ASME Journal of Computing and Information Science in Engineering
Online journal at:
https://asmedigitalcollection.asme.org/computingengineering

Determination of Multi-Component Failure in Automotive System Using Deep Learning

John O'Donnell
Department of Mechanical Engineering,
College of Engineering,
The University of Alabama,
P. O. Box 870276,
Tuscaloosa, AL 35487-0276
e-mail: jlodonnell@crimson.ua.edu

Hwan-Sik Yoon[1]
Department of Mechanical Engineering,
College of Engineering,
The University of Alabama,
P. O. Box 870276,
Tuscaloosa, AL 35487-0276
e-mail: hyoon@eng.ua.edu

The connectivity of modern vehicles allows for the monitoring and analysis of a large amount of sensor data from vehicles during their normal operations. In recent years, there has been a growing interest in utilizing this data for the purposes of predictive maintenance. In this paper, a multi-label transfer learning approach is proposed using 14 different pretrained convolutional neural networks retrained with engine simulation data to predict the failure conditions of a selected set of engine components. The retrained classifier networks are designed such that concurrent failure modes of an exhaust gas recirculation, compressor, intercooler, and fuel injectors of a four-cylinder diesel engine can be identified. Time-series simulation data of various failure conditions, which include performance degradation, are generated to retrain the classifier networks to predict which components are failing at any given time. The test results of the retrained classifier networks show that the overall classification performance is good, with the normalized value of mean average precision varying from 0.6 to 0.65 for most of the retrained networks. To the best of the authors' knowledge, this work represents the first attempt to characterize such time-series data utilizing a multi-label deep learning approach. [DOI: 10.1115/1.4063003]

Keywords: predictive maintenance, machine failure prognostics, multi-component failure detection, transfer learning, machine learning for engineering applications

Introduction and Background

The introduction of high-bandwidth communication systems for vehicle-to-vehicle and vehicle-to-infrastructure connections is enabling new technologies and services including optimal traffic control, car-sharing service, and connected and automated vehicles [1–5]. An area of interest that has not been explored much, however, is utilizing the data collected from these vehicles for predictive diagnostics or prognostics of the vehicle components [6]. Being able to predict component failure of a vehicle would be of great benefit as it could allow for reduction of maintenance time and effort, lower operation costs, and improved user safety [7].

The standard approach for handling component failure in automotive systems has been to employ a preventive maintenance, or scheduled maintenance program. Utilizing statistical data, preventative maintenance can be implemented such that critical components are replaced before failures typically occur [7]. However, due to the variation in each individual vehicle, it is difficult to apply this approach to a fleet of vehicles, nor is it economically viable to replace every component on a staggered schedule. For such cases, reactive maintenance can be employed [7]. In reactive maintenance, the failure of vehicle components is detected using an onboard diagnostic system [8]. For example, electronic control units (ECU) utilize heuristic and model-based methods to determine what constitutes a failure [9]. Then, techniques such as failure mode and effects analysis are applied utilizing sensor data and analytical models to determine potential failure modes of the components. The resulting failure modes and their related parameters are used in rule-based algorithms in onboard diagnostic systems [10,11]. This is a general approach without utilizing any particular data unique to a specific vehicle to generate appropriate predictions.

Although preventive and reactive maintenance approaches are currently prevalent in the automotive industry, predictive maintenance (PdM) is anticipated to offer considerable advantages, including reduced machine downtime and maintenance costs [12]. For this reason, PdM has attracted significant interest from researchers in the field of automotive systems. However, PdM has not seen any real application on the individual vehicle scale due to the difficulties associated with data collection [12,13]. Obtaining the necessary data for automotive systems requires the following at minimum: numerous sensors to measure physical signals, a means of transmitting and storing this data, expert knowledge of the system, and sufficient vehicle operation time to cover the progression of a failure mode from beginning to end [7]. Obtaining this data is further complicated by the fact that the lifecycle of any set of vehicles can involve dissimilar drive cycles and environments, which can have a significant effect on the progression of failure. For example, it has been shown that adding geographical data can provide more accurate maintenance scheduling for fleet management [14]. At the component level, active failure detection using machine learning

[1] Corresponding author.

Manuscript received May 9, 2022; final manuscript received June 23, 2023; published online August 14, 2023. Assoc. Editor: Bin He.

methods has been applied in some studies [6,7]. Among various automotive components, batteries have been the primary focus for hybrid-electric and fully electric vehicles [15–17]. Failures of other mechanical components such as bearings, brakes, gearboxes, and suspension systems have also been investigated. However, the aforementioned data limitations remain unresolved, and these limitations have led to a reliance on simplified experiments or models as a workaround [18–21]. Due to these challenges, the implementation of predictive maintenance for individual vehicles has not been extensively researched, making it the primary focus of this paper.

There have been many different data-driven approaches for component diagnostics and prognostics to date. These include physics-based models, knowledge-based approaches, statistical methods, machine learning approaches, and deep learning approaches [22–28]. However, most of the approaches found in the literature involve either binary or multi-class problems focused on isolated failure modes. Such binary and multi-class problems operate under the assumption of mutual exclusion, which restricts their ability to classify concurrently occurring component failures effectively. The research problems that do include multiple failure modes have treated them as a multi-class problem, which is not a viable approach as the number of components, failure modes of each component, and combinations of the two increase to the full scale on a real-world vehicle. In this paper, it is hypothesized that a multi-label approach has the potential to resolve this issue. While the multi-class classification picks only one output class requiring 2^n different output nodes for n different failing components, the multi-label classification will generate multiple output classes using just n different output nodes. This is because, in a multi-label approach, a single data case can correspond to multiple labels representing different failed components. However, a general approach of applying multi-label methods to time-series data, particularly automotive data, has not been studied to date. In addition, the approach of reshaping such time-series data into images for use in convolutional neural networks is lacking in the literature. Therefore, this paper demonstrates the feasibility of such a general approach by evaluating the performance of various pretrained learning models using automotive system data. This serves as an example for potential diverse applications in future work.

Methodology for Predicting Multi-Component Failure

The goal of this paper is to propose a new approach to determine the binary failure states of multiple components in an automotive system by employing transfer learning with time-series data. To accomplish this goal, an automotive engine system with at least one existing component failure is considered in this paper. To obtain the necessary system data, a high-fidelity diesel engine simulation model employing an exhaust gas recirculation (EGR) system and a variable-geometry turbocharger was developed in commercial software, GT-SUITE, as shown in Fig. 1. Within this model, four subsystems are assumed to have a single potential failure mode: the EGR, the compressor (Comp), the injectors (Inj1, Inj2, Inj3, Inj4), and the intercooler (Inter). The EGR is assumed to fail due to accumulation of soot along the EGR inner wall. For the modeling purpose, this soot accumulation is assumed to be of uniform thickness along the EGR pipe and remains constant throughout each

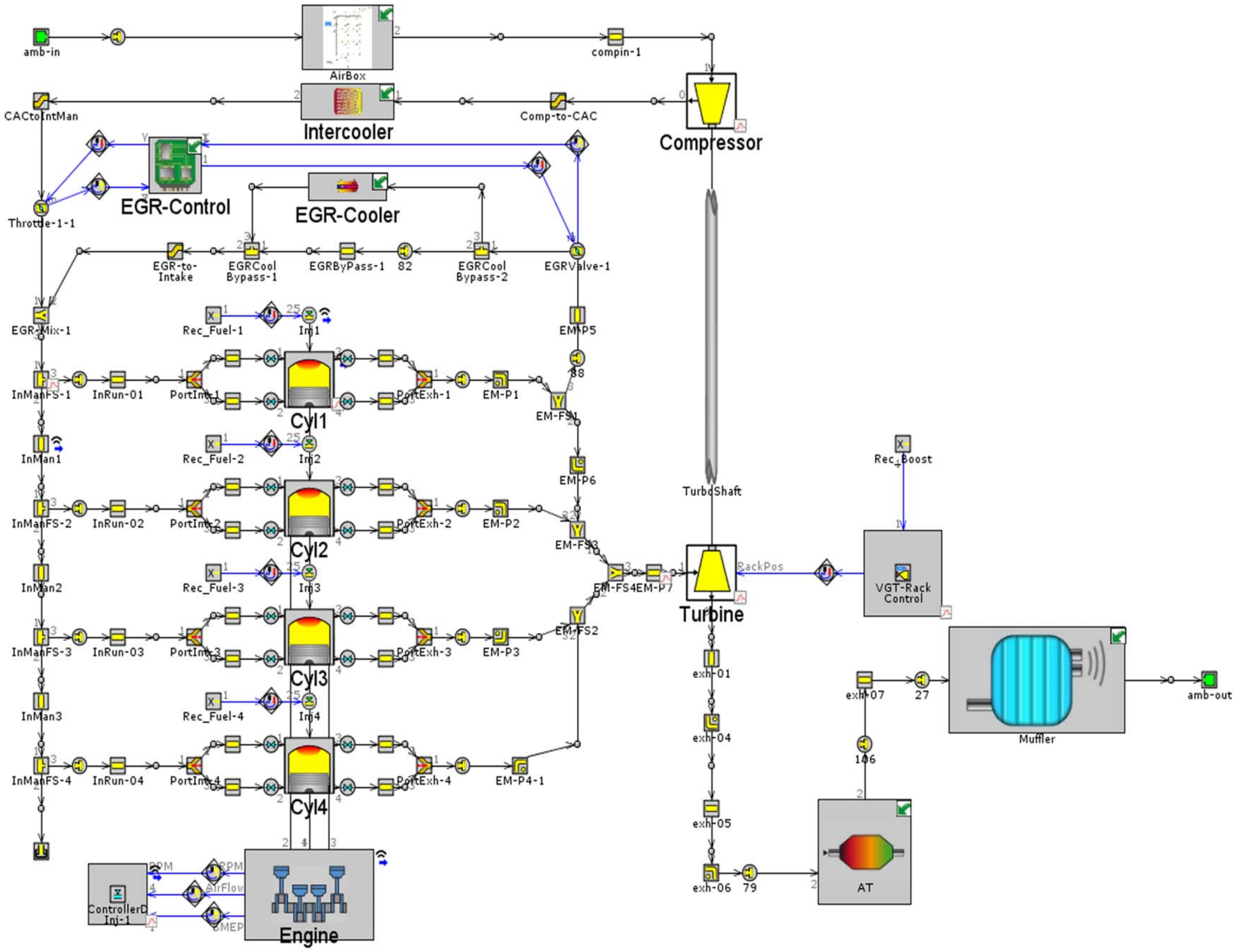


Fig. 1 GT-SUITE engine simulation model for generating time-series data

simulation. The compressor is assumed to fail due to progressive buildup of obstructions in the intake filter, represented as a decrease in the intake pipe diameter. The intercooler is assumed to fail due to the accumulation of debris and other material, represented as an obstruction on the intake side of the intercooler channel. Finally, the injectors are assumed to fail over time due to clogging from impurities in the fuel and exhaust gas particulates, represented by a uniform restriction in the diameter of the fuel injector output channels. Each individual injector is considered to have its own unique failure mode, meaning that there are seven total individual failure modes considered in this study.

In this study, the steady-state behavior of the system is of primary interest since it is regarded that steady-state behavior would provide the best means of isolating the effect of each failure mode. This means that the possible effects of transient behavior during drive-cycle simulation need to be minimized. To accomplish this, the engine simulation model is gradually brought to a target speed over 2 s and then held constant thereafter. Each specific steady-state engine operation with a combination of failure modes is then simulated for a minimum of 15 s, with a step size of 0.01 s, until steady-state is reached or a maximum simulation time of 2 min has passed, whichever occurs first. In the case that steady-state cannot be reached in this time frame, it is assumed that it is not possible to reach a true steady-state. Ten representative cases for steady-state engine operation, shown in Table 1, are chosen to cover the entire operational range of the engine. For the simulation study, the EGR failure due to the internal soot accumulation is modeled as decreasing diameter of the EGR pipe from valve to outlet as described in Table 2. To prevent the simulated gas flow from reaching supersonic speed, the EGR pipe clogging is limited to a maximum of 80%, which is regarded as the component failure point.

To include the thermal effect of accumulated soot in the simulation, the overall thermal conductivity in the pipe is modeled based on a pseudo-composite "material" consisting of deposited soot and stainless-steel pipe. Utilizing a lumped parameter modeling approach, the overall thermal conductivity is calculated as the weighted mean of the stainless steel's thermal conductivity and the soot's thermal conductivity, which is assumed to be 1 W/mK [29]. The density of the composite material is calculated by dividing the total mass by the combined volume of the materials. Utilizing these two material effects, the engine model is run for each case shown in Table 2 for the ten operational conditions specified in Table 1. Similarly, the diameter and respective failure percentage of the compressor, intercooler, and injector are shown in Tables 3–5, respectively. The thermal characteristics of these components are ignored as the failure modes involve only clogging obstructions at specific points in the channel.

Table 1 Ten cases chosen to represent the operational range of the simulated engine

Case	Speed (rpm)	BMEP (bar)
1	4500	14
2	4000	16
3	3000	16
4	2000	16
5	1500	13
6	1000	9.5
7	3000	3
8	2000	2
9	1500	1
10	800	0

Table 2 Diameter of EGR inner pipes relative to clogging percentage

Diameter (mm)	20	18	16	14	12	10	8	6	4
Percent clogged	0	10	20	30	40	50	60	70	80

Table 3 Diameter of compressor intake relative to health condition

Diameter (mm)	60	50	40	30	20	10
Health condition (%)	100	80	60	40	20	0

Table 4 Diameter of intercooler intake relative to health condition

Diameter (mm)	8	7	6	5	4	3	2
Health condition (%)	100	83.6	66.7	50	33.4	16.7	0

Table 5 Diameter of single injector output relative to health condition

Diameter (mm)	0.18	0.17	0.16	0.15	0.14	0.13	0.12
Health condition (%)	100	83.4	66.7	50.1	33.4	16.7	0

Utilizing the failure conditions shown in Tables 2–5, various combinations of multiple component failures can be simulated. Each combination of failing components, with varying health conditions (HC), simulated under each specific operational conditions specified in Table 1 represents a specific simulation. These simulations produce time-series data of 672 unique signals representing sensor output and other information from the simulation model. For example, these signals include the engine speed, engine torque, spark timing, and EGR valve opening, which can be obtained from the ECU and other control units in a vehicle. As previously mentioned, the cases where no component failure occurred are not included in the data as the no-failure case represents a mutually-exclusive case.

As the goal of this study is to predict the failure modes, independent of time scale, a convolutional neural network (CNN) is chosen to classify the aforementioned time-series data. To utilize this data in a CNN, a process has been developed to transform each simulation dataset into an image. While transforming time-series data into images has been reported in the literature, the approach presented here deviates from conventional approaches [30,31]. To accomplish this, the global maximum and minimum values of each signal are obtained and utilized to normalize each respective time-series data signal to the range of 0–255. Then, a sliding window average is used to compress each time-series signal to 224 instances in time regardless of the overall signal length in time. From the 672 normalized and compressed time-series data signals, 224 signals are randomly selected to form a gray-scale image of 224 × 224 pixels. Then, an additional 224 signals are randomly selected to form the second gray-scale image, and the remaining 224 signals are used to form the third gray-scale image. In this manner, three gray-scale images of square size are formed and they are designated as red, green, and blue (RGB) color images. This approach is adopted to harness all the information encapsulated in the 672 signals, and present it in a format typically recognized as input by CNNs. With the three images combined, each simulation data set can be completely represented by an RGB image of 224 × 224 pixels as shown in Fig. 2. The image data generated in this manner contain diverse time scales from 15 s to 120 s depending on how quickly steady-state is reached in each simulation. Since CNNs can extract hidden characteristics in the data regardless of the time scale that the image represents, the data compression does not deteriorate CNNs' classification capability.

A drawback of such a method, however, is that it is challenging to obtain sufficient amount of data for the training of machine

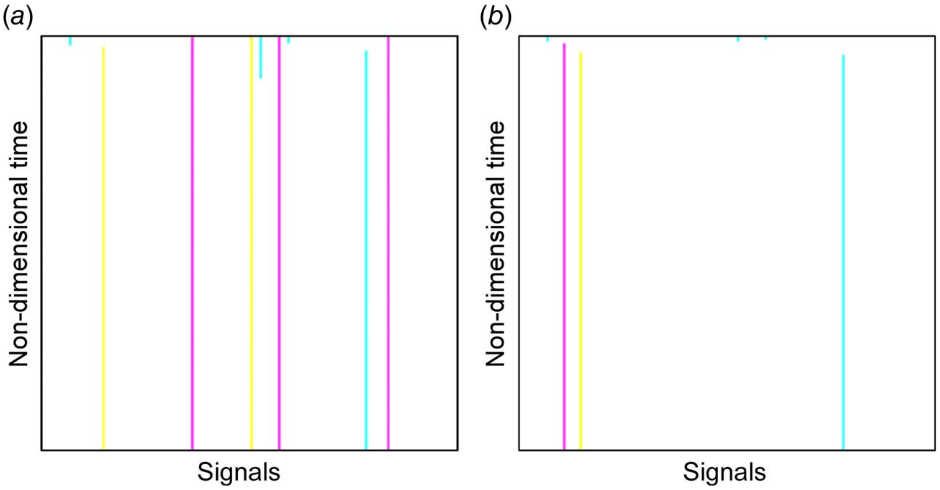


Fig. 2 Example images for time-series data with (*a*) compressor failure and (*b*) injector failure

learning-based prediction models. The deterministic simulation model shown in Fig. 1 does not provide sufficient randomness between simulations, and traditional image transformation techniques such as rotation are not applicable as they inherently mischaracterize the dataset. To mitigate the issue of the limited and homogeneous data size, white Gaussian noise is added to each data signal with a signal-to-noise ratio randomly selected between 40 dB and 60 dB. This is done ten times for each original image, increasing the total number of images from 1660 to 16,600.

One characteristic of this RGB image formation method, as shown in Fig. 2, is the lack of large variations in the data. Most parts of the image are white due to the combination of signals near a steady-state maximum value. As a consequence, the areas of meaningful information on a single image are relatively small and narrow, making it difficult to develop a prediction model with the optimal hyperparameters and well-tuned model parameters using the image data only. Therefore, a multi-label transfer learning approach utilizing pretrained CNNs is applied to develop failure modes classification models in this research. Since this data/image type represents a unique input for publicly available CNNs pretrained on photographs of real-life situations, a comparative analysis to determine how such data would perform is conducted by retraining the following 14 CNNs with the data: AlexNet (Alex), VGG, ResNet (Res), SqueezeNet (Squeeze), DenseNet (Dense), InceptionNet V3 (Incept), GoogLeNet (GoogLe), ShuffleNet (Shuffle), MobileNet (Mobile), ResNeXt (NeXt), Wide ResNet (Wide), MNASNet (MNAS), EfficientNet (Efficient), and RegNet (Reg) [32–45]. For each network, the first half of the layers have their weights frozen, the final half of the layers are unfrozen, and the classification output layer is adjusted to fit the new output space. If a network cannot be divided evenly, the extra layers are frozen. Then, each pretrained network is retrained over 30 epochs with the image data set from the engine model with a distribution of 70% for training (11,620), 20% for validation (3320), and 10% for testing (1660). This process is repeated five times with different validation and training sets each time, with the results averaged to account for overfitting. To center the pixel values in the images, and reduce the possible range of the network features, the pixel values in the images are normalized by subtracting the image mean and dividing by the standard deviation obtained from the ImageNet dataset. The mean and standard deviation of the RGB images are [0.485, 0.456, 0.406] and [0.229, 0.224, 0.225], respectively. The batch size for each network is chosen to be 32, while the learning rate is selected as 1×10^{-4}. An ADAM optimizer is employed for training each network.

Due to the nature of multi-label classification, various complications exist in data preparation including inter-class imbalances due to label co-occurrence and the positive-negative label imbalance [46,47]. As shown in Fig. 3, there are significant imbalances in the occurrence of labels. However, unlike multi-class imbalance, labels cannot be guaranteed to be balanced during preprocessing without causing imbalance in the classes due to the uneven representation of labels within the classes themselves, as shown by the 20 representative multi-mode failure classes in Table 6. For example, while Class 1 has one label representing the Intercooler failure mode, Class 17 also shares this label with the addition of the Compressor label. As a result, upsampling for balancing labels will likely lead to greater skew in the underlying classes within the data. Similarly, common techniques such as SMOTE to upsample minority classes, can lead to greater label imbalance due to the uneven number of labels contained within individual classes [48]. This is further complicated within this data as two additional class structures, the 166 health condition cases and the ten times operational range multiplier, are also hidden within the

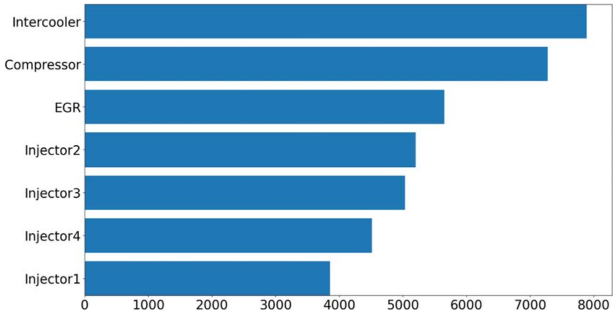


Fig. 3 Label distribution within training and validation sets

Table 6 Multi-mode failure classes within the dataset with assigned class ID

Class	ID	Class	ID
Inter	1	EGR	11
Inj4	2	EGR, Inj3	12
Inj3	3	EGR, Inj2, Inj3	13
Inj2	4	EGR, Inj1, Inj4	14
Inj2, Inj4	5	EGR, Inj1, Inj2, Inj3	15
Inj2, Inj3, Inter	6	Comp	16
Inj2, Inj3, Inj4	7	Comp, Inter	17
Inj1	8	Comp, Inj1, Inj2, Inj3, Inj4, Inter	18
Inj1, Inj2, Inj3, Inj4	9	Comp, EGR	19
Inj1, Inj2, Inj3, Inj4, Inter	10	Comp, EGR, Inter	20

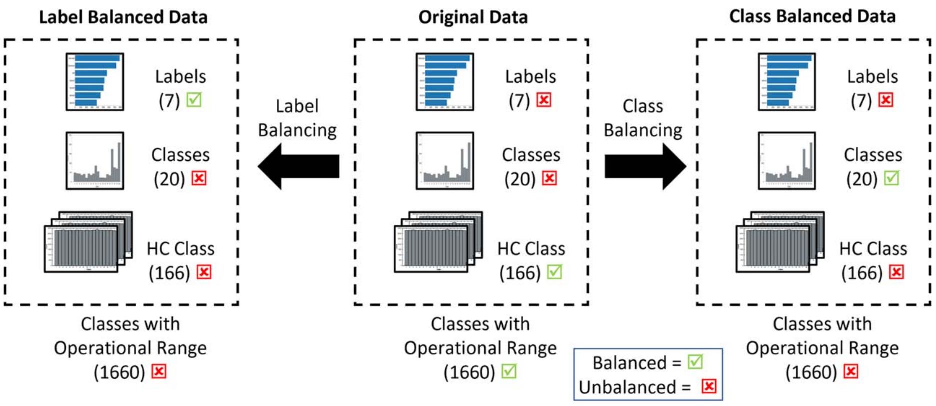


Fig. 4 Effect of class and label balancing on overall data balance

data as shown in Fig. 4. As the health condition classes and operational range condition are balanced within the original data, attempting to balance either labels or failure mode classes will cause all other conditions to become imbalanced, as demonstrated in Fig. 4.

For these reasons, it is a common practice to manage label imbalance within the loss function during training, such as with weighting or similar processes. If the focus of training is on the label-level utilizing such a process, the imbalance in the classes can be ignored if there is sufficient class representation within the data to allow an approximately even split between the training, validation, and test data. As shown in Figs. 5–7, this distribution of combinations of labels is relatively consistent within the three sets for training, validation, and test. Therefore, this inter-class imbalance is ignored for this study and label-based weighting is applied. However, weighting the loss function only is considered insufficient to account for the label imbalance given the dependency of the labels and the class distribution. This is demonstrated by the positive-negative label imbalance, as the data are biased toward negative labels (62.3% of labels are 0 representing the healthy cases). Therefore, the optimized asymmetric loss function developed by Ben-Baruch et al. is also utilized to reduce the effect of imbalance during training [47]. This loss function utilizes the ratio of positive and negative labels within an image to dynamically down-weight trivial classifications to improve the probability of learning minority label cases.

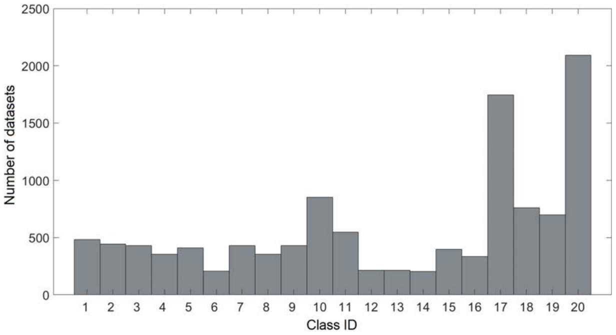


Fig. 5 Class distribution of multiple failure modes within training set

Multi-Component Failure Prediction Results

For the evaluation of the performance of different CNNs, the following metrics are used: the precision-recall area under the curve (PR AUC), the receiver operator curve area under the curve (ROC AUC), the mean average precision (mAP), the exact match ratio (EMR), the Hamming Loss, and the F1-Score [49–52]. The PR AUC and ROC AUC are determined from numerical integration of their respective curves via a trapezoidal method, while the mAP is defined as the average of the PR AUC of each individual label for a particular CNN. The EMR, Hamming Loss, and F1-Score are presented as well for the respective precision and recall values to describe the nuances in behavior between the different CNN models.

The primary metric used for describing classification performance in this research is the PR AUC, which is commonly referred to as average precision (AP). However, as there is imbalance in the labels, it is necessary to normalize the PR AUC for each label to compare the relative performance of training. This is accomplished by determining the unachievable region of each PR curve based on the proportion of positive labels within the label set, as shown in Table 7 [53]. From this table, considering that a perfectly balanced label has a value of 0.5, it is clear that the injectors have higher average skew than the other components, meaning the minimum

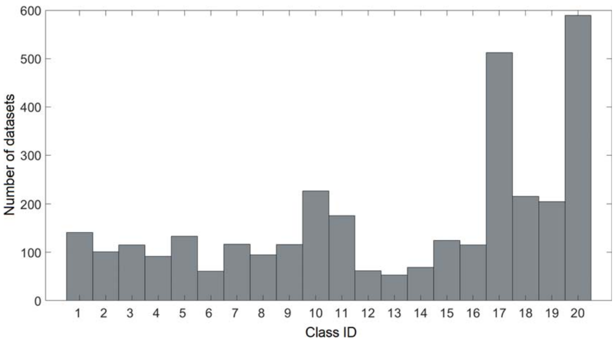


Fig. 6 Class distribution of multiple failure modes within validation set

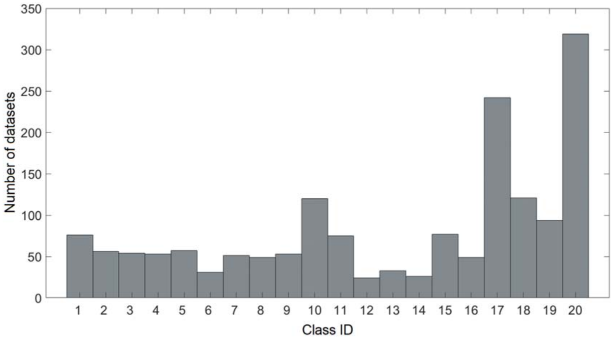


Fig. 7 Class distribution of multiple failure modes within test set

Table 7 Minimum PR AUC for given component and skew term for training data

	Comp	EGR	Inj1	Inj2	Inj3	Inj4	Inter
Skew term	0.496	0.367	0.246	0.351	0.337	0.307	0.534
Minimum PR AUC	0.304	0.211	0.135	0.201	0.192	0.172	0.334

Table 8 Normalized PR AUC for different components using different CNNs

Net	Comp	EGR	Inj1	Inj2	Inj3	Inj4	Inter
Alex	0.727	0.643	0.562	0.593	0.579	0.567	0.754
Dense	0.742	0.632	0.577	0.610	0.582	0.580	0.733
Efficient	0.750	0.701	0.562	0.591	0.565	0.576	0.783
GoogLe	0.747	0.708	0.563	0.577	0.604	0.573	0.770
Incept	0.526	0.553	0.382	0.426	0.404	0.403	0.541
MNAS	0.262	0.296	0.329	0.303	0.275	0.347	0.398
Mobile	0.751	0.683	0.574	0.593	0.564	0.576	0.784
Reg	0.744	0.710	0.585	0.625	0.604	0.589	0.778
Res	0.731	0.712	0.572	0.621	0.604	0.586	0.789
NeXt	0.750	0.721	0.583	0.626	0.607	0.590	0.798
Shuffle	0.727	0.633	0.565	0.536	0.521	0.565	0.692
Squeeze	0.740	0.655	0.570	0.599	0.581	0.578	0.788
VGG	0.713	0.680	0.568	0.612	0.586	0.576	0.747
Wide	0.748	0.725	0.587	0.625	0.606	0.592	0.786

PR AUC for these components is lower. Normalized PR AUC results from the testing set, shown in Table 8, correspond well with the label distribution in Fig. 3, with the models having the highest by-label performance placed in the order of intercooler, compressor, EGR, and then injectors. The average ROC AUC results, shown in Table 9, also correspond well with the label distribution. In general, the CNNs are more proficient in properly classifying intercooler and compressor failures, having normalized AP values above 0.7 for most of the trained models. Notably, the InceptionNet and MNAS show particularly poor performance, likely attributed to an inability to converge on a solution within the number of training epochs. When comparing the normalized AP to the minimum AP, it is clear that the MNAS is no better than random guessing in most cases, with the InceptionNet having much lower performance compared to other CNNs. The EGR appears to be one of the primary labels that differentiate the performance of the CNNs as shown by comparing the overall mAP in Table 10 to the by-label PR AUC in Table 6. In particular, the RegNet, ResNet152, ResNeXt, and Wide ResNet have normalized mAP values over 0.65 as a result of EGR PR AUCs of approximately 0.7. EfficientNet and GoogLeNet have similar proficiency with EGR compared to the aforementioned CNNs, but have poorer classification performance with the Injectors. The injectors, on average, are appeared to be classified with lower precision than the other system components. This is an expected result as the injectors have high linear dependence among themselves, and share similar signal characteristics with EGR failure. Notably, the pattern shown by the biased-label distribution in Fig. 3 is less consistent among the injectors, sometimes favoring injectors with less sample data over others. However, comparing the normalized AP values to the minimum AP values shows that this effect is significantly lower than the proportional amount of skew, confirming that the loss function effectively handles the label imbalance.

The average ROC AUC for each classification model shows relatively good performance for each label. The EGR and intercooler appear to have the most distinct failure mode among all types of failures simulated for the purposes of classification. The compressor and injectors appear to have a similar level of distinctiveness among the classifiers, with Injector 3 standing out as less distinct on average compared to the other labels. Most of the previously mentioned CNNs with high mAP values show high ROC AUC performance in distinguishing between labels. Others, such as the InceptionNet and DenseNet also show relatively higher ROC AUC performance although their mAP is noticeably lower than the aforementioned CNNs. This can primarily be attributed to a lower EGR AP and Intercooler AP for the InceptionNet and a lower EGR AP for the DenseNet, which implies that these models have poor recall for higher precision values.

For determining the optimal threshold value for classifying a binary health condition of each component, the Maximum F1-Score, the EMR, and the Hamming Loss were investigated. The calculated mAP, EMR, and Hamming Loss based on the optimal thresholds can be found in Table 10. The micro (m), macro (M), weighted (W), and samples (S) averages of the thresholds associated with each metric are also presented in Figs. 8 and 9 [54]. The thresholds based on the F1-Score appear to be more balanced, with values between 0.5 and 0.6 for most cases. The EMR and Hamming Loss thresholds appear to be more restrictive, having values in the range of 0.6–0.7. For maximizing the F1-Score, the samples average is considered to be the most representative for the multi-label case. For most CNNs, excluding the InceptionNet and MNAS network, the F1-Score is around 0.7 as shown in Figs. 10 and 11. As such, all CNNs are shown to have significantly improved performances over a baseline classifier. The F1-Scores corresponding to the maximum EMRs and minimum Hamming Losses, are presented in Figs. 10 and 11, also show improvement beyond the baseline. However, they show poorer performance

Table 9 Average ROC AUC for different components using different CNNs

Net	Comp	EGR	Inj1	Inj2	Inj3	Inj4	Inter
Alex	0.791	0.809	0.774	0.762	0.758	0.767	0.821
Dense	0.811	0.826	0.795	0.782	0.776	0.787	0.819
Efficient	0.795	0.820	0.791	0.779	0.776	0.785	0.816
GoogLe	0.810	0.800	0.797	0.782	0.778	0.785	0.825
Incept	0.696	0.732	0.648	0.647	0.651	0.646	0.713
MNAS	0.547	0.567	0.545	0.514	0.514	0.552	0.598
Mobile	0.812	0.815	0.789	0.775	0.775	0.780	0.818
Reg	0.813	0.823	0.793	0.780	0.773	0.786	0.822
Res	0.802	0.816	0.790	0.777	0.772	0.780	0.819
NeXt	0.807	0.823	0.794	0.783	0.778	0.783	0.819
Shuffle	0.811	0.782	0.791	0.779	0.772	0.781	0.820
Squeeze	0.802	0.807	0.780	0.771	0.762	0.773	0.818
VGG	0.800	0.817	0.785	0.769	0.762	0.775	0.819
Wide	0.811	0.824	0.796	0.784	0.777	0.790	0.824

Table 10 Average multi-label performance metrics

Net	Normalized mAP	EMR	Hamming
Alex	0.632	0.291	0.242
Dense	0.637	0.289	0.240
Efficient	0.647	0.305	0.239
GoogLe	0.649	0.298	0.239
Incept	0.462	0.240	0.289
MNAS	0.316	0.175	0.309
Mobile	0.646	0.288	0.241
Reg	0.662	0.303	0.238
Res	0.659	0.284	0.241
NeXt	0.668	0.306	0.238
Shuffle	0.606	0.296	0.240
Squeeze	0.644	0.286	0.244
VGG	0.640	0.313	0.238
Wide	0.667	0.309	0.240

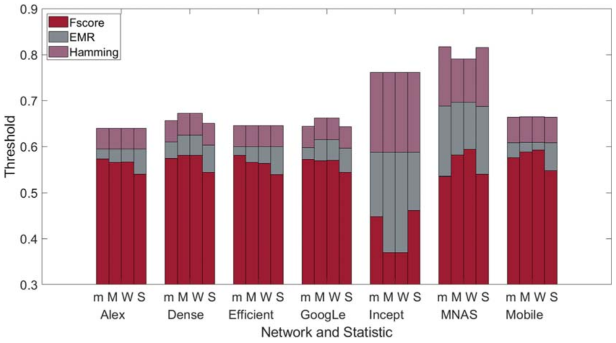


Fig. 8 Optimal thresholds determined by Maximum F1-Score, EMR, and Hamming Loss for the first seven CNNs

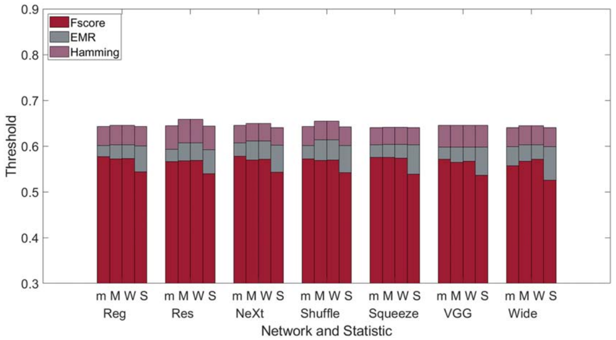


Fig. 9 Optimal thresholds determined by Maximum F1-Score, EMR, and Hamming Loss for the last seven CNNs

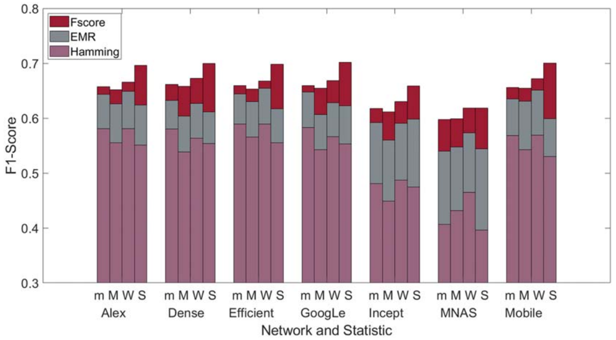


Fig. 10 Average F1-Scores determined by Maximum F1-Score, EMR, and Hamming Loss for the first seven CNNs

when compared to maximizing the F1-Score, with the EMR showing better results as compared to the Hamming Loss.

The precision and recall for each metric, shown in Figs. 12–15, provide more detailed characterization of each classifier. From the figures, maximizing the F1-Score reveals that the CNNs are generally biased toward high recall in terms of performance. This is expected behavior for data that are imbalanced and biased toward negative values, showing that they are more cost-efficient for the classifiers to accept wrong predictions in certain situations as opposed to learning to differentiate between the positive and negative cases. The EMR thresholding produced a more balanced approach between precision and recall as compared to maximizing the F1-Score. As the recall is reduced significantly to achieve the higher precision, this approach confirms that the recall bias is present within the classifiers. Minimizing the Hamming Loss

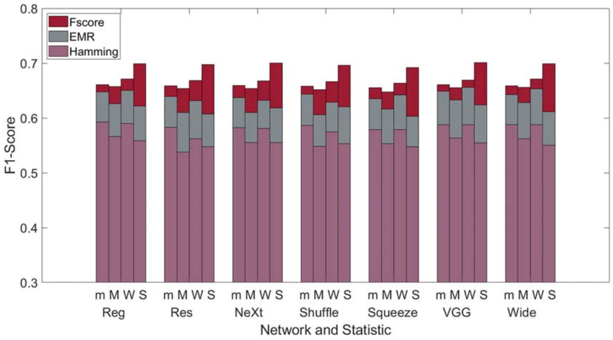


Fig. 11 Average F1-Scores determined by Maximum F1-Score, EMR, and Hamming Loss for the last seven CNNs

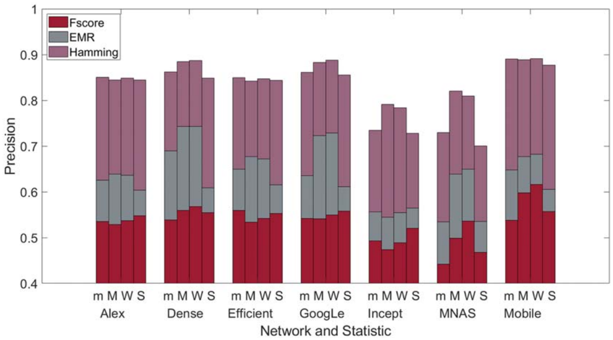


Fig. 12 Average precisions determined by Maximum F1-Score, EMR, and Hamming Loss for the first seven CNNs

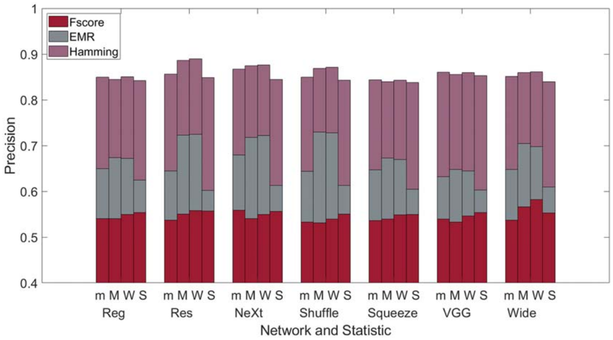


Fig. 13 Average precisions determined by Maximum F1-Score, EMR, and Hamming Loss for the last seven CNNs

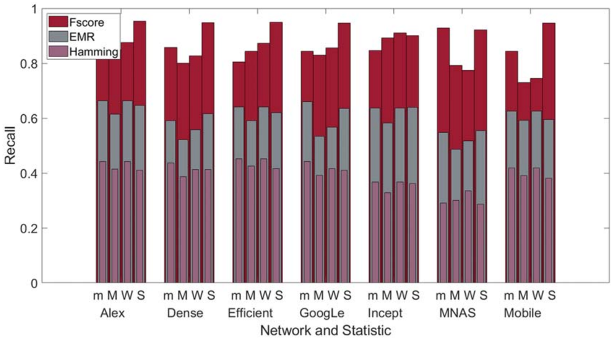


Fig. 14 Average recalls determined by Maximum F1-Score, EMR, and Hamming Loss for first the seven CNNs

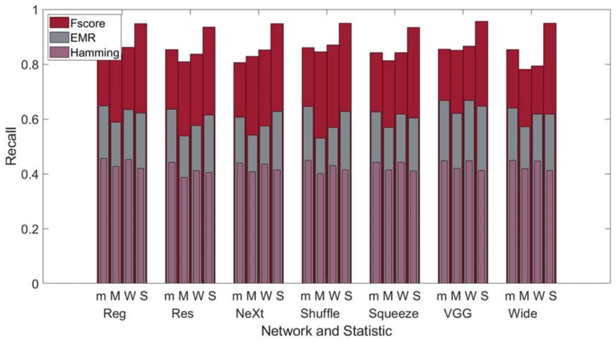
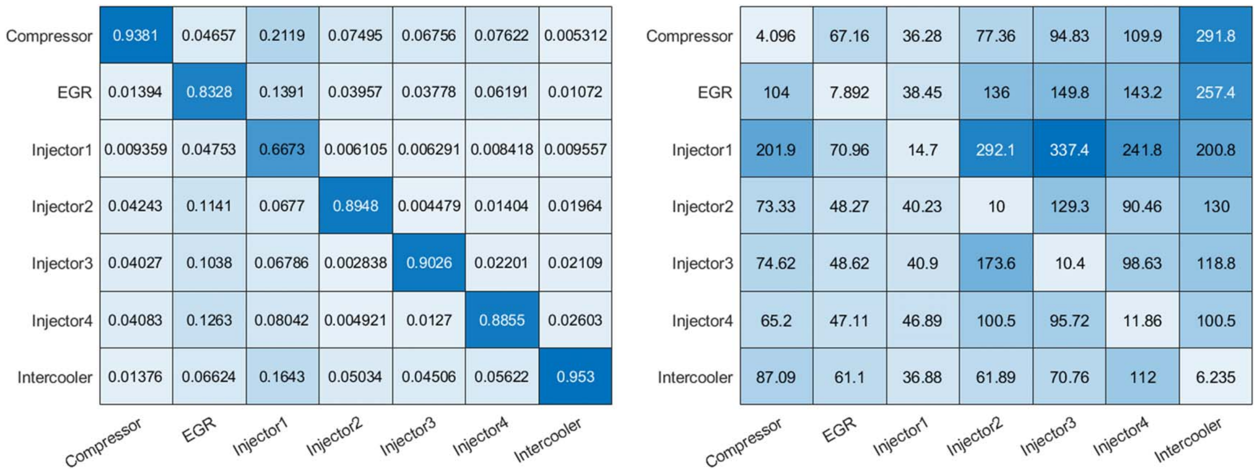


Fig. 15 Average recalls determined by Maximum F1-Score, EMR, and Hamming Loss for the last seven CNNs

shows that improving precision leads to a disproportionately greater reduction in recall compared to the cost of reducing precision when choosing to increase recall, as observed in the results from the maximum F1-Scores and EMRs.

Finally, it is of interest to know which the CNN has the highest amount of classification variance. To accomplish this, the mean and coefficient of variation of all pretrained CNNs' confusion matrices normalized with ground truth for each thresholding approach are presented in Figs. 16–18. The true positive distribution percentage for the mean values is represented by the diagonal of the confusion matrix. In Fig. 16, it can be seen that the predictions by the maximum F1-Score thresholding best match the ground truth, with a relatively uniform offset corresponding to incorrect predictions. Fig. 17 shows that the EMR approach sacrifices EGR and injector accuracies for improved classification accuracies of the compressor and intercooler. The Hamming Loss thresholding, presented in Fig. 18, shows a similar trend to Fig. 17 without the trade-off in EGR prediction accuracy.

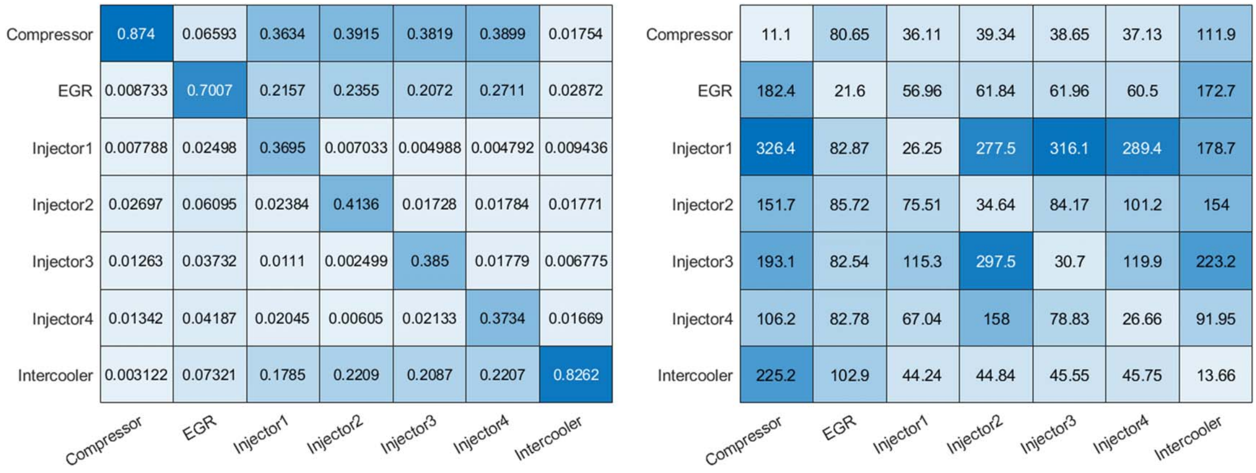
The coefficient of variation matrices in Figs. 16–18 shows where the CNNs have the largest variance in prediction distribution. From Fig. 16, a threshold chosen by maximizing the F1-Score reveals that most variation in the correct answers involves failure modes related to the compressor, Injector 1, and Injector 4. For the compressor, the variance is associated with EGR failure. For Injector 1, most differentiation involves Injectors 2 and 3, while Injector 4 is related to variance involving Injector 3. For thresholding based on the maximum F1-Score, the variance shows that the models are biased toward predicting Intercooler failure even when it is not present. This behavior can also be shown in Figs. 17 and 18. For the EMR thresholding, there is greater variance in the true positive predictions, implying that the CNNs tend to prioritize learning distinct labels for classification. From this approach, it can be seen that most variation in erroneous predictions appears to involve Injector 1, which is associated with all failure modes except the EGR. The Hamming



	Compressor	EGR	Injector1	Injector2	Injector3	Injector4	Intercooler
Compressor	0.9381	0.04657	0.2119	0.07495	0.06756	0.07622	0.005312
EGR	0.01394	0.8328	0.1391	0.03957	0.03778	0.06191	0.01072
Injector1	0.009359	0.04753	0.6673	0.006105	0.006291	0.008418	0.009557
Injector2	0.04243	0.1141	0.0677	0.8948	0.004479	0.01404	0.01964
Injector3	0.04027	0.1038	0.06786	0.002838	0.9026	0.02201	0.02109
Injector4	0.04083	0.1263	0.08042	0.004921	0.0127	0.8855	0.02603
Intercooler	0.01376	0.06624	0.1643	0.05034	0.04506	0.05622	0.953

	Compressor	EGR	Injector1	Injector2	Injector3	Injector4	Intercooler
Compressor	4.096	67.16	36.28	77.36	94.83	109.9	291.8
EGR	104	7.892	38.45	136	149.8	143.2	257.4
Injector1	201.9	70.96	14.7	292.1	337.4	241.8	200.8
Injector2	73.33	48.27	40.23	10	129.3	90.46	130
Injector3	74.62	48.62	40.9	173.6	10.4	98.63	118.8
Injector4	65.2	47.11	46.89	100.5	95.72	11.86	100.5
Intercooler	87.09	61.1	36.88	61.89	70.76	112	6.235

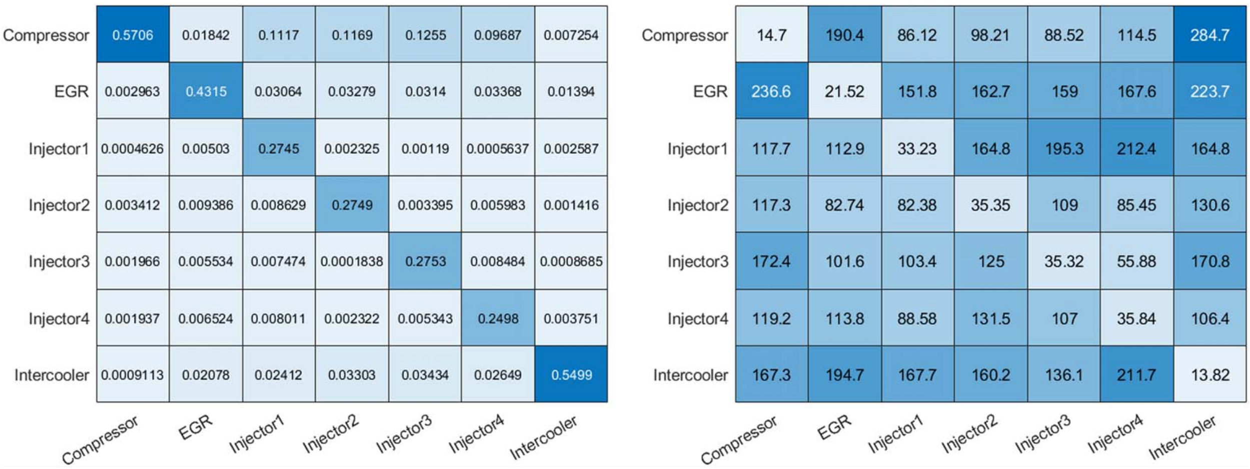
Fig. 16 Mean of all pretrained CNNs' confusion matrices (left) and coefficient of variation (right) from by-label maximum F1-Score thresholding



	Compressor	EGR	Injector1	Injector2	Injector3	Injector4	Intercooler
Compressor	0.874	0.06593	0.3634	0.3915	0.3819	0.3899	0.01754
EGR	0.008733	0.7007	0.2157	0.2355	0.2072	0.2711	0.02872
Injector1	0.007788	0.02498	0.3695	0.007033	0.004988	0.004792	0.009436
Injector2	0.02697	0.06095	0.02384	0.4136	0.01728	0.01784	0.01771
Injector3	0.01263	0.03732	0.0111	0.002499	0.385	0.01779	0.006775
Injector4	0.01342	0.04187	0.02045	0.00605	0.02133	0.3734	0.01669
Intercooler	0.003122	0.07321	0.1785	0.2209	0.2087	0.2207	0.8262

	Compressor	EGR	Injector1	Injector2	Injector3	Injector4	Intercooler
Compressor	11.1	80.65	36.11	39.34	38.65	37.13	111.9
EGR	182.4	21.6	56.96	61.84	61.96	60.5	172.7
Injector1	326.4	82.87	26.25	277.5	316.1	289.4	178.7
Injector2	151.7	85.72	75.51	34.64	84.17	101.2	154
Injector3	193.1	82.54	115.3	297.5	30.7	119.9	223.2
Injector4	106.2	82.78	67.04	158	78.83	26.66	91.95
Intercooler	225.2	102.9	44.24	44.84	45.55	45.75	13.66

Fig. 17 Mean of all pretrained CNNs' confusion matrices (left) and coefficient of variation (right) from by-label maximum EMR thresholding



	Compressor	EGR	Injector1	Injector2	Injector3	Injector4	Intercooler
Compressor	0.5706	0.01842	0.1117	0.1169	0.1255	0.09687	0.007254
EGR	0.002963	0.4315	0.03064	0.03279	0.0314	0.03368	0.01394
Injector1	0.0004626	0.00503	0.2745	0.002325	0.00119	0.0005637	0.002587
Injector2	0.003412	0.009386	0.008629	0.2749	0.003395	0.005983	0.001416
Injector3	0.001966	0.005534	0.007474	0.0001838	0.2753	0.008484	0.0008685
Injector4	0.001937	0.006524	0.008011	0.002322	0.005343	0.2498	0.003751
Intercooler	0.0009113	0.02078	0.02412	0.03303	0.03434	0.02649	0.5499

	Compressor	EGR	Injector1	Injector2	Injector3	Injector4	Intercooler
Compressor	14.7	190.4	86.12	98.21	88.52	114.5	284.7
EGR	236.6	21.52	151.8	162.7	159	167.6	223.7
Injector1	117.7	112.9	33.23	164.8	195.3	212.4	164.8
Injector2	117.3	82.74	82.38	35.35	109	85.45	130.6
Injector3	172.4	101.6	103.4	125	35.32	55.88	170.8
Injector4	119.2	113.8	88.58	131.5	107	35.84	106.4
Intercooler	167.3	194.7	167.7	160.2	136.1	211.7	13.82

Fig. 18 Mean of all pretrained CNNs' confusion matrices (left) and coefficient of variation (right) from by-label minimum Hamming loss thresholding

Loss shows similar behavior to the EMR although its true positive predictions are more stable.

Conclusion

This paper presented a multi-label transfer learning approach, which involves retraining fourteen different pretrained CNNs with multi-mode component failure data to predict the binary failure states of targeted components. In this approach, a preprocessing procedure was proposed to represent non-mutually exclusive multi-mode component failure data from an automotive system, in a suitable form for retraining the pretrained CNNs. The retrained CNNs were designed such that the failure modes of an EGR, compressor, injectors, and intercooler of a diesel engine can be identified. Although there have been many studies to determine a single specific component failure using machine learning and deep learning algorithms, to the best of the authors' knowledge, this is the first attempt to predict concurrent failures of multiple components utilizing a multi-label deep learning approach.

When the retrained classifier models based on CNNs were applied to the preprocessed simulation data, the test results showed good overall classification performance. The normalized mAP varied from 0.6 to 0.65 for most CNNs, which is comparable to performance with conventional image datasets. To characterize the prediction results, three thresholding procedures were presented: maximum F1-Score, maximum EMR, and minimum Hamming Loss. The results revealed a preference of higher recall over precision with the CNNs, which is in agreement with previous reports concerning biased multi-label data. However, the CNNs showed significant improvement over baseline classification models for the expected unachievable regions. Finally, the mean and coefficient of variation of the normalized confusion matrices were presented to characterize the differences in learning performance within the CNNs for each label. The results showed a true positive distribution in line with the expected distribution, with a bias toward predicting Intercooler failure. Injectors 1 and 4 were found to have the highest labeling variance among CNNs, with most differentiation involving the relation to the EGR. Therefore, for accurate prediction of component failure, it is important that the training dataset includes a balanced label distribution as much as possible. However, as the performance metrics between the injectors do deviate from this trend, it is suggested that the dependencies between similar components play a much more significant role than label imbalance for the purposes of classifying failure modes in this regime.

The process presented in this paper is for detecting concurrent failure of multiple components that are inherently correlated and can have different health conditions over time within the operational range of a vehicle. This paper has demonstrated that it is possible to determine which components are failing regardless of their severity in the vehicle when the health conditions are hidden, though this predictive capability decreases as failure modes become increasingly ambiguous such as seen with the Injectors. Based on the results of this research, more thorough analyses on the effects of the health condition, signal randomness, and RGB image randomness as well as increasing the dataset size and improving the label and class balance of the training and validation datasets will be conducted in the future. Also, the studied CNNs will be extended in future work to improve the classification performance by combining the CNNs into a consensus learning model. Finally, the effectiveness of different preprocessing techniques will also be examined.

Acknowledgment

This work was supported by the National Science Foundation under Grant No. 1650564.

Conflict of Interest

There are no conflicts of interest.

Data Availability Statement

The datasets generated and supporting the findings of this article are obtainable from the corresponding author upon reasonable request.

References

[1] Kim, M., Schrader, M., Yoon, H.-S., and Bittle, J. A., 2023, "Optimal Traffic Signal Control Using Priority Metric Based on Real-Time Measured Traffic Information," Sustainability, **15**(9), p. 7637.

[2] Elliott, D., Keen, W., and Miao, L., 2019, "Recent Advances in Connected and Automated Vehicles," J. Traffic. Transp. Eng. (Engl. Ed.), **6**(2), pp. 109–131.

[3] Killeen, P., Ding, B., Kiringa, I., and Yeap, T., 2019, "IoT-Based Predictive Maintenance for Fleet Management," Proc. Comput. Sci., **151**, pp. 607–613.

[4] Lu, N., Cheng, N., Zhang, N., Shen, X., and Mark, J. W., 2014, "Connected Vehicles: Solutions and Challenges," IEEE Internet Things J., **1**(4), pp. 289–299.

[5] Talebpour, A., and Mahmassani, H. S., 2016, "Influence of Connected and Autonomous Vehicles on Traffic Flow Stability and Throughput," Transp. Res. Part C: Emerg. Technol., **71**, pp. 143–163.

[6] Theissler, A., Pérez-Velázquez, J., Kettelgerdes, M., and Elger, G., 2021, "Predictive Maintenance Enabled by Machine Learning: Use Cases and Challenges in the Automotive Industry," Reliab. Eng. Syst. Saf., **215**, p. 107864.

[7] Arena, F., Collotta, M., Luca, L., Ruggieri, M., and Termine, F. G., 2022, "Predictive Maintenance in the Automotive Sector: A Literature Review," Math. Comput. Appl., **27**(1), p. 2.

[8] Isermann, R., 2005, "Model-Based Fault-Detection and Diagnosis—Status and Applications," Annu. Rev. Control, **29**(1), pp. 71–85.

[9] Ermagan, V., Krueger, I., Menarini, M., Mizutani, J. I., Oguchi, K., and Weir, D., 2007, "Towards Model-Based Failure-Management for Automotive Software," Fourth International Workshop on Software Engineering for Automotive Systems (SEAS '07), Minneapolis, MN, May 20–26, p. 8.

[10] Cho, D., and Paolella, P., 1990, "Model-Based Failure Detection and Isolation of Automotive Powertrain Systems," 1990 American Control Conference, San Diego, CA, May 23–25, pp. 2898–2907.

[11] Pickard, K., Leopold, T., Muller, P., and Bertsche, P., 2007, "Electronic Failures and Monitoring Strategies in Automotive Control Units," 2007 Annual Reliability and Maintainability Symposium, Orlando, FL, Jan. 22–25, pp. 17–21.

[12] Shivakarthik, S., Bhattacharjee, K., Mithran, M. S., Mehta, S., Kumar, A., Rakla, L., Aserkar, S., Shah, S. and Komati, R., 2021, "Maintenance of Automobiles by Predicting System Fault Severity Using Machine Learning," *Sustainable Communication Networks and Application*, P. Karuppusamy, I. Perikos, F. Shi, and T. N. Nguyen, eds., Springer Singapore, Singapore, pp. 263–274.

[13] Saufi, S. R., Ahmad, Z. A. B., Leong, M. S., and Lim, M. H., 2019, "Challenges and Opportunities of Deep Learning Models for Machinery Fault Detection and Diagnosis: A Review," IEEE Access, **7**, pp. 122644–122662.

[14] Chen, C., Liu, Y., Sun, X., Cairano-Gilfedder, C. D., and Titmus, S., 2019, "Automobile Maintenance Prediction Using Deep Learning With GIS Data," Proc. CIRP, **81**, pp. 447–452.

[15] Toosi, S. B., and Chaoui, H., 2021, "Lithium-Ion Batteries Long Horizon Health Prognostic Using Machine Learning," IEEE Trans. Energy Convers., **37**(2), pp. 1–1.

[16] Catelani, M., Ciani, L., Fantacci, R., Patrizi, G., and Picano, B., 2021, "Remaining Useful Life Estimation for Prognostics of Lithium-Ion Batteries Based on Recurrent Neural Network," IEEE Trans. Instrum. Meas., **70**, pp. 1–11.

[17] Quintián, H., Casteleiro-Roca, J.-L., Perez-Castelo, F. J., Calvo-Rolle, J. L., and Corchado, E., 2016, "Hybrid Intelligent Model for Fault Detection of a Lithium Iron Phosphate Power Cell Used in Electric Vehicles," Hybrid Artificial Intelligent Systems, HAIS 2016, Seville, Spain, Apr. 18–20.

[18] Aye, S. A., and Heyns, P. S., 2017, "An Integrated Gaussian Process Regression for Prediction of Remaining Useful Life of Slow Speed Bearings Based on Acoustic Emission," Mech. Syst. Signal Process., **84**, pp. 485–498.

[19] Jeong, K., and Choi, S., 2019, "Model-Based Sensor Fault Diagnosis of Vehicle Suspensions With a Support Vector Machine," Int. J. Automot. Technol., **20**(5), pp. 961–970.

[20] Praveenkumar, T., Saimurugan, M., Krishnakumar, P., and Ramachandran, K. I., 2014, "Fault Diagnosis of Automobile Gearbox Based on Machine Learning Techniques," Proc. Eng., **97**, pp. 2092–2098.

[21] Alamelu Manghai, T. M., and Jegadeeshwaran, R., 2019, "Vibration Based Brake Health Monitoring Using Wavelet Features: A Machine Learning Approach," J. Vib. Contr., **25**(18), pp. 2534–2550.

[22] Tinga, T., and Loendersloot, R., 2019, "Physical Model-Based Prognostics and Health Monitoring to Enable Predictive Maintenance," *Predictive Maintenance in Dynamic Systems: Advanced Methods, Decision Support Tools and Real-World Applications*, E. Lughofer, and M. Sayed-Mouchaweh, eds., Springer International Publishing, Cham, pp. 313–353.

[23] Yi, L., Tie Qi, C., and Hamilton, B., 2000, "A Fuzzy System for Automotive Fault Diagnosis: Fast Rule Generation and Self-Tuning," IEEE Trans. Veh. Technol., **49**(2), pp. 651–660.

[24] Ashok Raj, J., Singampalli, R. S., and Manikumar, R., 2021, "Application of EMD Based Statistical Parameters for the Prediction of Fault Severity in a Spur Gear Through Vibration Signals," Adv. Mat. Proc. Technol., **8**(2), pp. 1–19.

[25] Gao, Q., Duan, C., Fan, H., and Meng, Q., 2008, "Rotating Machine Fault Diagnosis Using Empirical Mode Decomposition," Mech. Syst. Signal Process., **22**(5), pp. 1072–1081.

[26] Vasavi, S., Aswarth, K., Sai Durga Pavan, T., and Anu Gokhale, A., 2021, "Predictive Analytics as a Service for Vehicle Health Monitoring Using Edge Computing and AK-NN Algorithm," Mater. Today: Proc., **46**, pp. 8645–8654.

[27] O'Donnell, J., and Yoon, H.-S., 2020, "Determination of Time-to-Failure for Automotive System Components Using Machine Learning," ASME J. Comput. Inf. Sci. Eng., **20**(6), p. 061003.

[28] Guo, J., Lao, Z., Hou, M., Li, C., and Zhang, S., 2021, "Mechanical Fault Time Series Prediction by Using EFMSAE-LSTM Neural Network," Measurement, **173**, p. 108566.

[29] Ismail, B. I., Zhang, R., Ewing, D., Cotton, J. S., and Chang, J.-S., 2002, "The Heat Transfer Characteristics of Exhaust Gas Recirculation (EGR) Cooling Devices," ASME 2002 International Mechanical Engineering Congress and Exposition, New Orleans, LA, Nov. 17–22, Vol. 7, pp. 539–546.

[30] Hatami, N., Gavet, Y., and Debayle, J., 2017, "Classification of Time-Series Images Using Deep Convolutional Neural Networks," International Conference on Machine Vision, Vienna, Austria, Nov. 13–15.

[31] Wang, Z., and Oates, T., 2014, Encoding Time Series as Images for Visual Inspection and Classification Using Tiled Convolutional Neural Networks.

[32] Krizhevsky, A., Sutskever, I., and Hinton, G. E., 2017, "ImageNet Classification With Deep Convolutional Neural Networks," Commun. ACM, **60**(6), pp. 84–90.

[33] Simonyan, K., and Zisserman, A., 2015, "Very Deep Convolutional Networks for Large-Scale Image Recognition." 3rd International Conference on Learning Representations, ICLR 2015, San Diego, CA, May 7–9.

[34] He, K., Zhang, X., Ren, S., and Sun, J., 2016, "Deep Residual Learning for Image Recognition," Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, Las Vegas, NV, June 27–30, pp. 770–778.

[35] Iandola, F. N., Han, S., Moskewicz, M. W., Ashraf, K., Dally, W. J., and Keutzer, K., 2016, "SqueezeNet: AlexNet-Level Accuracy With 50X Fewer Parameters and < 0.5MB Model Size," arXiv preprint arXiv:1602.07360.

[36] Huang, G., Liu, Z., Van Der Maaten, L., and Weinberger, K. Q., 2017, "Densely Connected Convolutional Networks," Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, Honolulu, HI, July 21–26.

[37] Szegedy, C., Vanhoucke, V., Ioffe, S., Shlens, J., and Wojna, Z., 2016, "Rethinking the Inception Architecture for Computer Vision," Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, Las Vegas, NV, June 27–30, pp. 2818–2826.

[38] Szegedy, C., Liu, W., Jia, Y., Sermanet, P., Reed, S., Anguelov, D., Erhan, D., Vanhoucke, V., and Rabinovich, A., 2015, "Going Deeper With Convolutions," Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, Boston, MA, June 7–12, pp. 1–9.

[39] Ma, N., Zhang, X., Zheng, H.-T., and Sun, J., 2018, "ShuffleNet v2: Practical Guidelines for Efficient CNN Architecture Design," Proceedings of the European Conference on Computer Vision (ECCV), Munich, Germany, Sept. 8–14, pp. 116–131.

[40] Howard, A., Sandler, M., Chu, G., Chen, L. C., Chen, B., Tan, M., Wang, W., et al., 2019, "Searching for Mobilenetv3," Proceedings of the IEEE/CVF International Conference on Computer Vision, Seoul, South Korea, Oct. 27–Nov. 2, pp. 1314–1324.

[41] Xie, S., Girshick, R., Dollár, P., Tu, Z., and He, K., 2017, "Aggregated Residual Transformations for Deep Neural Networks," Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, Honolulu, HI, July 21–26, pp. 1492–1500.

[42] Zagoruyko, S., and Komodakis, N., 2016, "Wide Residual Networks," arXiv preprint arXiv:1605.07146.

[43] Tan, M., Chen, B., Pang, R., Vasudevan, V., Sandler, M., Howard, A., and Le, Q. V., 2019, "MNASNET: Platform-Aware Neural Architecture Search for Mobile," Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition, Long Beach, CA, June 15–20, pp. 2820–2828.

[44] Tan, M., and Le, Q., 2021, "Efficientnetv2: Smaller Models and Faster Training," International Conference on Machine Learning, Virtual, July 18–24, pp. 10096–10106.

[45] Radosavovic, I., Kosaraju, R. P., Girshick, R., He, K., and Dollár, P., 2020, "Designing Network Design Spaces," Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition, Seattle, WA, June 13–19, pp. 10428–10436.

[46] Wu, T., Huang, Q., Liu, Z., Wang, Y., and Lin, D., 2020, "Distribution-Balanced Loss for Multi-label Classification in Long-Tailed Datasets," European Conference on Computer Vision, Glasgow, UK, Aug. 23–28, Springer, pp. 162–178.

[47] Ridnik, T., Ben-Baruch, E., Zamir, N., Noy, A., Friedman, I., Protter, M., and Zelnik-Manor, L., 2021, "Asymmetric Loss For Multi-Label Classification," 2021 IEEE/CVF International Conference on Computer Vision (ICCV), Montreal, QC, Canada, Oct. 10–17.

[48] Chawla, N. V., Bowyer, K. W., Hall, L. O., and Kegelmeyer, W. P., 2002, "SMOTE: Synthetic Minority Over-Sampling Technique," J. Artif. Intell. Res., **16**, pp. 321–357.

[49] Davis, J., and Goadrich, M., 2006, "The Relationship Between Precision-Recall and ROC Curves," Proceedings of the 23rd International Conference on Machine Learning, Pittsburgh, PA, June 25–29.

[50] Fawcett, T., 2006, "An Introduction to ROC Analysis," Pattern Recogn. Lett., **27**(8), pp. 861–874.

[51] Everingham, M., Van Gool, L., Williams, C. K. I., Winn, J., and Zisserman, A., 2010, "The Pascal Visual Object Classes (VOC) Challenge," Int. J. Comput. Vis., **88**(2), pp. 303–338.

[52] Sorower, M. S., 2010, A Literature Survey on Algorithms for Multi-Label Learning.

[53] Boyd, K., Costa, V. S., Davis, J., and Page, D., 2012, "Unachievable Region in Precision-Recall Space and Its Effect on Empirical Evaluation," Proceedings of the 29th International Coference on International Conference on Machine Learning, Edinburgh, Scotland, UK, June 26–July 1.

[54] Developers, S.-L., 2019, "Metrics and Scoring: Quantifying the Quality of Predictions," Scikit-Learn 0.22. 1 Documentation.