.00	0.00	49

Table 4.2: Normalized NAB scores for anomaly detection on the 3D printer dataset.

Chapter 5

Discussion

5.1 Overall Performance and Adaptability

Interestingly, algorithms that performed well on the bearing dataset, such as EXPoSE and Etsy Skyline performed worse on the 3D printer dataset. Additionally, algorithms that performed worse on the bearing dataset, such as LSTM and BC performed much better on the 3D printer dataset. Our HTM-based methodology using HD anomaly likelihood achieved consistently high performance on both applications without any hyperparameter tuning, demonstrating that this configuration can generalize and adapt to different applications without domain-specific tuning. This result also suggests that HTMs significantly benefit from the inclusion of an HD anomaly likelihood block.

Also, HTM+LP was the best performing model on the Low FP profile for the bearing dataset. However, this performance was not replicated in the 3D printer dataset. Similarly, LSTM beat HTM on the Standard and Low FP profile for the 3D printer dataset while performing worse than HTM on the bearing dataset. Hence, our results suggest that LSTMs are highly data-dependent and need to be re-tuned for every machine and/or application. Thus, the

LSTM approach is time-consuming, expensive, and impractical for real-world applications.

The benefits of HTM's continuous learning capability are clearly shown in Figure 4.1: after identifying earlier anomalies, the HTM-based approaches learn the new baseline for the signal and can pinpoint the future anomalies despite higher signal amplitudes. CAD-OSE also appears to learn continuously, but not as well as the HTMs.

5.2 Real-Time Detection Capability

In addition to detection accuracy and precision, an optimal PM system should be able to detect failure symptoms in real-time to allow adequate time for repairs to be scheduled and performed.

However, part failures are infrequent and generally present progressive symptoms before failure, so a hard real-time requirement for processing raw sensor data may unnecessarily limit the complexity (and subsequently the performance) of anomaly detection methods. Thus, we evaluate the anomaly detectors in the context of "soft real-time," where we determine if each detector can process a subsampled data segment before the next subsampled data segment arrives. For example, 1 second of data can be recorded each minute as a data segment to reduce data size while still ensuring that a wide range of vibration frequencies are captured at frequent intervals.

Both HTM+HD and TM-HTM+HD were able to process the complete bearing failure dataset in under 100 minutes; since the bearing dataset contains several months' worth of vibration data and minimal data preprocessing was performed (subsampling and timestamping), this demonstrates that HTMs can accurately detect failure symptoms in real-time, meaning that machine operators can be notified of degradation promptly. Other complex algorithms such as CAD-OSE, KNN-CAD, and EXPoSE had execution time on the same order of magnitude as HTMs and are thus also capable of real-time anomaly detection. Although HTM+LP, TM-HTM+LP, and LSTM took longer to process the dataset than HTM+HD and TM-HTM+HD, they can still be considered real-time due to the aforementioned dataset characteristics. However, the significant training time associated with the LSTM (over 12 hours on our hardware platform) and the need for application-specific hyperparameter tuning put LSTM at a disadvantage in terms of applicability to practical use cases.

5.3 Tunability, and Robustness to Noise

Figure 4.1 clearly shows HTM's ability to pinpoint anomalies while remaining robust to noise in the input. This is likely due to HTMs use of sparse encodings, making it unlikely that bit errors in the input due to noise will affect the bits corresponding to the input pattern, making them robust to noise. From the figure, it is also clear that the HTM implementations using anomaly likelihood blocks were more robust to noise outside of the anomaly windows than the HTM or TM-HTM alone. This is likely because the anomaly likelihood components filter out smaller detections to isolate only the most plausible anomalies. The HTM+HD and TM-HTM+HD detected anomalies earlier than the other configurations, albeit with slightly more false positives. The outputs of the different HTMs starkly contrast with the highly variable anomaly score outputs of Windowed Gaussian, EXPoSE, KNN-CAD, and BC, among others. These detectors record high anomaly scores even when there is relatively low noise in the input, meaning that they will likely suffer from false positives at higher noise levels.

A detector's threshold can be tuned to account for higher noise levels; however, for detectors such as Windowed Gaussian, which used the maximum detection threshold of 1.0, the threshold cannot be increased further to reduce its sensitivity. In contrast, TM-HTM+HD used a threshold of 0.5497 on the Standard profile. Thus, although Windowed Gaussian out-performed TM-HTM+HD on the Low FP scoring profile, it lacks tunability and will likely perform much worse than this HTM configuration in more noisy environments.

LSTM appears to have good robustness to noise, as shown in Figure 4.1. However, it is clear from the figure that it missed some of the earlier anomaly windows completely. In the context of PM, this can mean that an observer will only be warned of degradation later and will not have much time to organize repairs. Overall, our methodology demonstrates significant noise-robustness, better tunability, and the ability to detect early anomalies as well as larger, late-stage anomalies.

5.4 Limitations and Future Work

Another related PM problem is Remaining Useful Life (RUL) estimation. In many cases, RUL and anomaly detection go hand in hand as part of a comprehensive PM system. Although we did not evaluate the performance of HTM at RUL estimation, the core architecture of HTM is good at sequence prediction and could likely be used to solve this problem. We leave this for future work.

Another limitation of our work is the use of synthesized 3D printer anomalies instead of real-world examples of 3D printer failures. Due to resource constraints, we opted not to perform these experiments and used synthetic failure data instead. The question of whether HTM's performance on synthetic anomalies translates to real-world PM remains an open research problem.

5.5 Feasibility

The idea of predicting machine failures in advance is not brand new; many variants of PM systems have already been implemented in real-world manufacturing applications. However, based on our results, we believe that HTM is a better solution than current state-of-the-art methods. Our results demonstrate that HTMs are efficient enough to run on consumer-grade processors while learning and adapting continuously. Additionally, HTMs can be easily installed on existing PM systems as they only require time-series sensor inputs, which likely already exist in the system. As shown by our results, the industry-standard LSTM requires a significant amount of time for training (over 1000 epochs) as well as application-specific tuning. In contrast, HTMs do not require any application-specific parameter tuning and are essentially plug-and-play since they only need to be trained with a single pass on normal sensor data. These characteristics make HTMs an extremely viable, out-of-the-box solution for industrial PM.

Chapter 6

Conclusion

Existing methods for predicting machine failures from sensor data are limited in their practicality due to shortcomings, including poor noise resistance, efficiency, and adaptability. Our experiments demonstrated that our methodology outperforms state-of-the-art approaches at detecting anomalies in both bearing and 3D printer failure data with minimal to no preprocessing or application-specific tuning. On the Standard scoring profile, our methodology using HD anomaly likelihood achieved an average NAB score of **64.71**. In comparison, the other top algorithms: LSTM and Windowed Gaussian, achieved average scores of 49.38 and 61.06, respectively. Furthermore, our qualitative results show that our methodology is significantly more noise-resistant than the Windowed Gaussian, KNN-CAD, EXPoSE, and BC detectors, which we attribute to the use of SDRs and an anomaly likelihood component. We also demonstrated that our methodology is real-time capable, with an execution time on the same order of magnitude as state-of-the-art methods. Consequently, we conclude that HTM-based anomaly detection is a novel, practical solution for a wide range of industrial PM applications.

Bibliography

[1] J Lee, H Qiu, G Yu, J Lin, et al. Bearing data set. *IMS, University of Cincinnati, NASA Ames Prognostics Data Repository, Rexnord Technical Services*, 2007.

[2] A. V. Malawade, N. D. Costa, D. Muthirayan, P. P. Khargonekar, and M. A. Al Faruque. Neuroscience-inspired algorithms for the predictive maintenance of manufacturing systems. *IEEE Transactions on Industrial Informatics*, pages 1–1, 2021.

[3] Cornelius Scheffer and Paresh Girdhar. *Practical machinery vibration analysis and predictive maintenance*. Elsevier, 2004.

[4] R Keith Mobley. *An introduction to predictive maintenance*. Elsevier, 2002.

[5] Jonas Fausing Olesen and Hamid Reza Shaker. Predictive maintenance for pump systems and thermal power plants: State-of-the-art review, trends and challenges. *Sensors*, 20(8):2425, 2020.

[6] Jeff Hawkins and Sandra Blakeslee. *On intelligence: How a new understanding of the brain will lead to the creation of truly intelligent machines*. Macmillan, 2007.

[7] OR Seryasat, F Honarvar, Abolfazl Rahmani, et al. Multi-fault diagnosis of ball bearing using fft, wavelet energy entropy mean and root mean square (rms). In *2010 IEEE International Conference on Systems, Man and Cybernetics*, pages 4295–4299. IEEE, 2010.

[8] Fabio Immovilli, Marco Cocconcelli, Alberto Bellini, and Riccardo Rubini. Detection of generalized-roughness bearing fault by spectral-kurtosis energy of vibration or current signals. *IEEE Transactions on Industrial Electronics*, 56(11):4710–4717, 2009.

[9] Bin Zhang, Chris Sconyers, Carl Byington, Romano Patrick, Marcos E Orchard, and George Vachtsevanos. A probabilistic fault detection approach: Application to bearing fault detection. *IEEE Transactions on Industrial Electronics*, 58(5):2011–2018, 2010.

[10] Ameeth Kanawaday and Aditya Sane. Machine learning for predictive maintenance of industrial machines using iot sensor data. In *2017 8th IEEE International Conference on Software Engineering and Service Science (ICSESS)*, pages 87–90. IEEE, 2017.

[11] Diego Alejandro Tobon-Mejia, Kamal Medjaher, Noureddine Zerhouni, and Gerard Tripot. A data-driven failure prognostics method based on mixture of gaussians hidden markov models. *IEEE Transactions on reliability*, 61(2):491–503, 2012.

[12] Cheng Feng, Tingting Li, and Deeph Chana. Multi-level anomaly detection in industrial control systems via package signatures and lstm networks. In *2017 47th Annual IEEE/IFIP International Conference on Dependable Systems and Networks (DSN)*, pages 261–272. IEEE, 2017.

[13] Tayaba Abbasi, King Hann Lim, and Ke San Yam. Predictive maintenance of oil and gas equipment using recurrent neural network. *IOP Conference Series: Materials Science and Engineering*, 495:012067, jun 2019.

[14] Jae Yoon, David He, and Brandon Van Hecke. A phm approach to additive manufacturing equipment health monitoring, fault diagnosis, and quality control. In *Proceedings of the prognostics and health management society conference*, volume 29, pages 1–9. Citeseer, 2014.

[15] Chih-Ta Yen and Ping-Chi Chuang. Application of a neural network integrated with the internet of things sensing technology for 3d printer fault diagnosis. *Microsystem Technologies*, pages 1–11, 2019.

[16] Subutai Ahmad, Alexander Lavin, Scott Purdy, and Zuha Agha. Unsupervised real-time anomaly detection for streaming data. *Neurocomputing*, 262:134–147, 2017.

[17] Luis Rodriguez-Cobo, P Beatriz Garcia-Allende, Adolfo Cobo, José Miguel Lopez-Higuera, and Olga M Conde. Raw material classification by means of hyperspectral imaging and hierarchical temporal memories. *IEEE Sensors Journal*, 12(9):2767–2775, 2012.

[18] Amna Bamaqa, Mohamed Sedky, Tomasz Bosakowski, and Benhur Bakhtiari Bastaki. Anomaly detection using hierarchical temporal memory (htm) in crowd management. In *Proceedings of the 2020 4th International Conference on Cloud and Big Data Computing*, pages 37–42, 2020.

[19] Afaf Almehmadi, Tomasz Bosakowski, Mohamed Sedky, and Benhur Bakhtiari Bastaki. Htm based anomaly detecting model for traffic congestion. In *Proceedings of the 2020 4th International Conference on Cloud and Big Data Computing*, pages 97–101, 2020.

[20] Benhur Bakhtiari Bastaki. *Application of Hierarchical Temporal Memory to Anomaly Detection of Vital Signs for Ambient Assisted Living*. PhD thesis, Staffordshire University, 2019.

[21] Anomadarshi Barua, Deepan Muthirayan, Pramod P Khargonekar, and Mohammad Abdullah Al Faruque. Hierarchical temporal memory based one-pass learning for real-time anomaly detection and simultaneous data prediction in smart grids. *IEEE Transactions on Dependable and Secure Computing*, 2020.

[22] Sina Faezi, Rozhin Yasaei, Anomadarshi Barua, and Mohammad Abdullah Al Faruque. Brain-inspired golden chip free hardware trojan detection. *IEEE Transactions on Information Forensics & Security*, 2021.

[23] Weizhong Yan, Hai Qiu, and Naresh Iyer. Feature extraction for bearing prognostics and health management (phm)-a survey (preprint). Technical report, AIR FORCE RESEARCH LAB WRIGHT-PATTERSON AFB OH MATERIALS AND MANUFACTURING . . . , 2008.

[24] Dong Wang, Kwok-Leung Tsui, and Qiang Miao. Prognostics and health management: A review of vibration based bearing and gear health indicators. *IEEE Access*, 6:665–676, 2017.

[25] Jinjiang Wang, Yulin Ma, Laibin Zhang, Robert X Gao, and Dazhong Wu. Deep learning for smart manufacturing: Methods and applications. *Journal of Manufacturing Systems*, 48:144–156, 2018.

[26] Mirosław Kordos and Andrzej Rusiecki. Reducing noise impact on mlp training. *Soft Computing*, 20(1):49–65, 2016.

[27] ISO 15243:2017. Rolling bearings — damage and failures — terms, characteristics and causes. Standard, International Organization for Standardization, March 2017.

[28] Patrick Nectoux, Rafael Gouriveau, Kamal Medjaher, Emmanuel Ramasso, Brigitte Chebel-Morello, Noureddine Zerhouni, and Christophe Varnier. Pronostia: An experimental platform for bearings accelerated degradation tests. In *IEEE International Conference on Prognostics and Health Management, PHM'12.*, pages 1–8. IEEE Catalog Number: CPF12PHM-CDR, 2012.

[29] Sujit Rokka Chhetri and Mohammad Abdullah Al Faruque. Side channels of cyber-physical systems: Case study in additive manufacturing. *IEEE Design & Test*, 34(4):18–25, 2017.

[30] Jeff Hawkins and Subutai Ahmad. Why neurons have thousands of synapses, a theory of sequence memory in neocortex. *Frontiers in neural circuits*, 10:23, 2016.

[31] numenta. Numenta Temporal Memory Implementation, Feb 2020. [Online; accessed 10. Feb. 2020].

[32] Jinman Park. RNN based Time-series Anomaly Detector Model Implemented in Pytorch, 2018. [Online code repository].

[33] Markus Schneider, Wolfgang Ertel, and Fabio Ramos. Expected similarity estimation for large-scale batch and streaming anomaly detection. *Machine Learning*, 105(3):305–333, 2016.

[34] Mikhail Smirnov. Contextual Anomaly Detector, Aug 2016. [Online code repository].

[35] Chengwei Wang, Krishnamurthy Viswanathan, Lakshminarayan Choudur, Vanish Talwar, Wade Satterfield, and Karsten Schwan. Statistical techniques for online anomaly detection in data centers. In *12th IFIP/IEEE International Symposium on Integrated Network Management (IM 2011) and Workshops*, pages 385–392. IEEE, 2011.

[36] Abe Stanway. Etsy Skyline, Oct 2015. [Online code repository].

[37] Evgeny Burnaev and Vladislav Ishimtsev. Conformalized density-and distance-based anomaly detection in time-series data. *arXiv preprint arXiv:1608.04585*, 2016.

[38] Ryan Prescott Adams and David JC MacKay. Bayesian online changepoint detection. *arXiv preprint arXiv:0710.3742*, 2007.