Research Square

Preprints are preliminary reports that have not undergone peer review. They should not be considered conclusive, used to inform clinical practice, or referenced by the media as validated information.

Fog Computing Application of Cyber-Physical Models of IoT Devices with Symbolic Approximation Algorithms

Deok-Kee Choi (dkchoi@dankook.ac.kr)
Dankook University

Research Article

Keywords: Cyber-Physical Model, Smart Manufacturing, IoT, Fog computing, Machine Learning, Symbolic Approximation Algorithms, Streaming Data Analytics

Posted Date: June 4th, 2021

DOI: https://doi.org/10.21203/rs.3.rs-385990/v1

License: 🅭 🅯 This work is licensed under a Creative Commons Attribution 4.0 International License.
Read Full License

RESEARCH

Fog Computing Application of Cyber-Physical Models of IoT Devices with Symbolic Approximation Algorithms

Deok-Kee Choi

Abstract

Smart manufacturing systems transmit out streaming data from IoT devices to cloud computing; however, this could bring about several disadvantages such as high latency, immobility, and high bandwidth usage, etc. As for streaming data generated in many IoT devices, to avoid a long path from the devices to cloud computing, Fog computing has drawn in manufacturing recently much attention. This may allow IoT devices to utilize the closer resource without heavily depending on cloud computing. In this research, we set up a three-blade fan as IoT device used in manufacturing system with an accelerometer installed and analyzed the sensor data through cyber-physical models based on machine learning and streaming data analytics at Fog computing. Most of the previous studies on the similar subject are of pre-processed data open to public on the Internet, not with real-world data. Thus, studies using real-world sensor data are rarely found. A symbolic approximation algorithm is a combination of the dictionary-based algorithm of symbolic approximation algorithms and term-frequency inverse document frequency algorithm to approximate the time-series signal of sensors. We closely followed the Bayesian approach to clarify the whole procedure in a logical order. In order to monitor a fan's state in real time, we employed five different cyber-physical models, among which the symbolic approximation algorithm resulted in about 98% accuracy at a 95% confidence level with correctly classifying the current state of the fan. Furthermore, we have run statistical rigor tests on both experimental data and the simulation results through executing the post-hoc analysis. By implementing micro-intelligence with a trained cyber-physical model unto an individual IoT device through Fog computing we may alienate significant amount of load on cloud computing; thus, with saving cost on managing cloud computing facility. We would expect that this framework to be utilized for various IoT devices of smart manufacturing systems.

Keywords: Cyber-Physical Model; Smart Manufacturing; IoT; Fog computing; Machine Learning; Symbolic Approximation Algorithms; Streaming Data Analytics

Introduction

Smart manufacturing with IoT brings about rapid manufacturing by networking machines, sensors, and actuators as a whole [1–3]. The economics of Industrial IoT is expected to grow up to $6.2 trillion by 2025 [4]. IoT structure can be divided into three levels: IoT device (embedded computing), Fog computing, and Cloud computing. Each level is distinguished by the use of the three different data analytics shown in Fig. 1. Fog computing is facility that uses edge devices to carry out fast streaming data analytics [5–8]. The characteristics of Fog computing are a) low latency, b) low data bandwidth, and c) sensor heterogeneity [9, 10]. Cloud computing with big data analytics demands much expensive computing resources; therefore, downsizing the process being sent to cloud computing may be quite beneficial.

What smart manufacturing concerns most is to access the state of installed devices in real-time. Implementing any complexity of intelligence that can recognize the state of devices into embedded computing is often not possible due to its lack of built-in computing resources. This calls for a cyber-physical model (CPM) [11, 12] that can make an important decision on its own at Fog computing. However, no matter how simple an IoT device operates in a certain way, it is very difficult to create a CPM that can accurately tell the state on its own subject to unknown external or internal elements.

Correspondence: dkchoi@dankook.ac.kr
Department of Mechanical Engineering, Dankook University, 152, Jukjeon-ro, Suji-gu, 16890 Yongin-si, Gyeonggi-do, Republic of Korea
Full list of author information is available at the end of the article

Yet another difficult issue can be how to process enormous streaming data from sensors with great efficiency. This type of data is called time series (TS), which is a collection of ordered values over time. Since TS has a temporal structure, it needs to be treated differently from other types of data [13]. One big challenge in dealing with TS is its vast size [14]. In general, by applying approximation or transformation to data, some amount of size and noise can be cut down. Being used as a baseline algorithm for TS, K-Nearest Neighbors (KNN) with Dynamic Time Warping (DTW) is quite slow and requires much computing resources [15]. Recently, two Symbolic Approximation Algorithms that serve to transform a TS into symbols have drawn good attention: One is Symbolic Aggregate approXimation (SAX) [16]. The other is Symbolic Fourier Approximation (SFA) [17], through which we carried out simulations in this study. SFA consists of Discrete Fourier Transform (DFT) for approximation and Multiple Coefficient Binning (MCB) for quantization.

Given TS, time series classification (TSC) is of determining to which of a set of designated classes in this TS belongs to, the classes being defined by multiple set of training data [18, 19]. As the heart of the intelligence of IoT, CPM is evolving in such a way that it can manipulate the physical processes based on the domain-specific knowledge on its own, not blindly transmitting data to a cloud system for a request for an important decision. Recently, a couple of interesting algorithms for TSC have been rolled out: the BOSS model [20] and the WEASEL MUSE model [21] both apply a supervised symbolic representation to transform subsequences to words for classification. Several machine learning techniques have been used such as logistic regression, support vector machine, random forest, artificial neural networks (ANN) [22], and LSTM [23]. In this study, we employed the BOSS model as an algorithm for the CPM. Furthermore, we adopt an extended algorithm: BOSS VS (Bag-Of-Symbolic Fourier Symbols in Vector Space) based on a term-frequency inverse-term-frequency vector for each class [24, 25].

Figure 2 shows a workflow, on which we divided it into three phases: observation, machine learning, and status update. In the observation phase, streaming data is being stored for $3T$ sec. Once it passes $3T$ sec, then Machine Learning trains the model for $2T$ sec. It should be noted that even while learning the model, the new data is still coming in to be stored. As soon as the training is up, the model conducts classification over new data for T sec. It is important to note that Machine Learning and Status Update must be completed within $3T$ sec, otherwise, this process can be out of work. This completes a single process loop in the repetitive process for classification.

In this study, we built a three-blade fan with an accelerometer installed on it. We defined three states of the fan for classification: normal state, counter-wind state, and mechanical failure state. Fog computing is set up with a system: Intel Core i3-8100 CPU 3.60 GHz at 4 GB memory, and the OS being Ubuntu 20.04.1 LTS.

In summary, herein we pursued three objectives: along with real-world experimental data of the fan, we proposed a cyber-physical model that meets the following requirements:

- To be able to efficiently process streaming data,
- To be able to accurately classify the state of the fan,
- To complete classification within the designated time at Fog computing.

Cyber-Physical Models for IoT Device

In probabilistic modeling, we pay good attention to a model for real-world phenomena with uncertainty. Unlike with well-organized and preprocessed data, the complexity of the real-world phenomenon as shown in Fig. 3 can easily go out of the bound of our comprehension. For example, such are air velocity, rotation, vibration, counter-wind occurrence, and pressure changes, etc. This is because there are too many unknowns relating physical interactions for even a simple IoT device like a fan.

We have sought a cyber-physical model that can classify the state (or class) of a fan. The class set $\mathbf{C} = \{C_1, \cdots, C_K\}$ with K being the number of the classes, which the classes are normal, counter-wind, and mechanical failure states of the fan in this study. Fast Streaming data from sensors can be regarded as time series. A time series T_i is defined as an ordered sequence $T_i = \{t_1, \ldots, t_n\}$. The multivariate time series have several features, that is $\forall j, t_j \in \mathcal{R}^d$, where d is the number of features. The model is expected to provide a relatively simple function $f : \mathbf{T} \rightarrow \mathbf{C}$, where $\mathbf{T} = \{T_1, \ldots, T_N\}$, and is expressed as a joint probability distribution $p(\mathbf{C}, \mathbf{T})$.

A prime purpose of the use of the Bayesian approach is to infer probability for what we are interested in. As shown in Fig. 3, because the inference in such a phenomenon is quite complex, the model demands a huge amount of training data. For such a practical reason, the number of cases within the model of the joint probability must be drastically reduced. That is, firstly we worked with the joint probability distribution $p(\mathbf{C}, \mathbf{T})$ of the class $\mathbf{C}$ and a time series $\mathbf{T}$. We wanted to calculate how likely a given data is. Thus, we defined the posterior $p(\mathbf{C}|\mathbf{T})$ to infer the hidden structure with Bayesian inference:

$$\begin{aligned} f(\mathbf{T}) &= \underset{C \in \mathbf{C}}{\operatorname{argmax}}\; p(\mathbf{C}|\mathbf{T}) \\ &= \underset{C \in \mathbf{C}}{\operatorname{argmax}}\; \frac{p(\mathbf{T}|\mathbf{C})p(\mathbf{C})}{\int p(\mathbf{T}|\mathbf{C})p(\mathbf{C})d\mathbf{C}} \\ &\propto \underset{C \in \mathbf{C}}{\operatorname{argmax}}\; p(\mathbf{T}|\mathbf{C})p(\mathbf{C}) \end{aligned} \tag{1}$$

By marginalizing the joint probability distribution over $\mathbf{C}$ we are not interested in, then we can make the resulting marginal distribution $p(\mathbf{T}) = \int p(\mathbf{T}|\mathbf{C})p(\mathbf{C})d\mathbf{C}$. This calls for the conditioning on the joint probability and prior: $p(\mathbf{T}|\mathbf{C})p(\mathbf{C})$. Once given the training data set, the likelihood $p(\mathbf{T}|\mathbf{C})$ in Eq. (1) can be calculated. Thereby, we can then answer: given the data $\mathbf{T}$, what are the most likely parameters of the model or class $\mathbf{C}$? With the help of the Bayesian approach, we may come up with a logical procedure empolyed in the present study.

Classification refers to labeling a new time series $Q = \{q_1, \cdots, q_m\}$ and $\forall i$, $q_i \in \mathcal{R}^d$ for $i = 1 \ldots m$ with d being the feature shown in Eq.(2). In other words, classification indicates the calculation of $p(\mathbf{C}|\mathbf{Q})$ subjected to new streaming data $\mathbf{Q} = \{Q_1, \ldots, Q_N\}$:

$$\begin{aligned} \texttt{label}(\mathbf{Q}) &= \underset{C_k \in \mathbf{C}}{\operatorname{argmax}}\; p(\mathbf{C}|\mathbf{Q}) \\ &= \underset{C_k \in \mathbf{C}}{\operatorname{argmax}}\; p(\mathbf{Q}|\mathbf{C})p(\mathbf{C}) \end{aligned} \tag{2}$$

Symbolic Fourier Approximation (SFA)

This section introduces Symbolic Fourier Approximation (SFA) used in the BOSS VS model. SFA consists of Discrete Fourier Transformation (DFT), and Multiple Coefficient Binning (MCB).

Discrete Fourier Transformation (DFT)

Discrete Fourier Transform (DFT) extracts Fourier coefficients from each time series T:

$$\texttt{DFT}(T) = \{a_1, b_1, \ldots, a_m, b_m\} \tag{3}$$

where a_i and b_i are the real and the imaginary element of Fourier coefficients. Figure 4 shows that low-pass filtering and smoothing of a sample of acceleration in x-axis upon Discrete Fourier Transform (DFT), where DFT result is obtained by taking first two Fourier coefficients in Eq.(3).

Multiple Coefficient Binning (MCB)

Next, the Multiple Coefficient Binning (MCB) quantization is carried out with training data. $\mathbf{M}$ matrix is constructed using the Fourier transform of N training time series with the first l of Fourier coefficients being equivalent to an SFA word of length l as defined in Eq.(4).

$$\begin{aligned} \mathbf{M} &= \begin{pmatrix} \texttt{DFT}(T_1) \\ \vdots \\ \texttt{DFT}(T_N) \end{pmatrix} \\ &= \begin{pmatrix} (a_1, b_1)_1 & \ldots & (a_{l/2}, b_{l/2})_1 \\ \vdots & \vdots & \vdots \\ (a_1, b_1)_N & \ldots & (a_{l/2}, b_{l/2})_N \end{pmatrix} \end{aligned} \tag{4}$$

where M_j being $j-$th column of $\mathbf{M}$ matrix for all of N training data. M_i is then divided into intervals of c and is sorted by value and then divided into c bins of equi-depth policy. That is, the $i-$th row of $\mathbf{M}$ corresponds to the Fourier transform of the ith time series T_i. With the columns M_j for $j = 1, \ldots, l$, and an alphabet space $\mathcal{A}^l$ of size c, the breakpoints $\beta_j(0) < \cdots < \beta_j(c)$ for each column M_j are generated. Each bin is labeled by applying the ath symbol of the alphabet $\mathcal{A}^l$ to it. For all combination of (j, a) with $j = 1, \ldots, l$ and $a = 1, \ldots, c$, the labeling $symbol(a)$ for M_j can be done by

$$[\beta_j(a-1), \beta_j(a)] \approx symbol(a) \tag{5}$$

It is noted that this process in Eq.(5) applies to all training data.

SFA Working Example

SFA word can be obtained from $\texttt{SFA}(T) = s_1, \ldots, s_l$ with DFT where $\texttt{DFT}(T) = t'_1, \ldots, t'_l$ and t's are transformed time series with Fourier transform. That is, $\texttt{SFA}$: $\mathcal{R}^l \to \mathcal{A}^l$, where $\mathcal{A}^l$ is the alphabet set of which size is c. For a working example, in Fig. 5, we set $l = 1$ and $c = 2$. Six samples as shown Fig. 5(a) are randomly selected from the experimental data. The data then is transformed via DFT, resulting in the Fourier coefficients for each sample. A vector of the Fourier coefficient values of the first sample reads (-1.352, 5.043) as shown in Fig. 5(b). Next, MCB is conducted with an alphabet set $\mathcal{A}^1 = \{\mathbf{aa}, \mathbf{ab}, \mathbf{ba}, \mathbf{bb}\}$ as shown in Fig. 5(c). Thereby, an SFA word of the first sample is mapped into a word **ab** shown in Fig. 5(d). Likewise, the other samples can be transformed into their respective SFA words.

BOSS: Bag-of-SFA-Symbols model

The Bag-Of-SFA-Symbols (BOSS) model is of the time series representation with the structure-based representation of the bag-of-words model. The sequence of SFA words for six samples in Fig. 5(d) reads as follows:

$$\mathbf{S} = \{\mathbf{ab}, \mathbf{ba}, \mathbf{bb}, \mathbf{aa}, \mathbf{aa}, \mathbf{bb}\} \tag{6}$$

The values that count the apperance of SFA words in Eq.(6) are expressed upon numerosity reduction:

$$B : \mathbf{aa} = 2,\ \mathbf{ab} = 1,\ \mathbf{ba} = 1,\ \mathbf{bb} = 2 \tag{7}$$

It is noted that SFA words in Eq.(6) now results in the BOSS hisgogram shown in Eq.(7). Therefore, the BOSS model B can be regarded as a random variable, that is, $B : \mathbf{S} \to \mathcal{N}$. The probability mass function $p(B)$ can be addressed by $p(B = \mathbf{aa}) = 1/3$, $p(B = \mathbf{ab}) = 1/6$, $p(B = \mathbf{ba}) = 1/6$, and $p(B = \mathbf{bb}) = 1/3$. This provides us with quite important information about the structure of the samples, which structure is being used as features for machine learning.

BOSS VS: Bag-of-SFA-Symbols in Vector Space

BOSS VS model is an extented BOSS model. A time series $T = \{t_1, \dots, t_n\}$ of length n is divided into sliding windows of length of w is $S_{i,w}$, where $w \in \mathcal{N}$. The SFA word is defined as $\mathtt{SFA}(S_{i,w}) \in \mathcal{A}^l$, with $i = 1, 2, \dots, (n-w+1)$, where $\mathcal{A}$ is the SFA word space and $l \in \mathcal{N}$ is the SFA word length. The BOSS histogram $B(\mathbf{S}) : \mathcal{A}^l \to \mathcal{N}$. The number in the histogram is the count of appearance of an SFA word within T upon numerosity reduction. BOSS VS model allows frequent updates, such as fast streaming data analytics. As shown in Fig. 6(a) and Fig. 6(b), the BOSS VS model operates sliding windows unto each time series resulting in multiple windowed subsequences. Next, each subsequence is tranformed into the SFA words shown in Fig. 6(c). All of the subsequences eventually then result in the BOSS histogram shown in Fig. 6(d). However, since the BOSS histogram itself is not suitable for performing multiple of matrix calculations, it is vectorized through Term Frequency Inverse Document Frequency (TF-IDF) algorithm shown in Fig. 6(e).

TF-IDF: Term Frequency Inverse Document Frequency

The BOSS VS model employs Term Frequency Inverse Document Frequency (TF-IDF) algorithm to weight each term frequency in the vector. This assigns a higher weight to signify words of a class. The term frequency $\mathtt{tf}$ for SFA words $\mathbf{S}$ of a time series T within class $\mathbf{C}$ is defined as

$$\mathtt{tf}(\mathbf{S}, \mathbf{C}) = \begin{cases} 1 + \log\left(\sum_{T\in\mathbf{C}} B(\mathbf{S})\right), & \text{if } \sum_{\mathrm{T}\in\mathbf{C}} \mathrm{B}(\mathbf{S}) > 0 \\ 0, & \text{otherwise} \end{cases} \tag{8}$$

where $B(\mathbf{S})$ is the BOSS histogram in Eq.(7). The inverse document frequency $\mathtt{idf}$ is given by

$$\mathtt{idf}(\mathbf{S}, \mathbf{C}) = \log \frac{|\mathbf{C}|}{|\{\mathbf{C}|T \in \mathbf{C} \cap B(\mathbf{S}) > 0\}|} \tag{9}$$

In this study, for classification purposes, we employed three different states of a running fan, which is presented as a set of classes (states) $\mathbf{C}$. The elements of the set are $\mathbf{C} = \{C_1, C_2, C_3\}$. It is noted that each element C_k for $k = 1, 2, 3$ represents a certain state of the fan. As for the human-readable format, we have assigned name-tags to each class such as C_1 = "*Normal*", C_2 = "*Counter Wind*", and C_3 = "*Mechanical Failure*", respectively. The inverse document frequency indicates the frequency of an SFA word in a class C_k. Therefore, in this study, the numerator of Eq.(9) of $|\mathbf{C}|$ denotes a numeric value 3. Multiplying Eq.(8) by Eq.(9), the $\mathtt{tfidf}$ of an SFA word $\mathbf{S}$ within class $\mathbf{C}$ is defined as

$$\begin{aligned} \mathtt{tfidf}(\mathbf{S}, \mathbf{C}) &= \mathtt{tf}(\mathbf{S}, \mathbf{C}) \cdot \mathtt{idf}(\mathbf{S}, \mathbf{C}) \\ &= \left[1 + \log\left(\sum_{T\in\mathbf{C}} B(\mathbf{S})\right)\right] \cdot \\ &\log \frac{|\mathbf{C}|}{|\{\mathbf{C}|T \in \mathbf{C} \cap B(\mathbf{S}) > 0\}|} \end{aligned} \tag{10}$$

The result of $\mathtt{tfidf}(\mathbf{S}, \mathbf{C})$ on three states is displayed in Fig. 6(e). It is noted that a high weight in Eq.(10) is obtained by a high term frequency in the given class.

Classification

Classification of new data Q can be carried out using the cosine similarity metric $\mathtt{CosSim}$:

$$\begin{aligned} &\mathtt{CosSim}(\mathbf{Q}, \mathbf{C}) \\ &= \frac{\sum_{\mathbf{S}\in\mathbf{Q}} \mathtt{tf}(\mathbf{S}, \mathbf{Q}) \cdot \mathtt{tfidf}(\mathbf{S}, \mathbf{C})}{\sqrt{\sum_{\mathbf{S}\in\mathbf{Q}} \mathtt{tf}^2(\mathbf{S}, \mathbf{Q})}\sqrt{\sum_{\mathbf{S}\in\mathbf{C}} \mathtt{tfidf}^2(\mathbf{S}, \mathbf{C})}} \end{aligned} \tag{11}$$

It is noted that in Eq.(11) $\mathtt{tf}(\mathbf{S}, \mathbf{Q})$ is of the term frequency of $\mathbf{Q}$ as shown in Fig.7(b), which is the BOSS histogram of $\mathbf{Q}$. Then, $\mathtt{CosSim}(\mathbf{Q}, \mathbf{C})$ is calculated in Eq.(11). Upon maximizing the cosine similarity, a query $\mathbf{Q}$ is thus classified into the class C_k as shown in Eq.(12):

$$\mathtt{label}(\mathbf{Q}) = \arg\max_{C_k\in\mathbf{C}} \left(\mathtt{CosSim}(\mathbf{Q}, C_k)\right) \tag{12}$$

In conclusion, BOSS VS algorithm of which foundation is composed of two notions: Bag-of-words and TF-IDF. What makes BOSS VS be different from other algorithms is a way of taking features of data. This algorithm does not construct a loss function like other machine learning algorithms but simply use Bag-of-Words instead. With time series are transformed into sequences of symbols, Bag-of-words approaches are then used to extract features from these sequences. Time series are presented as histograms with designated symbols. And then each histogram is transformed into TF-IDF vectors for classification. What we have discussed for building a model is quite involved, thereby we sorted out the procedure step by step with a lookup table. Table 1 displays the lookup table for the probabilistic models and corresponding algorithms.

Results and Discussion

Experiments

An experimental apparatus is a three-blade fan on the wheels with a low-power digital accelerometer made in Analog Device (ADXL345 GY-80) installed as shown in Fig. 8. The dimension of the apparatus is the width 18.5 mm, length of 12.3 mm, and height of 30 mm. For classification, we considered three of the most probable states of the fan we can think of in a real-world situation: The normal state where the fan running without any noticeable event (see left pane in Fig. 8), the counter-wind state in which occurrence of intermittency of counter-wind against the fan takes place (see center pane in Fig. 8), and the mechanical failure state where one of the blades is broken off (see right pane in Fig. 8). The average rotational speed of the fan was 114 rpm at normal state, 41 rpm at counter-wind state, and 107 rpm at mechanical-failure state, respectively.

Each sample was collected at a sampling rate of 100 Hz for 3 seconds from the accelerometer, so the length of each sample is 300. For example, 900 samples for each state of the fan were collected via x and y channels, so the number of data points sums 900 samples × 2 channels × 300 × 3 states = 1, 620, 000. It took 2 hours and 15 minutes for collecting 1,620,000 of data points at each measurement. The samples of the experimental data are shown as a set of time series along with mean and strandard deviation into three states in Fig.9.

Exploratory Data Analysis

Exploratory Data Analysis (EDA) is carried out to identify the property of data. In Fig. 9, the raw data, its rolling mean, and the standard deviation are overlaid. Since raw data contains much noise, it is necessary to filtered out for a better analysis. The rolling mean is one such filtering tool. The standard deviation can be used for estimating variance of data.

In addition, we need to know how much trends, repetition over intervals, white noise, and another uncertainty are. These characteristics of data should be never taken lightly because the authenticity of the experiment can be determined by them. We employed the Augmented Dickey-Fuller (ADF) test with the null hypothesis of whether it being stationarity. The test results of p-value ≤ 0.05 as shown in Table 2 for the three states with all six time-series from experimental data; therefore, we can reject the null hypothesis at a 95% confidence level. Thus, we may affirm that the experiment data is stationary.

Comparison of Models

We empoyed five diferent models: WEASEL MUSE, BOSS VS, random forest (RF), logistic regression (LR), and one-nearest neibor DTW (1-NN DTW). Table 3 describes characteristics of five models according to temporal structure (Temporal), low-pass filtering (Filter), transformation (Transform), and key features (Features). Only 1-NN DTW keeps the temporal structure of data, and the others do not consider the order of data points over time. Algorithms for feature extraction are χ^2 test for WEASEL MUSE, Term-Frequency Inverse Document Frequency algorithm for BOSS VS, Entropy for RF, the cost function for LR, and Euclidean distance for 1-NN DTW.

Classification with BOSS VS

Table 4 shows the numerical expression of the trained model $p(\mathbf{C}|\mathbf{T})$ in Table 1, which is the result of vector $\texttt{tfidf}(\mathbf{S}, \mathbf{C})$ calculated using training data. The symbolic algorithm SFA converts the whole training data to $\mathbf{S} = \{\mathbf{aa}, \mathbf{ab}, \mathbf{ba}, \mathbf{bb}\}$. For example, the features of the normal state (C_1) are **aa**, **ab**, **ba**, and **bb** with the numerical values (3.5649, 3.5649, 3.5649, 3.6390) as displayed in Table 4. For the counter-wind state (C_2), the value reads (3.1972, 2.9459, 3.3025, 3.3025), which is clearly distinguished from those of the normal state.

Table 5 shows the classifier $p(\mathbf{C}|\mathbf{Q})$ in Table 1 for $\mathbf{Q}$. For example, the first sample Q_1 is predicted as the normal state because of the largest value of 0.9990 throughout the column to which it belongs. In the same fashion, the classification is performed for the remaining time series such as the counter-wind state for Q_2, and the mechanical failure state for Q_5 and so on.

Post-Hoc Analysis

Often in many studies, the results tend to be presented without statistical rigor. However, it is important to check if it being statistically significant before further discussion, which is called Post-Hoc analysis.

As shown in Table 6, the BOSS VS model indicates the highest accuracy for all data sizes. In addition, even though the number of data is increased from 180 k to 1.6 million, it shows a little change in accuracy, so we may conclude that the BOSS VS model is not significantly affected by the size of data. The smaller the number of data used, the shorter the run time, but on the other hand, the model tends to be overfitted to the data. For example, in scenario I, 180,000 data points was used; for the BOSS VS model, the accuracy turned out 100%. This can clearly indicates overfitting, where too little data was used for training. If the number of data is increased, the time for preprocessing and calculation also increases accordingly. Therefore, it is necessary to manipulate the data size so that Fog computing may handle a whole process within a designated time. In this study, we put a time limit on the simulation, where the processing time t_{ML} does not exceed 1/10 of the data collection time t_{obs}.

Table 6 does not tell whether the difference in the accuracy of each model is statistically meaningful. Another ambiguity arises in the results of the run time. Thus, we carried out the ANOVA (Analysis of Variance) test that provides statistical significance among differences. Table 7 shows the ANOVA test result for accuracy with $F = 60.8$, and p-value ≤ 0.05 at a 95% confidence level. This indicates that the null hypothesis is rejected. Therefore, it can be said that the mean values for the accuracy of each model differ significantly. In addition, the difference in run time for each model is statistically significant, with $F = 4.58$, and p-value $= 0.008$. In conclusion, the simulation over five scenarios for accuracy and run time with five models can be confirmed to be statistically significant.

However, the ANOVA test results shown in Table 7 alone cannot tell which model is different in accuracy and run time from those of others. Thereby, another test should be carried out to see which model is significantly different from the others. We employed Tukey's Honest Significant Difference test (Tukey Test) for all pariwise comparisons while controlling multiple comparisons. In this study, the suitability of models was sought statistically in two aspects: accuracy and scalability.

Accuracy

The result of the Tukey Test, which being multiple comparisons of means of accuracy from five models, is summarized in Table 8. It is noted that two cases, 1-NN DTW vs WEASEL MUSE and LR vs RF, are not statistically significant upon a 95% confidance level. This implies that two cases may have a great similarity in the way to make poor predictions. On the contrary, all pairwise comparisons with the BOSS VS model are proven statistically significant at a 95% confidence level. Figure 10 shows yet another aspect of the trend of accuracy over run time for all five models, where the BOSS VS model outputs a far great performance in both accuracy and run time.

Scalability

In general, a scalable model shows consistently good performance despite an increase in the amount of data. Multiple comparisons of run time for five models are summarized in Table 9. All pairwise cases for 1-NN DTW model vs the other models turn out to be significant. Thus, we may conclude that 1-NN DTW model is far less scalable. Figure 11 shows the scalability of models in which except the 1-NN DTW model the other models keep relatively small changes in the runtime subject to increasing data size. Figure 12 shows the comparison of the 95% confidence interval (CI) of accuracy of each model using experimental data of different sizes. The accuracy of the BOSS VS model fell into CI $= 0.9872 \pm 0.0073$, of which statistical behavior is much better compared to CI $= 0.8205 \pm 0.1319$ in the second-place RF model. Moreover, the deviation 0.0073 of the BOSS VS model is quite small compared to 0.1319 that of the RF model. This explains good scalability, which indicates that the BOSS VS model is robust to changes in data size.

Conclusion

Analyzing a huge amount of data transmitted in real time from a networked IoT device, which is a core of smart manufacturing, to properly classify the state of the device is interesting and practically important. Recently, there has been a growing tendency to solve this problem not only in cloud computing but rather at Fog computing close to IoT devices. To this end, two issues must be resolved: one is is of a cyber-physical model that can represent the state of an IoT device, and the other is about how to properly process streaming sensor data in real-time.

A major goal in this study is to build a good cyber-physical model with significant accuracy in classification. Taking advantage of machine learning and statistical inference with a vast amount of data, data-driven modeling approach can alienate quite a burden of such complicated theoretical domain-specific knowledge. While most literature publishes the results of the simulation using well-preprocessed public data, in this study, we implemented noisy real-world data. A three-blade fan with an accelerometer installed is considered for an IoT device to create a cyber-physical model that can classify the state of the fan into three states. Using several algorithms including the most recent out ones, upon the classification performance of five models with real-world data; we achieved an accuracy of about 98% with BOSS VS model.

For further studies, we need to challenge a couple of tasks for better accuracy and scalability. Thus, more studies should be conducted for efficient models and algorithms with machine learning against ever-increasing sensors at smart manufacturing, in order not to wholly depending on cloud computing.

Acknowledgements

The present research was supported by the research fund of Dankook university in 2021.

Availability of data and materials

Experimental data are included.

Competing interests

The authors declare that they have no competing interests.

Authors' information

The author is professor of deparment of mechanical engineering at Dankook University, Korea. His area of research is application of machine learning into smart manufacturing and application on various computing enviroments such as cloud computing, fog computing , and on-device computing.

References

1. Kang, H.S., Lee, J.Y., Choi, S., Kim, H., Park, J.H., Son, J.Y., Kim, B.H., Do Noh, S.: Smart manufacturing: Past research, present findings, and future directions. International journal of precision engineering and manufacturing-green technology **3**(1), 111–128 (2016)
2. Herrmann, C., Schmidt, C., Kurle, D., Blume, S., Thiede, S.: Sustainability in manufacturing and factories of the future. International Journal of precision engineering and manufacturing-green technology **1**(4), 283–292 (2014)
3. Manyika, J., Chui, M., Bisson, P., Woetzel, J., Dobbs, R., Bughin, J., Aharon, D.: Unlocking the potential of the internet of things. McKinsey Global Institute (2015)
4. Mohammadi, M., Al-Fuqaha, A., Sorour, S., Guizani, M.: Deep learning for iot big data and streaming analytics: A survey. IEEE Communications Surveys Tutorials **20**(4), 2923–2960 (2018)
5. Bonomi, F., Milito, R., Zhu, J., Addepalli, S.: Fog computing and its role in the internet of things. In: Proceedings of the First Edition of the MCC Workshop on Mobile Cloud Computing, pp. 13–16 (2012)
6. Dastjerdi, A.V., Buyya, R.: Fog computing: Helping the internet of things realize its potential. Computer **49**(8), 112–116 (2016)
7. Stojmenovic, I., Wen, S.: The fog computing paradigm: Scenarios and security issues. In: 2014 Federated Conference on Computer Science and Information Systems, pp. 1–8 (2014). IEEE
8. Yi, S., Hao, Z., Qin, Z., Li, Q.: Fog computing: Platform and applications. In: 2015 Third IEEE Workshop on Hot Topics in Web Systems and Technologies (HotWeb), pp. 73–78 (2015). IEEE
9. Gassais, R., Ezzati-Jivan, N., Fernandez, J.M., Aloise, D., Dagenais, M.R.: Multi-level host-based intrusion detection system for internet of things. Journal of Cloud Computing **9**(1), 1–16 (2020)
10. Zatwarnicki, K.: Two-level fuzzy-neural load distribution strategy in cloud-based web system. Journal of Cloud Computing **9**(1), 1–11 (2020)
11. Oks, S.J., Jalowski, M., Fritzsche, A., Möslein, K.M.: Cyber-physical modeling and simulation: A reference architecture for designing demonstrators for industrial cyber-physical systems. Procedia CIRP **84**, 257–264 (2019)
12. Xu, Z., Zhang, Y., Li, H., Yang, W., Qi, Q.: Dynamic resource provisioning for cyber-physical systems in cloud-fog-edge computing. Journal of Cloud Computing **9**(1), 1–16 (2020)
13. Žliobaitė, I., Bifet, A., Read, J., Pfahringer, B., Holmes, G.: Evaluation methods and decision theory for classification of streaming data with temporal dependence. Machine Learning **98**(3), 455–482 (2015)
14. Keogh, E., Chakrabarti, K., Pazzani, M., Mehrotra, S.: Dimensionality reduction for fast similarity search in large time series databases. Knowledge and information Systems **3**(3), 263–286 (2001)
15. Salvador, S., Chan, P.: Toward accurate dynamic time warping in linear time and space. Intelligent Data Analysis **11**(5), 561–580 (2007)
16. Senin, P., Malinchik, S.: Sax-vsm: Interpretable time series classification using sax and vector space model. In: 2013 IEEE 13th International Conference on Data Mining, pp. 1175–1180 (2013). IEEE
17. Schäfer, P., Högqvist, M.: Sfa: a symbolic fourier approximation and index for similarity search in high dimensional datasets. In: Proceedings of the 15th International Conference on Extending Database Technology, pp. 516–527 (2012)
18. Karimi-Bidhendi, S., Munshi, F., Munshi, A.: Scalable classification of univariate and multivariate time series. In: 2018 IEEE International Conference on Big Data (Big Data), pp. 1598–1605 (2018). IEEE
19. Raza, A., Kramer, S.: Accelerating pattern-based time series classification: a linear time and space string mining approach. Knowledge and Information Systems **62**(3), 1113–1141 (2020)
20. Schäfer, P.: The boss is concerned with time series classification in the presence of noise. Data Mining and Knowledge Discovery **29**(6), 1505–1530 (2015)
21. Schäfer, P., Leser, U.: Fast and accurate time series classification with weasel. In: Proceedings of the 2017 ACM on Conference on Information and Knowledge Management, pp. 637–646 (2017)
22. Wang, Z., Yan, W., Oates, T.: Time series classification from scratch with deep neural networks: A strong baseline. In: 2017 International Joint Conference on Neural Networks (IJCNN), pp. 1578–1585 (2017). IEEE
23. Karim, F., Majumdar, S., Darabi, H., Chen, S.: Lstm fully convolutional networks for time series classification. IEEE access **6**, 1662–1669 (2017)
24. Schäfer, P.: Scalable time series classification. Data Mining and Knowledge Discovery **30**(5), 1273–1298 (2016)
25. Hatami, N., Gavet, Y., Debayle, J.: Bag of recurrence patterns representation for time-series classification. Pattern Analysis and Applications **22**(3), 877–887 (2019)

Figures

Figure 1 Three levels of IoT system. As the top-level, IoT employing cloud system associated with big data analytics. Fog computing resides in the middle equipped with fast streaming data analytics. At the bottom level, IoT devices such as sensors are located with consecutive temporal data requiring real-time data analytics.

Figure 2 The workflow of Fog computing in the present study. Fog computing is composed of three distinctive modes in a single time sequence: observation, machine learning, and status update. Observation phase executes streaming data storing. Machine Learning phase does data processing and learning the model. Status update phase carries out a classification of a fan status with the trained model. A red box refers to a sliding window corresponding to a process of the workflow on the timeline. For example, the box which is on the second from the left indicatese data storing. Likewise, the fifth box from the left is of learning the model.

Figure 3 A cyber-physical model is expected to well explain the effect of real-world elements such as acoustic noise, mechanical failure, temperatue change, revolution (rpm), pressure distribution of blades, vibration, counter-wind occurence, and wind velocity etc.

Figure 4 Low-pass filtering and smoothing of a sample of acceleration in x-axis upon Discrete Fourier Transform (DFT). In this plot, DFT result is obtained by taking only first two Fourier coefficients.

Figure 5 A pictorial diagram of symbolic Fourier approximation (SFA) procedure: a) Incoming sensor data of six time-series, b) The data is then transformed via Fourier transform, c) The Fourier coefficients are quantizied via Multiple Coefficient Binning (MCB), and d) Each time series has been mapped into its respective SFA word.

Figure 6 BOSS model and BOSS VS: a) Samples are being scanned with a sliding window, b) multiple windowed subsequences are generated, c) all of the subsequences are transformed into SFA words, d) SFA words are summarized in the form of BOSS histogram (BOSS model), and e) the BOSS histogram is vectorized through Term Frequency Inverse Document Frequency (TF-IDF) model, which finally results in TF-IDF vectors for training data.

Figure 7 Schematic diagram of classification with the consine similarity: a) New data for query is first transformed into SFA words, b) the SFA words of the new data is tranformed into the BOSS histogram, c) the trained model in the form of tf-idf algorithm is given, and d) the classificaiton is carried out through calculating the cosine similarity between the trained model and the query.

Figure 8 Photos of the three-blade fan in the three states: Normal state (left), Counter-wind state (center), and Mechanical failure state (right). The counter-wind state indicates the state where counter-wind being blown by another fan in front of the fan. The mechanical failure refers to the state in which one of the blades having been removed off.

Figure 9 Experimental time series data for three states of the fan: Normal state (top row), counter-wind state (middle row), and mechanical failure state (bottom row). Raw data from the accelerometer overlaid with the rolling mean and standard deviation. Each row represents both x (left) and y (right) acceleration in g unit.

Figure 10 Accuracy comparison of five models (WEASEL MUSE, BOSS VS, Random Forest, Logistic Regression, and 1-Nearest-Neighbor DTW). 1-NN DTW model shows the worst performance both in accuracy and run time. On the contrary, the BOSS VS model shows excellent accuracy over the others. Note: the upper left being the overall best performance.

Figure 11 Scalability comparison of five models. As the amount of data is increased, the 1-NN-DTW model shows the worst scalability. On the contrary, the other models show reasonable scalability. The BOSS VS model performs excellent scalability yet keeping the best accuracy.

Figure 12 The result of comparing the 95% confidence interval (CI) of the accuracy of five models using five scenarios of data size. This illustates the scalability of each model's performance in classification. The accuracy of the BOSS VS model fell into CI $= 0.9872 \pm 0.0073$ resulting in the best performance.

Tables

Table 1 Lookup table for the probabilistic models and corresponding algorithms.

Probability	$p(\mathbf{T}\|\mathbf{C})$	$p(\mathbf{C})$	$p(\mathbf{C}\|\mathbf{T})$	$p(\mathbf{Q}\|\mathbf{C})$	$p(\mathbf{C}\|\mathbf{Q})$
Algorithm	$\texttt{tf}(\mathbf{S},\mathbf{C})$	$\texttt{idf}(\mathbf{S},\mathbf{C})$	$\texttt{tfidf}(\mathbf{S},\mathbf{C})$	$\texttt{tf}(\mathbf{Q},\mathbf{C})$	$\texttt{CosSim}(\mathbf{Q},\mathbf{C})$
Note	Model	Prior	Trained Model	Transformed	Classifier

Table 2 Augmented Dickey-Fuller (ADF) Test Results: The results show that all of six time-series are found to be stationary of being statistically significant owing to p-value ≤ 0.05 (95% confidence level), which indicates evidence against the null hypothesis H_0.

Time Series	ADF statistic	P-value	Critical Value (5%)
x-acc (Normal)	-4.629	1.1×10^{-4}	-2.871
y-acc (Normal)	-6.137	8.1×10^{-8}	-2.871
x-acc (Counter Wind)	-6.486	1.2×10^{-8}	-2.871
y-acc (Counter Wind)	-5.839	3.8×10^{-7}	-2.871
x-acc (Mechanical Failure)	-4.577	1.4×10^{-4}	-2.871
y-acc (Mechanical Failure)	-4.459	2.3×10^{-4}	-2.871

Table 3 Comparison of characteristics of five models via conducting normalization (Norm.), keeping temporal structure (Temporal), carrying out low-pass filtering (Filter), executing transformation (Transform), and key features (Features).

	Temporal	Filter	Transform	Features
WEASEL MUSE	No	Yes	Yes	χ^2 test
BOSS VS	No	Yes	Yes	Term Frequency
RF	No	No	No	Entropy
LR	No	No	No	Cost
1-NN DTW	Yes	No	No	Distance

Table 4 The trained model $p(\mathbf{C}|\mathbf{T})$ with equivalent of TF-IDF vector $\texttt{tfidf}(\mathbf{S},\mathbf{C})$ for the training data. $\mathbf{S} = \{\mathbf{aa}, \mathbf{ab}, \mathbf{ba}, \mathbf{bb}\}$ is the SFA words and $\mathbf{C} = \{C_1, C_2, C_3\}$ is three states of the fan.

Class	C_1	C_2	C_3
	Normal	Counter Wind	Mechanical Failure
$\mathbf{aa}$	3.5649	3.1972	2.9459
$\mathbf{ab}$	3.5649	2.9459	2.3862
$\mathbf{ba}$	3.5649	3.3025	2.0986
$\mathbf{bb}$	3.6390	3.3025	2.3862

Table 5 The classifier $p(\mathbf{C}|\mathbf{Q})$ with equivalent to Cosine similarity between the trained model $p(\mathbf{C}|\mathbf{T})$ for each class and new samples $\mathbf{Q} = \{Q_1, \ldots, Q_6\}$ as a query. $\mathbf{C} = \{C_1, C_2, C_3\}$ is three states of the fan. The similarity resutls in the prediction for the new samples. The maximum value of the cosine similarity for each sample is boldfaced.

	Q_1	Q_2	Q_3	Q_4	Q_5	Q_6
Normal	**0.9990**	0.9958	0.9987	**0.9963**	0.9943	0.9970
Counter Wind	0.9964	**0.9977**	**0.9988**	0.9942	0.9909	**0.9991**
Mechanical Failure	0.9908	0.9791	0.9924	0.9855	**0.9985**	0.9868

Table 6 Comparison of performace of five models according to five scenarios: The number of data points is increased from 180k up to 1.6 Million. As for Fog computing: The processor: Intel Core i3-8100 CPU 3.60GHz at 4GB memory, and the OS being Ubuntu 20.04.1 LTS. BOSS VS model shows excellent scalability while keeping the highest accuracy among the other models.

Scenario	Data points (samples)	Model	t_{obs} (sec)	t_{ML} (sec)	Accuracy
I	180,000 (300)	WEASEL MUSE	900	0.398	0.200
		BOSS VS	900	0.464	1.000
		RF	900	0.443	0.567
		LR	900	0.253	0.633
		1-NN DTW	900	10.409	0.266
II	540,000 (900)	WEASEL MUSE	2,700	1.009	0.333
		BOSS VS	2,700	1.422	0.977
		RF	2,700	1.163	0.822
		LR	2,700	2.540	0.655
		1-NN DTW	2,700	89.546	0.377
III	900,000 (1,500)	WEASEL MUSE	4,500	1.810	0.346
		BOSS VS	4,500	2.456	0.986
		RF	4,500	1.961	0.906
		LR	4,500	7.153	0.740
		1-NN DTW	4,500	246.794	0.400
IV	1,080,000 (1,800)	WEASEL MUSE	5,400	2.193	0.366
		BOSS VS	5,400	2.966	0.983
		RF	5,400	2.472	0.877
		LR	5,400	12.921	0.766
		1-NN DTW	5,400	352.067	0.411
V	1,620,000 (2,700)	WEASEL MUSE	8,100	3.851	0.370
		BOSS VS	8,100	4.667	0.988
		RF	8,100	3.910	0.929
		LR	8,100	32.183	0.711
		1-NN DTW	8,100	793.221	0.388

Table 7 ANOVA test result of accuracy and run time among five models.

	sum_{sq}	df	F	PR(>F)
Algorithms (Accuracy)	1.64	4.0	60.80	6.56×10^{-11}
Residual	0.13	20.0	-	-
Algorithms (Run time)	346268.91	4.0	4.58	0.008631
Residual	377628.33	20.0	-	-

Table 8 Multiple Comparison of Means of Accuracy - Tukey HSD test

Group1	Group2	meandiff	p-adj	lower	upper	reject
1-NN DTW	BOSS VS	0.614	0.001	0.458	0.770	True
1-NN DTW	LR	0.328	0.001	0.172	0.484	True
1-NN DTW	RF	0.447	0.001	0.292	0.603	True
1-NN DTW	WEASEL MUSE	-0.049	0.862	-0.205	0.106	False
BOSS VS	LR	-0.286	0.001	-0.441	-0.130	True
BOSS VS	RF	-0.166	0.032	-0.322	-0.011	True
BOSS VS	WEASEL MUSE	-0.663	0.001	-0.819	-0.508	True
LR	RF	0.119	0.187	-0.036	0.274	False
LR	WEASEL MUSE	-0.377	0.001	-0.533	-0.222	True
RF	WEASEL MUSE	-0.497	0.001	-0.652	-0.341	True

Table 9 Multiple Comparison of Means of run time - Tukey HSD

Group1	Group2	meandiff	p-adj	lower	upper	reject
1-NN DTW	BOSS VS	-296.01	0.020	-556.08	-35.93	True
1-NN DTW	LR	-287.39	0.025	-547.47	-27.32	True
1-NN DTW	RF	-296.41	0.020	-556.49	-36.34	True
1-NN DTW	WEASEL MUSE	-296.55	0.020	-556.62	-36.48	True
BOSS VS	LR	8.61	0.900	-251.45	268.68	False
BOSS VS	RF	-0.405	0.900	-260.47	259.66	False
BOSS VS	WEASEL MUSE	-0.542	0.900	-260.61	259.53	False
LR	RF	-9.02	0.900	-269.09	251.05	False
LR	WEASEL MUSE	-9.15	0.900	-269.23	250.91	False
RF	WEASEL MUSE	-0.137	0.900	-260.21	259.93	False

Additional Files

Additional files

Experimental Raw Data file ($statefan.csv$)

All figure files

Figures

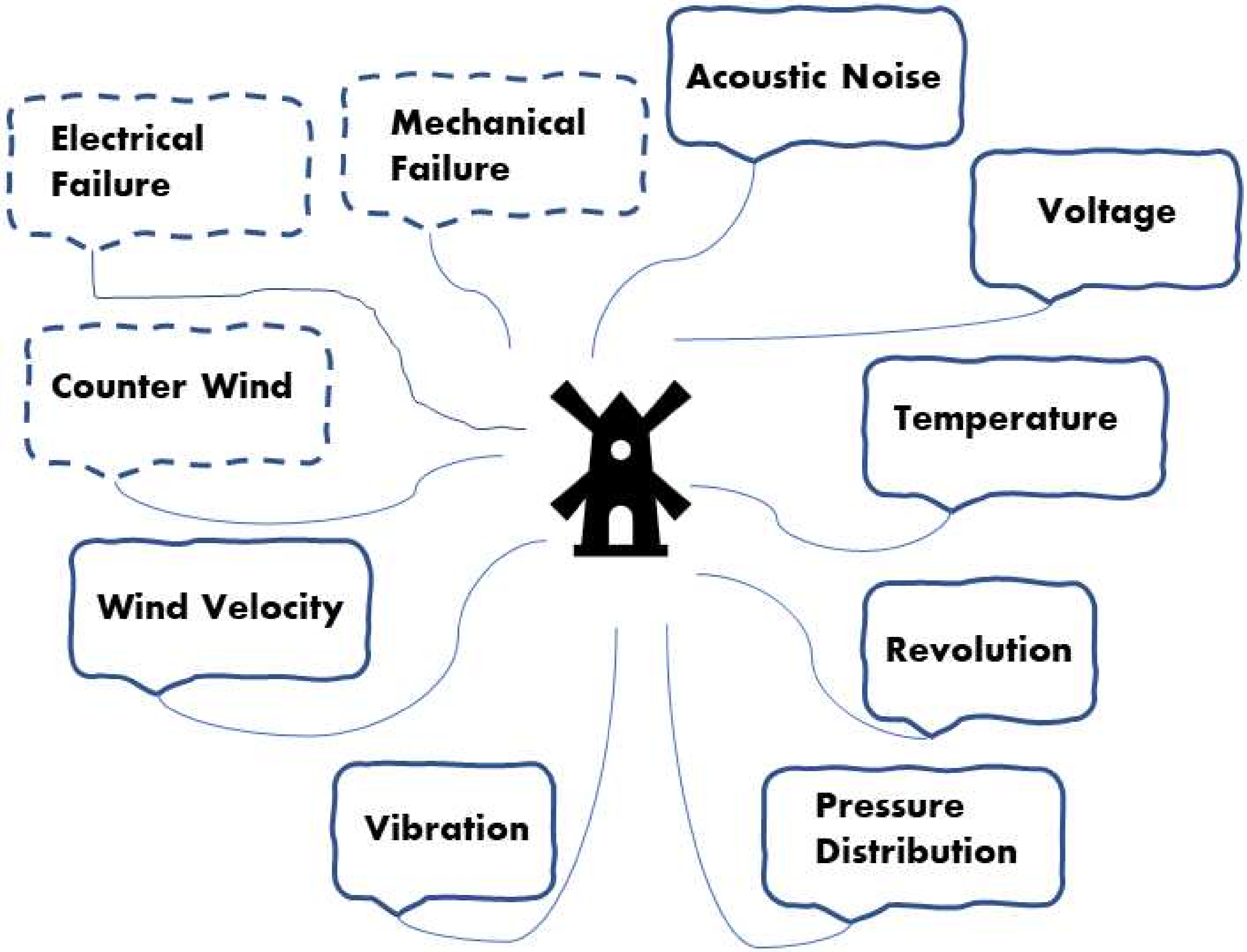


Figure 1

Three levels of IoT system. As the top-level, IoT employing cloud system associated with big data analytics. Fog computing resides in the middle equipped with fast streaming data analytics. At the bottom level, IoT devices such as sensors are located with consecutive temporal data requiring real-time data analytics.

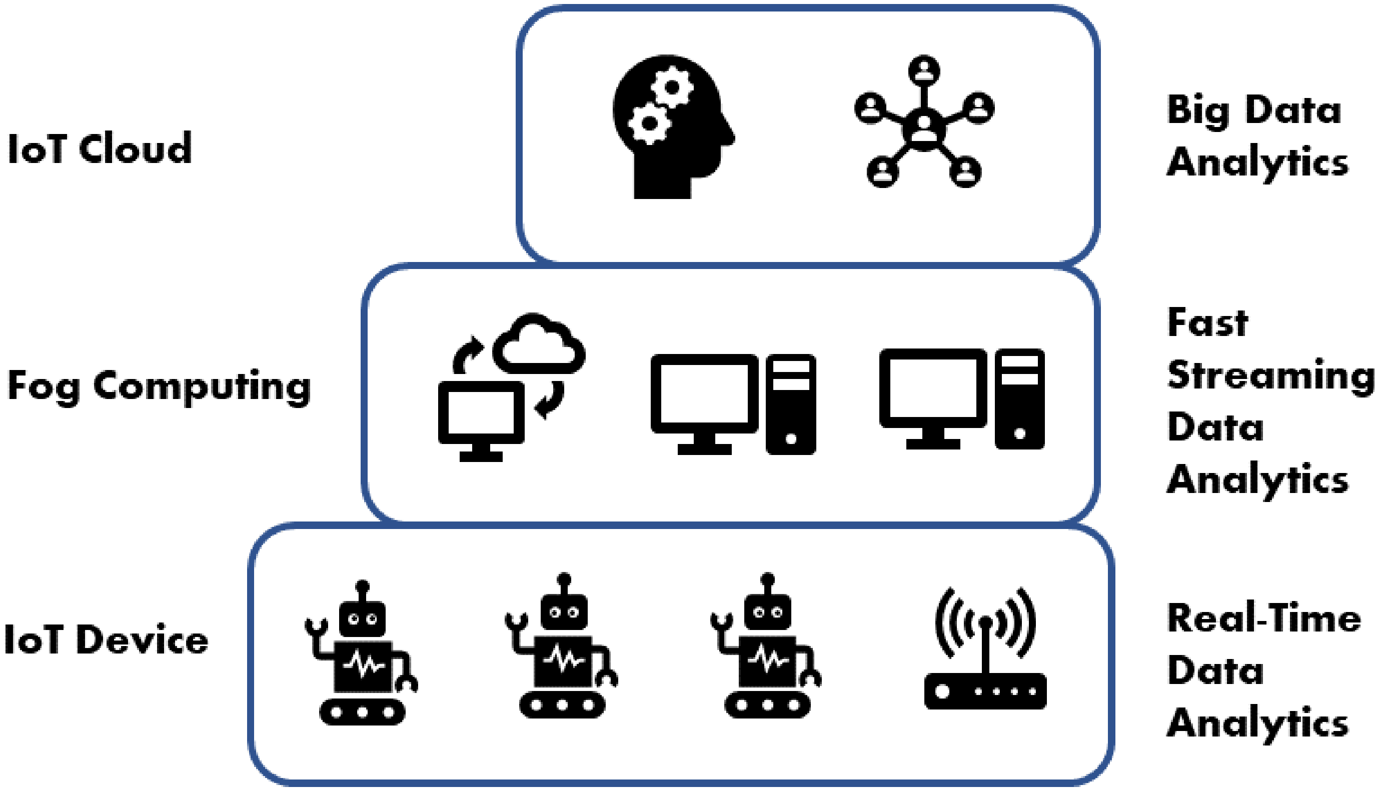


Figure 2

The workflow of Fog computing in the present study. Fog computing is composed of three distinctive modes in a single time sequence: observation, machine learning, and status update. Observation phase executes streaming data storing. Machine Learning phase does data processing and learning the model. Status update phase carries out a classification of a fan status with the trained model. A red box refers to a sliding window corresponding to a process of the workflow on the timeline. For example, the box which is on the second from the left indicatese data storing. Likewise, the fifth box from the left is of learning the model.

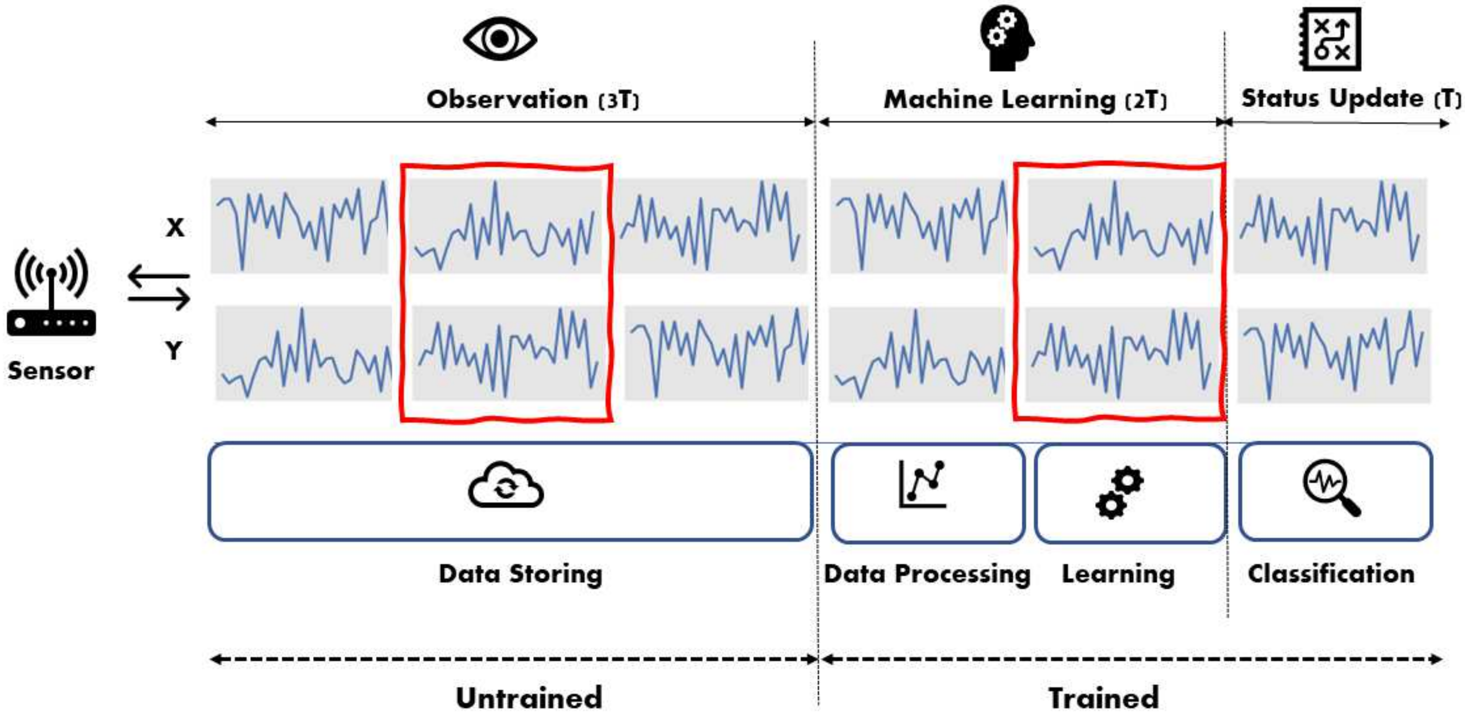


Figure 3

A cyber-physical model is expected to well explain the effect of real-world elements such as acoustic noise, mechanical failure, temperatue change, revolution (rpm), pressure distribution of blades, vibration, counter-wind occurence, and wind velocity etc.

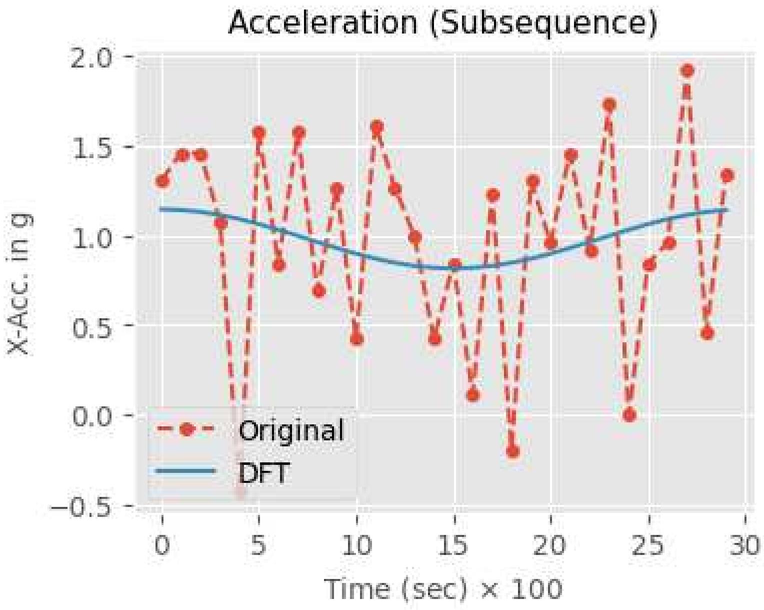
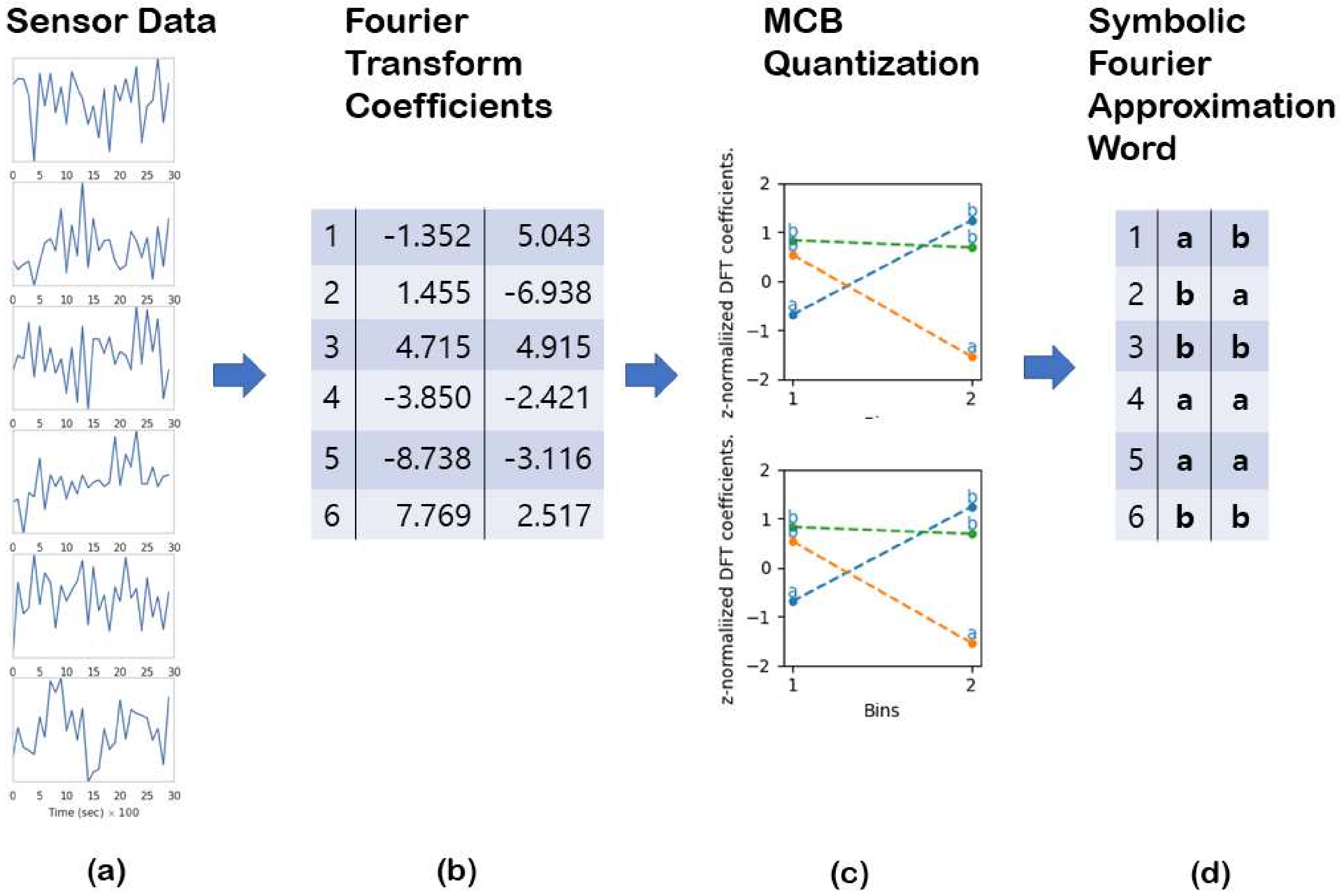


Figure 4

Low-pass filtering and smoothing of a sample of acceleration in x-axis upon Discrete Fourier Transform (DFT). In this plot, DFT result is obtained by taking only first two Fourier coefficients.

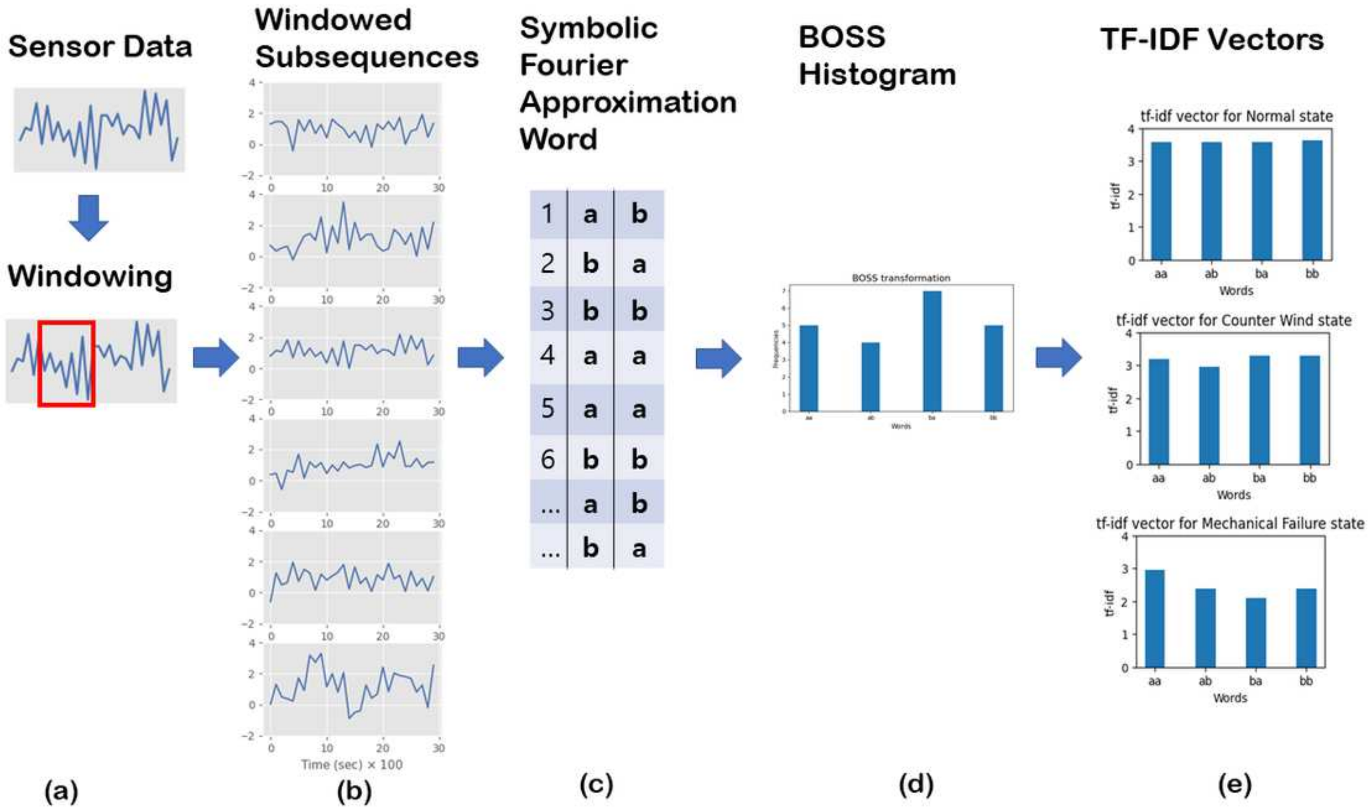

Sensor Data
0 5 10 15 20 25 30
Time (sec) × 100
Fourier Transform Coefficients
MCB Quantization
z-normalized DFT coefficients.
2
1
0
−1
−2
1
2
Bins
a
b
Symbolic Fourier Approximation Word
(a)
(b)
(c)
(d)

1	-1.352	5.043
2	1.455	-6.938
3	4.715	4.915
4	-3.850	-2.421
5	-8.738	-3.116
6	7.769	2.517

1	a	b
2	b	a
3	b	b
4	a	a
5	a	a
6	b	b

Figure 5

A pictorial diagram of symbolic Fourier approximation (SFA) procedure: a) Incoming sensor data of six time-series, b) The data is then transformed via Fourier transform, c) The Fourier coefficients are quantizied via Multiple Coefficient Binning (MCB), and d) Each time series has been mapped into its respective SFA word.



1	a	b
2	b	a
3	b	b
4	a	a
5	a	a
6	b	b
...	a	b
...	b	a

Figure 6

BOSS model and BOSS VS: a) Samples are being scanned with a sliding window, b) multiple windowed subsequences are generated, c) all of the subsequences are transformed into SFA words, d) SFA words are summarized in the form of BOSS histogram (BOSS model), and e) the BOSS histogram is vectorized through Term Frequency Inverse Document Frequency (TF-IDF) model, which finally results in TF-IDF vectors for training data.

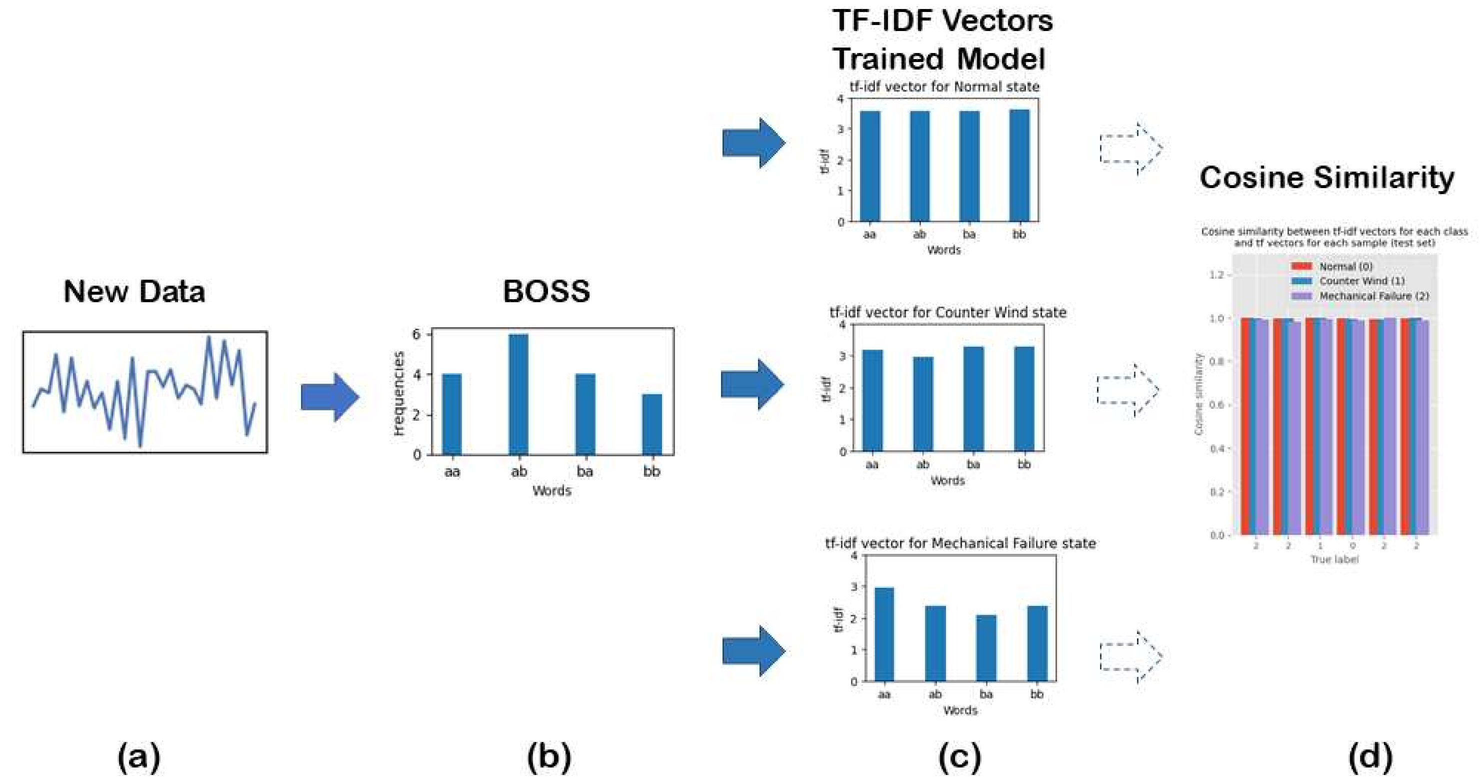


Figure 7

Schematic diagram of classification with the consine similarity: a) New data for query is first transformed into SFA words, b) the SFA words of the new data is tranformed into the BOSS histogram, c) the trained model in the form of tf-idf algorithm is given, and d) the classificaiton is carried out through calculating the cosine similarity between the trained model and the query.

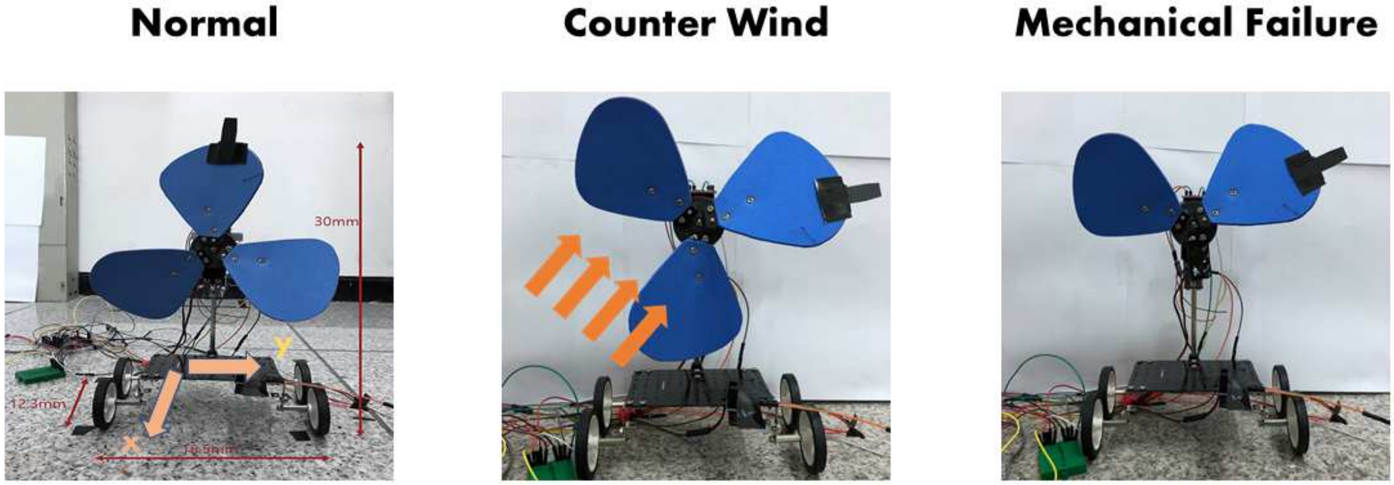
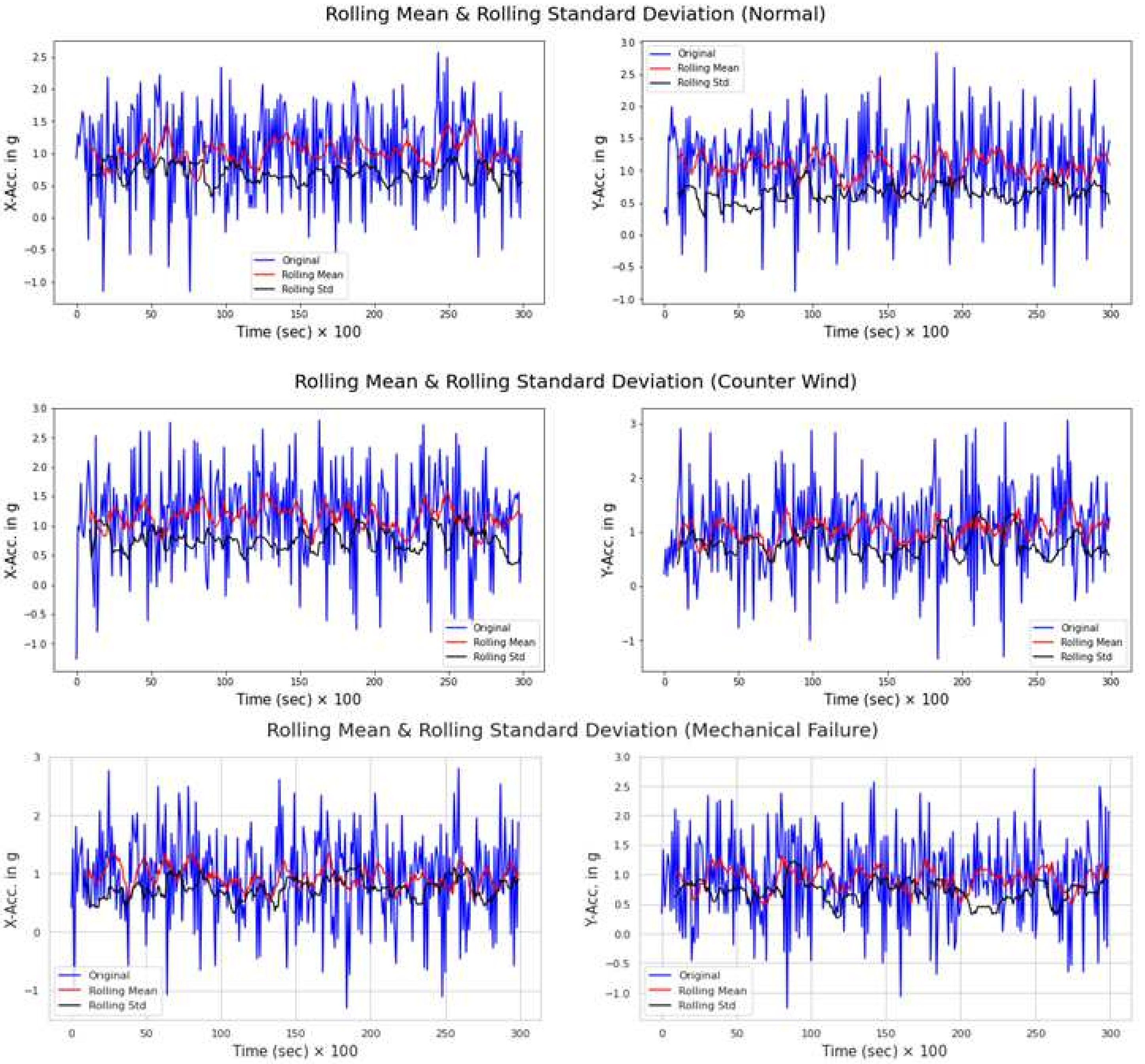


Figure 8

Photos of the three-blade fan in the three states: Normal state (left), Counter-wind state (center), and Mechanical failure state (right). The counter-wind state indicates the state where counter-wind being blown by another fan in front of the fan. The mechanical failure refers to the state in which one of the blades having been removed off.



Rolling Mean & Rolling Standard Deviation (Normal)

Rolling Mean & Rolling Standard Deviation (Counter Wind)

Rolling Mean & Rolling Standard Deviation (Mechanical Failure)

Figure 9

Experimental time series data for three states of the fan: Normal state (top row), counter-wind state (middle row), and mechanical failure state (bottom row). Raw data from the accelerometer overlaid with the rolling mean and standard deviation. Each row represents both x (left) and y (right) acceleration in g unit.

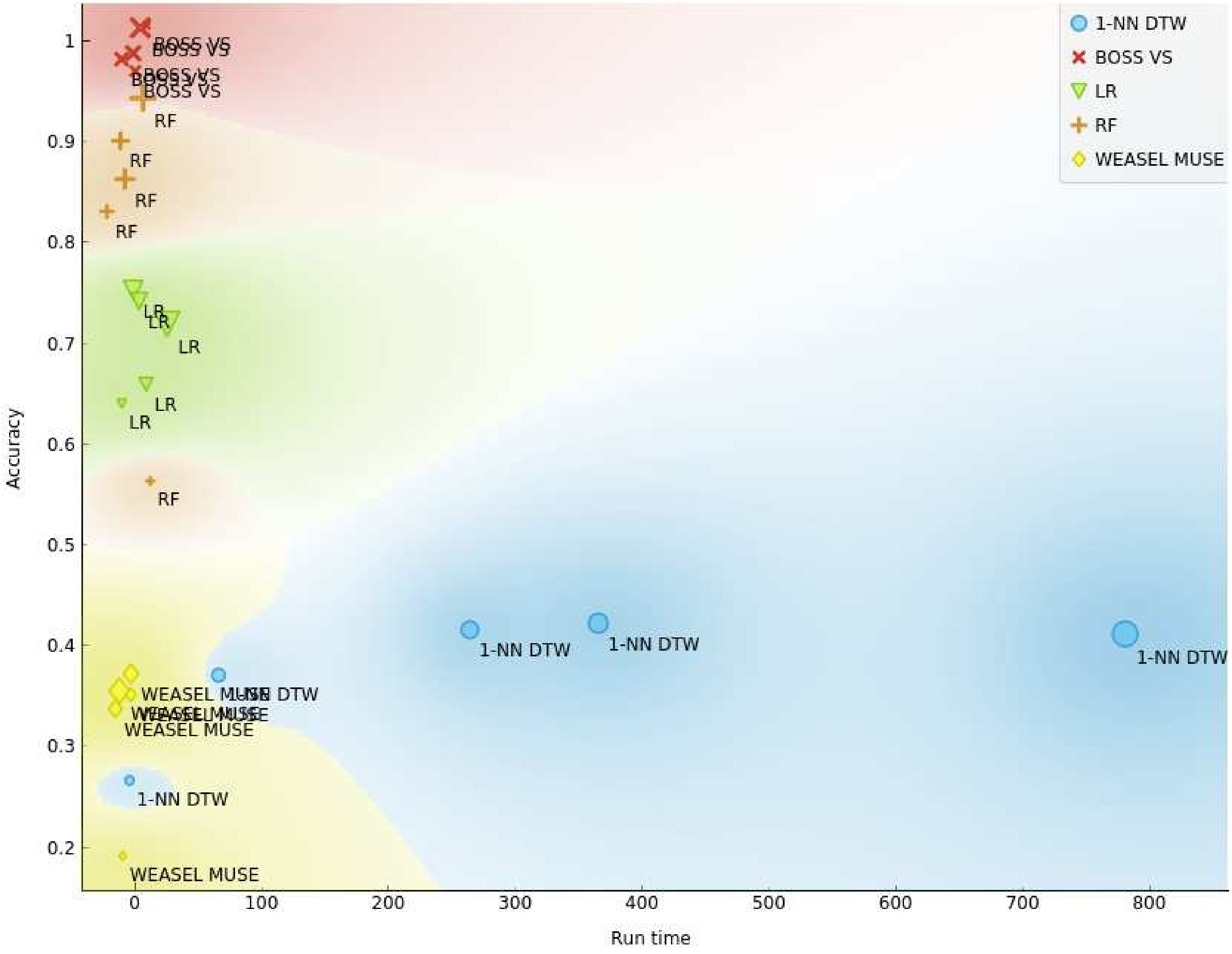


Figure 10

Accuracy comparison of five models (WEASEL MUSE, BOSS VS, Random Forest, Logistic Regression, and 1-Nearest-Neighbor DTW). 1-NN DTW model shows the worst performance both in accuracy and run time. On the contrary, the BOSS VS model shows excellent accuracy over the others. Note: the upper left being the overall best performance.

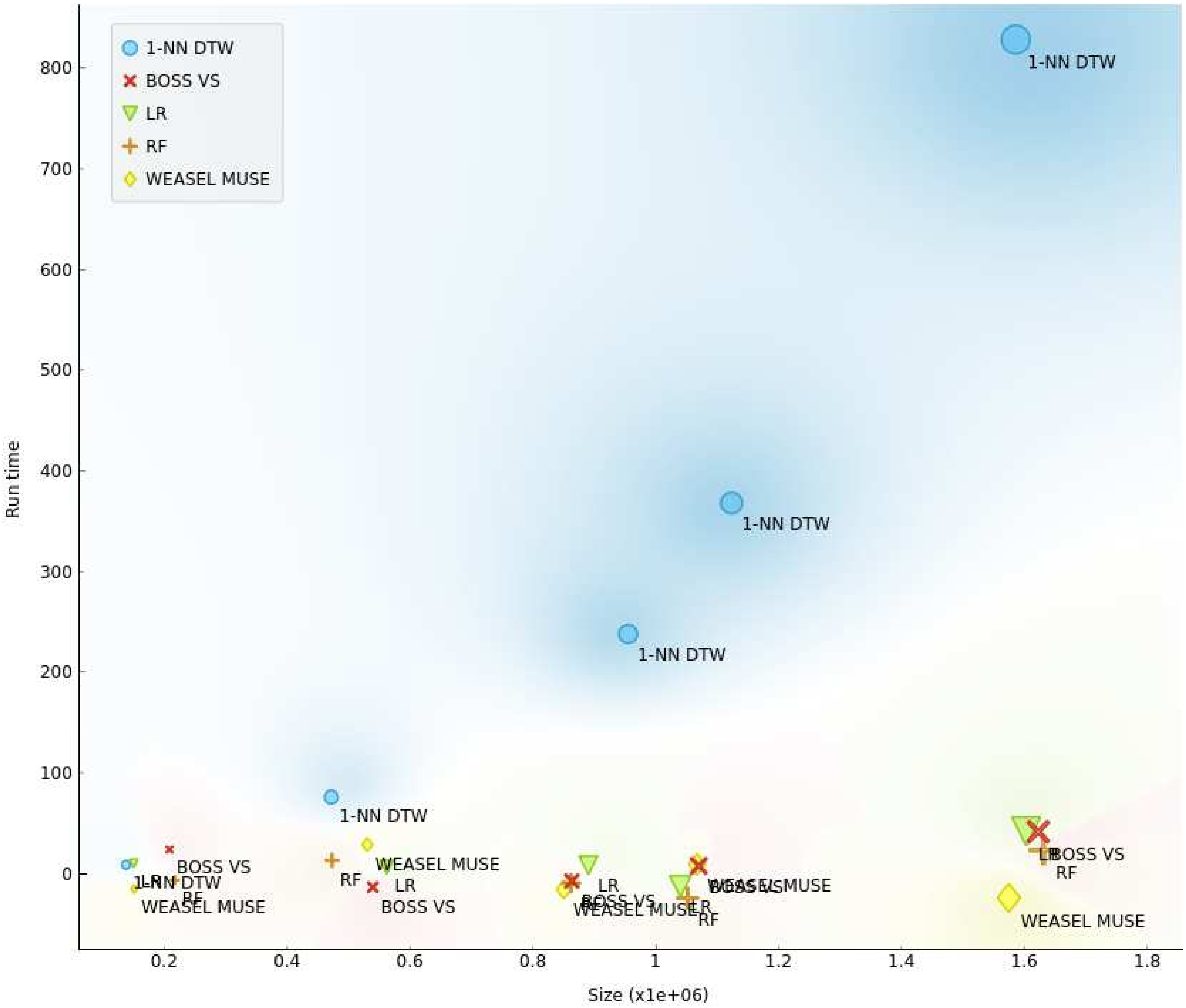


Figure 11

Scalability comparison of five models. As the amount of data is increased, the 1-NN-DTW model shows the worst scalability. On the contrary, the other models show reasonable scalability. The BOSS VS model performs excellent scalability yet keeping the best accuracy.

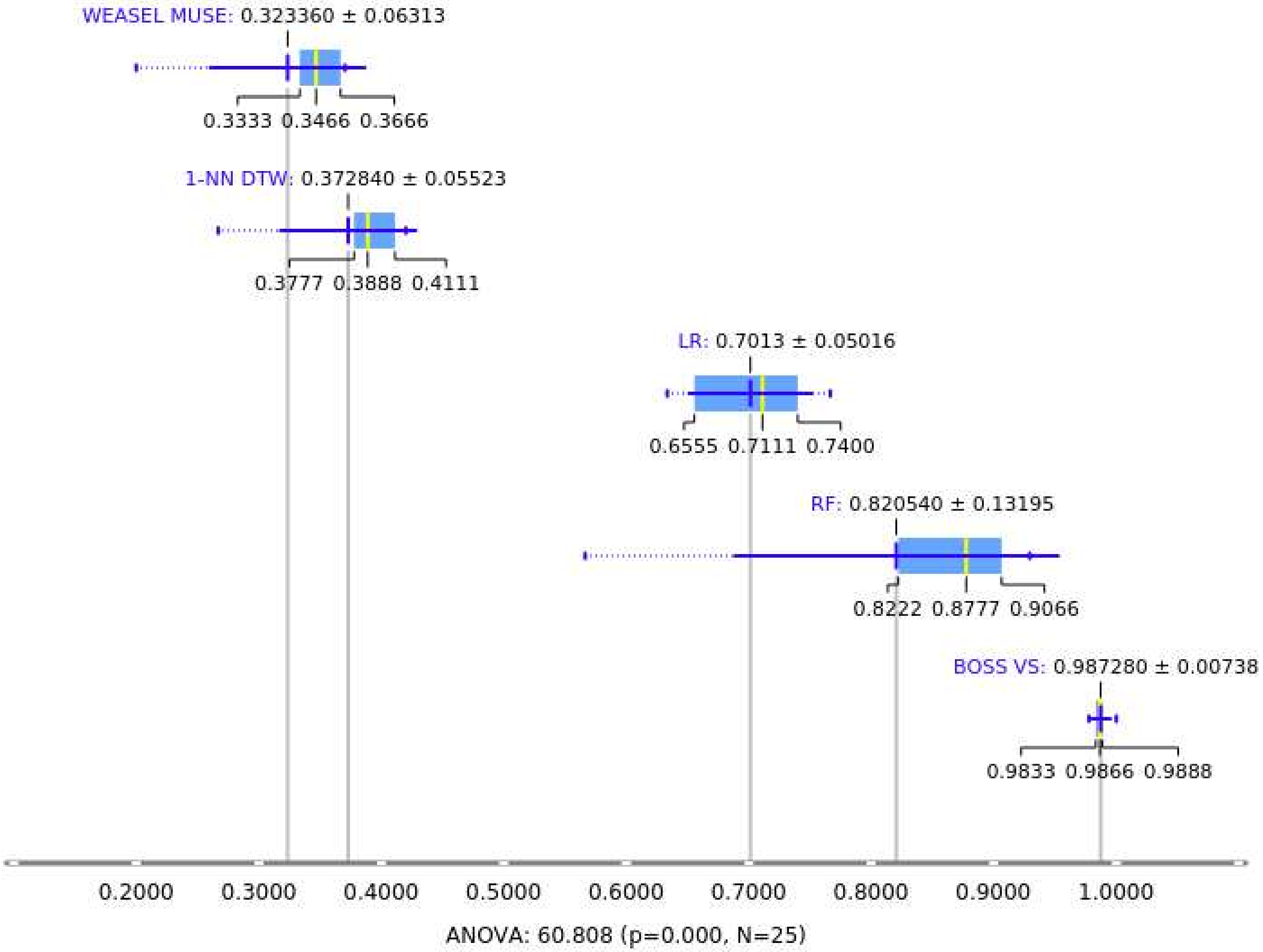


Figure 12

The result of comparing the 95% confidence interval (CI) of the accuracy of five models using five scenarios of data size. This illustates the scalability of each model's performance in classification. The accuracy of the BOSS VS model fell into CI = 0:9872 ± 0:0073 resulting in the best performance.

Supplementary Files

This is a list of supplementary files associated with this preprint. Click to download.

- statefan.csv