2018 IEEE International Conference on Robotics and Automation (ICRA)
May 21-25, 2018, Brisbane, Australia

Inertial Machine Monitoring System for automated failure detection

Jens Windau, Laurent Itti

Smart manufacturing technologies are emerging which combine industrial equipment with Internet-of-Things (IoT) sensors to monitor and improve productivity of manufacturing. This allows for new opportunities to explore algorithms for predicting machine failures from attached sensor data. This paper presents a solution to non-invasively upgrade an existing machine with an Inertial Machine Monitoring System (IMMS) to detect and classify equipment failure or degraded state. We also provide a strategy to optimize the amount, placement locations, and efficiency of the sensors. In experiments, the system collected data from 36 inertial sensors placed at multiple locations on a 3D printer. Normal operation vs. 10 types of real-world abnormal equipment behavior (loose belt, failures of machine components) were detected and classified by Support Vector Machines and Neural Networks. Using under 1 minute of recording while running a test print, a recursively discovered best subset of 4 to 9 sensors yielded 11-way classification accuracy over 99%. Our results suggest that even a small sensor network and short test program can yield effective detection of machine degraded state and can facilitate early remediation.

I. Introduction

Unplanned downtime of machines costs industrial manufacturers an estimated $50 billion annually [Emerson, 2017]. Equipment failure is the cause of 42 percent of this unplanned downtime. The results are excessive maintenance, time-consuming repair or equipment replacement cost. However, worst case scenarios can be avoided by upgrading industrial equipment with smart predictive machine failure systems. Such systems monitor equipment with a sensor network and recognize abnormal machine behavior during run-time. In case of an identified failure, the system could schedule on-demand maintenance service to avoid larger economical damage to manufacturers. This is particularly important for factory machines, of which more than 90% are not yet network connected, and the vast majority are more than 15-years-old [Business Insider, 2016]. Upgrading equipment with predictive failure systems could save cost and increase efficiency and productivity, not only for the manufacturing industry. It is predicted that by 2025, the industrial IoT market will be worth $11 trillion and predictive maintenance can help companies save $630 billion over the next 15 years [McKinsey Global Institute Report, June 2015]. Although predictive maintenance has the most potential economic impact in factories (e.g. CNC, assembly robots, Laser Cutters, Pick-and-Place Machines), there is a wide field of potential applications for predictive failure systems. This ranges from Transportation (vehicle monitoring, condition-based maintenance of engines ranging from scooters to jumbo jets), Offices (e.g., elevator surveillance), Retail (e.g., commercial washing machines), but also could be implemented in high-end consumer and hobby products such as espresso machines or 3D printers. Fig. 1 shows an example.

Jens Windau, Laurent Itti are with the iLab and Computer Science Department at the University of Southern California, Los Angeles CA 90089-2520, USA. (website: http://ilab.usc.edu phone: +1(213)740-3527 email: jenswindau@gmail.com, itti@poluux.usc.edu)

Machine Monitoring System / Predicted Class → / Sample Actions ↓	Normal Operation	X-Axis Failure	Y-Axis Failure	Z-Axis Failure	X-Axis Belt is loose	Y-Axis Belt is loose	Tool scratches surface	Tool is jammed	Interfering Vibrations	Ground unstable	X/Y Axis damaged
No Action Needed	■										
Grease Machine Axes		■	■	■			■				■
Tighten/Exchange Belt					■	■					
Recalibrate Machine Axes							■				
Untangle Supply Tubes / Pipes								■			
Change Machine Location									■	■	
Exchange Machine Axis											■
Check Motor Cable Connection		■	■	■				■			
Reduce Actuator Speed							■			■	

Figure 1: Sample decisions of a customized Machine Monitoring Setting (3D printer with belt-driven print head that moves along 3 axes)

IoT-based, network-connected, and offline machines, could all benefit from a predictive machine failure system. What are the critical challenges to predicting equipment failure? First, abnormal behavior of a machine needs to be detected and classified. Critical for the success of this step is the right combination of sensor hardware and data processing technique. Particularly, two types of sensors have proven to be valuable for equipment failure detection. First, microphones have been used to detect equipment failure with sound vibrations. One example is the Fault Detection and Diagnosis of Railway Point Machines. A microphone records moving noise of switchblades during operations of railway infrastructure. Statistical Features are used to detect and classify three failure scenarios with an accuracy exceeding 94.1% [1]. Microphones have also been used for Acoustic Processing for Joint Fault Diagnosis of Legged Robots [2]. Microphone-based approaches mainly focus on sound in the high-frequency spectrum [3]. Our approach rather focuses on low-frequency vibrations (up to 25 Hz) which covers the major frequency spectrum of motions in linear axis machines. For this reason, we decided to design our hardware with a network of Inertial Measurement Units (IMUs). These devices are sensor cluster which embed three-axes accelerometers and gyroscopes (and sometimes magnetometers). They are well suited to capture low-frequency vibrations and motions of mechanical machine components. IMU sensor networks have been used for many applications including Wind-turbine Blade Condition Monitoring [4]. IMU-based approaches have

shown to measure deformations of a single linear axis by 5 µm (linear error) or 15 µrad (angular error) due to axis degradation [5]. A research team from the Chongqing Technology and Business University developed a system for fault identification and classification in gearboxes. Statistical features of inertial vibration data were analyzed in a convolutional neural network to distinguish between 12 discrete gear and bearing fault combinations. Although this approach performs well on the selected fault patterns (96.8%), it is limited to a number of discrete fault combinations (e.g., single gear failure, pair of gears failure). Our approach isolates each failure as an independent class and thus does not require a machine failure to occur in a certain combination of multiple faults [6]. With IMMS, we are building a universal sensor network solution that also uses vibrations as critical signals. IMMS is flexible to apply to various machinery, primarily focusing on linear and rotational motions. The approach presented in this paper has a set of unique contributions: (1) a robust hardware design to record inertial sensor data from 36 locations (2) an effective communication protocol to collect sensor data at 50 Hz (3) the identification of critical features (4) a database with ten recorded failure scenarios (5) a technique to classify and learn future failure scenarios in a short time and (6) a strategy to optimize amount, placement, and efficiency of sensors.

II. The IMMS Approach

1. *Robust Hardware Design of the sensor network*

The network consists of two types of modules: The Sensor Module [Fig. 2] and the Master Module. The network counts 36 daisy-chainable Sensor Modules that are connected over a bus system to the Master Module. The sensor network is split in three independent bus branches to limit bus noise and to provide a stable communication speed of 50 Hz [Fig. 3].

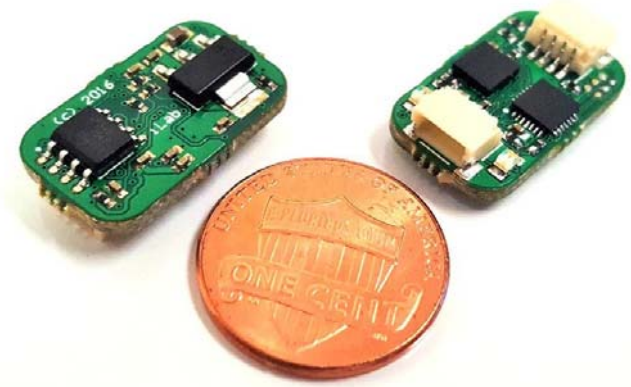

Figure 2: Design of the Sensor Modules to capture machine vibrations

Each **Sensor Module** consists of an IMU sensor (LSM9DS0), to collect gyroscope and accelerometer data. The IMU sensor is connected via I2C to an on-board microcontroller (ATmega328). The Sensor Module operates on stabilized 5V (LD1117AS33TR) and powered via the 9V supply bus. The RS485 serial communication is half-duplex and performed with the SP3485 IC. The IMU measures inertial motions as unsigned int16 (2 bytes), the Sensor Module communicates the data bytewise to the Master Board. A total of 12 bytes is transmitted (3x acceleration, 3x gyroscope) in each measurement cycle.

The **Master Module** collects data of all 36 Sensor Modules and forwards it to the PC for further processing. We use the Arduino Due which is equipped with four UART (Universal Asynchronous Receiver-Transmitter) ports as well as a native USB port that provides sufficient bandwidth for the transmission of all sensor data. Three UART ports receive data from the three sensor branches (250kHz Baud Rate), while the USB port forwards data to the PC (4,000kHz Baud Rate).

2. *Communication Protocol*

Every 20 ms (50Hz), the Master Module collects IMU data of all 36 Sensor Modules (432 bytes), adds a measurement timestamp (4 bytes), combines all data to a large package (436 bytes) and passes it on to the PC. To collect the data efficiently, each Sensor Module is programmed with a unique ID. In every 20 ms cycle, the Master Module broadcasts a measurement command. All Sensors Modules synchronously collect IMU data and consecutively send it to the Master in ascending ID order. The serial transmission standard is RS485. The bus consists of four wires: Two communication lines, 9V supply, and GND.

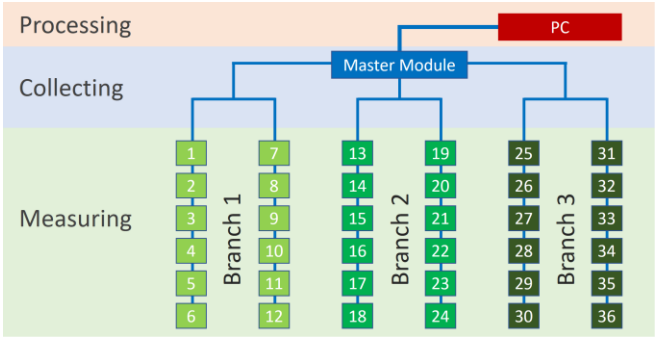

Figure 3: Wiring architecture of the sensor network. The green boxes represent Sensor Modules with unique IDs

3. *Identification of critical features*

To reduce the dimension of the sensor data and to prepare it for classification, features are extracted from the data. In this paper, we are testing our sensor network on a linear axes machine and only select the accelerometer sensor data for feature extraction. Accelerometers measure particularly linear motions and suit well for machines with movements along linear axes. For machines with rotational motions, we recommend using gyroscope data as well. For our experiments, we tested two approaches: (1) Statistical Features and (2) Vibration Frequency Diagrams. **Statistical Feature Vectors** represent the sensor data as a vector of length 1008. Each vector includes statistical features such as mean, kurtosis, skewness, variance, range, min, max, interquartile range, median absolute deviation, median, and most frequent value of the sensor data. **Vibration Frequency Diagrams** represents the sensor data in the frequency domain with the Fast Fourier Transformation (FFT). The data is stored in a matrix in which the vertical axis represents frequencies in the range between 0 and 25 Hz. In consideration of the Nyquist sampling theorem, the maximum measurable vibration is 25 Hz (half of the 50 Hz sensor sampling rate). The horizontal axis describes the numbers of sliding FFT-windows. Each FFT-window represents a FFT over sensor data of fixed length. The color of each cell in the Diagram describes the vibration

frequency occurrence. Brighter colors mean higher occurrence (FFT amplitude) [Fig. 4].

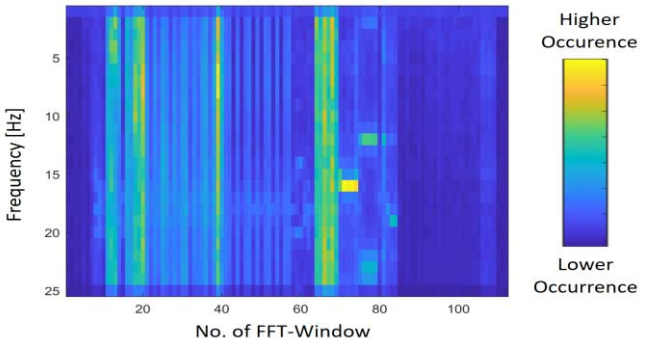

Figure 4: FFT-Diagrams describe vibration frequency occurrence over time. A 224 seconds dataset was described with 112 FFT-windows of each two-second sensor data. This FFT-Diagram displays the class "Tool scratches on surface" [Table 1].

4. Creating an extendable database

IMMS must have information about the failure scenarios of the machine, in advance, to classify them. Therefore, ten failure or degraded machine state scenarios were created under real-world settings, and sample data was recorded and stored in the IMMS database. This database can easily be extended by recording sensor data of new failure classes.

5. Classification technique for machine failure

We tested classifiers with the IMMS database using the Matlab Classification Learner App and Neural Network Toolbox. Among all supported machine learning algorithms, the two best-performing classifiers are compared in detail in this paper: (1) A Support-Vector-Machine with six different Kernel variations, and (2) a Feedforward Neural Networks with various Hidden Layer Sizes.

6. Optimization Strategy for sensors

IMMS uses a greedy algorithm to identify the subset of sensors that lead to the best classification result. This algorithm was applied to four independent sections of a machine test program. This optimization approach provides information on required number of installed sensors and placement location. It also gives suggestions on how a machine test program should be composed for the optimized sensor cluster and can be optimized for duration.

III. The Measurement in Detail

A. Overview

The architecture of IMMS consists of three major processing steps [Fig. 5]. In Step 1, the Master Module collects the IMU data of all Sensor Modules, combines it to one large package and forwards it with a measurement timestamp to the PC. Step 2 prefilters the sensor data and prepares it for feature extraction (Statistical Features or FFT Diagrams). All features are stored in a feature vector. Step 3 performs classification between 11 classes (normal operation + 10 failure scenarios). The feature vector is processed by a classifier (Support Vector Machine or Feedforward Neural Network). IMMS outputs the result of the classifier that determines if the machine operates properly or shows one of the 10 abnormal behaviors.

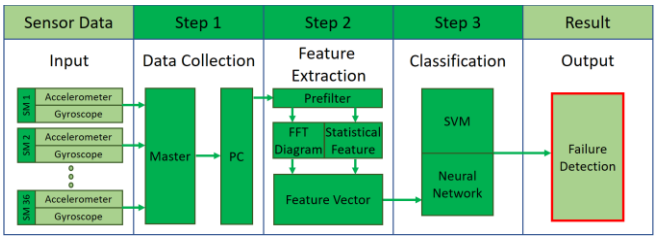

Figure 5: Overview of the system architecture

B. Steps in detail

Collection of IMU data: The Master Module collects 10,800 raw sensor values per second. For each collection cycle, the Master Module broadcasts a measurement command to all three sensor branches. Once received, each Sensor Module collects and locally stores the latest IMU data in its microcontroller. After all Sensor Modules processed the measurement command, the transmission process of the IMU data to the Master Module begins [Fig. 6].

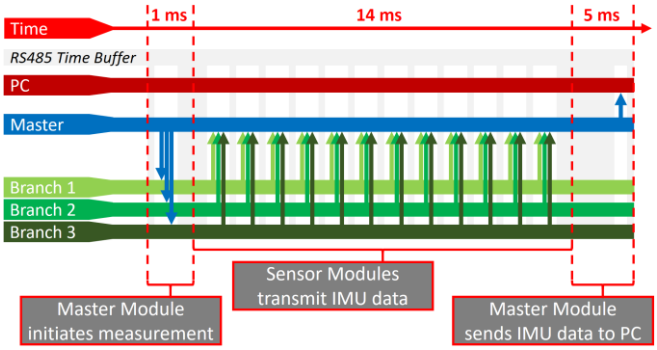

Figure 6: The Master Module initiates a 20ms data collection cycle

All sensor modules have a unique identity (ID), and know their place in the transmitting queue, which is being processed in ascending ID number. Each Sensor Module is counting the number of transmitted bytes in the branch. Once it is the turn of the Sensor Module in the queue, it releases the data and waits for the next measurement command of the Master Module to arrive. Each cycle takes 20 ms. To guarantee a sampling rate of 50 Hz, the Master Board has mechanisms to compensate for transmitting errors. In case the Master Module does not receive all 432 bytes (36x12 bytes) of sensor data, it stops the cycle after waiting for 15 ms and transmits only the received data to the PC. Should the transmission of data between the Master Board and PC last longer than 5 ms, it interrupts the transmission and starts a new data collection cycle on-time. Missing data is repaired or interpolated during prefiltering in Step 2.

Feature Extraction

In this paper, we record datasets of 224 seconds per class. This amount is equal 1.2M data points of accelerometer data. We reduce the amount of sensor data significantly by extracting features in two ways: (1) Extracting statistical features from the dataset reduces the sensor data to 1008 data points. (2) FFT Feature Diagrams varies depending on the FFT-window size. The amount of data points for FFT diagrams ranges from 350 (16 sec window) to 28,000 (0.2 sec window). To proceed with classification, FFT Feature Dia-

grams of size [N x M] will be converted into one-dimensional FFT Feature Vectors of length N x M.

Classification

IMMS uses the extracted feature vector to detect and classify a potential machine failure. Two of all tested classifier techniques showed particularly promising results on the given IMMS dataset and were further explored: First, we trained a Support-Vector-Machine (SVM). This classifier tends to perform well for classification tasks in high dimensions. We tested 6 different Kernel types: Linear, Quadratic, Cubic, Fine Gaussian, Medium Gaussian, and Coarse Gaussian. Second, we used a Feedforward Neural Network. The Hidden Layer was modified and tested for sizes between 1 – 100 Hidden Neurons. The output of the Network contained 11 Neurons, one for each possible outcome class [Fig. 7].

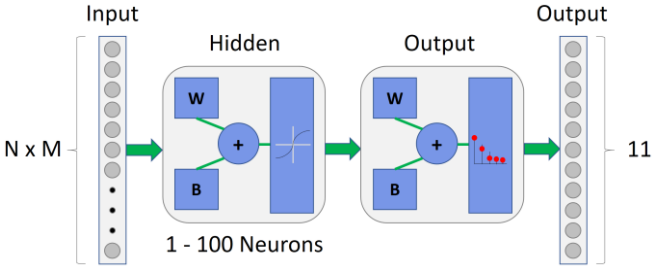

Figure 7: The architecture of the Feedforward Neural Network

IV. Test Results

A. Test Machine

To record real-world performance data, a linear axis 3D Printer was equipped with 36 IMMS Sensor Modules. The sensors were placed in 3D-printed holders and epoxy-glued on the machine [Fig. 8].

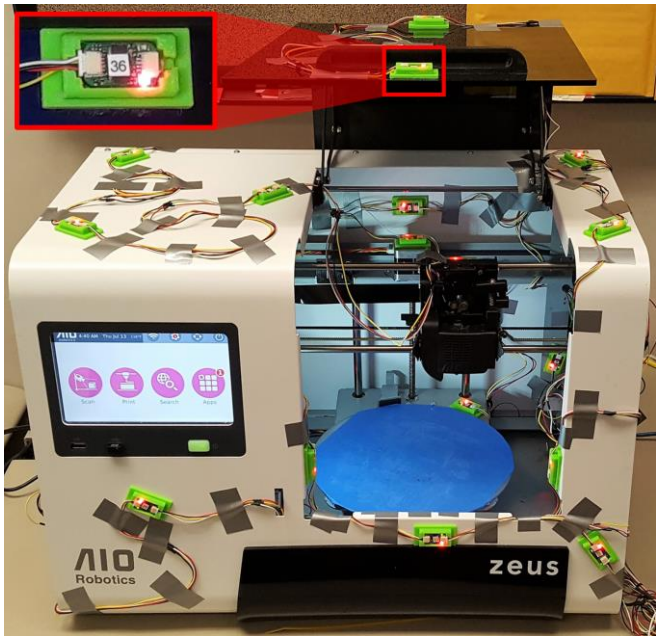

Figure 8: The testing machine is a 3D printer with motions along three linear axes

Locations and orientations of sensors are selected randomly but include all moving components of the machine (X, Y, and Z axis). First, the 3D Printer was operated in normal operation to record regular operation sensor data. Then, components were altered and modified to create sensor data of 10 common machine failure scenarios [Figure 9, Table 1].

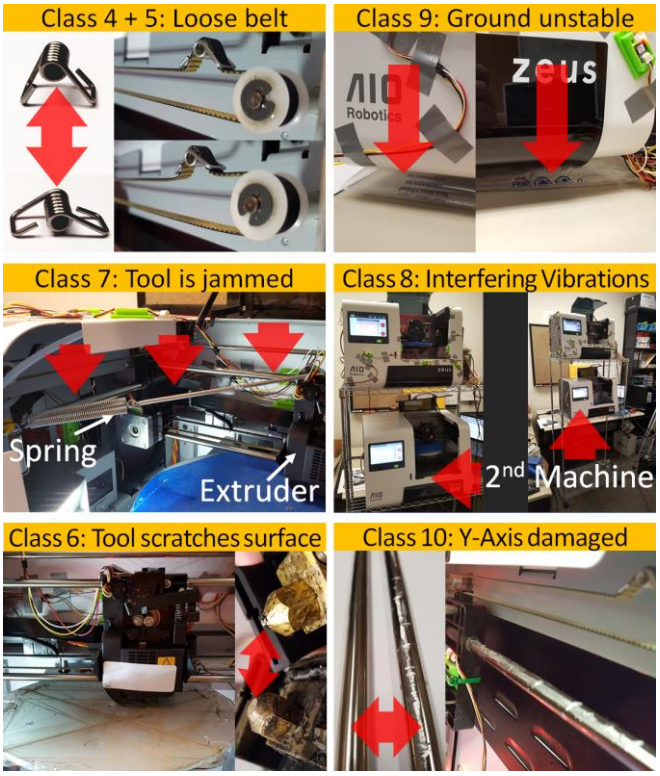

Figure 9: Altered machine components to record data of failure classes (as listed in Table 1)

For our experiments, we chose failure scenarios that not only occurred on 3D Printers, but can be found on many linear axes equipment with prices ranging from low-cost consumer equipment to expensive industrial machines. This includes machines for additive and subtractive manufacturing (e.g. CNC, Router, 3D Printers), cutting (e.g. Waterjet, Laser Cutter), or pick-and-place (e.g. electronic component placement machines).

Classes		Details
0	Normal Operation	No defects occurred
1	X-Axis Failure	X-Axis does not move
2	Y-Axis Failure	Y-Axis does not move
3	Z-Axis Failure	Z-Axis does not move
4	X-Axis Belt is loose	X-Axis Belt Tensioner is bent
5	Y-Axis Belt is loose	Y-Axis Belt Tensioner is bent
6	Tool scratches surface	Extruder nozzle is scratching turntable
7	Tool is jammed	Extruder is tied to a spring
8	Interfering Vibrations	Rack with two running printers
9	Ground unstable	Printer is placed on air bags
10	Y-Axis damaged	Y-Axis has file scratches

Table 1: Common failure scenarios of a 3D Printer

B. Test Program

We composed and installed a test program for IMMS on the 3D printer to create motions along all linear axes with varying speed and length. The total runtime is 224 seconds, but the program can be described to pass four characteristic sections, each with a length of 56 sec [Fig. 10].

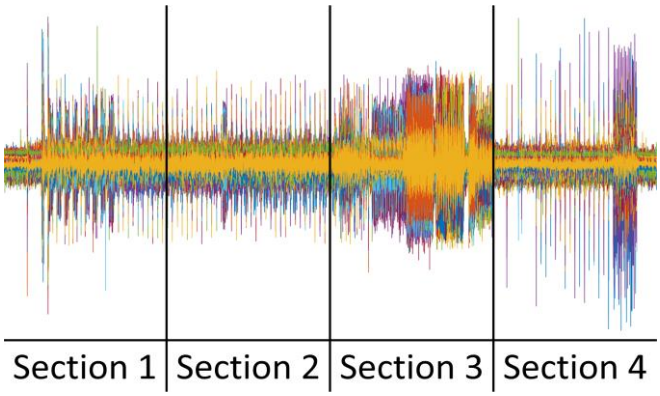

Figure 10: Test Program of 224 sec split into 4 sections. Each color shows the data of individual Sensor Modules

The four sections can be described with a set of unique motion directions and frequencies [Table 2].

Section	(Major) Motion Direction	Frequency
1	Single X/Y/Z axis motions	Low
2	Combined X/Y axis motions	Low
3	Single & Combined X/Y axis motions	Medium & High
4	Single Z axis motions	Low & Medium

Table 2: Composition of a machine program to cover various frequencies and motion directions

The total collected sensor data for the test program is 4.88 Mbytes. For each of the 11 classes, the test program was recorded 30 times over multiple days and in varying failure order. With a total of 330 test program recordings, we created a data package of 1.61 Gbytes. For our experiments, we are only using accelerometer data (0.81 Gbytes). The acceleration data was prefiltered, normalized and stored in double-precision format (8 bytes per number) which created a total amount of 3.21 Gbytes acceleration data. This data package was split in 2/3 training data and 1/3 testing data.

C. Performance Tests

Classification Test *(Statistical Features)*

In our first experiment, we analyzed the performance of statistical features. Two scenarios were tested: (1) Data from all 36 Sensors and (2) Data from four sensors, of which three were attached to moving machine components (X, Y, Z axis) and one was placed on the inner metal case.

For scenario (1), the Coarse Gaussian Kernel performed best with 27.3% error among all SVM classifiers. The Feedforward Neural Network achieved a similar result. For scenario (2) the overall performance error of SVM and Neural Network increased. This is an indication that sensors attached to moving machine components are not sufficient to properly classify all failure scenarios [Table 3].

Classifier (with Statistical Features)		Error Performance (1) All Sensors	(2) Moving Parts
Random Guess		90.9%	90.9%
Support Vector Machine	SVM - Linear Kernel	37.3%	36.4%
	SVM - Quadratic Kernel	38.2%	36.4%
	SVM - Cubic Kernel	38.2%	36.4%
	SVM - Fine Gaussian Kernel	90.9%	71.8%
	SVM - Medium Gaussian K.	45.5%	36.4%
	SVM - Coarse Gaussian K.	27.3%	45.5%
Feedforward Neural Network		27.3%± 0.8% *20 Neurons*	38.2% ± 4.5% *91 Neurons*

Table 3: Failure Classification via Statistical Features

Classification Test *(FFT Features)*

First, FFT-feature vectors from various FFT-diagram sizes were created. FFT-windows ranged from 0.2 sec. (Diagram dim. 25x1120) to 16 sec. (Diagram dim.: 25x14).

The SVM classifier achieved best results (4.5% error) using the Coarse Gaussian Kernel with 0.5 sec windows. Linear, Quadratic, and Cubic Kernels performed lower and scored slightly below 10% error. Fine and Medium Gaussian Kernel stayed above 25% error and are not listed in [Fig. 11].

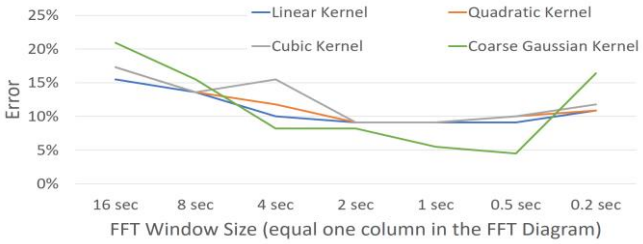

Figure 11: Performance of the SVM Classifier using FFT Features for various window sizes

The best performing window size for the Forward Neural Network is between 0.5 sec - 4 sec. The classifier achieved errors ranging between 0.7% - 0.0%. The 2 sec window with either 28, 31, 36, 38 or 42 Neurons in the Hidden Layer was best performing [Fig. 12].

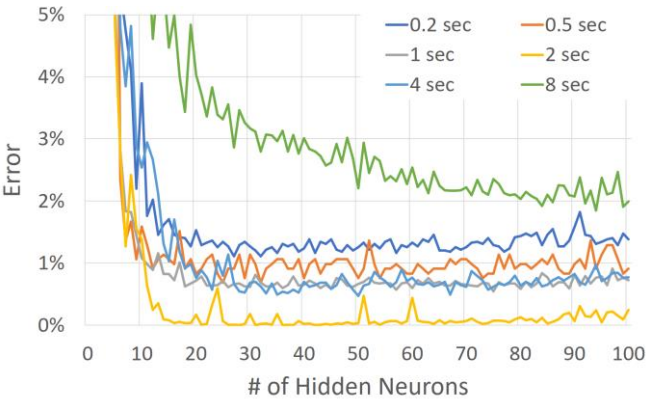

Figure 12: Feedforward Neural Network classification error using various window sizes

In summary, it turns out that the Feedforward Neural Network is superior over the SVM classifier for window sizes of 8 sec and shorter [Table 4].

Window Size	Feature Vector Size	SVM Error	SVM Kernel	Neural Network Error (100 runs)	Neural Network # Neurons
0.2 sec	28,000	10.9%	Linear/Quadratic	1.1% ± 0.5%	27
0.5 sec	11,200	4.5%	Coarse Gaussian	0.7% ± 0.4%	25
1 sec	5,600	5.5%	Coarse Gaussian	0.5% ± 0.5%	75
2 sec	2,800	8.2%	Coarse Gaussian	0.0% ± 0.0%	28,31,36,38,42
4 sec	1,400	8.2%	Coarse Gaussian	0.5% ± 0.6%	50
8 sec	700	13.6%	Lin./Quadr./Cubic	1.8% ± 1.1%	94
16 sec	350	15.5%	Linear	16.1% ± 4.1%	36

Table 4: Best performing SVM and Feedforward Neural Network parameters

Optimization Strategy

In this experiment, the goal is to find a subset of highest performing sensors. Therefore, a greedy algorithm is used to eliminate the least performing sensors. We temporarily re-

move one sensor from a set of *N* sensors and measure the classification performance of the subset *N-1*. A total of N subsets need to be tested. The temporarily removed sensor of the best performing subset will be eliminated, because it contributes least to the classification success. The algorithm starts with *N = 36* and is repeated until *N = 1*. This greedy algorithm might not strictly give us the best performing subset of sensors because we are only eliminating one sensor at a time. However, the algorithm generates u-shaped performance curves [Fig 13]. When initially removing sensors from the set, the performance continuously improves until a turning point is reached. At this point, the greedy algorithm minimized the error and all redundant sensors have been removed (e.g. sensors with redundant vibration measurements, because of similar placement location). After passing the turning point, the error increases again. This is because valuable sensor data is now being removed that is necessary for reliably distinguishing between failure scenarios. The experiment was performed on sections (half duration, quarter duration) and the whole test program. The Feedforward Neural Network was used with a best performing hidden layer size of 42 Neurons.

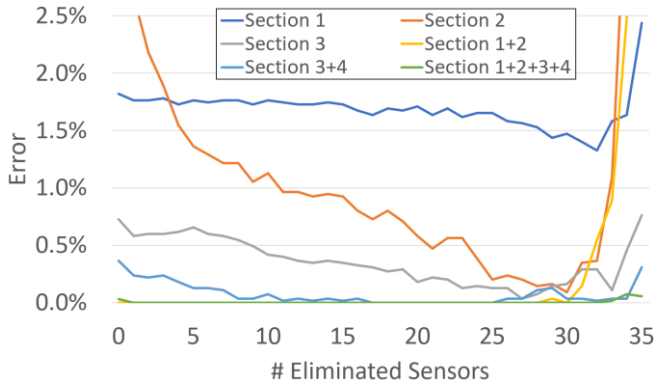

Figure 13: Applying a Greedy Algorithm to find the best performing subset of sensors. Section 4 is not displayed, since it only reached a performance of 7.65% error

Section	Best Performance	Length of Test Program	# Sensors left
1	1.33%	56 sec	4
2	0.09%	56 sec	6
3	0.04%	56 sec	9
4	7.65%	56 sec	15
1+2	No error	112 sec	6
3+4	No error	112 sec	9
1+2+3+4	No error	224 sec	4

Table 5: Classification results of best performing subsets

This optimization strategy shows that machine failures can reliably be detected by a small subset of sensors (four to nine). Below is a picture showing the placement location of the subset sensors of section 3 [Fig 14]. We observed that sensors of the best performing subset are commonly found on the inside and outside of the sheet-metal case of the 3D printer, but not necessarily on moving components. It also turned out that the highest performing sections of the test program contained combined motions over all machine axes. This observation should be considered when designing further machine testing programs. This experiment also proved that sufficient performance can be achieved with test programs lasting less than 1 minute. Because our neural network is very small, it could easily run on a low-cost embedded processor.

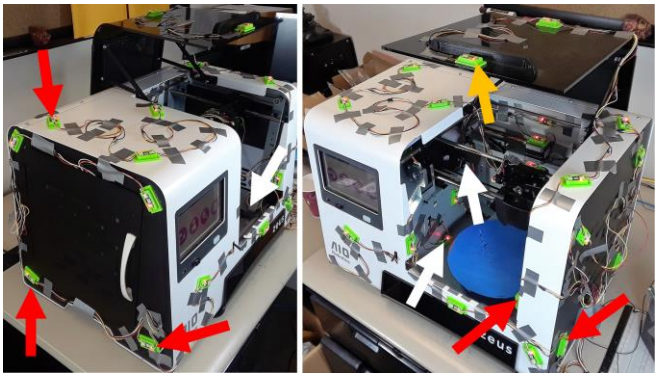

Figure 14: The subset of best performing sensors consists of sensors placed on the outside case (5 red arrows), inside case (3 white arrows) and the lid (1 orange arrow).

V. Conclusion

IMMS showed that inertial sensor networks can be successfully used to reliably detect and classify between 10 machine failures or degraded states of operation. With an increasing number of IoT sensors entering the manufacturing floor, predictive maintenance of industrial equipment guarantees higher efficiency and productivity. By providing valuable failure classification and early detection of machine degradation, we hope that IMMS made a contribution to the development of smart monitoring systems to help minimizing machine downtime.

Acknowledgment

This work was supported by the National Science Foundation (grants CCF-1317433 and CNS-1545089) and Intel Corporation. The authors affirm that the views expressed herein are solely their own, and do not represent the views of the United States government or any agency thereof.

References

[1] J. Lee, H. Choi, D. Park, Y. Chung, H. Kim, S. Yoon, "Fault Detection and Diagnosis of Railway Point Machines by Sound Analysis", Sensors, April 2016

[2] S. Chattunyakit, Y. Kobayashi, T. Emaru, "Joint Fault Diagnosis of Legged Robot based on Acoustic Processing', IEEE/SICE International Symposium on System Integration (SII), Meijo University, Nagoya, Japan, December 2015

[3] J Li, W. Dai, F. Metze, S. Qu, S. Das, "A comparison of Deep Learning methods for environmental sound detection", IEEE Int. Conference on Acoustics, Speech and Signal Processing (ICASSO), 2017

[4] O. Esu, J. Flint, S. Watson, "Static Calibration of Microelectromechanical Systems (MEMS) Accelerometers for In-Situ Wind Turbine Blade Condition Monitoring", Proceedings of 33rd IMAC, A Conference and Exposition on Structural Dynamics, Special Topics in Structural Dynamics, Volume 6, pp 91-98, 2015

[5] G. Vogl, M Donmez, A. Archenti, B. Weiss, "Inertial Measurement Unit for On-Machine Diagnostics of Machine Tool Linear Axes", Annual Conference of the Prognostics and Health Management Society, 2016

[6] Z. Chen, C. Li, R. Sanchez: "Gearbox Fault Identification and Classification with Convolutional Neural Networks", Research Article, Shock and Vibration, Hindawi Publishing Corporation, Volume 2015, Article ID 390134, 10 pages