

T.R.
GEBZE TECHNICAL UNIVERSITY
GRADUATE SCHOOL OF NATURAL AND APPLIED SCIENCES

SIAMESE NETWORKS FOR PREDICTIVE MAINTENANCE

UĞUR CEYLAN
A THESIS SUBMITTED FOR THE DEGREE OF
MASTER OF SCIENCE
DEPARTMENT OF COMPUTER ENGINEERING

GEBZE
2022

T.R.
GEBZE TECHNICAL UNIVERSITY
GRADUATE SCHOOL OF NATURAL AND APPLIED SCIENCES

**SIAMESE NETWORKS FOR
PREDICTIVE MAINTENANCE**

UĞUR CEYLAN

**A THESIS SUBMITTED FOR THE DEGREE OF
MASTER OF SCIENCE
DEPARTMENT OF COMPUTER ENGINEERING**

**THESIS SUPERVISOR
ASSIST. PROF. DR. YAKUP GENÇ**

**GEBZE
2022**



YÜKSEK LİSANS JÜRİ ONAY FORMU

GTÜ Fen Bilimleri Enstitüsü Yönetim Kurulu'nun 03/02/2022 tarih ve 2022/06 sayılı kararıyla oluşturulan jüri tarafından 16/02/2022 tarihinde tez savunma sınavı yapılan Uğur CEYLAN'ın tez çalışması Bilgisayar Mühendisliği Anabilim Dalında YÜKSEK LİSANS tezi olarak kabul edilmiştir.

JÜRİ

ÜYE

(TEZ DANIŞMANI) : Dr. Öğr. Üyesi Yakup GENÇ

ÜYE

: Prof. Dr. Fatih Erdoğan SEVİLGİN

ÜYE

: Doç. Dr. Behçet Uğur TÖREYİN

ONAY

Gebze Teknik Üniversitesi Fen Bilimleri Enstitüsü Yönetim Kurulu'nun
...../...../..... tarih ve/..... sayılı kararı.

SUMMARY

Predictive maintenance reduces engine maintenance costs in power plants, aircraft, and factories by predicting when maintenance is needed (or by estimating the remaining useful life). Recently, with significant advances in deep learning and the availability of high volumes of data extracted from manufacturing processes, data-driven methods to estimate remaining useful life (RUL) have received great interest. A major problem with data-based prognosis is that it is costly and requires large amounts of training data that could be difficult to obtain in large numbers of failure cases (often impossible and expensive to obtain in the real world).

This thesis proposes a learning-based method for fault diagnosis requiring fewer failure data. To this end, we use a Siamese network architecture founded on a specific deep Convolutional Neural Network (CNN) termed InceptionTime. The Siamese part of the network allows repeated use of the existing data to establish a similarity metric for two separate time windows. The turbofan engines C-MAPSS dataset supplied by NASA is used to verify the proposed model.

Experiments are conducted using various sizes of the turbofan engines data to compare the performance on small datasets between the proposed model and a baseline model. In addition, studies were carried out to improve the proposed model's performance via applying various literature approaches. The results demonstrate that our model can be used in fault diagnosis and provide satisfying prediction results with fewer data with comparable performances against state-of-art methods for RUL prediction.

Keywords: Prognostics and Health Management (PHM), Remaining Useful Life (RUL), C-MAPSS, Siamese Networks, Deep Learning (DL).

ÖZET

Tahmine dayalı bakım, ne zaman bakım gerektiğini tahmin ederek (veya kalan kullanım ömrünü tahmin ederek) enerji santrallerinde, uçaklarda ve fabrikalarda motor bakım maliyetlerini azaltır. Son zamanlarda, derin öğrenmedeki önemli ilerlemeler ve üretim süreçlerinden çıkarılan yüksek hacimli verilerin kullanılabilirliği ile birlikte, kalan faydalı ömrü (KFÖ) tahmin etmeye yönelik veriye dayalı yöntemler büyük ilgi gördü. Veriye dayalı prognozla ilgili önemli bir sorun, maliyetli olması ve çok sayıda hata durumunda elde edilmesi zor olabilecek büyük miktarda eğitim verisi gerektirmesidir (genellikle gerçek dünyada elde edilmesi imkansız ve pahalıdır).

Bu tez, daha az arıza verisi gerektiren arıza teşhisi için öğrenmeye dayalı bir yöntem önermektedir. Bu amaçla, InceptionTime olarak adlandırılan belirli bir derin Evrişimli Sinir Ağı (ESA) üzerine kurulmuş bir Siyam ağ mimarisi kullanıyoruz. Ağın Siyam kısmı, iki ayrı zaman penceresi için bir benzerlik ölçüsü oluşturmak için mevcut verilerin tekrar tekrar kullanılmasına izin verir. Önerilen modeli doğrulamak için NASA tarafından sağlanan turbofan motorları C-MAPSS veri seti kullanılmıştır.

Önerilen model ve bir temel model arasındaki küçük veri kümelerindeki performansı karşılaştırmak için çeşitli boyutlarda turbofan motor verileri kullanılarak deneyler yapılmıştır. Ayrıca, çeşitli literatür yaklaşımları uygulanarak önerilen modelin performansının iyileştirilmesine yönelik çalışmalar yapılmıştır. Sonuçlar, modelimizin arıza teşhisinde kullanılabileceğini ve RUL tahmini için son teknoloji yöntemlere kıyasla karşılaştırılabilir performanslarla daha az veri ile tatmin edici tahmin sonuçları sağladığını göstermektedir.

Anahtar Kelimeler: Prognostik ve Sağlık Yönetimi (PSY), Kalan Faydalı Ömür (KFÖ), C-MAPSS, Siyam Ağları, Derin Öğrenme (DÖ).

ACKNOWLEDGEMENTS

I would like to state my deepest indebtedness to my supervisor Assist. Prof. Yakup Genç for his help, advice, and motivation through my graduate education. I have acquired a better research view thanks to him patiently sharing his comprehensive knowledge and experience. Besides his help in completing my thesis, he has always advised me about the future of my journey in machine learning and computer vision research. It has been an enjoyment for me to work with him and to learn from him. I also would like to thank the thesis jury members who contributed to the formation of my thesis for their suggestions.



TABLE of CONTENTS

	<u>Page</u>
SUMMARY	v
ÖZET	vi
ACKNOWLEDGEMENTS	vii
TABLE of CONTENTS	viii
LIST of ABBREVIATIONS and ACRONYMS	ix
LIST of FIGURES	x
LIST of TABLES	xi
1. INTRODUCTION	1
2. BACKGROUND	3
3. METHODOLOGY	8
3.1. Data Overview	8
3.2. Siamese InceptionTime Network	12
4. EXPERIMENTS AND RESULTS	18
4.1. The Siamese InceptionTime Results	18
4.2. Experiments on Small Datasets	20
4.3. Model Improvements	26
5. CONCLUSION AND FUTURE WORK	29
REFERENCES	30
BIOGRAPHY	33
APPENDICES	34

LIST of ABBREVIATIONS AND ACRONYMS

<u>Abbreviations and</u>	<u>Explanations</u>
<u>Acronyms</u>	
C-MAPSS	: Commercial Modular Aero-Propulsion System Simulation
CNN	: Convolutional Neural Network
DAG	: Directed Acyclic Graph
DL	: Deep Learning
DNN	: Deep Neural Network
exp	: Euler's number
LSTM	: Long Short-Term Memory
ML	: Machine Learning
PdM	: Predictive Maintenance
PHM	: Prognostics and health management
ReLU	: Rectifier Linear Unit
RMSE	: Root Mean Square Error
RNN	: Recurrent Neural Network
RUL	: Remaining Useful Life

LIST OF FIGURES

<u>Figure No:</u>	<u>Page</u>
3.1: A simplified diagram of simulation engine in C-MAPSS.	8
3.2: Some sensor outputs of engine id 1 in C-MAPSS.	10
3.3: InceptionTime network residual connection/Inception module details.	13
3.4: Inception network for time series classification.	13
3.5: The architecture of the proposed Siamese InceptionTime Network.	14
3.6: Extracting health/failure samples.	15
4.1: Predicted RUL for each engine in the FD001 test sub-dataset.	19
4.2: BINC and RUL outputs of engines #41 and #42 in FD001 test sub-dataset.	19
4.3: Classification results input of pairs healthy/failure samples of test data.	20
4.4: Hyperparameters RMSE (top) and Score (bottom) results.	23
4.5: Parameters insensitive RMSE (top) and Score (bottom) results.	24
4.6: Performance (top – RMSE, bottom – Score) change when the first 15% of each data is ignored during training.	25
4.7: Changes on the proposed model architecture for improvement.	26

LIST OF TABLES

<u>Table No:</u>	<u>Page</u>
3.1: C-MAPSS inputs to simulate diverse degradation scenarios in any of the five rotating components of the simulated engine.	9
3.2: 21 sensor outputs of simulation engine running.	11
3.3: Description FD001 sub-dataset of the C-MAPSS.	11
3.4: Healthy/failure sample sizes for each engine after converting the FD001 of the C-MAPSS to a metric-learning task.	16
4.1: Engine RUL prediction comparison in RMSE on FD001 test dataset.	21
4.2: Changes in model architecture results.	27
4.3: Model results with magnitude pruning.	28

1. INTRODUCTION

Prognostics and health management (PHM) of engineering systems, mechanical materials, and their critical components has attracted attention and predicting remaining useful life (RUL) which is the total amount of cycles/time a given machinery/component can still work before failure is the core of PHM [1], [2]. By estimating the RUL of the machinery, it is possible to maximize the time of usage and operation of equipment, delay/reduce maintenance activities and reduce material and labor costs as well as it is likely to provide preventive mechanical actions and offer operation and maintenance planning activities in a targeted way [3]. There are three main methods to estimate mechanical equipment RUL: data-driven, model-based and hybrid approaches. Basically, in the former, machinery is run until a fault occurs and the runs are repeated multiple times to collect data. The latter system requires knowledge of the physics of the breakdown progression while hybrid approaches use the properties of the two methods together [4]. In recent years, the RUL studies for data-driven approaches have drawn considerable attention with the significant collection of the high volume of sensor data with IoT devices and improving computing systems when physical modeling is difficult to obtain for complex systems.

In this thesis, RUL estimation is predicted using a data-driven method. Data-driven approaches offer fault detection solutions after certain faults occur (diagnostics) and forecasts of future operating situations and the time to failure (prognosis). The main challenge in data-driven approaches is that it is usually difficult to obtain a high volume number of samples for failure progression which is expensive and labor demanding [4]. This can appear in a few different scenarios: (1) industry systems are not permitted to work until failure due to the outcomes, particularly for crucial systems and failures; (2) most of the mechanical breakdowns happen gradually and result in a degradation path such that failure degradation of a system might take months or even years. Various techniques have been used to address this challenge. The first technique is to quicken aging by operating the system in a lab with excessive loads, increased speed, or doing simulations of real components that are made by weak materials so that a failure proceeds faster than normal [5]. Another technique is organizing failure progress unnaturally using exponential degeneration to model natural failure progression [6]. Both techniques have their benefits and limitations with the ability to

interpret the failure degradation. But, in real-world applications where the circumstance is very distinct from the lab conditions, these techniques are challenging to perform and to obtain sufficient predicting achievement.

Recently, it has been shown that machine learning approaches, especially deep learning methods, perform powerful outcomes in areas such as computer vision, image processing, and natural language processing [7]. Deep learning methods have been employed for fault diagnosis and applied to the RUL estimation problem, in particular using ensemble networks (different types of networks combination). Using a combination of a Convolutional Neural Network (CNN) and a Long Short-Term Memory (LSTM) neural network proposes an encouraging method to estimate RUL with a convenient and precise method that can be extended and adapted to obtain more accurately. However, one of the biggest limitations is that they require a large number of labeled failure cases data. In many applications, it is sometimes not possible to collect a lot of data to determine the condition of an asset. We also need to retrain the model whenever we want to classify class or estimate the RUL of a new failure case data. Therefore, the proposed method aims to solve those problems by using a Siamese network.

The Siamese networks try to model the discriminant nature of the data from two opposite samples. This can be leveraged to generalize the network's predictive ability not only to new data but to entirely new types from unknown distributions [8]. By allowing repeated use of the existing data to establish a similarity metric for two separate time windows, the network can generalize predictive ability on small datasets and address the problem of lacking training samples.

This thesis proposes a Siamese network that is based on a deep Convolutional Neural Network (CNN), called Inception-Time to increase the precision of RUL prediction performance. The problem is posed as modeling a metric between two time-windows observed data, similar being on the same side of the failure start point, and different being on the opposite sides.

The rest of the thesis is organized as Section 2 provides a brief literature review for RUL estimation methods with a focus on machine learning (ML) methods and deep learning (DL) approaches. Section 3 describes the pre-processing of input data to the Siamese InceptionTime network. Also, in this section details about the proposed Siamese InceptionTime Network are given. Experiments and results are explained in Section 4 with Section 5 concluding the thesis.

2. BACKGROUND

The state-of-art for predictive maintenance shows that the newest studies on the topic are data-driven approaches and model-driven studies are out of date. Data-driven studies can be classified into two main groups. The first group tries to predict the remaining useful life. Historical sensor data from the machinery/engine is employed as a time series and the problem is converted into a regression problem [9], [10], [13], [23]. In the second group study, the status of the machinery/engine is characterized as healthy, slightly damaged, or very damaged, and transformed into a classification problem [12], [25], [26], [30]. Many distinct methodologies have been stated in the literature to estimate RUL in this manner.

Early studies on predictive maintenance for RUL estimation are based on statistical methods and ML-based approaches. For instance, Ordóñez et al. [9] proposed a data-based approach on turbofan engine dataset using auto-regressive integrated moving average (ARIMA) forecasting prediction as input of the support vector machine (SVM) model. The best kernel function selection is performed using genetic algorithms strategies. The predictive capability of their proposed hybrid method is shown to be good compared to the ones obtained solving the problem once using multivariate Vector Autoregressive Moving-Average (VARMA) models. Khelif et al. [10] used SVR-RUL to model the direct relationship between sensor values or health index (HI) and predicted RUL directly from sensor values without estimating deterioration state or failure threshold.

Durbhaka et al. [11] use k-means to analyze the behavior of wind turbines using vibration signal analysis. In this study, k-NN and SVM algorithms are compared with the k-means algorithm to classify failure types in wind turbines. S. Behera et al. [12] proposed a predictive maintenance model that applies a data-driven method for identification of the condition engine and solution formulated as classification manner with one of the three possible values critical, warning, and normal depending upon the RUL value. Utilizing advanced feature engineering, they obtained more insight and used ensemble tree learning. Comprehensive experiments are conducted on a widely used C-MAPSS dataset. It has been seen that Gradient Boosted Trees (GBT) accuracy performs better than Random Forest accuracy. However, RF competitively performed with a much faster compute time in comparison to the GBT.

Recently as an ML-based approach, Ayvaz et al. [13] developed an end-to-end predictive maintenance system for detecting potential failure signals in production lines of manufacturing with employed data in real-time from IoT sensors of the real-world dataset by performing multiple machine learning algorithms. Since data high dimensional and very highly correlated variables were derived from sensors, to avoid the curse of dimensionality problem applied PCA as dimensional reduction during feature selection, and then some features were eliminated based on the predictive power of features in the Random Forest algorithm. They considered a problem as a regression task rather than classification. Ultimately, they obtained the best performance with Random Forest and XGBoost method with measure R^2 , MAE, MAPE, RMSE metrics among other various ML algorithms. Jia et al. [14] worked on the first turbofan engine dataset using XGBoost and compared its results with other traditional machine learning algorithms such as Decision Tree, Random Forest, and SVM. Results of XGBoost outperform other algorithms.

Deep neural network (DNN) based methods have gained dominance in the last decade. For instance, Xu et al. [15] the healthy status of the engine with the data obtained from 21 sensors belonging to 100 different engines was tried to be monitored. The degradation behaviors of the 21 sensory signals observed and seven of them were selected in this study. The remaining data were estimated by a mixed system consisting of Dempster-Shafer Regression (DSR), SVM, and Recurrent neural networks (RNN).

Babu et al. [16] built the first attempt of a deep CNN for RUL estimation across two public data sets. The CNN structure is employed to extract the local data features through the deep learning network for better prognostics. Their approach outperformed SVM and MLP based methods. Zhang et al. [17] proposed a neural network-based approach that a multi-objective deep belief networks ensemble (MODBNE) method. Deep belief networks (DBNs) are used to handle two issues. The first issue is utilizing handcrafted features which can hardly adjust to different predictive applications. It is employed DBNs as a solution since DBNs offer a promising solution to extracting the most useful features relevant to the estimation task via learning powerful hierarchical attribute representations from data. The other issue is utilizing one single model which can hardly keep good generalization performance across diverse predictive methods. They handle this problem with a multi-objective evolutionary ensemble learning framework incorporated with the traditional DBN.

Li, et al. [18], proposed a deep learning CNN architecture to estimate RUL. For better feature extraction with CNN, the sliding the time window approach is used for sample preparation. The turbofan engine raw sensor data measurements with scaled between $[-1, 1]$ and then used as input into the proposed network deep CNN architecture. This helps the industrial applications that do not need any prior expertise in signal processing. The high-level abstract features can be extracted with deep CNN architecture and related RUL can be estimated from the basis of the representation learned. The proposed method using time window, data normalization, and deep CNN structure is expected to achieve higher prognostic accuracy compared to traditional machine learning methods.

Hanga et al. [19], proposed an applied LSTM network, an architecture that specializes in exploring fundamental patterns embedded in time series to monitor system degeneration and eventually predict RUL. The purposes of this paper firstly explain raw sensor data into an interpretable health index to better characterize the health status, and secondly monitor system corruption of the past to predict the future health situation correctly. For those purposes they used the C-MAPSS engine dataset and results show that LSTM produces more robust and accurate in explaining degeneration patterns, enabled by discovering the variation pattern underlying time-series is examined to track the system degradation.

Hinchi et al. [20] proposed a deep neural network framework that is based on CNN and LSTM recurrent units for RUL estimation. First, the neural network subtracts the local features directly from sensor data using the convolutional layer, followed by an LSTM layer to catch the degradation process is added, RUL estimated finally using LSTM output and estimated time value. Experiments used ball-bearing data with good performances. Once again, LSTM models require representative large training data sets.

Zhang et al. [21] proposed for RUL estimation, a transfer learning algorithm based on Bi-directional Long Short-Term Memory (Bi-LSTM) recurrent neural networks that models can be trained on different but related datasets first and then fine-tuned by the target dataset. Experimental results show that transfer learning can improve prediction models in the dataset with a small number of samples with one exception when transferring learning leads to a worse result when changing from multi-type to single operating conditions. Transfer learning methods however rely on a large training data.

Zhang et al. [22] used a temporal convolutional network (TCN) with Autoencoder on the turbofan engine dataset. Autoencoder network employed for data dimension reduction and since TCN capable process sequence model task, they compared its results with other sequence modeling such as RNN, GRU, LSTM, etc. and TCN outperforms all other methods.

Li et al. [23] proposed a directed acyclic graph (DAG) network combining LSTM and a CNN for RUL estimation. The proposed method rather than just using CNN for extracting features it combines CNN and LSTM. The resulting model shows improved performance. We will use this method as our baseline model. Residual connections existed in specified layers of the network for the gradient vanishing problem.

Finally, Siamese networks have been used in the literature quite a lot. Most of these are in the image processing domain. Such as Koch et al. [8] use a Siamese network and employed it for identifying characters that consist of comprehensive alphabet sets. By using the Siamese convolutional architecture, they can achieve powerful results on one-shot classification tasks that exceed those of other deep learning models with state-of-the-art performance. Generally, with the Siamese neural networks, they learn image representations through a supervised metrics-based strategy and then reuse the properties of that network without any retraining. Hsiao et al. [24] applied Siamese neural networks to the malware image classification task. They implemented a method for identifying distinct types of malware for malware propagation using malware image classification. They also address the issue of lack of training samples at the early stage of new malware appearance by implementing Siamese networks.

Besides, recently some articles have mentioned the Siamese networks for fault diagnosis as a one-shot learning task. For instance, Xing et al. [25] employed the Siamese network with CNN as its sub-network to issue address the lack of unlabeled bearing data. Despite Euclidean distance usage to measure similarity between samples of the Siamese network inputs, they measured this distance with a maximum mean difference (MMD) based on probability distribution. Basically, training data was collected with specific frequency from six bearings and three of them were used as training and the remaining separated for testing. They handled the data with a classification approach with three states that health, outer race defect, and inner race defect of condition bearings. The pairing of combination dataset samples was given

into sub-network CNN of the Siamese and extracted low-dimensional features measure similarity differences using MDD. Lastly, output this passed through the sigmoid function to detect if belongs to the same or different states. The experimental results demonstrated that the Siamese network can be used for lack of data for fault diagnosis with high performance.

Xu et al. [26] proposed a convolutional Siamese network with a cross-entropy-based loss function for fault diagnosis and method name termed as cross-category fault diagnosis method (CFDM). Since the Siamese network learning distance distribution between input pairs rather than training examples, the employed network can distinguish not just fault categories samples in train data but also for new fault categories samples. They used three datasets to evaluate the performance of the network and results showed if there were only a few labeled samples for each fault category, the network make a good generalization on a completely new type target category without retraining or refinement.

However, the Siamese network employed for fault diagnosis as approach classification manner in mentioned papers, in our approach we combine regression and classification tasks in one network for RUL estimation and distinguish of failure/health samples and address the problem lack of labeled failure data.

3. METHODOLOGY

This thesis presents a Siamese network based on InceptionTime architecture for RUL estimation using time series data. It is used in the turbofan engine degradation dataset. We first provide the details of the turbofan engine degradation dataset. Then we present the proposed Siamese InceptionTime network, as well as its key components, i.e., InceptionTime and Siamese networks.

3.1. Data Overview

The turbofan engine degradation dataset is a well-known data set based on a simulation by Commercial Modular Aero-Propulsion System Simulation (C-MAPSS) tool developed by NASA [27]. A simplified diagram of the simulation engine is shown in Figure 3.1.

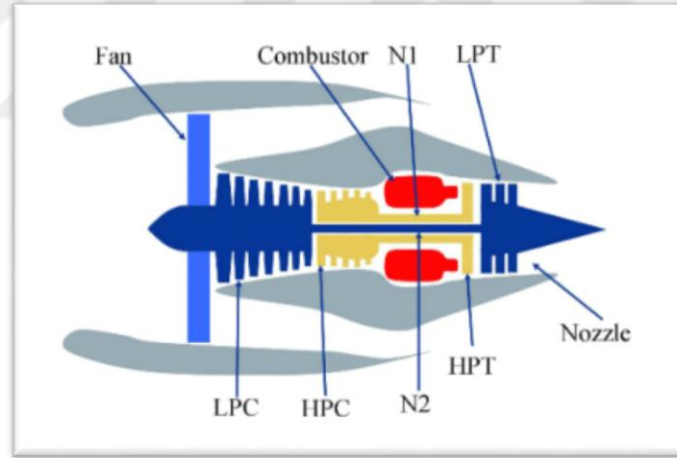


Figure 3.1: A simplified diagram of simulation engine in C-MAPSS.

C-MAPSS is coded in the MATLAB and Simulink environment for simulating a realistic large commercial turbofan engine. The tool has several editable inputs to let the user enter specific values of their preferences concerning Fan pressure-ratio modifier, Fuel flow, Fan flow modifier, environmental conditions, etc. A complete control system's main elements consist of a high-pressure turbine (HPT), low-pressure turbine (LPT), low-pressure compressor (LPC), high-pressure compressor (HPC), fan, combustor, and nozzle. C-MAPSS has 14 facets (Table 3.1) that possess fuel flow and

a set of 13 health-parameter inputs that permit the user to mimic the outcomes of faults and degradation of the engine's five rotating elements (Fan, LPC, HPC, HPT, and LPT) and can bring out several outputs (Table 3.2) that symbolizes the health condition of the turbofan engine.

Table 3.1: C-MAPSS inputs to simulate diverse degradation scenarios in any of the five rotating components of the simulated engine. For example, to simulate HPC degradation HPC flow and efficiency modifiers were used (highlighted in gray).

Name	Symbol
Fuel flow	Wf
Fan efficiency modifier	fan eff mod
Fan flow modifier	fan flow mod
Fan pressure-ratio modifier	fan PR mod
LPC efficiency modifier	LPC eff mod
LPC flow modifier	LPC flow mod
LPC pressure-ratio modifier	LPC PR mod
HPC efficiency modifier	HPC eff mod
OPC flow modifier	HPC flow mod
HPC pressure-ratio modifier	HPC PR mod
HPT efficiency modifier	HPT eff mod
HPT flow modifier	HPT flow mod
LPT efficiency modifier	LPT eff mod
HPT flow modifier	LPT flow mod

In Table 3.2, column 3 'Trend' legend illustrates the trend of the output to deteriorate where $\uparrow$ parameter which implies that the incline over time, $\downarrow$ denotes that the parameter is decline with time, and remains express $\sim$ that the parameter is irregular with time.

Visualization of some output sensors data with normalized belongs to engine id 1 is illustrated in Figure 3.2 where data belongs to some output sensors in the first row has no regular trend and in the second row has declines trend while third-row has incline trends over time.

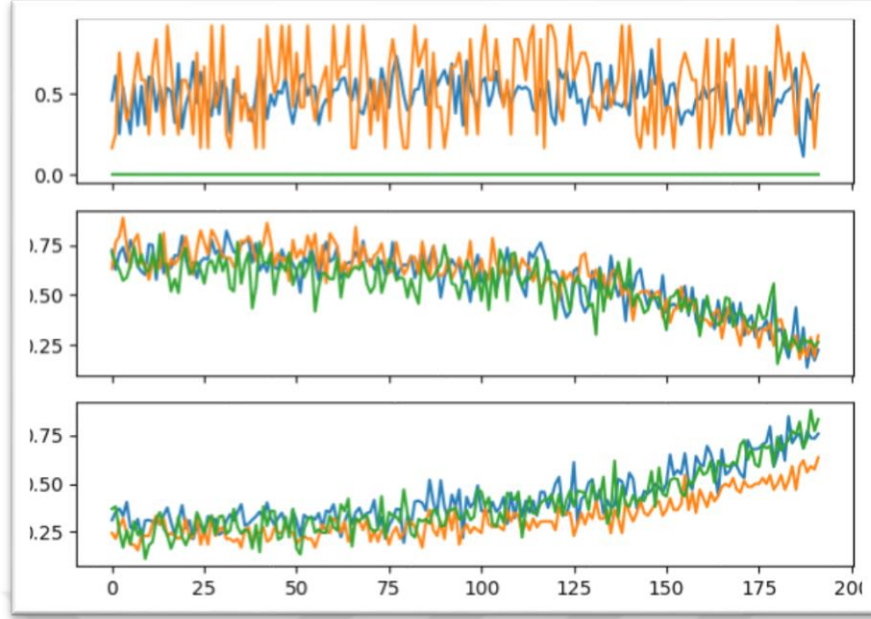


Figure 3.2: Some sensor outputs of engine id 1 in C-MAPSS.

The dataset consists of four sub-datasets based on different operating conditions and fault modes. Each sub-dataset consists of multiple multivariate time series which is a 24-dimensional feature vector consisting of 3 operational settings and 21 sensor values. Each sub-dataset includes training data, test data, and actual RUL that corresponds to the last cycle of the test data. The training data comprises a certain number of engines data from a specific healthy state to failure, while the test data includes a certain number of engine data that is a bit before operating failure. Furthermore, the training and test data engines data with different initial healthy states, respectively. Engines because of different initial healthy states, the operating cycles of distinct engines are different in the same database. Data also include some noise.

In this study, sub-dataset FD001 was used. A short description of the FD001 sub-datasets is given in Table 3.3. In sub-dataset FD001 of the database, both the training and test datasets include 100 engines. The training dataset engine life with a minimum operating cycle of 128 and a maximum operating cycle of 362 and the test dataset includes 100 engines, with a minimum operating cycle of 31 and a maximum operating cycle of 303, and collected under one operating condition with one fault mode.

Table 3.2: 21 sensor outputs of simulation engine running.

Symbol	Description	Trend
T2	Total temperature at fan inlet	~
T24	Total temperature at LPC outlet	↑
T30	Total temperature at HPC outlet	↑
T50	Total temperature at LPT outlet	↑
P2	Pressure at fan inlet	~
P15	Total pressure in bypass-duct	~
P30	Total pressure at HPC outlet	↓
Nf	Physical fan speed	↑
Nc	Physical core speed	↑
epr	Engine pressure ratio	~
Ps30	Static pressure at HPC	↑
phi	Ratio of fuel flow	↓
NRf	Corrected fan speed	↑
NRc	Corrected core speed	↓
BPR	Bypass Ratio	↑
farB	Burner fuel-air ratio	~
htBleed	Bleed Enthalpy	↑
Nf_dmd	Demanded fan speed	~
PCNfR_dmd	Demanded corrected fan speed	~
W31	HPT coolant bleed	↓
W3	LPT coolant bleed	↓

Table 3.3: Description FD001 sub-dataset of the C-MAPSS.

Sub-dataset	FD001
Engines # in the training set	100
Engines # in the test set	100
Max/Min cycles for training	362/128
Max/Min cycles for test	303/31
Operating condition/Fault mode	1/1

3.2. Siamese InceptionTime Network

InceptionTime is inspired by the Inception-v4 architecture and an ensemble of deep CNN models [28] for time series classification (TSC), each created by cascading multiple Inception modules, where each module has the same architecture [29]. The main building block of InceptionTime is the inception module (illustrated in Figure 3.3, b). The core idea of an inception module is that utilize multiple filters simultaneously to an input time series. The module involves filters of different lengths enabling the network to automatically extract related features from both long and short time series. This consists of the following layers:

- The dimensionality (i.e. depth) of the input is reduced with the bottleneck layer. This reduces the number of parameters, the computational cost and speeds up training, and improves generalization.
- The bottleneck output is fed into three one-dimensional convolutional layers with kernel sizes which in our case are 3, 5, and 7.
- The input of the inception module is also passed through a MaxPooling layer of size 3 and then, through a bottleneck Layer.
- The last layer is a depth concatenation layer where the outputs of the four convolutional layers at step 2 are concatenated along the depth dimension. All convolutional layers have 64 filters.

The network consists of a sequence of Inception modules followed by a Global Average Pooling layer and a Dense layer with a softmax activation function as illustrated in Figure 3.4.

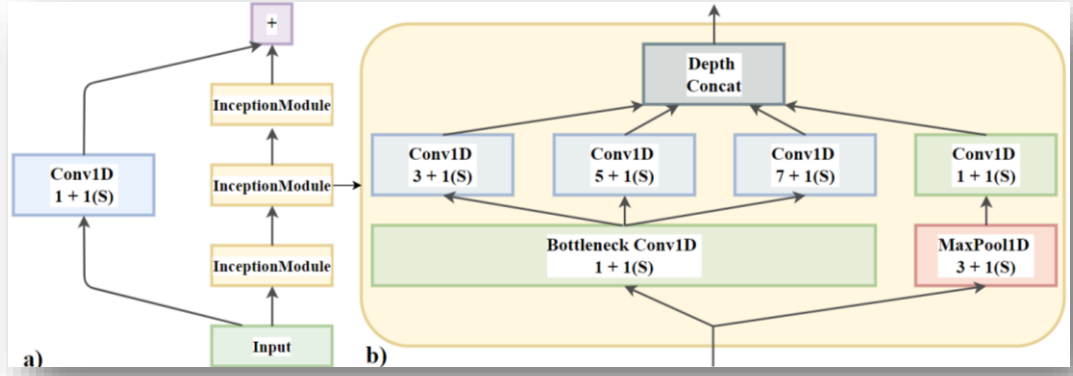


Figure 3.3: a) InceptionTime network residual connection
b) Inception module details.

We make some changes after the Global Average Pooling layer in the framework of the proposed method to adapt our case. Furthermore, InceptionTime introduces a further element within its network's layers: residual connections at every third inception module. Thus decreasing the vanishing gradient problem by allowing a direct flow of the gradient (see in Figure 3.3, a).

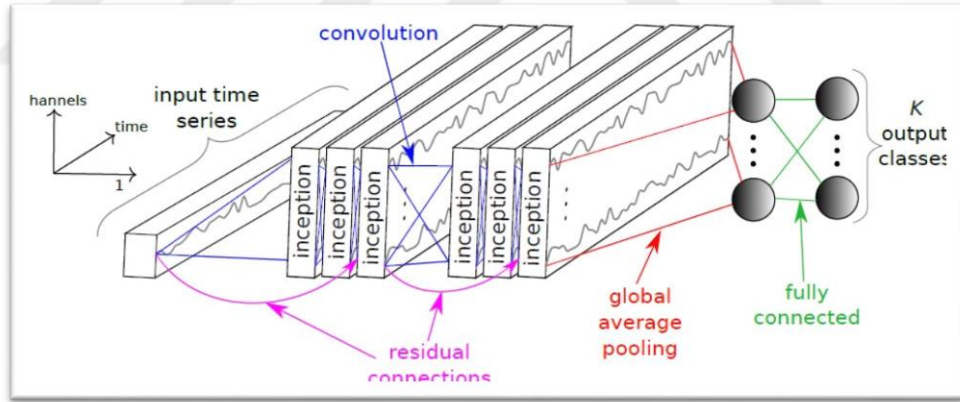


Figure 3.4: Inception network for time series classification.

The main idea of the Siamese network is that two input signals go through a shared network generating fixed-length feature vectors. Assuming the appropriately trained neural network model, we can make the following hypotheses: If two input instances belong to the same instance classes, their feature vectors must be similar, if two input instances belong to different instance classes, their feature vectors will also be different. So absolute differences of the two feature vectors must be very different in both the above cases. The proposed network is shown in Figure 3.5. The FD001

sub-dataset data has simply been processed and through into the Siamese to train network. The specific steps are explained as follows:

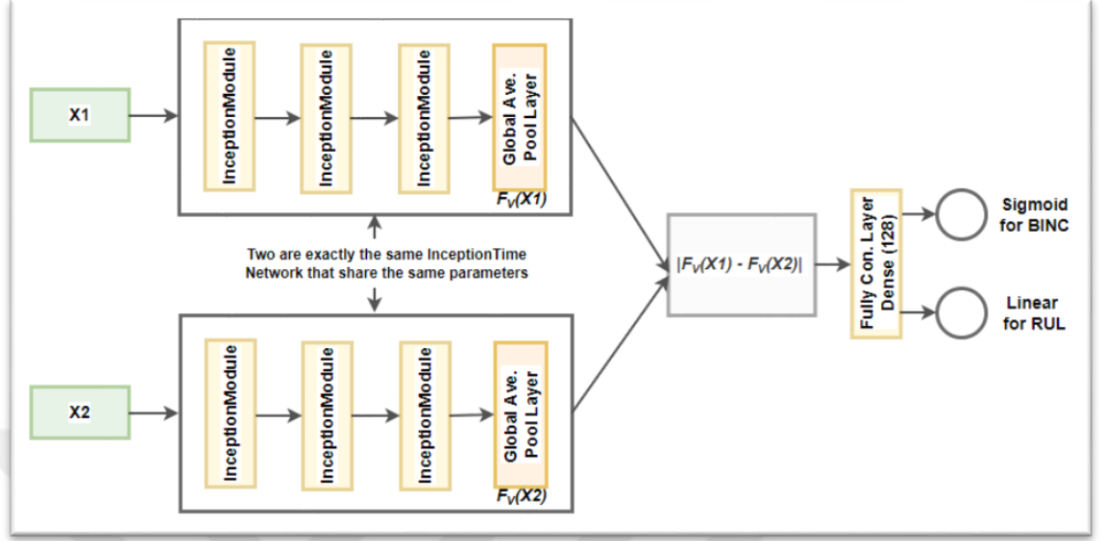


Figure 3.5: The architecture of the proposed Siamese InceptionTime Network.

i) Data Preparation: We first prepare pairs of healthy and failure samples as input for the Siamese network. The sub-dataset has N attributes and the lifetime of the sensors are L_t cycles, i.e., machinery lifetime (see Figure 3.6). The data is obtained by sliding the time window with the length of w_l (length of time window) cycles, and the sliding step size is one cycle. The size of the array extracted each time by time window is $w_l \times N$, and the number of arrays is $L_t - w_l$. So the corresponding RUL $[L_t - w_l, L_t - w_l - 1, \dots, 0]$. The w_l is selected as 30 for the FD001 sub-dataset because it gives the highest precision between results from other time window sizes as proposed [30] and with large time window size includes more data points within the time frame, so the degradation pattern may be simpler. To evaluate all engine's performance in sub-test data and make the comparison between the compared and the results of the proposed method, we can't increase the time window size because of min cycle in the test engine is 31. In our experiment we select 14 sensor outputs with regular trend [23] are chosen as input to the Siamese network, and these 14 sensors range output value varies according to each one, so we scaled data between $[0, 1]$. As a result, our input samples are 14×30 size time-series data.



Figure 3.6: Extracting health/failure samples.

Then, a breakpoint RUL value which is 30 specified (which is illustrated with a solid red vertical line in Figure 3.6) to discriminate between healthy and failure states for the FD001 sub-dataset, and according to this breakpoint value, healthy and failure samples were prepared. If the RUL value of input data is less than the breakpoint value, it was determined as a failure sample and other cases as an example of healthy samples. So, the input of the Siamese InceptionTime network is pairs of the combination of healthy and failure samples. For binary classification (BINC output, see Figure 3.5), data labeled as if a pair input data in the same status (in this study only pairing healthy sample vs healthy sample applied) then the label is 1 other case which is failure vs healthy samples then the label is 0. On the other hand, for output RUL (see Figure 3.5), for each pair of input data labeled as minimum RUL value of input pair samples. According to this manner when the data is prepared, the number of failure/healthy samples size of each engine in the FD001 dataset is as shown in Table 3.4.

ii) Siamese InceptionTime Network: In this method, the InceptionTime network is used as a sub-network of proposed the Siamese network for RUL estimation. The network includes two identical paths that share weights, and each is composed of three Inception modules followed by a Global Average Pooling layer. So, two input samples (x_1 and x_2) are passed through the Siamese InceptionTime Network to generate a fixed-length feature vector for each ($f_v(x_1)$ and $f_v(x_2)$), then, we compute the absolute distance between these feature vectors and passed to a fully connected layer (with a rectified linear unit activation) followed by another fully connected layer (with a sigmoid activation) determining the similarity of the two input.

Depth of inception network set to 3 with 64 filters at each and three convolutions with [3, 5, 7] filters lengths. For training our network, we used Adam optimizer with learning rate 10^{-5} and batch size of 64.

Table 3.4: Healthy/failure sample sizes for each engine after converting the FD001 of the C-MAPSS to a metric-learning task.

engine_id	train_health_size	train_failure_size	test_health_size	test_failure_size
18	136	30	102	2
20	175	30	141	14
24	88	30	147	10
31	175	30	145	22
34	136	30	151	23
35	122	30	150	19
36	99	30	86	11
37	111	30	83	9
40	129	30	102	2
41	157	30	82	12
42	137	30	107	20
49	156	30	265	9
52	154	30	159	1
53	136	30	131	4
56	216	30	92	15
61	126	30	121	9
64	224	30	137	2
66	143	30	102	16
68	140	30	136	22
76	151	30	156	20
81	181	30	162	22
82	155	30	112	21
90	95	30	115	2
92	282	30	111	10
100	141	30	159	10

To evaluate the performance of the proposed Siamese Inceptiontime Network, root mean square error (RMSE) and Score metrics were used. RMSE is a frequently utilized metric for RUL prediction that gives the same penalties while the score has different penalties for early and late predictions [23]. If there is a prediction error e and the error between the predicted RUL and actual RUL is $e = RUL_{\text{predict}} - RUL_{\text{actual}}$ then RMSE and Score formulas are given in Eqs. (3.1) and (3.2) separately:

$$RMSE = \sqrt{\frac{1}{N} \sum_{t=1}^N e_t^2} \quad (3.1)$$

$$Score = \sum_{t=1}^N s_t, s_t = \begin{cases} \exp^{-e_t/13} - 1 & e_t < 0 \\ \exp^{e_t/10} - 1 & e_t \geq 0 \end{cases} \quad (3.2)$$

where e_t is the prediction error, $\exp$ denotes Euler's number and N is the test sample size. Results of the proposed Siamese InceptionTime network are discussed in the following section.

4. EXPERIMENTS AND RESULTS

We conducted some experiments in the first dataset (FD001) of C-MAPSS to show the performance of the proposed method. In this section, firstly it has been demonstrated the performance of the proposed model on a small dataset and besides compared its results with other methods in the literature. Then, the performances of the proposed method and the baseline method [23] that one of the best models in the state-of-art were compared on the small datasets prepared. Finally, some changes were made to the proposed model to improve the results and the results of these changes were compared.

4.1. The Siamese InceptionTime Results

Firstly, we only used the data from 15 engines that were randomly selected from the FD001 train dataset to train the proposed method. Selected engines ids are [18, 35, 37, 40, 41, 42, 49, 52, 53, 61, 66, 68, 76, 81 and 100] and randomly selected. We obtained 30000 training data pairs of data from the selected engines with the balanced version of the dataset by weighting randomly selecting samples from each engine according to the total dataset size to train the proposed network. We use 80 % samples of the data as the training set, 20 % for the validation set of the training dataset in our experiments.

The trained proposed network was employed to estimate the RUL of the FD001 test dataset. For testing, the last sample (last time window) is paired with the first healthy sample for each engine respectively in the test dataset and pairs are given to the proposed network for prediction. For the FD001 database, the RMSE is 16.50 and the score is 460.77 (the baseline method [23] results are 11.96 and 229.00 respectively). The predicted RUL of each test engine in the FD001 sub-dataset is shown in Figure 4.1.

The green bar shows the predicted RUL value and the vertical solid red line denotes differences of RUL value between actual and predicted. With the proposed method generally, the estimated value of the RUL can be observed close to the true value.

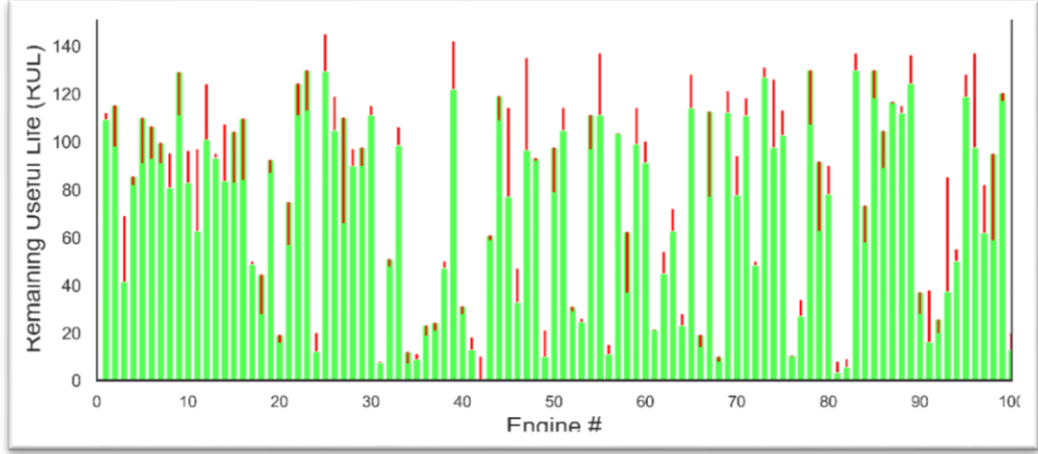


Figure 4.1: Predicted RUL for each engine in the FD001 test dataset.

Figure 4.2 demonstrates the predicted RUL and binary similarities generated by the proposed method on the two of the engines for the FD001 test dataset. The green line shows the binary similarity (the first healthy sample is compared sequentially with other samples of the engine) and gives some signals before the failure point indicated in dashed red via decreasing similarity score of the pair samples. Also, the RUL prediction results are shown in blue and successfully estimate predictions as close to actual values and give the alarm signal before the failure point by lowering the predicted RUL value.

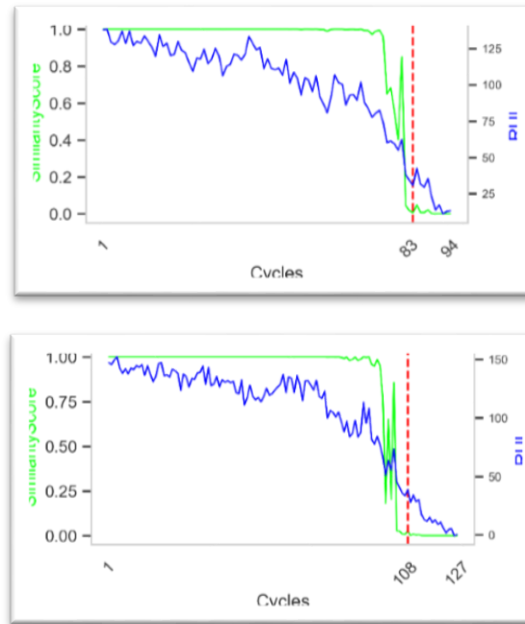


Figure 4.2: BINC and RUL outputs of engine #41 and #42 in FD001 test sub-dataset.

Classification results of the FD001 test dataset are illustrated in Figure 4.3 in a confusion matrix. The accuracy of our classifier is 95% while the F1 score is 96%. This shows the discriminant nature of the windows before and during failure. The Siamese network captures this very accurately.

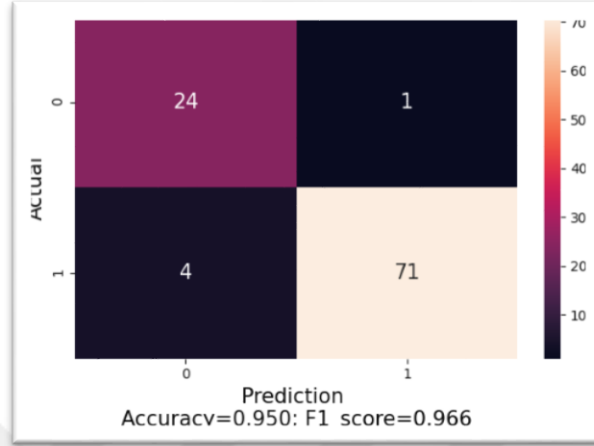


Figure 4.3: Classification results input of pairs healthy/failure samples of test data.

The comparison result is quantified using RMSE of RUL prediction for the FD001 test dataset as listed in Table 4.1. It is noted that the developed proposed model outperforms traditional machine learning approaches such as SVR and ensemble methods with other DL approaches such as MLP, DCNN, GRU, Bi-LSTM, RNN DLSTM sequential network types. Also, the results of the proposed method are close to GB and MODBNE. DAG LSTM-CNN gives better than the proposed result. However, it should not be forgotten that the proposed method uses only 15 engines data for training.

4.2. Experiments on Small Datasets

To evaluate compare the proposed method to a baseline system when a smaller amount of data is used, we have implemented the network described in [23]. For both proposed and compared models we obtained the best hyperparameters and run the model over one thousand epochs. We train these models with random hyperparameters and compared their results in RMSE and Score metrics.

Table 4.1: Engine RUL prediction comparison in RMSE on FD001 test dataset.

Methods	RMSE	Methods	RMSE
ARIMA – SVR [9]	39.68	GRU [22]	20.60
MLP [16]	37.56	Bi-LSTM [22]	19.94
SVR [14]	32.7	DLSTM [22]	19.53
MLP [22]	25.93	TCN [22]	18.74
Decision Tree [14]	25.2	DCNN [16]	18.44
Random Forest [14]	23.8	DSAE-TCN [22]	18.01
SVR [22]	23.75	RF [17]	17.91
CNN [22]	22.93	GB [17]	15.67
RNN [22]	22.59	MODBNE [17]	15.04
SVR [16]	20.96	LSTM-CNN [23]	11.96
XGBoost [14]	20.9	The proposed method (trained only 15 machinery)	16.50

First of all, we carried out the training process by using the data of a different number of engines from the training dataset of 25 engines that have at least one failure sample as a training procedure, and engine ids are [18, 20, 24, 31, 34, 35, 36, 37, 40, 41, 42, 49, 52, 53, 56, 61, 64, 66, 68, 76, 81, 82, 90, 92 and 100] as specified in Table 3.4. We simulate the amount of data (smaller to larger) by using a fixed number of engines in the training (i.e., 5 engines randomly chosen from the training set). We change the number of engines from 5 to 24 indicating an increasing amount of data. We expect that 5 engines for training will be challenging and as the number increases, the models will perform better. For a fixed number of engines, models are trained 30 times. We call each of these an experiment, and for each iteration, we select engines randomly and used the same engine's data for both models to compare performances on test data and recorded their results with both hyperparameters and parameters insensitive. For each experiment, we prepare 30000 data pairs from selected engines and train the proposed model for one thousand epochs with a simple early stopping

strategy. The selected engines data for the baseline model is trained as described in [23]. The training procedure was repeated 30 times for both models.

RMSE and Score results that were obtained from hyperparameters on test data for both models are illustrated in Figure 4.4 while parameters insensitive results are shown in Figure 4.5 with boxplots. Box plots result cut of values bigger than 50 for RMSE and 5000 for Score metrics for hyperparameter results while those values are 50 for RMSE and 15000 for Score metrics for parameters insensitive. The proposed model gives a reasonable performance compared to the baseline model when optimal hyperparameters are used. For parameters insensitive, the proposed model gives better results. In Figure 4.4, RMSE results were obtained with hyperparameters, 15 out of 20 experiments the proposed model gives better results, and for all experiments, the proposed model gives better results with parameters insensitive RMSE and Score metrics.

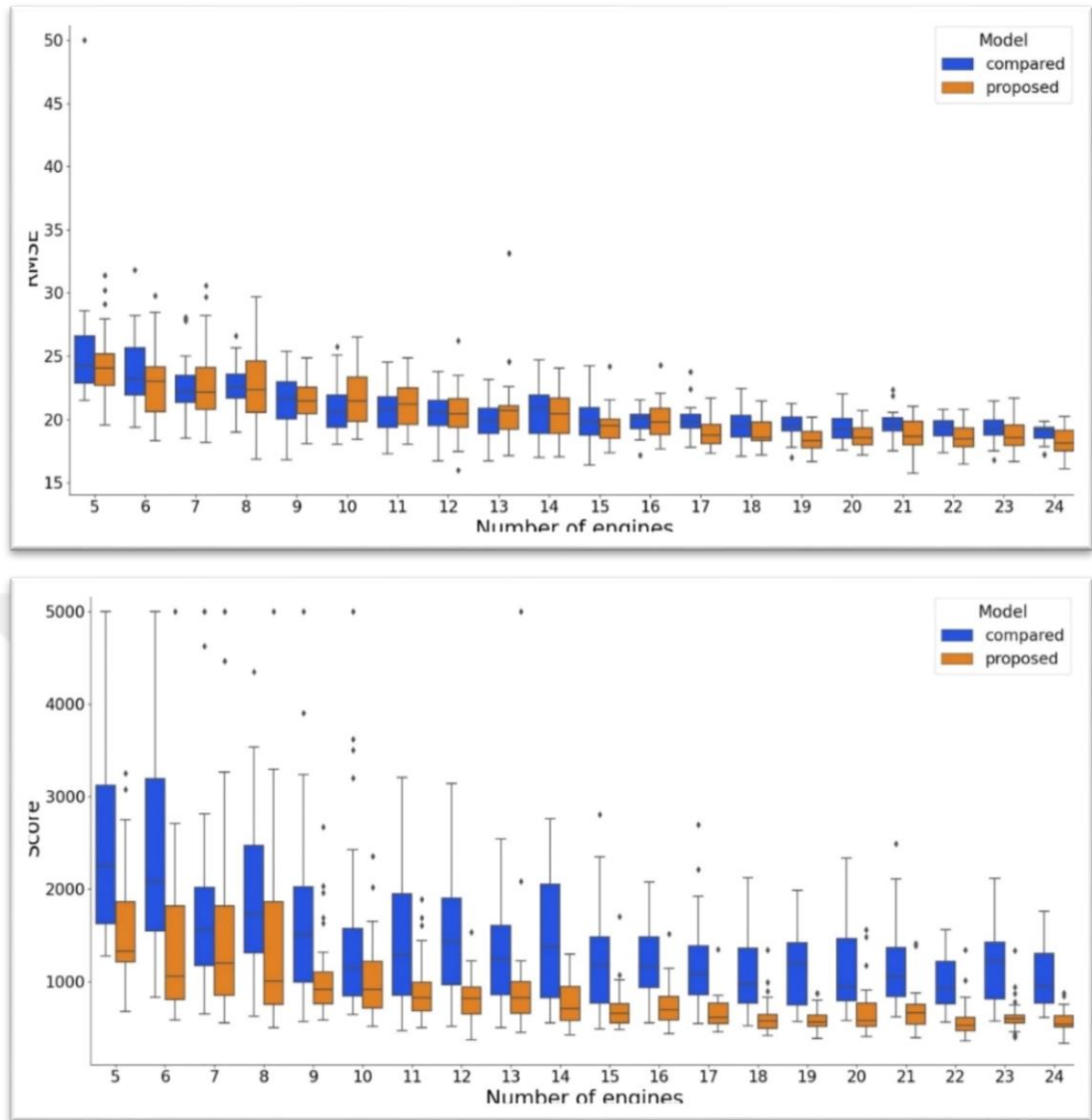


Figure 4.4: Hyperparameters RMSE (top) and Score (bottom) results. Boxplot results were obtained from 30 experiments for changing the number of engines.

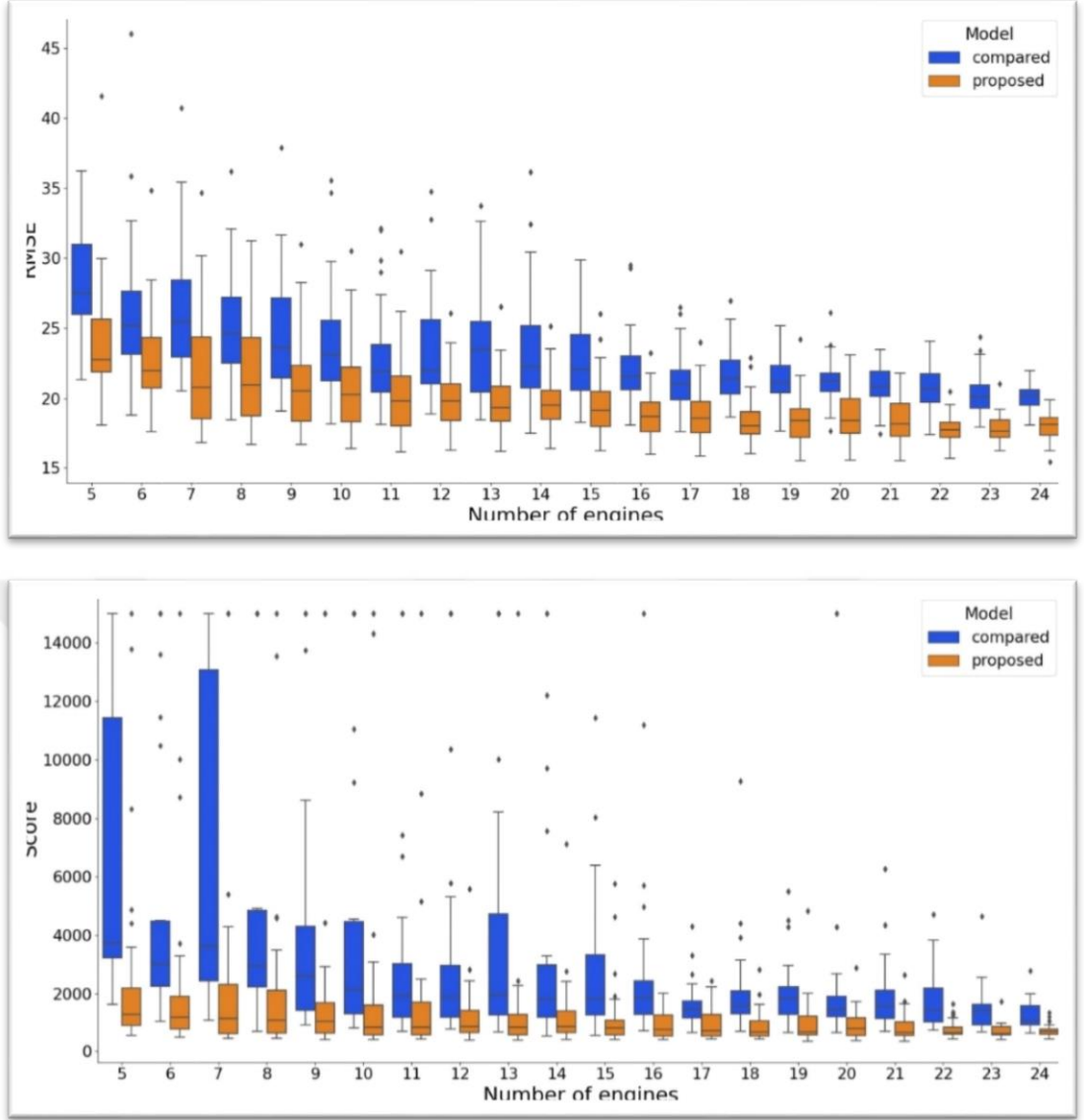


Figure 4.5: Parameters insensitive RMSE (top) and Score (bottom) results. Boxplot results were obtained from 30 experiments for changing the number of engines.

To see the effect of shortening the available sequence hence reducing the training data, we have trained the two models first with full data and then with the first 15% of the data from the beginning is removed. The RMSE and Score metrics show slight improvements as seen in Figure 4.6. The proposed method performs slightly better when the sequence is shortened. This confirms our expectation that the model performs well with fewer data.

The improvement is slightly more with the baseline algorithm with shorter sequences. This might seem a bit counter intuitive. Yet, this result can still be explained by the fact that we are using a linear model for machine degradation.

Ignoring the begging of the sequence removes any potential noise before the linear degradation starts.

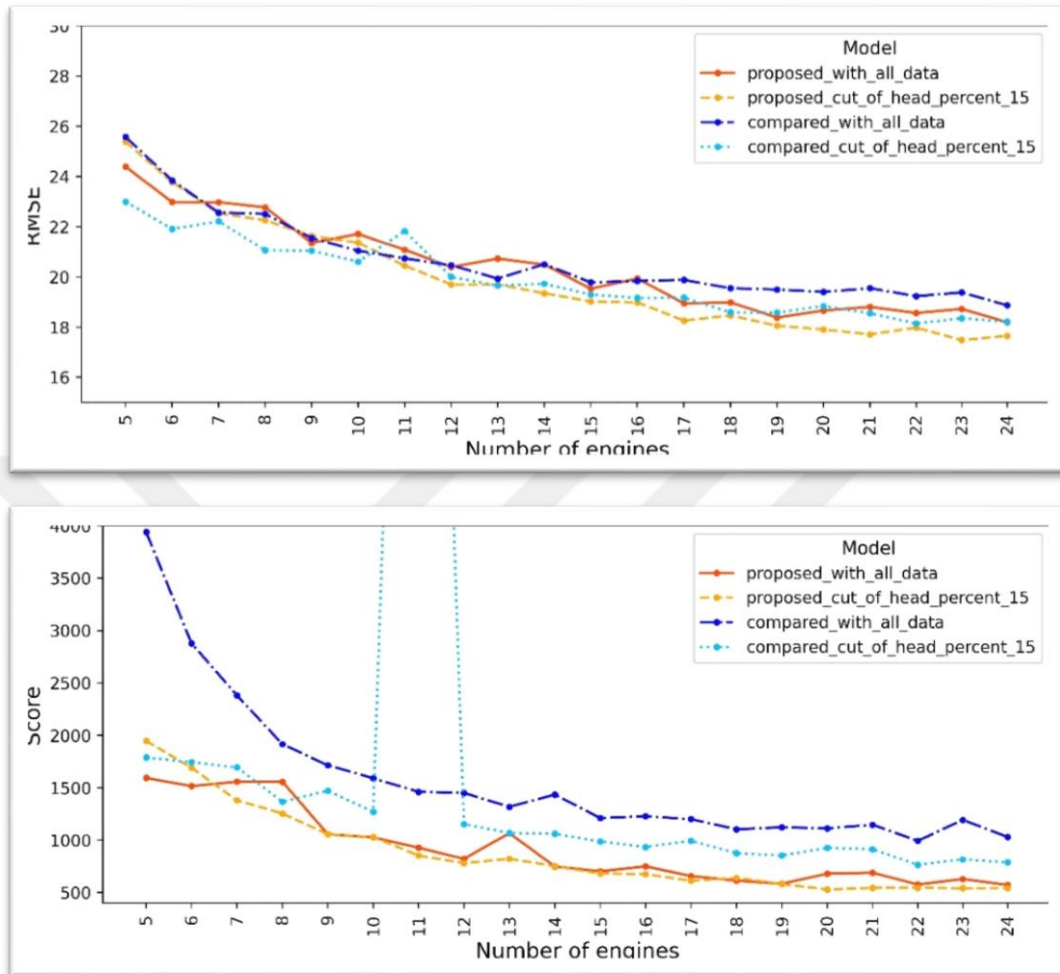


Figure 4.6: Performance (top – RMSE, bottom – Score) changes when the first 15% of each data is ignored during training. Mean of the metrics are given for changing the number of engines used in training. Ignoring data at the beginning of a sequence improves results slightly.

Specifically, as we see from the line plot here, training with the small size of engines data (approximately 10 and lower size engines), the proposed model cut off head 15 percent performance lower than trained with all data. By small size data obtained with the cut of head 15 percent of engines data up to approximately 10 engine size data, the model can't generalize as with trained with all data. But after some point (approximately bigger 10 engine size data), since the distribution of trained data can make more generalized on test data and the network push to learn to distinguish of the discriminative feature of failure samples and near failure samples, the network

performs better trained with all data and its results are better than the compared both trained with all data and the cut of head 15 percent of each engine data results. The same results were obtained for Score metric results.

4.3. Model Improvements

For sake of improvement on the performance of the proposed model, some changes were made to the model architecture as illustrated in Figure 4.7 and their results recorded.

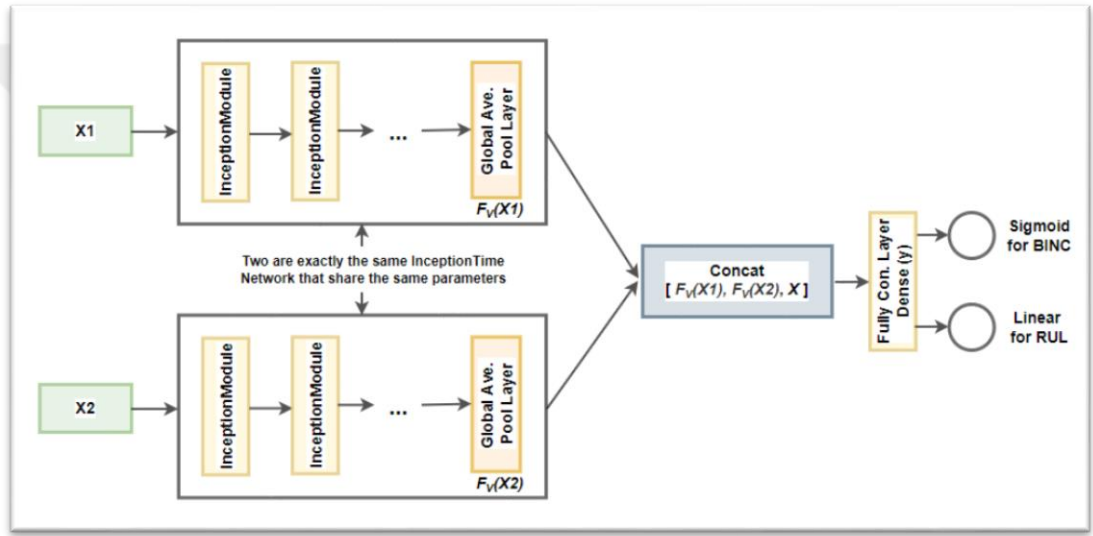


Figure 4.7: Changes in the proposed model architecture for improvement.

Instead of only taking the absolute difference of feature vector of x_1 and x_2 that is $((f_v(x_1) \text{ and } f_v(x_2)))$, it is also concatenated their features and also adds some other special forms of the feature vectors (illustrated as X in Figure 4.7 in Concat layer) and then dense layer unit sizes (illustrated as y in Figure 4.7 in Dense layer) changed according to the size of the concatenated output. Additionally, to optimize the model in the changes made for X and y , the number of inception modules, the number of filters used in inception modules, and the learning rate of the model were searched with hyperparameters. The range of depth of inception module in the interval $[2, 7]$ and it is searched for filter size in inception modules are $[32, 64, 96, \text{ and } 128]$ and learning rates in $[0.01, 0.001, 0.0001, \text{ and } 0.00001]$. Changes made on X and y and

their RMSE and Score results are specified in Table 4.2. For training, it is used 15 engines data as the same in section 4.1.

Table 4.2: Changes in model architecture results.

X	y	RMSE	Score
Base model	1 * 64	16.50	460.77
(only concatenated)	2 * 128	17.05	586.51
Absolute difference of FV(X1) and FV(X2)	3 * 96	17.43	545.14
Element wise multiplication of FV(X1) and FV(X2)	3 * 128	16.88	464.76
Sin2π(FV(X1)), Sin2π(FV(X2)), Cos2π(FV(X1)), Cos2π(FV(X2))	6 * 128	16.08	541.19
Depth Concat [Sin2 π (FV(X1)), Sin2 π (FV(X2)), Cos2 π (FV(X1)), Cos2 π (FV(X2))] + Conv1D (1 + 1(S))	6 * 128	17.79	631.11

From Table 4.2 as we see, the best RMSE obtained from using the positional encoder (Sin2 π (FV(X1)), Sin2 π (FV(X2)), Cos2 π (FV(X1)), Cos2 π (FV(X2))) and while the best Score obtained with element multiplication of the feature vectors.

Another work we have done to improve our model performance zeroed out or pruned and retrained the weights of our trained base model within certain threshold intervals on the same training data specified in Section 4.1. This operation called magnitude pruning which sparse the network and makes a regularization effect on the model. Such that works weights lowest magnitude are masked to zero and training continues from where it left off [31].

In our experiments, we try to some threshold values and record their model performance on test data. Results obtained from experiments are demonstrated in Table 4.3. Best results are taken from the threshold with 0.1 when trained with 50 epochs. While the model shows an increase in performance from a regularization effect as it comes towards a certain threshold it is observed that this performance decreases afterward.

Table 4.3: Model results with magnitude pruning.

Threshold	Epoch #	RMSE	Score
Base model	-	16.50	460.77
0.001	10	22.90	2012.55
	50	17.47	584.92
0.01	10	22.66	1837.12
	50	17.46	580.53
0.1	10	43.47	7371.29
	50	16.09	413.65
0.25	10	>>	>>
	50	>>	>>

5. CONCLUSION AND FUTURE WORK

Accurate estimation of RUL has important advantages in many industrial applications where safety efficiency and reliability are among primary concerns. In the absence of a lot of failure data in data-driven prognostics, we propose a Siamese InceptionTime network with temporal data processing capabilities for fault diagnosis classification and regression. InceptionTime network allows temporal processing. The network try to model the discriminant nature of the data from two opposite time window samples. Since typical deep learning models do not train well with small sets of training samples, the proposed method allows multiple uses of the data for learning a window-based similarity metric.

We conducted some experiments on the turbofan engine degradation dataset. We prepare pairs of healthy and failure samples as input of the network with determine a value to RUL as threshold. Pairs of samples fed into the network and conduct experiments. The experiments comparing the proposed method with a state-of-art baseline method show improved performance when less data is available. To improve the results of the proposed method, some techniques were applied such as different harmonics, mask pruning. The performance was increased via the mask pruning method.

The results of this thesis presented a potential solution to early-staged fault diagnosis appearance scenarios as well and enables the engineered systems to increase reliability, safety and reduce costs. The proposed network has shown its advantage in performance and is promising for industrial applications such as obtaining large-size data is difficult. While these results are promising, further architecture and hyperparameters optimizations are still needed as the reported performance is slightly lower than the best state-of-art methods. Further experimental validations will be done on the other C-MAPSS sub-datasets.

REFERENCES

- [1] Wang D., Tsui K., Miao Q., (2018), “Prognostics and Health Management: A Review of Vibration Based Bearing and Gear Health Indicators”, IEEE Access, 665–676.
- [2] Zhao R., Yan R., Chen Z., Mao K., Wang P., Gao R., (2019), “Deep learning and its applications to machine health monitoring”, Mechanical Systems and Signal Processing, 115(1), 213–237.
- [3] Si X., Wang W., Hu C., Zhou D., (2011), “Remaining useful life estimation - A review on the statistical data driven approaches”, Eur. J. Oper. Res. 213(1), 1–14.
- [4] Eker Ö.F., Camci F., Jennions I.K., (2012), “Major challenges in prognostics: study on benchmarking prognostic datasets”, Prognostics and Health Management Society European Conference, 8, July Dresden/German.
- [5] Gebraeel N., Elwany A., and Pan J., (2009), “Residual Life Predictions in the Absence of Prior Degradation Knowledge”, IEEE Transactions on Reliability 58(1), 106–117.
- [6] Eker Ö. F., Camci F., Guclu A., Yilboga H., Şevkli M., Baskan S., (2011), “A Simple State-Based Prognostic Model for Railway Turnout Systems”, IEEE Transactions on Industrial Electronics 58(5), 1718–1726.
- [7] Bengio Y., LeCun Y., Hinton G. E., (2021), “Deep learning for AI”, Communications of the ACM, 64(7), 58–65.
- [8] Koch G., Zemel R., Salakhutdinov R., (2015), “Siamese Neural Networks for One-shot Image Recognition”, ICML deep learning workshop, 2-8.
- [9] Ordóñez C., Lasheras F. S., Roca-Pardiñas J., Juez F. J. C., (2019), “A hybrid ARIMA-SVM model for the study of the remaining useful life of aircraft engines”, Comput. Appl. Math. 346, 184–191.
- [10] Kelif R., Chebel-Morello B., Malinowski S., Laajili E., Fnaiech F., Zerhouni N., (2017), “Direct Remaining Useful Life Estimation Based on Support Vector Regression”, IEEE Transactions on Industrial Electronics 64(3), 2276–2285.
- [11] Durbhaka G. K., Selvaraj B., (2016), “Predictive maintenance for wind turbine diagnostics using vibration signal analysis based on collaborative recommendation approach”, 2016 International Conference on Advances in Computing, Communications and Informatics (ICACCI), 1839–1842.
- [12] Behera S., Choubey A., Kanani C. S., Patel Y. S., Misra R., Sillitti A., (2019), “Ensemble trees learning based improved predictive maintenance using IIoT for turbofan engines”, Proceedings of the 34th ACM/SIGAPP Symposium on Applied Computing, 842-850, April.

- [13]Ayvaz S., Alpay K., (2021), “Predictive Maintenance System for Production Lines in Manufacturing: A Machine Learning Approach Using IoT Data in Real-Time”, Expert Systems with Applications. 173, 114598.
- [14]Jia Z., Xiao Z., Shi Y., (2021), “Remaining Useful Life Prediction of Equipment Based on XGBoost”, 5th International Conference on Computer Science and Application Engineering (CSAE 2021). Association for Computing Machinery, New York/USA, Article 59, 1–6.
- [15]Xu J., Wang Y., Xu L., (2014), “PHM Oriented Integrated Fusion Prognostics for Aircraft Engines Based on Sensor Data”, IEEE Sensors Journal 14(4), 1124–1132.
- [16] Babu G. S., Zhao P., Li X-L., (2016), “Deep Convolutional Neural Network-based Regression Approach for Estimation of Remaining Useful Life”, Proc. International Conference on Database Systems for Advanced Applications, 9642, 214–28.
- [17]Zhang C., Lim P., Qin AK., Tan KC., (2016), “Multi-objective Deep Belief Networks Ensemble for Remaining Useful Life Estimation in Prognostics”. IEEE Transactions on Neural Network and Learning System, 28(10), 1–13.
- [18] Li X., Ding Q., Sun J., (2018), “Remaining useful life estimation in prognostics using deep convolution neural networks”. In: Reliab. Eng. Syst. Saf. 172, 1–11.
- [19]Zhang J., Peng W., Yan R., Gao R. X., (2018), “Long short-term memory for machine remaining life prediction”. In: Journal of Manufacturing Systems 48, 78–86.
- [20]Hinchi A. Z., Tkiouat M., (2018), “Rolling element bearing remaining useful life estimation based on a convolutional long-short-term memory network”. In: Procedia Computer Science 127, 123–132.
- [21] Zhang A. Wang H. Shaobo L., Cui Y., Liu Z., Yang G., Hu J., (2018), “Transfer Learning with Deep Recurrent Neural Networks for Remaining Useful Life Estimation”. In: Applied Sciences 8(12), 2416-2425.
- [22] Zhang Y., Liu X., (2021), “Remaining Useful Life Prediction for Turbofan Engine using SAE-TCN Model”, 40th Chinese Control Conference (CCC), 8280-8285, Shanghai, China, 26-29 July.
- [23] Li J., Li X., He D., (2019), “A Directed Acyclic Graph Network Combined with CNN and LSTM for Remaining Useful Life Prediction”, IEEE Access, 7(1), 75464–75475.
- [24]Hsiao S. C., (2019), “Malware Image Classification Using One-Shot Learning with Siamese Networks”, Proc. Knowledge-Based and Intelligent Information & Engineering Systems (KES2019). 159, 1863-1871.

- [25] Xing X., Guo W., Wan X., (2021), “An Improved Multidimensional Distance Siamese Network for Bearing Fault Diagnosis with Few Labelled Data”, Global Reliability and Prognostics and Health Management (PHM-Nanjing), 1-6, Nanjing, China, 15-17 Oct.
- [26] Xu J., Shi Y., Yuan X., Lu S., (2021), “Cross-Category Mechanical Fault Diagnosis Based on Deep Few-Shot Learning”, IEEE Sensors Journal, 21(24), 27698-27709.
- [27] Saxena A., (2008), “Damage Propagation Modeling for Aircraft Engine Run-to-failure Simulation”, Proc. International Conference on Prognostics and Health Management, 1–9, Denver, CO, USA, 6-9 Oct.
- [28] Fawaz I., Lucas H., Forestier B., (2020), “InceptionTime: Finding AlexNet for Time Series Classification”, Data Mining and Knowledge Discovery, 34(6), 1936–1962.
- [29] Szegedy C., Liu W., Jia Y., Sermanet P., Reed S., Anguelov D., Erhan D., Vanhoucke V., Rabinovich A., (2015), “Going Deeper with Convolutions”. Proc. IEEE Conference on Computer Vision and Pattern Recognition (CVPR), 1–9.
- [30] Zhang C., Sun J. H., Tan K., (2015), “Deep Belief Networks Ensemble with Multi-objective Optimization for Failure Diagnosis”, IEEE International Conference on Systems, Man, and Cybernetics, 32–37, Hong Kong, China, 9-12 Oct.
- [31] Movva R., Frankle J., Carbin M., (2021), “Studying the Consistency and Composability of Lottery Ticket Pruning Masks”.

BIOGRAPHY

Uğur Ceylan took bachelor's degree from the Computer Engineering Department of Marmara University in June 2017. Since the following semester, he has been a master student at the Computer Engineering Department of Gebze Technical University. In addition to his career, he has been working on an AI project based on time-series data analysis, computer vision, and deep learning as a data scientist at Doruk Automation and Software Inc. for approximately four years.



APPENDICES

Appendix A: The Publications about the Thesis

Ceylan U., Genc Y., (2021), "Siamese Inception Time Network for Remaining Useful Life Estimation", J. Artif. Intell. Data Sci. 1(2), (2021), 165-175.

Ceylan U., Genc Y., (2021), "Siamese Networks for Predicting Remaining Useful Life of Machinery", 1st International Congress on Artificial Intelligence and Data Science (ICADA), İzmir, Turkey, 26-28 Nov.

