Ensemble Deep Learning for Detecting Onset of Abnormal Operation in Industrial Multi-component Systems

Balaji Selvanathan, Sri Harsha Nistala, Venkataramana Runkana*, Saurabh Jaywant Desai, and Shashank Agarwal

TCS Research, Tata Consultancy Services, Pune – 411028, Maharashtra, India

balaji.selvanathan@tcs.com
sriharsha.nistala@tcs.com
**venkat.runkana@tcs.com* (Corresponding Author)
sj.desai@tcs.com
shashank.agarwal2@tcs.com

ABSTRACT

Breakdowns and unplanned shutdowns in industrial processes and equipment can lead to significant loss of availability and revenue. It is imperative to perform optimal maintenance of such systems when signs of abnormal behavior are detected and before they propagate and lead to catastrophic failure. This is particularly challenging in systems with interconnected multiple components as it is difficult to isolate the effect of one component on the operation of other components in the system. In this work, an ensemble approach based on Cascaded Convolutional neural network and Long Short-term Memory (CC-LSTM) network models is proposed for detecting and predicting the time of onset of faults in interconnected multicomponent systems. The performance of the ensemble CC-LSTM model was demonstrated on an industrial 4-component system and was found to improve the accuracy of onset time predictions by ~15% compared to individual CC-LSTM models and ~25-40% compared to commonly used deep learning techniques such as dense neural networks, convolutional neural networks and LSTMs. The CC-LSTM and the ensemble models also had the lowest missed detection rates and zero false positive rates making them ideal for real-time monitoring and fault detection in multicomponent systems.

1. INTRODUCTION

Prognostics and Health Management (PHM) deals with optimal maintenance of industrial equipment and systems to minimize breakdowns and unplanned shutdowns, lower the cost of maintenance and inventory, and reduce the time to repair critical assets. In complex industrial equipment comprising multiple interconnected components faults in one component may propagate to other components leading to accelerated degradation and failure of the entire equipment resulting in significant loss of availability and revenue. Examples of interacting multicomponent systems include mechanical equipment such as motors, gearboxes and bearings used in industrial machinery, power grids, water distribution systems, wind turbines and gas turbines. The interactions among the components could be:

- Direct physical contact, e.g., in gearboxes where multiple gears mesh together, a crack or pitting in one of the gears could impact the working of the other gears and lead to accelerated failure of the system
- Indirect via material or energy streams, e.g., in gas turbines, faults in the combustors can lead to improper temperature control of the combusted gas which can lead to overheating/erosion of the turbine blades downstream

Such systems typically have evolving operating environment that start in a normal mode of operation but go into abnormal mode due to faults or degradation of one or more components. It is important to detect the earliest possible signs of faults (as close as possible to the onset of abnormal operation) so that appropriate operational changes or maintenance activities can be undertaken before the faults lead to catastrophic failure of the system.

In recent years, with abundant availability of sensor data and advances in computing power, data-driven models have been used extensively for abnormal behavior detection in industrial systems (Chalapathy & Chawla, 2019). An extensive survey of data-driven techniques used for predictive maintenance of industrial equipment, especially for fault diagnosis and estimation of remaining useful life (RUL) has been presented by (Zhang, Yang, & Wang, 2019). Among the data-driven models, neural networks are of particular interest due to their ability to model complex

https://doi.org/10.36001/IJPHM.2022.v13i2.3093

nonlinearities in industrial systems (Zhao, Zhang, Ge, & Liu, 2016) and learn abstract features from sensor data automatically without expert knowledge (Yuan & Tian, 2019) and their capability to deal with the inherently changing nature of industrial processes (Shang, Yang, Huang, & Lyu, 2014). Several networks such as shallow/deep neural networks (DNNs), autoencoders, sequence-based recurrent neural networks (RNNs) and its variants such as gated recurrent units (GRUs) and long short-term memory (LSTM), and convolutional neural networks (CNNs) have been used for fault detection and diagnosis, and RUL estimation.

Of these networks, LSTM network-based approaches have been used extensively for anomaly detection in industrial processes as they are capable of learning long-term temporal dependencies from operations data (Zhou, Sun, Liu, & Lau, 2015). For example, a LSTM-based encoder-decoder scheme was used to detect anomalies in real-world multi-sensor turbofan engines (Malhotra, et al., 2016) as well as anomaly detection in discrete manufacturing plants (Lindemann, Jazdi, & Weyrich, 2020). An unsupervised LSTM based technique was proposed to detect abnormal operation in mechanical systems (Li, Li, Wang, & Wang, 2019). Recently, Bi-directional LSTM (Bi-LSTM), which is an extension of LSTM, has been used to perform anomaly detection in cyber-networks (Aljbali & Roy, 2021). A survey of the use of LSTM networks for anomaly detection in industrial systems was presented by (Lindemann, Maschler, Sahlab, & Weyrich, 2021).

While largely used for image processing and analysis (Krizhevsky, Sutskever, & Hinton, 2017), CNNs have been extended for industrial anomaly detection as the network can perform 1-Dimensional (1D) convolution and extract generalized abstract features from time series data without having to perform hand-crafted feature engineering (Kiranyaz, et al., 2021). CNN-based techniques have been used extensively for industrial bearing anomaly detection and fault diagnosis using vibration data (Eren, Ince, & Kiranyaz, 2019) (Wang, Guo, Song, Gao, & Li, 2019) (Hasan, Sohaib, & Kim, 2019) (Wang, Mao, & Li, 2020), fault diagnosis of rotating machinery (Li, Zou, Jiang, & Zhou, 2019) and anomaly detection in industrial control systems (Lai, Zhang, & Liu, 2019).

Recently, hybrid CNN-LSTM networks have been proposed to combine the strengths of CNNs and LSTMs for modeling spatial and temporal patterns respectively. Kim and Cho (2018) have used a hybrid CNN-LSTM model for detecting anomalies in web traffic and demonstrated that the performance of the hybrid model is better than individual LSTMs and CNNs (Kim & Cho, 2018). Zheng et al. (2019) have predicted the steam temperature after the primary de-superheating system using a hybrid CNN-LSTM model which is then used for fault prediction in a boiler in an industrial power plant unit and found that the CNN-LSTM network improved the accuracy of fault detection compared to the standalone LSTM network. Hybrid CNN-LSTM models have also been used for industrial abnormal behavior detection by (Canizo, Triguero, Conde, & Onieva, 2019) and (Ullah, et al., 2020).

In multi-component industrial systems, a faulty or degraded component can accelerate the degradation of other components due to interdependencies among the components and lead to faster system failure. It is therefore imperative to consider such component interactions while developing data-driven models for fault detection and diagnosis and RUL estimation in multicomponent systems. However, sensor data would be influenced by all the components, and it is challenging to isolate the effect of individual components on the data and thereby estimate the state of health of each component. In mechanical multicomponent systems where high frequency vibration data from accelerometers is available, time and/or frequency domain analysis may be carried out to isolate the behavior of each component (Assaf, Do, Scarf, & Nefti-Meziani, 2017). However, in most industrial systems, high frequency data may not be available continuously or available at all as it results in enormous amounts of data that is difficult to maintain and analyze. Only low frequency data related to temperatures, pressures, flow rates, voltages, currents, etc. may be available. In such cases, researchers have used various other approaches to model and isolate inter-component interactions. Such approaches include stochastic dependence modeling (Bian & Gebraeel, 2014), adaptive degradation modeling demonstrated for an electrical system (Prakash, Samantaray, Bhattacharyya, & Ghoshal, 2018), and Naïve Bayesian method (Lin, Zakwan, & Jennions, 2020) and model-free clustering analysis (Liu, Zhao, Zaporowska, & Zakwan, 2020) demonstrated for an aircraft fuel rig.

In this context, the problem posed by the Advanced Reliability Availability and Maintenance for Industries and Services (ARAMIS) Group is very pertinent (Cannarile, Compare, Bareldi, Yang, & Zio, 2020). They simulated the behavior of multiple industrial 4-component systems under evolving conditions and provided the data for fault detection and predicting the time of onset of abnormal operation in each of the components. Several researchers solved this problem using different approaches. Siahpour, Ainapure, Li and Lee (2020) have used an ensemble of CNN and LSTM models to predict the onset of abnormal condition but did not consider dependency among the components. Rocchetta, Petkovic and Gao (2020) have used an ensemble of support vector machine classifiers but did not specify if the inter-component dependencies were considered. Altarabichi et al. (2020) have used a stacked generalization approach comprising base models composed of heterogenous ML-based classifiers and a meta model based on random forest for estimation onset of abnormal operation in the multicomponent systems. Gupta et al. (2021) have used

several LSTM-based models with and without interdependence among the components while Yang, Baraldi and Zio (2022) have proposed a sparse autoencoder based deep learning network to capture the dependencies among the components and predict the onset of abnormal operation.

In this work, we propose to utilize the hybrid CNN-LSTM network, hereby referred to as a Cascaded CNN-LSTM (CC-LSTM) network for abnormal operation detection in multi-component industrial systems. It is hypothesized that the spatial features abstracted from all the sensors by the CNN layers in the CC-LSTM network carry information related to the spatial dependencies among the sensors and the condition of health of each of the components. This information when combined with modeling of temporal dependencies by the LSTM layer is expected to improve the accuracy of fault detection in multicomponent systems. Further, some deep learning models perform better than others even when trained on the same dataset due to the stochasticity associated with training such models, and the overall performance of such models could be improved by using an ensemble of multiple models. Hence, we propose an approach called the Ensemble Cascaded CNN-LSTM (ECC-LSTM) technique which uses a weighted ensemble of multiple CC-LSTM models for estimating the abnormal onset times. The efficacy of CC-LSTM and ECC-LSTM models has been demonstrated on the abnormal onset time prediction problem posed by the ARAMIS Group (Cannarile, Compare, Bareldi, Yang, & Zio, 2020). The contributions of this work are as follows:

- An ECC-LSTM approach is proposed for detecting early signatures of faults and predicting the onset of abnormal operation in multi-component industrial systems
- The effect of inter-component dependency was evaluated by training CC-LSTM models assuming dependence as well as independence among the components
- The performance of CC-LSTM and ECC-LSTM models was compared with that of commonly used deep learning models
- ECC-LSTM models improved the performance metric by ~15% compared to CC-LSTM models and ~25-40% compared to common deep learning techniques

The rest of the paper is organized as follows: The problem statement and the dataset used are described in Section 2. The data pre-processing strategy is explained in Section 3. The CC-LSTM models and proposed ECC-LSTM methodology are explained in Section 4. Results from the proposed approach are presented and compared against those from other deep learning algorithms in Section 5. The findings of the study are summarized in Section 6.

2. PROBLEM DESCRIPTION

2.1. System of Interest

The industrial system of interest comprises $J = 4$ identical interconnected components, with operation time T, in arbitrary time units (*atu*). Data from $M = 200$ such industrial systems was provided. The components in the systems undergo random degradation during their operation. When the degradation of a component exceeds a threshold, it enters an abnormal state of operation. This abnormal state does not correspond to component failure but makes the operation of the system suboptimal. When all the four components start operating in abnormal operation, system failure is said to occur at time, T_f. The simulation of the degradation paths of all four components of all 200 systems had been done till the end of operation time T or till system failure occurs, whichever comes first. $K = 10$ sensors were installed on each component from which the level of degradation of the components could be estimated. The operating conditions and the degradation levels impact the sensor signals, $s_t^{j,m,1}, \dots, s_t^{j,m,K}$. A fixed frequency of $f_s = 1$ atu^{-1} was used for recording sensor signals.

2.2. Problem Statement

The K measurements taken from the j^{th} component of the m^{th} system at time t are represented by the vector $x_t^{j,m} = [s_t^{j,m,1}, \dots, s_t^{j,m,K}]$. The label of the component is denoted as $y_t^{j,m} \in 0$, 1 where 0 and 1 indicate normal and abnormal operation respectively at time t. The time at which the component enters first into an abnormal state, referred to as Time of Onset of Abnormal Behavior (TOAB) is represented as $\tau^{j,m}$. This means that when $t < \tau^{j,m}$, $y_t^{j,m} = 0$ and when $t >= \tau^{j,m}$, $y_t^{j,m} = 1$. It is possible that $\tau^{j,m} > T$, which means that the component has not entered abnormal operation in its lifetime. Given a training set of sensor data $x_t^{j,m}$ and its labels $y_t^{j,m}$, the objective is to detect the TOAB of all J components in the test systems.

2.3. Performance Metric

For evaluating the performance metric, testing dataset containing the sensor data of M_{test} four-component systems was considered. The ground truth time at which the j^{th} component of the m^{th} system first enters into abnormal operation is denoted as $\tau^{j,m}$. $\tau^{j,m}$ is fixed to be *NaN* (Not A Number) if the component had not entered abnormal state in its entire lifetime.

$\hat{\tau}^{j,m}$ is the estimate of $\tau^{j,m}$ that is to be found, for any $m = 1, \dots, M_{test}$ and $j = 1, \dots, J$. $\hat{\tau}^{j,m}$ should be set to *NaN*, if a component's entry into abnormal operation had not been detected. The error of estimating the time $\tau^{j,m}$ with $\hat{\tau}^{j,m}$ is defined by Eq. (1):

$$\Delta^{j,m}=\begin{cases}\tau^{j,m}-\hat{\tau}^{j,m} & \tau^{j,m}\neq NaN,\hat{\tau}^{j,m}\neq NaN.\\ 0 & \tau^{j,m}=NaN,\hat{\tau}^{j,m}=NaN.\\ k_{false} & \tau^{j,m}=NaN,\hat{\tau}^{j,m}\neq NaN.\\ -k_{missed} & \tau^{j,m}\neq NaN,\hat{\tau}^{j,m}=NaN.\end{cases} \quad (1)$$

False alarms are penalized with an error, $k_{false} > T$ and missed alarms are penalized with an error, $k_{missed} > T$. The average error of the solution on all test components is quantified by the Timeliness Error, TE (Cannarile, Compare, Bareldi, Yang, & Zio, 2020) that is desired to be as small as possible (ideally closed to zero). The expression for computing TE is given by Eq. (2).

$$TE = \frac{1}{4M_{test}}\sum_{m=1}^{M_{test}}\sum_{j=1}^{4}\varphi(\Delta^{j,m}) \quad (2)$$

where

$$\varphi(\Delta^{j,m})=\begin{cases}1 & \Delta^{j,m}<-T\\ (1-e^{\frac{\Delta^{j,m}}{a_1}})b_1 & -T\leq\Delta^{j,m}<0\\ (1-e^{\frac{\Delta^{j,m}}{a_1}})b_1 & 0\leq\Delta^{j,m}\leq T\\ 1 & \Delta^{j,m}>T\end{cases} \quad (3)$$

$$b_1=\frac{1}{1-e^{\frac{-T}{a_1}}} \quad (4)$$

$$b_2=\frac{1}{1-e^{\frac{-T}{a_2}}} \quad (5)$$

Parameters b_1 and b_2 are set to obtain $\varphi(T) = 1$ and $\varphi(-T) = 1$, respectively. Values of 13 and 10 are set for the parameters a_1 and a_2 respectively so that late estimates are subject to a higher penalty compared to early estimates of TOAB (Saxena, et al., 2008). In the paper, this proposed timeliness error was chosen for comparing the effectiveness of various techniques for detecting the onset of abnormal operation.

2.4. Data Description

The time series plots of 10 sensors for one of the components from a sample system are shown in Figure 1. The green line indicates normal operation while the red line indicates abnormal operation. As seen from Figure 1, the onset time of abnormal operation is not directly identifiable from individual sensors and could be a multivariate effect. The influence of other components on a component entering abnormal state is also not directly evident from visual inspection of data.

Out of 800 components from all systems, 411 components entered abnormal operation during their lifetime. The entire dataset is randomly partitioned into a training set consisting of 150 systems and a testing set consisting of 50 systems.

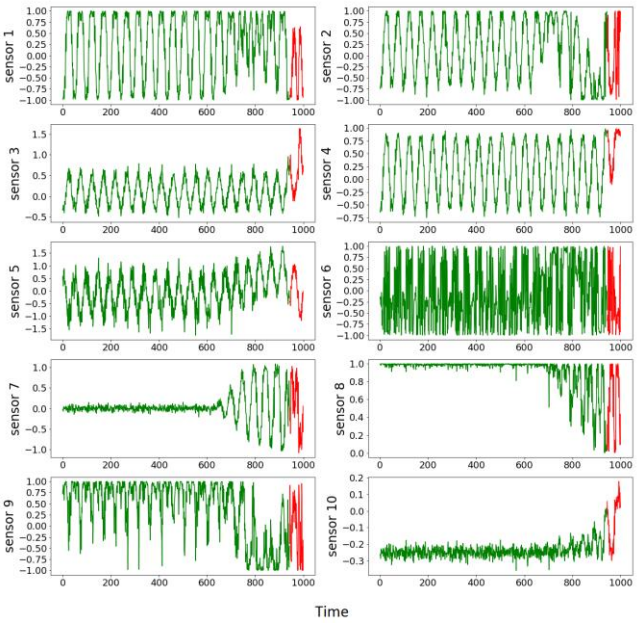


Figure 1. Time series plots of the 10 sensors of a component

The training set is used for development of model(s) for the detection of onset of abnormal operation and the testing dataset is used for the validation of the developed model(s).

3. DATA PRE-PROCESSING

The following data pre-processing steps were applied on the sensor signals in the training and testing datasets:

3.1. Data Denoising

The Butterworth Low-pass filter (Bansal, 2010) of order 2 with suitable cutoff frequency was used to remove any high frequency measurement noise present in each of the 10 sensor signals. The suitable cut-off frequency for each sensor signal was estimated from its frequency spectrum. Sample raw and denoised plots for a sensor signal are shown in Figure 2.

3.2. Data Normalization

All sensor signals were found to be adequately Gaussian-like. Z-normalization was applied on each sensor signal, to obtain normalized sensor data with a mean of zero and standard deviation of one. Data normalization was done component-wise, that is, the mean and standard deviation of sensors from component-1 of all systems were used to normalize component-1 signals in all systems and the same procedure was applied to components 2, 3 and 4. Sensor signals of the testing datasets were standardized using the mean and standard deviations obtained from the corresponding training datasets.

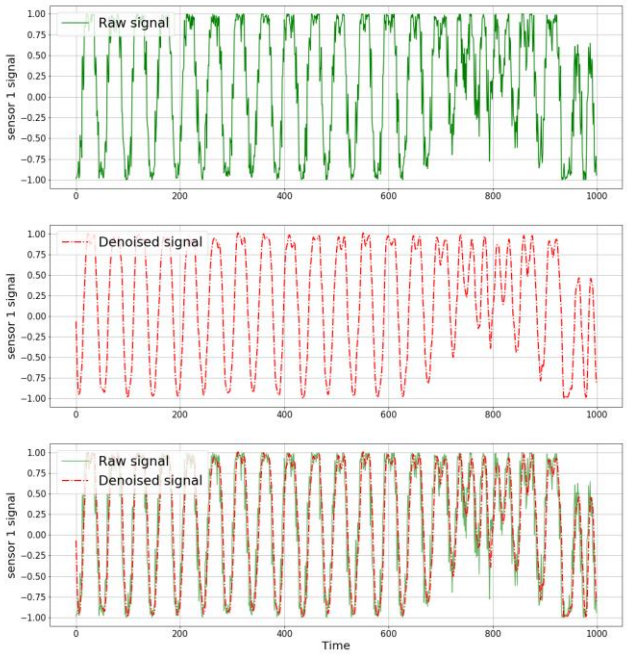


Figure 2. Raw vs Denoised time series plots of sensor 1 of component 1 and system 1

4. PROPOSED METHODOLOGY

4.1. Cascaded CNN-LSTM (CC-LSTM) Models

A time-series classification approach is proposed here in which trained cascaded CNN-LSTM or CC-LSTM classifier models classify the sensor data at each time step into 'normal behavior' (label 0) or 'abnormal behavior' (label 1). These predicted labels are then used to determine the TOAB. The architecture of CC-LSTM used in this work is shown in Figure 3. The architecture comprises a one-dimensional (1D) CNN network having two convolutional layers and a max pooling layer in sequence. These layers extract complex feature representations from the input windowed data without losing key spatial information. The first convolutional layer consists of F_1 number of filters each with a filter size of F_{s1}. The second convolutional layer consists of F_2 filters each with a filter size of F_{s2}. The two convolutional layers use ReLU (Rectified Linear Units) activation function. The max pooling layer consists of one filter with size P. The features extracted by the 1D-CNN network are input to the LSTM layer that consists of N_{LSTM} number of LSTM nodes with *tanh* activation function. A dropout layer (with dropout fraction, D) is used after the LSTM to mitigate overfitting which is a common problem with neural networks. A final dense output layer with sigmoid activation function and one neuron is used to classify the input window into '0' or '1' labels.

The CC-LSTM architecture shown in Figure 3 was used to train classification models using the training datasets from each of the four components of all systems. Hence, there were four classifier models corresponding to each of the four components. The methodology followed for training the models and using them for predictions is depicted in Figure 4 and explained as follows.

Preprocessed data in the form of two-dimensional windows (each window of size W_s x N_f) was given as input to train the CC-LSTM network shown in Figure 3 where W_s is the length of the window and N_f is the number of features considered in the model. A window shift of one time step was used during windowing of training as well as testing datasets. The label corresponding to the last time step in a window was taken as the label for the entire window.

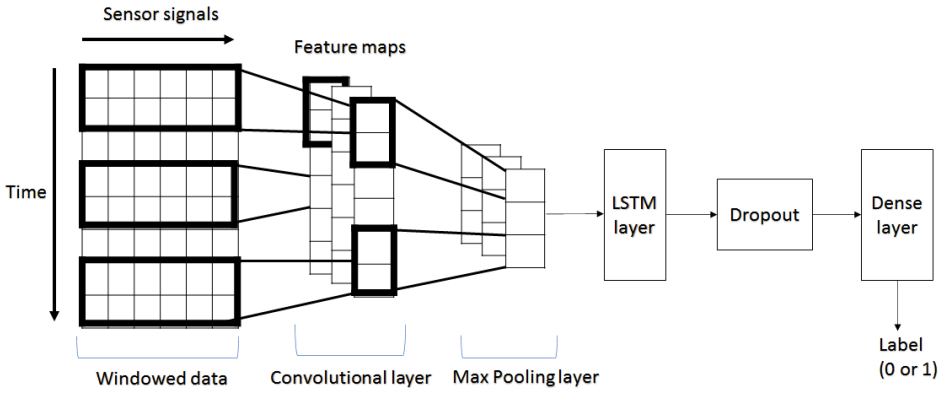


Figure 3. CC-LSTM architecture

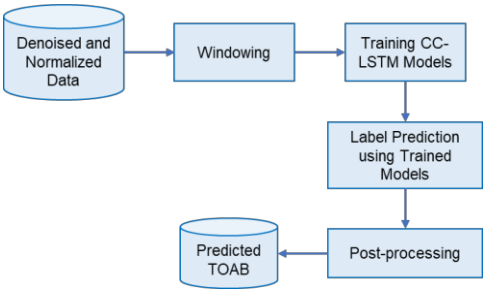


Figure 4. CC-LSTM implementation

To obtain the best possible models, the architecture of the CC-LSTM network was optimized via hyperparameter tuning using the Hyperband approach (Li, Jamieson, DeSalvo, Rostamizadeh, & Talwalkar, 2018). In this approach, hyperparameter tuning is performed in a sports championship style bracket wherein many models with different hyperparameter values are trained for a few epochs in every round and the top half of the models are considered for the next round. The hyperband algorithm uses early stopping and adaptive resource allocation to arrive at the best possible model quickly (Introduction to the Keras Tuner, 2022). Li et al. (2018) have demonstrated that the hyperband algorithm was just as effective, and 5-30 times faster compared to Bayesian optimization algorithms for several deep learning problems. Adam optimizer with variable learning rate and binary cross entropy loss function were used for training the network. 20% of training data was used as validation data and the value of loss function on validation data was monitored at every epoch during training. To prevent overfitting, an early stopping criterion was employed, i.e., training was terminated when the validation loss did not decrease for more than 'n' epochs (patience). In this work, network training was carried out for a maximum of 100 epochs and a patience value of n = 10 was selected after experimenting with patience values up to 50. The model with the lowest validation loss was considered the best model and used for predictions. The trends in training and validation loss for one of the CC-LSTM models are shown in Figure 5. It can be seen from the figure that the validation loss was minimum at epoch #14 and increased after that. Therefore, the model with the weights at epoch #14 is considered the best model in this trial.

As the system of interest comprises 4 interconnected components, the onset of abnormal operation in a certain component may be influenced by or be dependent on any of the other components. To determine if such an interdependency among the components was present, the following two cases of CC-LSTM models were developed for each of the four components:

1. **Independent Components (IC) type models:** In this case, no dependency among the components was assumed and the CC-LSTM model for each component was developed using the 10 sensors corresponding to that component only.
2. **Dependent Components (DC) type models**: In this case, each component was assumed to be dependent on the other three components. Therefore, the CC-LSTM model for each component was developed using 40 sensors, i.e., 10 sensors from the component for which the model was built combined with the 30 sensors from the other three components.

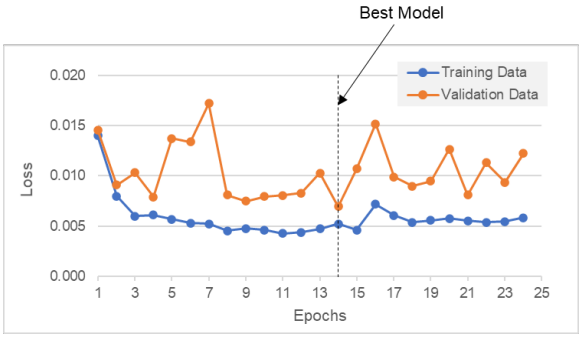


Figure 5. Trend in loss for training and validation data for a CC-LSTM model

The optimized parameter values of the CC-LSTM network for IC and DC type models along with the corresponding network training parameters are shown in Table 1. For IC and DC type models, the search spaces for hyperparameters were as follows: F_1 and F_2 were experimented at values of [32, 64, 128, 256, 512, 1024], F_{s1} and F_{s2} at values of [3, 5], N_{LSTM} at values of [32, 64, 128, 256, 512, 1024], D at values of [0.1, 0.2, 0.3] and batch size at values of [32, 64, 128, 256]. Here, it should be noted that the CC-LSTM network parameters were optimized for the first component and the same parameters were used for training the models for the other components as the components are identical in nature.

Each trained CC-LSTM model used a window/segment of sensor data for making a binary label prediction of 0 or 1. The predictions were then used for computing TOAB as follows: A window of size, W_{post} x N_f was considered with

Table 1. Optimized network and training parameters

Layer/Parameter	IC Type Models	DC Type Models
1D convolutional Layer 1	$F_1 = 64$, $F_{s1} = 5$	$F_1 = 128$, $F_{s1} = 3$
1D convolutional Layer 2	$F_2 = 32$, $F_{s2} = 3$	$F_2 = 256$, $F_{s2} = 5$
Max Pooling Layer, P	2	2
LSTM nodes, N_{LSTM}	64	256
Dropout, D	0.2	0.1
Batch size	256	128
Window size, W_s	50	50

W_{post} = 10. If all the predicted labels in the window were 1, then that window's start time was considered to be the TOAB. If no windows with all predicted labels of 1 were detected in a component's lifetime, then the component was classified as normal and TOAB was set to be *NaN*. For any system in the testing dataset, the TOAB was estimated for each of its four components.

4.2. Ensemble of Cascaded CNN (ECC)-LSTM Models

It was observed that all CC-LSTM models could not estimate the TOAB with a high degree of accuracy. This was largely due to the intrinsic randomness in the training of deep learning algorithms. As the training and performance of deep learning models depend on the initial random values of the weights and biases, few sets of random initializations of the weights and biases resulted in models with lower accuracy w.r.t estimation of TOAB. To overcome this issue and exploit the randomness to our advantage, a weighted ensemble of multiple CC-LSTM models is proposed for estimating the TOAB. Ensemble of CC-LSTM models gives less weightage to poorly performing CC-LSTM models and more weightage to better models ensuring reliable predictions of TOAB overall. The ECC-LSTM approach consists of two phases: the training phase and the testing phase.

The training phase of the ECC-LSTM approach is shown in Figure 6. Similar to the training of CC-LSTM models, component-wise modelling is performed for ECC-LSTM i.e., separate models were developed for each of the four components. For each component, N=10 number of CC-LSTM models were developed by training on the 150 systems using different random initializations for network weights and biases. The approach for training individual CC-LSTM models is discussed in Section 4.1. Each of the N CC-LSTM models was used to predict the TOAB of the components in the training dataset which was then used to compute the timeliness error (specified in Section 2.3) for each of the N CC-LSTM models. The computed timeliness errors were an indication of the training effectiveness of each model and are stored to be used as weights in the testing phase.

The testing phase of the ECC-LSTM approach is depicted in Figure 7. For a given component, TOAB predictions from the corresponding trained N CC-LSTM models of that component were considered. The procedure for estimation TOAB from individual CC-LSTM models was the same as that discussed in Section 4.1. From the N TOAB predictions, the number of models detecting the component as being in normal operation (N_{normal}) and the number of models detecting the component as being abnormal operation ($N_{abnormal}$) were calculated. If N_{normal} was greater than $N_{abnormal}$, then the test component was classified as being in normal operation throughout its life and TOAB for the component was set to be *NaN*. On the other hand, if N_{normal} is lower than $N_{abnormal}$, the test component was

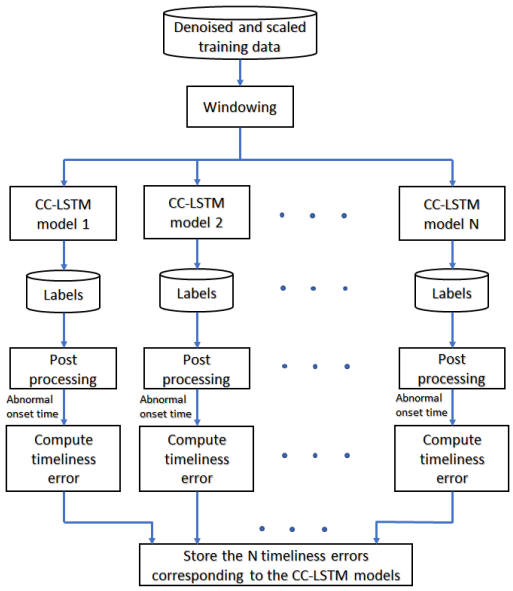


Figure 6. Training phase of ECC-LSTM approach

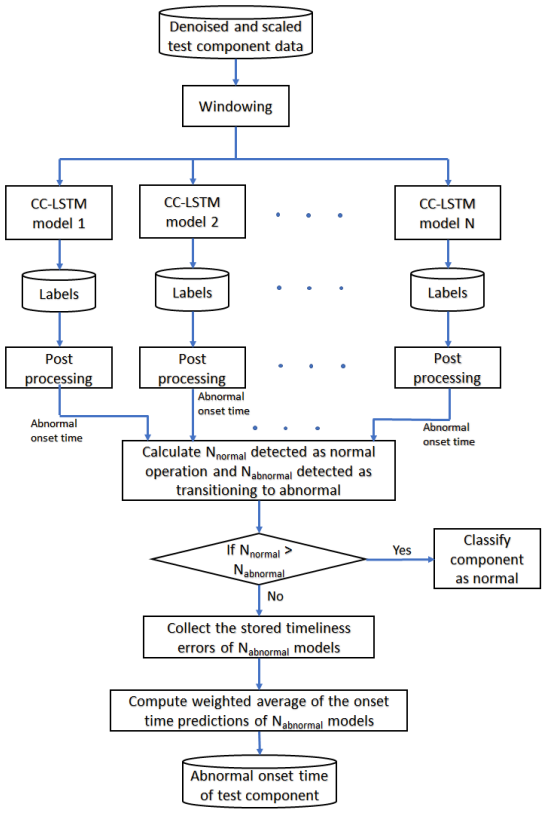


Figure 7. Testing phase of ECC-LSTM approach

classified as being in abnormal operation and the TOAB for the component was then obtained as a weighted average of the TOABs predicted by the $N_{abnormal}$ models using the

inverse of timeliness errors of the $N_{abnormal}$ models as weights.

5. RESULTS AND DISCUSSION

All the deep learning models in the paper were trained using the Keras 2.4.0 package (Keras API Reference, 2021). The results were obtained on a computing system with Intel Core i7-8650U CPU and 16 GB RAM.

5.1. Dependency among Components

For IC (10 sensors as inputs) and DC (40 sensors as inputs) type models, 10 trials with a different random initialization of the network in each trial were conducted to obtain reliable results. This resulted in 10 CC-LSTM models for each of the 4 components. Using each of the models, the timeliness error on the testing dataset consisting of 50 systems was computed. The mean and standard deviation of the timeliness errors for the two types of models for each of the 4 components and all components combined are shown in Table 2 and Figure 8. From the table, it can be observed that the timeliness error obtained from IC type models is in the range 0.059 to 0.095 and that obtained from DC type models is in the range 0.139 to 0.235. For each component and all components combined, IC models resulted in lower mean timeliness errors (at least 40% lower) compared to DC models. The differences in the timeliness errors from IC and DC models can be seen clearly in Figure 8. To confirm that the timeliness errors from both the types of models were statistically different, a two-sample t-test (Snedecor & Cochran, 1989) was conducted with the following null and alternate hypotheses:

H_0: $TE_{IC} = TE_{DC}$

H_a: $TE_{IC} \neq TE_{DC}$

Test Statistic, $$T = \frac{\bar{Y}_{IC} - \bar{Y}_{DC}}{\sqrt{\frac{s_{IC}^2}{N_{IC}} + \frac{s_{DC}^2}{N_{DC}}}}$$

where:
TE_{IC} and TE_{DC} are timeliness errors from IC and DC type models,
N_{IC} and N_{DC} are the sample sizes or number of trials of IC and DC type models,
$\bar{Y}_{IC}$ and $\bar{Y}_{DC}$ are means of timeliness errors from the trials of IC and DC type models, and
s_{IC}^2 and s_{DC}^2 are the variances of timeliness errors from the trials of IC and DC type models

In this case, the samples sizes N_{IC} and N_{DC} were equal to 10 as ten trials each were conducted for IC and DC type models. The null hypothesis that the timeliness errors from IC and DC type models are equal can be rejected if

$$|T| > t_{1-\alpha/2,\nu}$$

where $t_{1-\alpha/2,\nu}$ is the critical value of the t distribution with ν degrees of freedom and α is the significance level. The t-test was conducted for timeliness errors obtained for each of the components and all components combined. The values of the critical value of the t distribution at the 0.05 significance level and the test statistic from these tests are shown in Table 3. It can be observed from the table that the test statistic is greater than the critical value for all the components leading to the rejection of the null hypothesis. This leads to the conclusion that the difference between the mean timeliness errors from the IC and DC type models is statistically significant. The lower timeliness errors from IC models indicated that addition of more sensors in DC models did not improve the prediction accuracy leading us to believe that there was no interdependency or only a weak interdependency among the components in the given systems. This is an interesting observation since the inclusion of additional sensors is typically expected to improve or at least maintain the prediction accuracy of deep learning models. However, Gupta et al. (2021) have also reported a similar observation that incorporating sensors from all the components did not improve the predictions of TOAB from one component. Therefore, only IC models, i.e., CC-LSTM models with only 10 sensors corresponding to each component were considered for evaluating the ECC-LSTM approach.

5.2. Implementation of benchmark algorithms

The performance of the ECC-LSTM approach was compared with that of CC-LSTM models as well as models built using common neural networks such as Shallow Neural Network (SNN), Deep Neural Network (DNN), LSTM, Bi-LSTM and CNN. These models were trained for

Table 2. Mean timeliness errors on the testing dataset computed using IC and DC type models. Number in brackets is the standard deviation of timeliness error from 10 trials

Model Type	Mean Timeliness Error for Component				
	C1	C2	C3	C4	All Components
IC	0.095 (0.019)	0.059 (0.009)	0.091 (0.011)	0.075 (0.009)	0.080 (0.007)
DC	0.168 (0.017)	0.139 (0.029)	0.235 (0.036)	0.174 (0.012)	0.179 (0.016)

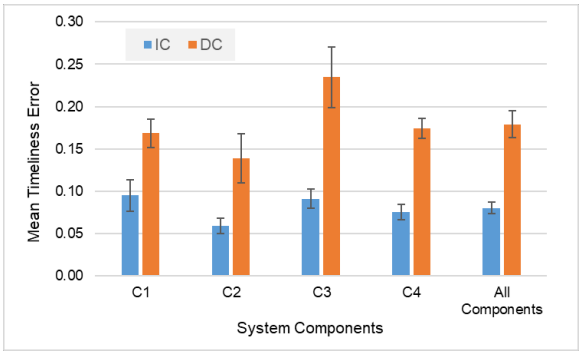


Figure 8. Comparison of timeliness errors from IC and DC type models. Error bars indicate standard deviation of timeliness error from 10 trials

Table 3. Results of two sample t-tests on timeliness errors from IC and DC type models

Component	Test Statistic, $\|T\|$	Critical value, $t_{1-\alpha/2,\nu}$
C1	7.731	2.262
C2	6.007	2.776
C3	8.790	2.776
C4	16.509	2.446
All Components	13.334	2.570

each component using 10 sensors as input. They were trained to predict a binary output of 0 or 1 corresponding to normal or abnormal behavior of the components. These labels were then used to estimate TOAB as described in Section 4.1. The SNN model consisted of 1 dense hidden layer while the DNN model comprised 3 dense hidden layers. The CNN model consisted of two 1D convolutional layers (F_1 filters with filter size of F_{s1} and F_2 filters with filter size of F_{s2}) followed by a max pooling layer with filter P, a dense layer with dropout and the output layer. The LSTM and Bi-LSTM models consisted of one hidden layer with dropout followed by the output layer. In all the models, the output layer was a dense layer with 1 neuron and sigmoid activation function. Hyperparameter tuning for these networks was also carried out using the hyperband approach detailed in Section 4.1. The optimized network parameters for all the models are shown in Table 4. For SNN and DNN models, the number of dense units in each of the layers and batch size were experimented at values of [32, 64, 128, 256, 512] and [32, 64, 128, 256] respectively. For the CNN model, F_1 and F_2 were experimented at values of [32, 64, 128, 256], F_{s1} and F_{s2} at values of [3, 5], number of dense units at values of [32, 64, 128], drop out at values of [0, 0.1, 0.2, 0.3, 0.4, 0.5] and batch size at values of [32, 64, 128, 256]. For the LSTM and Bi-LSTM models, the number of units were experimented at values of [32, 64, 128, 256], drop out at values of [0, 0.1, 0.2, 0.3, 0.4, 0.5] and batch size at values between 16 and 256 with a step size of 16. The window size, Ws = 50 for CNN, LSTM and Bi-LSTM models was chosen after extensive experimentation with varying window sizes ranging from 10 to 100.

Table 4: Optimized parameters for neural network models

Model	Optimized Network Parameters
SNN	Dense units: 128 Batch size: 64
DNN	Dense units: 128, 128, 64 Batch size: 256
CNN	F_1 = 64, F_{s1} =5 F_2 = 64, F_{s2} = 5 Dense units: 32 Dropout: 0.3 Batch size: 64
LSTM	LSTM Units: 256 Dropout: 0.1 Batch size: 16
Bi-LSTM	Bi-LSTM Units: 256 Dropout: 0.2 Batch size: 256

Due to the stochastic nature of the deep learning algorithms instead of relying on a single model, 10 instances of the models were trained for each of the techniques with a different random initialization of the network in each trial. The final timeliness error for each technique was obtained by averaging the timeliness errors from the 10 trials on the test data.

5.3. Comparison of CC-LSTM and ECC-LSTM with benchmark algorithms

The mean timeliness error from all the models for each of the components and all components combined in the test dataset is shown in Figure 8. It can be observed from the figure that the timeliness errors for component 2 are the lowest across all the models indicating that all the models were able to capture the onset of abnormal operation in component 2 across all systems quite well. The timeliness errors for components 1, 3 and 4 are higher than those of component 2 across all the models. It can also be seen that for each component, simpler network models such as SNN and DNN as well as CNN resulted in higher timeliness errors indicating that they were unable to model the degradation behavior of the components. On the other hand, sequence-based models such as LSTM, Bi-LSTM, CC-LSTM and ECC-LSTM captured the operating behavior of the components well as is evident from the lower timeliness errors for each component and all components combined. Of these, CC-LSTM and ECC-LSTM models performed better than all the other models with respect to timeliness error,

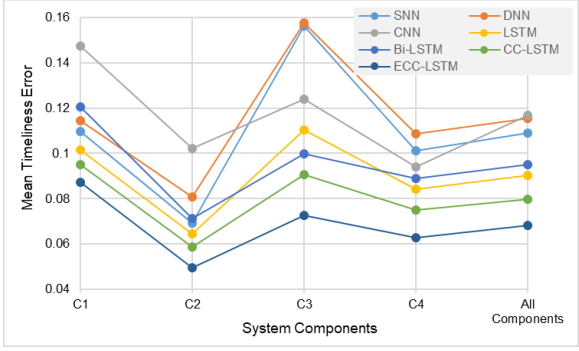


Figure 8. Component-wise comparison of timeliness errors of models

highlighting the efficacy of the proposed CC-LSTM and the ensemble methodology.

The mean timeliness errors from all the models for all components combined is shown in Figure 9 where the error bars represent the standard deviations of timeliness errors from 10 trials. It should be noted that the ECC-LSTM model has only one value of timeliness error and no standard deviation as the TOABs from ECC-LSTM model were estimated as a weighted mean of TOABs from the 10 CC-LSTM models. Figure 9 shows that the mean timeliness error for the CC-LSTM and ECC-LSTM models are the lowest amongst the timeliness errors from all the models. The improvement in timeliness error due to the CC-LSTM approach compared to SNN, DNN and CNN models is 26-32% while it is 11-16% compared to LSTM and Bi-LSTM models. A two-sample t-test was carried out to confirm that the timeliness error due to the CC-LSTM approach was statistically different from the timeliness error from to the other models using the following null and alternate hypotheses:

H_0: $TE_{CC\text{-}LSTM} = TE_M$

H_a: $TE_{CC\text{-}LSTM} \neq TE_M$

where:
$TE_{CC\text{-}LSTM}$ is the timeliness error from CC-LSTM model
TE_M is the timeliness errors from other models where the subscript 'M' is SNN, DNN, CNN, LSTM and Bi-LSTM

The values of the test statistic and the critical value of the *t* distribution at the 0.05 significance level from the tests are shown in Table 5. It can be observed from the table that the test statistic is greater than the critical value for all the models. Therefore, the null hypothesis can be rejected indicating that the improvement in the timeliness error due to the CC-LSTM approach is statistically significant.

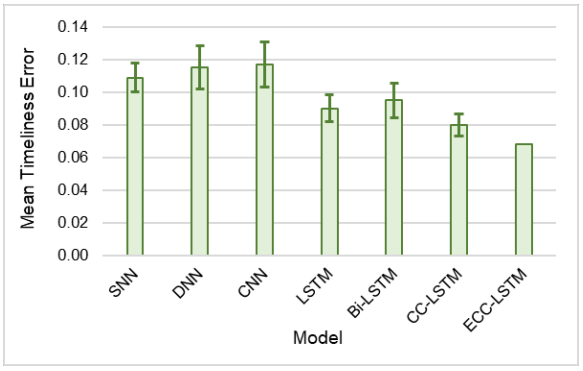


Figure 9. Mean timeliness error of models considering all components. Error bars indicate standard deviation of timeliness error from 10 trials

Table 5. Results of two sample t-tests comparing the timeliness errors from CC-LSTM approach and other models

Model	Test Statistic, $\|T\|$	Critical value, $t_{1-\alpha/2,\nu}$
SNN	8.38	2.11
DNN	7.91	2.14
CNN	7.61	2.16
LSTM	3.05	2.11
Bi-LSTM	2.89	2.44

The ECC-LSTM approach further improved the timeliness error by ~15% compared to the CC-LSTM models indicating that the ensemble approach estimated the TOAB in multi-component systems with a high degree of accuracy. Compared to the common deep learning models, the improvement in timeliness error due to ECC-LSTM is ~25-40%. The improvement in estimation of TOAB due to the CC-LSTM and ECC-LSTM approaches may be attributed to the use of spatial features extracted by the CNN layers that capture the spatial dependencies among the sensors as well as incorporation of temporal effects using the LSTM layer.

The models were also compared using common performance metrics such as Mean Absolute Error (MAE), Root Mean Square Error (RMSE), Hit Rate (HR), Missed Detection Rate (MDR) and False Positive Rate (FPR). HR is the percentage of model predictions that lie within a threshold of the actual onset times. In this work, HR was calculated for thresholds of 2.5, 5 and 10 *atu* (arbitrary time unit). MAE, RMSE and HR were calculated considering the components for which $\tau^{j,m}$ was available and a corresponding $\hat{\tau}^{j,m}$ was predicted by the model. MDR refers to the number of TOABs that are missed by the model and was expressed as a percentage of the number of components that entered abnormal operation. FPR refers to the number of falsely predicted TOABs and was expressed as a percentage of the number of components that did not enter

abnormal operation. In the test data, 92 components entered abnormal operation during their lifetime while 108 components did not. For the models, HR should be as high as possible whereas MAE, RMSE, MDR and FPR should be as low as possible.

Various performance metrics along with the mean and standard deviation (SD) of the timeliness error for all models considering all components together are shown in Table 6. The table shows that all performance metrics improved as one moved from simpler networks such as SNN and DNN to complex LSTM-based networks. This is expected in dynamic systems with evolving regimes of operation such as the one considered in this work. It can be seen that the CC-LSTM and ECC-LSTM models performed better than all the other models. Both these models have very low missed detection rates and zero false alarm rates, and their predictions of TOAB are fairly accurate as can be observed from MAE, RMSE and HR metrics. Compared to the CC-LSTM model, the MAE and RMSE of the ECC-LSTM model improved 17% and 19% respectively. This improvement can be observed clearly in the parity plots of TOAB from CC-LSTM and ECC-LSTM models shown in Figure 10(a) and 10(b) respectively. It can be seen from the figure that the TOAB predictions from ECC-LSTM model are closer to the diagonal and within the ±10 *atu* threshold compared to the predictions from one of the CC-LSTM models. This is more pronounced for actual abnormal onset times greater than 925 indicating that the ECC-LSTM model could provide reliable predictions of the abnormal operation onset times even closer to the end of the operation.

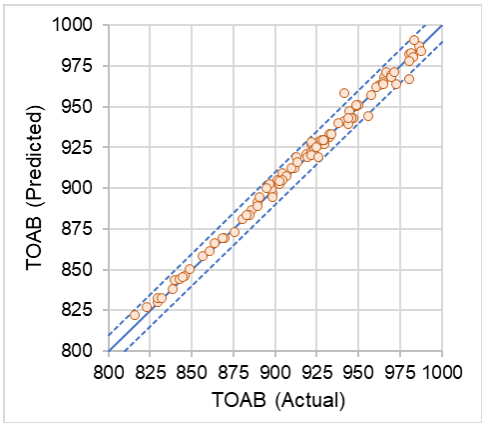
Results from the current work were compared with those reported by Siahpour et al. (2020), Gupta et al. (2021) and Yang et al. (2022) on the same problem. Rocchetta et al. (2020) and Altarabichi et al. (2020) too worked on the ARAMIS problem, but the performance metrics of the models developed by them were not available for comparison. The performance of the proposed ECC-LSTM approach is either on par or better than that reported by other researchers. The timeliness error of 0.068 obtained in this work is slightly better than the value of ~0.075 reported by Siahpour et al. (2020). The values of missed detection and false positive rates reported by Yang et al. (2022) were in the ranges of 0.473-1.08% and 0.011-0.036% respectively. In comparison, the missed detection rate of the ECC-LSTM model (0.98%) is in the same range while the false positive rate (0%) is slightly better. It should, however, be noted that Yang et al. had access to and used data from 5000 industrial systems instead of the 200 systems in this work. Lastly, Gupta et al. (2021) reported timeliness errors in the range of 0.0086-0.417. The timeliness error of 0.068 obtained for the ECC-LSTM model is better than those obtained for some of their models. However, the timeliness error of 0.0086 obtained from their best model appeared too low to be practical and will have to validated independently.

This study demonstrated that hybrid deep learning models such as CC-LSTM and their ensembles such as ECC-LSTM could improve fault detection in multicomponent systems. It is, however, interesting to note that the models that considered sensors from all the components to capture the inter-component dependencies (DC type) had an inferior performance compared to models that did not explicitly consider the dependencies (IC type). Such a behavior is unintuitive, and it is not immediately clear if it is an isolated characteristic of the industrial systems considered in this study. Additional work on other industrial multicomponent systems is necessary to judge the need for considering explicit dependencies among components when building models for predictive maintenance.

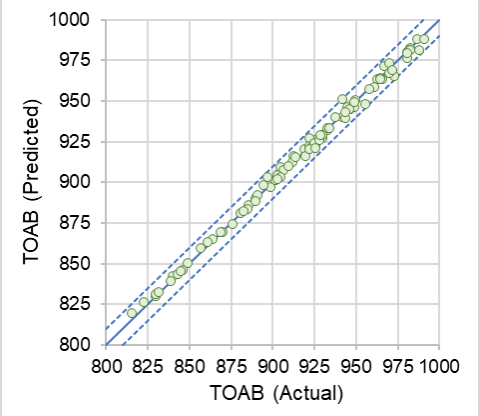
Due consideration must also be given to the amount of data and computational effort required for training these models. While deep learning models in general require larger amounts of data for effective training, simpler deep learning models such as SNNs and DNNs as well as CNNs could be trained well with reasonable amount of data and computational resources due to the lesser number of network weights and hyperparameters.

Table 6. Performance metrics of the models considering all components (*atu*: arbitrary time unit)

Models	Timeliness Error		MAE, *atu*	RMSE, *atu*	HR (2.5), %	HR (5), %	HR (10), %	MDR, %	FPR, %
	Mean	SD							
SNN	0.109	0.009	8.72	21.88	66.4	80.8	92.3	2.72	0.09
DNN	0.115	0.013	7.87	19.95	62.0	79.1	91.4	2.72	0
CNN	0.117	0.014	3.83	6.43	47.3	73.7	95.6	1.63	0
LSTM	0.090	0.008	2.76	4.71	64.2	84.0	97.7	1.52	0
Bi-LSTM	0.095	0.011	2.75	3.85	60.8	82.0	96.9	1.74	0
CC-LSTM	0.080	0.007	2.29	3.34	68.7	85.7	97.8	0.98	0
ECC-LSTM	0.068	-	1.90	2.70	72.5	90.0	100	1.09	0



(a)



(b)

Figure 10. Parity plot of TOABs for (a) CC-LSTM model (Trial #7) and (b) ECC-LSTM model. Dotted lines are ±10 *atu* from the diagonal

On the other hand, sequence-based models such as LSTM and Bi-LSTM and hybrid models such as CC-LSTM required large amounts of data to prevent overfitting as the number of network weights and hyperparameters was large. They also required considerable computational time which may be reduced by using GPUs. In this work, the training times for each SNN, DNN and CNN model were in the range of 1-4 hours while those for each LSTM, Bi-LSTM and CC-LSTM model were in the range of 12-20 hours on the same computing system.

The number of component-wise models to be built is another consideration. In this work, one fault detection model was trained for each of the 4 components. This increases the effort for developing and maintaining the models. For the industrial systems considered in this work, as the components and their sensors are identical, either a common fault detection model for all components or a multi-output (one output per component) deep learning model could be developed. The efficacy of such models needs to be evaluated. However, most industrial systems may not have identical components with the same number of sensors, in which case, multiple fault detection models and/or multi-output models will have to be developed and maintained.

6. CONCLUSION

In this paper, a deep learning-based Ensemble Cascaded CNN-LSTM approach was proposed to improve the performance of individual CC-LSTM models for detecting the onset time of abnormal behavior in industrial multi-component systems. ECC-LSTM predicted the TOABs as a weighted averaged of the onset time predictions from individual CC-LSTM models wherein the inverse of timeliness errors from the CC-LSTM models were used as weighting factors. The performance of the ECC-LSTM approach was compared with that of CC-LSTM and other common deep learning models. ECC-LSTM models performed better than all the other models in detecting the onset time of abnormal operation, with ~15% improvement in the timeliness error metric and ~17% drop in the mean absolute error of TOAB predictions compared to CC-LSTM models, and ~25-40% better than models based on commonly used deep learning techniques. Reliable predictions of abnormal behavior onset time from the proposed ensemble approach could be used to improve scheduling and inventory management for predictive maintenance of industrial equipment and prevent catastrophic failure of critical equipment.

ACKNOWLEDGEMENT

The authors thank the management of Tata Consultancy Services Ltd. for the permission to publish this paper.

REFERENCES

Aljbali, S., & Roy, K. (2021). Anomaly Detection Using Bidirectional LSTM. *Advances in Intelligent Systems and Computing* (pp. 612-619). Amsterdam: Springer.

Altarabichi, M., Mashhadi, P., Fan, Y., Pashami, S., Nowaczyk, S., Moral, P., . . . Rognvaldsson, T. (2020). Stacking ensembles of heterogeneous classifiers for fault detection in evolving environments. *The 30th European Safety and Reliability Conference and the 15th Probabilistic Safety Assessment and Management Conference* (p. 1068). Venice, Italy: Research Publishing.

Assaf, R., Do, P., Scarf, P., & Nefti-Meziani, S. (2017). Diagnosis for Systems with Multi-component Wear Interactions. *IEEE Intl Conf on Prognostics and Health Management.* Dallas, TX, USA: IEEE.

Bansal, M. (2010). Performance evaluation of butterworth filter for signal denoising. *International Journal of Electronics and Communication Technology*, 59-62.

Bian, L., & Gebraeel, N. (2014). Stochastic modeling and real-time prognostics for multi-component systems with degradation rate interactions. *IIE Transactions*, 470-482.

Canizo, M., Triguero, I., Conde, A., & Onieva, E. (2019). Multi-head CNN-RNN for multi-time series anomaly detection: An industrial case study. *Neurocomputing*, 246-260.

Cannarile, M., Compare, M., Bareldi, P., Yang, Z., & Zio, E. (2020, August). *The Aramis Challenge: Prognostics and Health Management in Evolving Environments.* Retrieved from ESREL2020-PSAM15: https://www.esrel2020-psam15.org/Aramis_Challenge.pdf

Chalapathy, R., & Chawla, S. (2019). Deep learning for anomaly detection: A survey.

Eren, L., Ince, T., & Kiranyaz, S. (2019). A generic intelligent bearing fault diganosis system using compact adpative 1D CNN classifier. *Journal of Signal Processing Systems*, 179-189.

Gupta, A., Masampally, V. S., Jadhav, V., Deodhar, A., & Runkana, V. (2021). Supervised Operational Change Point Detection using Ensemble Long-Short Term Memory in a Multicomponent Industrial System. *IEEE 19th World Symposium on Applied Machine Intelligence and Informatics (SAMI)*. Herl'any, Slovakia: IEEE.

Hasan, M., Sohaib, M., & Kim, J. (2019). 1D CNN based trasnfer learning based model for bearing fault diagnosis under variable working conditions. *Advances in Intelligent Systems and Computing*, 888.

Introduction to the Keras Tuner. (2022, March 25). Retrieved from Tensorflow: https://www.tensorflow.org/tutorials/keras/keras_tuner

Keras API Reference. (2021, May 05). Retrieved from Keras: https://keras.io/api/

Kim, T., & Cho, S. (2018). Web traffic anomaly detection using C-LSTM neural networks. *Expert Systems with Applications*, 66-76.

Kiranyaz, S., Avci, O., Abdeljaber, O., Ince, T., Gabbouj, M., & Inman, D. (2021). 1D convolutional neural networks and applications: A survey. *Mechanical Systems and Signal Processing*, 107398.

Krizhevsky, A., Sutskever, I., & Hinton, G. (2017). Imagenet classification with deep convolutional neural networks. *Communication of the ACM*, 84-90.

Lai, Y., Zhang, J., & Liu, Z. (2019). Industry anomaly detection and attack classification method based on convolutional neural network. *Security and Communication Networks*.

Li, L., Jamieson, K., DeSalvo, G., Rostamizadeh, A., & Talwalkar, A. (2018). Hyperband: A novel bandit-based approach for hyperparameter optimization. *Journal of Machine Learning Research*, 1-52.

Li, Y., Zou, L., Jiang, L., & Zhou, X. (2019). Fault diagnosis of rotating machinery based on combination of deep belief network and one dimensional convolutional neural network. *IEEE Access*, 7.

Li, Z., Li, J., Wang, Y., & Wang, K. (2019). A deep learning approach for anomaly detection based on SAE and LSTM in mechanical equipment. *International Journal of Advanced Manufacturing Technology*, 499-510.

Lin, Y., Zakwan, S., & Jennions, I. (2020). A Bayesian Approach to Fault Identification in the Presence of Multi-component Degradation. *International Journal of Prognostics and Health Management*, 1-9.

Lindemann, B., Jazdi, N., & Weyrich, M. (2020). Anomaly detection and prediction in discrete manufacturing based on cooperative LSTM networks. *IEEE 16th International Conference on Automation Science and Engineering (CASE)*, (pp. 1003-1010). Hong Kong.

Lindemann, B., Maschler, B., Sahlab, N., & Weyrich, M. (2021). A survey on anomaly detection for technical systems using LSTM networks. *Computers in Industry*, 103498.

Liu, H., Zhao, Y., Zaporowska, A., & Zakwan, S. (2020). A machine learning-based clustering approach to diagnose multi-component degradation of aircraft fuel systems. *Neural Computing and Applications*.

Malhotra, P., Ramakrishnan, A., Anand, G., Vig, L., Agarwal, P., & Shroff, G. (2016). LSTM-based Encoder-Decoder for Multi-sensor Anomaly Detection. *ICML Anomaly Detection Workshop*. New York, NY, USA.

Prakash, O., Samantaray, A., Bhattacharyya, R., & Ghoshal, S. (2018). Adaptive Prognosis for a Multi-component Dynamical System of Unknown Degradation Modes. *IFAC PapersOnLine*, 184-191.

Rocchetta, R., Petkovic, M., & Gao, Q. (2020). Scenario-based Generalization bound for Anomaly Detection Support Vector Machine Ensembles. *Proceedings of the 30th European Safety and Reliability Conference and the 15th Probabilistic Safety Assessment and Management Conference*, (pp. 1069-1076). Venice, Italy.

Saxena, A., Celaya, J., Balaban, E., Goebel, K., Saha, B., Saha, S., & Schwabacher, M. (2008). Metrics for evaluating performance of prognostics techniques.

International Conference on Prognostics and Health Management.

Shang, C., Yang, F., Huang, D., & Lyu, W. (2014). A multiscale feature learning scheme based on deep learning for industrial process monitoring and fault diagnosis. *Journal of Process Control*, 223-233.

Siahpour, S., Ainapure, A., Li, X., & Lee, J. (2020). A Deep Learning Framework for Health Anomaly Detection of Multi-component Systems in Evolving Environments: A case study in PHM. *30th European Safety and Reliability Conference and 15th Probabilistic Safety Assessment and Management Conference* (pp. 1077-1084). Venice, Italy: Research Publishing.

Snedecor, G., & Cochran, W. (1989). *Statistical Methods.* Iowa State University Press.

Ullah, W., Ullah, A., Haq, I., Muhammad, K., Sajjad, M., & Baik, S. (2020). CNN features with bi-directional LSTM real-time anomaly detection in surveillance networks. *Multimedia tools and applications*, 80.

Wang, D., Guo, Q., Song, Y., Gao, S., & Li, Y. (2019). Application of multiscale learning neural network based on CNN in bearing fault diagnosis. *Journal of Signal Processing Systems*, 91.

Wang, X., Mao, D., & Li, X. (2020). Bearing fault diagnosis based on vibro-acoustic data fusion and 1D-CNN network. *Measurement*.

Yang, Z., Baraldi, P., & Zio, E. (2022). A method for fault detection in multi-component systems based on sparse autoencoder-based deep neural networks. *Reliability Engineering and System Safety, 220*, 1-15.

Yuan, J., & Tian, Y. (2019). A multiscale feature learning scheme based on deep learning for industrial process monitoring and fault diagnosis. *IEEE Access*, 151189-151202.

Zhang, W., Yang, D., & Wang, H. (2019). Data-Driven Methods for Predictive Maintenance of Industrial Equipment: A Survey. *IEEE Systems Journal*, 2213-2227.

Zhao, G., Zhang, G., Ge, Q., & Liu, X. (2016). Research advances in fault diagnosis and prognostics based on deep learning. *Prognostics and System Health Management Conference*, (pp. 1-6). Chengdu.

Zheng, L., Xue, W., Chen, F., Guo, P., Chen, J., Chen, B., & Gao, H. (2019). A fault prediction of equipment based on CNN-LSTM network. *IEEE International Conference on Energy Internet*, (pp. 537-541).

Zhou, C., Sun, C., Liu, Z., & Lau, F. (2015). A C-LSTM Neural Network for Text Classification. *arXiv*, arXiv:1511.08630.