

Fault detection and classification by unsupervised feature extraction and dimensionality reduction

Praveen Chopra^{1,2}  · Sandeep Kumar Yadav¹

Received: 24 April 2015 / Accepted: 18 December 2015 / Published online: 18 January 2016
© The Author(s) 2016

Abstract A unique technique is proposed based on sparse-autoencoders for automated fault detection and classification using the acoustic signal generated from internal combustion (IC) engines. This technique does not require any hand-engineered feature extraction and feature selection from acoustic data for fault detection and classification, as usually done. The proposed technique uses sparse-autoencoder for unsupervised features extraction from the training data. The training and testing data sets are then transformed by these extracted features, before being used by the softmax regression for classification of unknown engines into healthy and faulty class. The use of sparse-autoencoder to learn fault features improves the classification performance significantly with a small number of training data. This technique is tested on industrial IC engine data set, with overall classification performance of 183 correct classifications out of 186 test cases for four different fault classes.

Keywords Autoencoders · Softmax regression · Fault detection · IC engine · ANN classifier

1 Introduction

In the automobile industry, most of the fault detection is done by skilled technicians and their decision is highly influenced by their training, experience and varies with time of the day.

✉ Praveen Chopra
praveenchopra@gmail.com

Sandeep Kumar Yadav
sy@iitj.ac.in

¹ Indian Institute of Technology Jodhpur, Jodhpur, Rajasthan, India

² DRDO, Delhi, India

This decision making is not reliable and requires long decision time. Most of the fault detection techniques developed so far are using the vibration and acoustics signals generated by the engine. It is reported in the literature that almost 99% of the mechanical faults have noticeable indicators in the form of vibration and acoustic signals [1]. These signals are then processed by signal processing technique to extract the desired fault features, to be used for classifier training and testing.

Most widely used feature extraction techniques are in time–frequency or in the frequency domain because the vibration or acoustic signals generated by an IC engine are highly dynamic and non-stationary. In the time–frequency domain Yen and Lin [2] has proposed a feature extraction technique from vibration data based on wavelet packet transform (WPT). The wavelet coefficients of this transformation are used as the features of the vibration data and are used for classification by an artificial neural network (ANN)-based classifier. Wu and Liu [3] also proposed WPT-based feature extraction, where energy distribution of the wavelet packets is used as the features of the acoustic signal. In this work different levels of wavelet packet decomposition with various types of mother wavelets are used to get different types of feature spaces. These features are then used to train ANN-based classifier.

In frequency domain feature extraction, Yadav and Kalra [4] has used spectrogram of the acoustic signal. In this technique they have used total nine statistical features like kurtosis, shape factor, crest factor, mean, median and the variance of spectrogram for classification by an ANN-based classifier.

In these techniques, the feature extraction and selection is based on some hand-engineered criteria, which restricts them to be used for any type of fault detection. The proposed technique in this work does not have any of these con-

straints. The selection of feature and their extraction is completely unsupervised, due to the use of sparse-autoencoder (SAE).

These extracted features by sparse-autoencoder are then used to reduce the dimensionality of the training and testing data before being used by the classifier. In this technique, the frequency domain approach FFT is used to transform these time domain, non-stationary signals into their freq. spectrums. By this conversion, the time domain information of the signal is lost, but it does not affect the efficiency of the technique. The sparse-autoencoder uses these spectrum data vectors for fault features extraction. The sparse-autoencoder is a variant of autoencoder (AE) with added sparsity in its cost function [5]. Autoencoder and its variants are used extensively for unsupervised feature learning from images [6–8] and audio signals [9]. Hinton and Salakhutdinov [7] demonstrated the nonlinear data dimensionality reduction by use of autoencoder. The autoencoder aims to learn a compressed representation for an input through minimizing its reconstruction error. The ability of autoencoder and its variants to learn meaningful features from different types of data is also demonstrated in [6, 10, 11].

Deng et al. [9] has used sparse-autoencoder for acoustic features extraction from human speech signal for human emotion recognition. Shu and Fyshe [12] has used sparse-autoencoder for feature extraction from magnetoencephalography signal. The learned features from autoencoders are in the form of its hidden layer weights.

In the proposed technique softmax regression is used as a classifier. The softmax regression is a generalized version of the logistic regression [13–16], where the output class labels are multi-class classification instead of binary classification. The softmax regression classifier is most suitable when the classes for classification are mutually exclusive. In this work, it was assumed that no two faults occur at the same time. In the area of deeplearning, softmax regression is the most widely used classifier. Zhang and Zhu [17] has used stacked-autoencoders for image feature extraction and softmax regression for classification. In the same area of image classification, Gao et al. [18] and Dong et al. [19] have used convolutional neural networks based feature extraction from images and classification by softmax regression. The softmax regression classifier requires very small training time as compared to widely used ANN-based classifier with the same level of accuracy.

The proposed technique was tested on acoustic data from industrial IC engine, with three different fault classes and one healthy class. The acoustic data are recorded at four different positions of the engine and data from each position are used independently to compute the performance of the technique. A majority voting-based criteria among all four positions was used to finally declare the type of fault in the engine.

2 Proposed technique

The proposed technique uses three stages for fault feature extraction and classification. The first stage is signal processing; the second is feature extraction and feature space dimensionality reduction and the third stage is classifier training and testing.

By analyzing the FFT spectrum of the faults signals, it was observed that the peaks in the FFT spectrum are always at the harmonics of the operating frequency of the engine. The repetition pattern of the peaks at different harmonics represents the faults features in the frequency domain. It is also observed from the spectrum data that most of the spectrum peaks are in the range of 5kHz, so spectrum data up to 6kHz is only used in this work. This resultant spectrum data vector is very small in size as compared to original time domain signal data and contains almost all the features of the data. This size of data vector can be handled by the sparse-autoencoder for feature extraction.

These spectrum vectors of fault signals are used to train the SAE. On training, the SAE updates its weight matrix, which was initialized by random values. This weight matrix represents the features of input training vectors [7].

The flow diagram of the proposed technique is shown in Fig. 1.

The proposed technique uses the following two data sets:

1. Labeled training data set $x_l^{(i)} \in R^m$ with v numbers of data vectors. $\{(x_l^{(1)}, y_l^{(1)}), (x_l^{(2)}, y_l^{(2)}), \dots, (x_l^{(v)}, y_l^{(v)})\}$, where the $y_l^{(i)} \in (1, 2, \dots, C)$ is the class label of each training data vector, where C is number of fault classes or labels.
2. Testing data set $x_t^{(i)} \in R^m$.

2.1 Principle of sparse-autoencoder for unsupervised feature extraction

The objective of the sparse-autoencoder is to solve the following optimization problem [5] to learn the features from the input data:

$$\min_{b, a} \sum_{i=1}^u \|x_l^{(i)} - \sum_{j=1}^m a_j^{(i)} b_j\|_1^2 + \lambda \|a^{(i)}\|_1 \quad (1)$$

Subject to constraint of $\|b_j\|_2 \leq 1$ for all $j = 1, 2, \dots, n$.

Here the term $\sum_{j=1}^m a_j^{(i)} b_j$ is an approximate reconstruction of the input $x_l^{(i)}$. The vector $a^{(i)}$ is activation of input vector $x_l^{(i)}$ and the b_j is learned feature vectors. On training of the SAE by data set $x_l^{(i)}$, it updates the feature vector b_j , to reduce the reconstruction error of $x_l^{(i)}$. The approximate

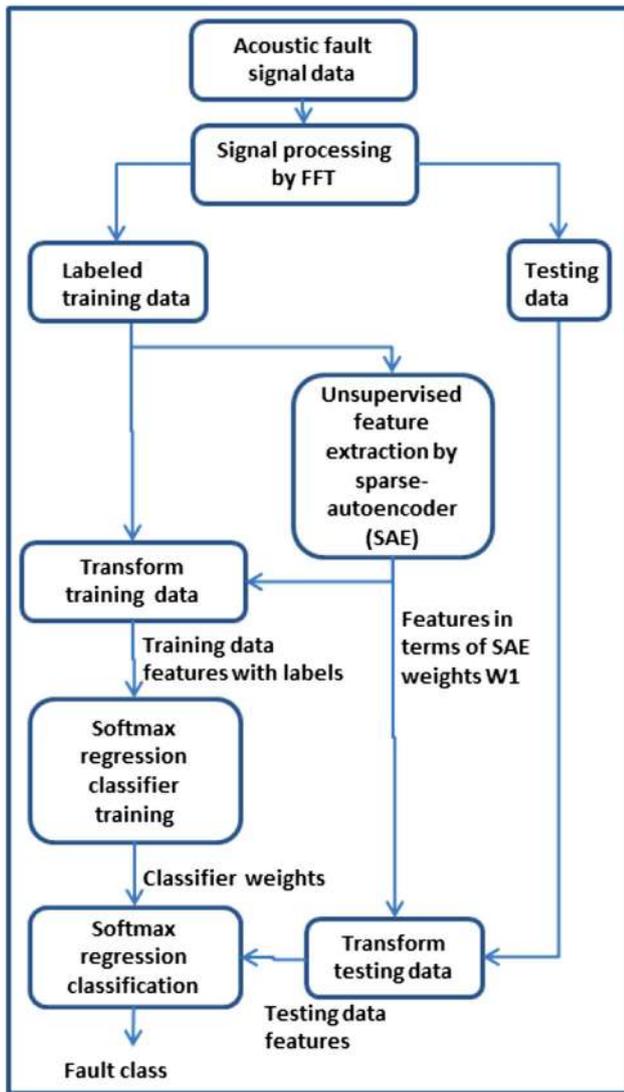


Fig. 1 Flow diagram of fault detection and classification by sparse-autoencoders and softmax regression classifier

reconstruction of the $x_l^{(i)}$, is represented by the $\hat{x}_l^{(i)}$. These feature vectors b_j are the columns of the weight matrix W_1 , which represents the features of data set $x_l^{(i)}$.

In this technique single-layer sparse-autoencoder is used. The SAE has added sparsity in its hidden layer activation [9]. An autoencoder is a three-layer neural network with input, output and a hidden layer. It learns the nonlinear approximation of the identity function at the output on training. The structure of autoencoder is shown in Fig. 2.

The activation of the hidden layer $a_h^{(k)} \in R^n$ for input $x_l^{(i)} \in R^m$ is defined as:

$$a_h(x) = f(W_1 x_l + B_1), \tag{2}$$

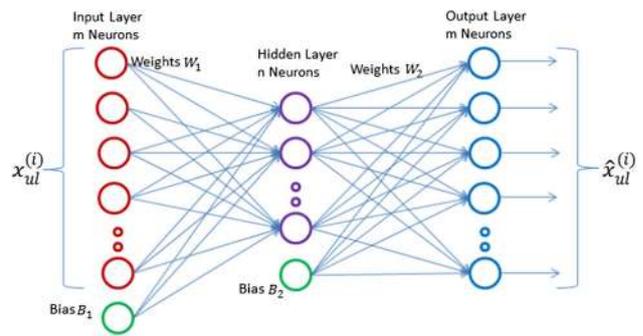


Fig. 2 Autoencoder

where $f(z) = \frac{1}{(1+exp(-z))}$ is a sigmoid function which provides the nonlinear activation and $B_1 \in R^n$ is bias vector. The size of the weight matrix $W_1 \in R^{n \times m}$, where the n is the number of hidden layer neurons and m is size of the input data vector. In the autoencoder, the number of hidden layer neurons are less than number of neurons in input or output layer, $n \ll m$. The activation of output layer is given as below:

$$\hat{x}_l = f(W_2 a_h + B_2), \tag{3}$$

where the $\hat{x}_l^{(i)} \in R^m$ is output vector, a nonlinear approximation of input vector $x_l^{(i)}$. The parameters $W_2 \in R^{m \times n}$ and $B_2 \in R^m$ are weight matrix and bias vector of the output layer.

On training by back-propagation algorithm with v number of vectors $x_l^{(i)}$, the AE updates its weight matrix W_1 and bias vector B_1 , to minimize the reconstruction error $\sum_{i=1}^v \|x_l^{(i)} - \hat{x}_l^{(i)}\|^2$. This nonlinear AE learns low-dimensional and complex features from input data in the form of weight matrix W_1 and bias B_1 . Further enhancement of the feature learning is done by adding the sparsity in the AE. The sparsity limits the number of activation in the hidden layer neurons. This makes the features space more compressed and increases the separability of the data. This sparsity constraint in AE is enforced in its cost function in terms of Kullback–Leibler (KL) divergence. The overall cost function to be minimized with sparsity is:

$$C(W, B) = \frac{1}{2v} \sum_{i=1}^v \|x_l^{(i)} - \hat{x}_l^{(i)}\|^2 + \lambda \|W\|^2 + \beta \sum_{j=1}^m KL(\rho \| \hat{\rho}_j), \tag{4}$$

where W is the sum of weights of both layers and the term $KL(\rho \| \hat{\rho})$ is defined as:

$$KL(\rho \| \hat{\rho}_j) = \rho \log \frac{\rho}{\hat{\rho}_j} + (1 - \rho) \log \frac{1 - \rho}{1 - \hat{\rho}_j}$$

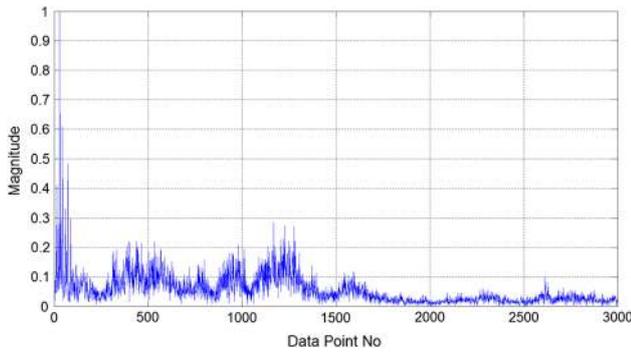


Fig. 3 Original pattern of data

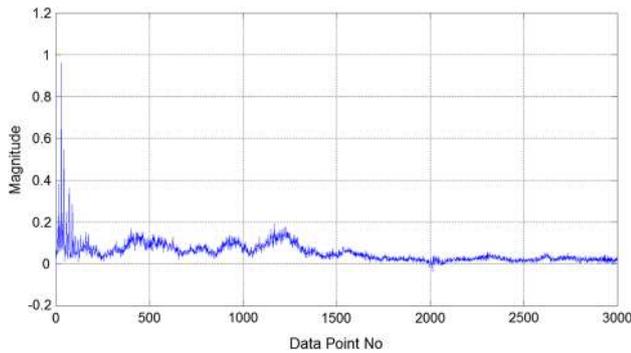


Fig. 4 Reconstructed pattern of data

The parameter ρ is desired sparsity and it controls the activation of hidden neurons.

The parameter $\hat{\rho}_j = \frac{1}{v} \sum_{i=1}^v a_j(x_i)$, is average activation of the hidden layer neuron j for all the input training examples $x_i^{(i)}$. The parameter β controls the weight of the sparsity penalty term and the parameter λ does the weight decay regularization.

The feature matrix W_1 has n linearly independent basis vectors and each represents a unique feature learned from data. In a typical case of SAE with 50 hidden neurons, there are 50 feature vectors in the features matrix W_1 . Figure 3 shows the typical pattern of fault data, and Fig. 4 shows its reconstruction by the weighted linear combination of these learned features [5]. Figure 5 shows the plots of some typical learned feature patterns from the feature matrix W_1 .

In all these figures, the X axis is the data point number, and the Y axis is the magnitude of that data point.

2.2 Transformation of training and testing data by extracted features

The SAE learns the feature matrix W_1 from $x_l^{(i)}$ training data set as described in the previous section. This feature matrix W_1 is then used to linearly transform the input training and testing data vectors into lower dimensional feature vectors.

The training data vector $x_l^{(i)} \in R^m$ is transformed into $\hat{x}_l^{(i)} \in R^n$ as follows:

$$\hat{x}_l = W_1 x_l \quad (5)$$

This transformed training vector $\hat{x}_l^{(i)}$, is weighted linear combination of the feature vectors from W_1 . In other words, the features of training vector $x_l^{(i)}$ are compressed and represented in terms of these learned features. The new training data set $\{(\hat{x}_l^{(1)}, y_l^{(1)}), (\hat{x}_l^{(2)}, y_l^{(2)}), \dots, (\hat{x}_l^{(v)}, y_l^{(v)})\}$ with v number of labeled training data vectors is used to train the softmax regression classifier.

Similarly the testing data vector $x_t^{(i)} \in R^m$, is also transformed into $\hat{x}_t^{(i)} \in R^n$ as follows:

$$\hat{x}_t = W_1 x_t \quad (6)$$

The size of the transformed training and testing data vectors is n , which is less than original size m , because the number of hidden layer neurons are less than the number of input layer neurons or $n \ll m$. This way the proposed technique improves the classification performance by enhancing the feature representation and reducing the size of the training and testing data vectors. In a typical case, the input training and testing data vector of size 6000, is reduced to 50, size of hidden unit.

2.3 Principle of the softmax regression classifier

The softmax regression is a generalization of the logistic regression, where the output class labels are multi-class $y_i \in (1, 2, \dots, k)$, instead of binary output classes [13–16]. The input training set for softmax regression with v numbers of data vectors $\{(x_1, y_1), (x_2, y_2), \dots, (x_v, y_v)\}$, where $x_i \in R^m$.

In the softmax regression-based classifier the probability $P(Y = j|X)$ of X belonging to each class from set of k classes is given as:

$$P(y_i = j|x_i; \theta) = \frac{e^{\theta_j^T x_i}}{\sum_{l=1}^k e^{\theta_l^T x_i}}, \quad (7)$$

where the parameter $j = 1, \dots, k$ and $Y = [y_1, y_2, \dots, y_k]$ is output class. The input variables to this cost function are feature vector $X = [x_1, x_2, \dots, x_v]$, and weight or model parameter of softmax regression model $\theta = [\theta_0, \theta_1, \dots, \theta_k]$.

The generalized softmax regression cost function is defined as:

$$J(\theta) = -\frac{1}{v} \left[\sum_{i=1}^v \sum_{j=0}^1 1(y_i = j) \log P(y_i = j|x_i; \theta) \right] \quad (8)$$

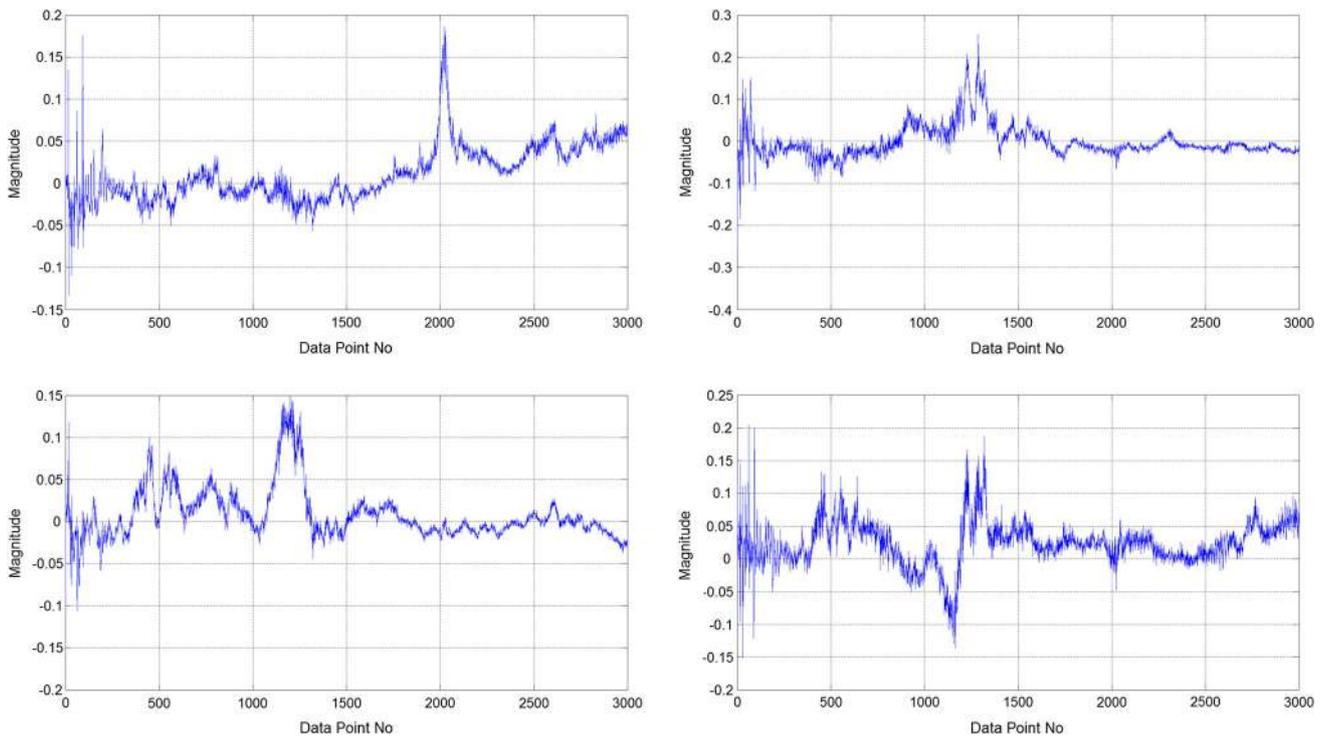


Fig. 5 Learned feature patterns

This softmax regression cost function has no closed form way to minimize the cost value, so the iterative algorithm, gradient descent is used.

The gradient of cost function $\nabla_{\theta_j} J(\theta)$ is given by following equation:

$$\nabla_{\theta_j} J(\theta) = -\frac{1}{v} \sum_{i=1}^v [x_i (1\{y_i = j\} - P(y_i = j|x_i; \theta))], \tag{9}$$

where weight parameters are updated by $\theta_j = \theta_j - \alpha \nabla_{\theta_j} J(\theta)$ for $j = 1, \dots, k$. To make the softmax regression cost function strictly convex, so that it can converge to a global minimum, a weight decay term is added. The modified cost function with its gradient is given blow:

$$J(\theta) = -\frac{1}{v} \left[\sum_{i=1}^v \sum_{j=1}^k 1(y_i = j) \log \frac{e^{\theta_j^T x_i}}{\sum_{l=1}^k e^{\theta_l^T x_i}} \right] + \frac{\lambda}{2} \sum_{i=1}^k \sum_{j=0}^n \theta_{ij}^2 \tag{10}$$

$$\nabla_{\theta_j} J(\theta) = -\frac{1}{v} \sum_{i=1}^v [x_i (1\{y_i = j\} - p(y_i = j|x_i; \theta))] + \lambda \theta_j,$$

where the weight decay parameter λ shall be always positive. All the input data for softmax regression shall be in the

range of 0–1, so the FFT spectrum data vector needs to be normalized. Initially, the weights θ of softmax regression are initialized with random values and these weights are updated with each training vector $\hat{x}_l^{(i)}$, to minimize the value of the cost function.

2.4 Parameters used for the sparse autoencoders and softmax regression

In feature extraction by SAE, following parameters are used:

1. Number of input/output layer neurons, $m = 6000$
2. Number of hidden layer neurons, $n = 50$
3. Sparsity parameter $\rho = 0.25$
4. Weight decay parameter $\lambda = 0.0025$
5. Weight of sparsity penalty term $\beta = 3$.

In classification by softmax regression, following parameters are used:

1. Weight decay parameter $\lambda = 0.001$
2. Number of the weights, $\theta = 50$.

These values of the parameters are arrived after parametric analysis.



Fig. 6 Experimental setup of IC engine

Table 1 Types of fault seeded and number of data sets

S. no.	Fault type	Number of data sets
1	PGW (primary gear whining)	64
2	MRN (magneto rotor noise)	65
3	TAPPET	59
4	Healthy engine	60

3 Experimental setup and results

The proposed sparse-autoencoder and softmax regression-based automated fault detection and classification technique was tested on industrial data from single cylinder IC engines of a commercial two wheeler manufacturing company.

These data were recorded in the industrial environment. In this setup, four PCB 130D20 piezoelectric microphones were placed at four different parts or assemblies of the engine to record these acoustic signals. The speed of rotation of the engine was kept at 40 Hz with the accuracy of $\pm 2\%$.

These acoustic signals were recorded from each position of the sensor for three different types of seeded faults and one normal operation. This technique was tested separately for each sensor position data (Fig. 6).

Table 1 shows the seeded faults and number of data sets recorded for each fault type. The details of each fault are described in [4, 20, 21].

For testing of this technique, each training and testing data set for each fault is divided in the different ratio of training and testing data set, as shown in Table 2. In this division, the selection of data set was completely random.

Table 2 shows the classification performance of each position with different division ratios with majority voting (MV) among all positions. The classification performance is

Table 2 Classification performance in % with different training and testing data set division ratio

Ratio (%)	Pos1	Pos2	Pos3	Pos4	Majority voting
5–95	65.25	55.08	57.63	77.12	53.81
15–85	90.48	76.67	80.0	90.95	86.67
25–75	89.78	95.16	89.25	97.85	98.39
35–65	95.65	90.68	89.96	96.89	98.14
50–50	95.12	95.12	95.12	96.75	98.73
75–25	91.94	93.55	100	100	98.39

depicted in %, total correct classification*100/total test cases, in all the tables.

From Table 2, it can be seen that the proposed technique has performed very well with small number of training data set in the industrial environment also. In this work, the majority voting is the majority of classification types among all four sensor positions. If the classification type has more than two votes for a class, then the classification belongs to that particular class. And if there is a tie between votes, then also the fault classification is assumed from incorrect class only.

In a typical division ratio of 25–75 % for training and testing data, the individual classification performance for each fault type is more than 90 %, as shown in Table 3. The classification performance was computed for each sensor position as well as for each fault type.

Table 4 shows the position-wise classification performance for all fault classes for a typical case of 25–75 % division ratio with majority voting.

For a typical division ratio of 25–75 %, the overall classification performance of 183 correct classifications out of 186 test cases was achieved, with majority voting. In all 186 test cases, three cases were wrongly classified by two or more classifiers on majority voting.

3.1 Comparison of softmax regression with ANN-based classifier

The ANN-based classifier is most widely used classifier in the field of fault detection [2–4, 21] and the softmax regression classifier is widely used in areas where the feature extraction is done by deep learning architectures [17–19]. The softmax regression has been compared with conventional ANN-based classifier on the same data set. In this comparison, a three-layer ANN with input, hidden and output layers was used. The classification performance of ANN classifier varies with the number of neurons in hidden layer and the processing time also varies, for a given size of training data set. To find an optimal configuration of ANN, which provides good classification performance for all sensor positions, different size of hidden layers [100, 150, 200, 250, 300] were tried. In all these configurations, the hidden layer with 200 neurons

Table 3 Position-wise classification performance of each fault

S. no.	Fault type	Pos 1	Pos 2	Pos 3	Pos 4	Majority voting
2	PGW	91.67	97.92	100	97.92	100
3	MRN	97.96	93.88	95.92	100	100
4	TAPPET	88.64	93.18	81.82	95.45	95.45
5	Healthy engine	84.44	95.56	77.78	100	97.78

Table 4 Position-wise classification performance

Position	Classification performance
Position 1	90.86
Position 2	95.16
Position 3	89.25
Position 4	97.85

Table 5 Position-wise majority voting classification performance for all four positions by ANN classifier

Ratio (%)	Pos1	Pos2	Pos3	Pos4	Majority voting
5–95	74.70	60.90	72.08	85.72	69.03
15–85	89.52	76.81	87.98	94.35	88.81
25–75	90.03	81.17	91.63	97.36	93.96
35–65	94.22	83.27	94.72	99.02	97.20
50–50	94.73	83.10	96.11	99.14	97.73
75–25	95.20	84.09	97.87	99.75	98.74

was found optimal. In this process gradient-based back-propagation algorithm was used to train the ANN, with a constant learning rate of 0.1.

Table 5 shows the average classification performance for 100 iterations for ANN classifier with different division ratios.

By comparing Tables 2 and 5, it can be concluded that softmax regression provides comparable classification performance than the ANN classifier for the same size of training and testing data set. The computation time for softmax regression is always less than 10 s for all the division ratios. But in the case of the ANN, the computation time was always in range of minutes for 1000 iterations of ANN training. In conclusion, softmax regression is more suited for real-time applications, with small computation time.

3.2 Comparison with existing techniques

Most of the fault detection techniques available in the literature are based on wavelet or FFT with supervised feature extraction. Yadav and Kalra [4] has used spectrogram for statistical feature extraction from a similar type of IC-engine

Test-Rig with acoustic data. They have used these statistical features to train an ANN-based classifier. The MV accuracy of their technique was less than 93 % for all fault classes. The ANN classifier was trained with 400 training data sets for seven different types of fault classes and was tested for 200 data sets. In another work, Yadav et al. [20] has used FFT and correlation for feature extraction from acoustic data for the same type of IC Engines. In this technique, the faulty engine features are correlated with a prototype engine and the achieved final classification accuracy for four different types of fault classes was less than 93 %. The classification accuracy for CHN fault was 80 % and for MRN fault it was 93 % (Table 6).

With the similar type of fault detection, Wu and Liu [3] has used WPT and energy distribution of the WPT coefficients as features of acoustic data of a GDI (gasoline direct-injection) engine. The claimed average classification accuracy with ANN classifier was around 95 %. For classification of engine fault in five classes, an ANN was trained with 30 training data sets and was tested for 120 data sets for each fault class.

All above discussed techniques use some form of pre-defined criteria for feature extraction and selection from engine signals. But the proposed technique does not require any such criteria and has the performance at par with these techniques with a small set of training data.

In the field of unsupervised feature extraction and selection, Chouchane and Ftoutou [22] has proposed a technique for IC engine fault detection with vibration signals. In this technique, unsupervised feature extraction was done by reducing the size of the matrix representation of the time-frequency image of the fault signal. Then an unsupervised feature selection was carried out to remove the redundancy in the feature set. But, this technique has got limited classification success with fuzzy clustering algorithms as classifiers.

From above analysis, it can be concluded that the proposed technique works very well in the industrial environment, with classification performance more than 98 %. In the industrial environment, where a lot of noise is there in sensor recording, the sparse-autoencoder based feature extraction is very much successful, without any noise filtering of the signals. The softmax regression classifier also performed very well with a small set of training data with these features. The performance of the technique can still improve if the initialization of SEA weight is done in some intelligent way so that

Table 6 Comparison with existing techniques for 25–75 % division ratio

S. no.	Technique	Classifier	No of faults	No of training data sets	Performance
1	Spectrogram-based feature extraction [4]	ANN	7	400	Less than 93 %
2	FFT and correlation based [20]	Comparison with prototype engine correlation matrix	4	NA	Less than 93 %
3	WPT-based feature extraction [3]	ANN	5	30	Over 95 % in various engine operating conditions
4	Reducing the size of the matrix representation of the time–frequency image of the fault signal [22]	Fuzzy clustering	NA	NA	Very limited
5	Proposed technique	Softmax regression	4	62	98 %

the cost function does not get trapped in poor local minima. The implementation of complete technique and analysis was done on Matlab-2013, on an Intel i5 CPU with 8GB RAM.

4 Conclusion

The proposed technique for automated fault detection and classification for IC engines uses sparse-autoencoders for unsupervised feature extraction. These extracted features from the FFT spectrum of the acoustic signals are used for classification by softmax regression. The complete process of feature extraction to feature selection is completely unsupervised. This technique has been tested for various sizes of training and testing data and performed very well. The performance of the technique for the four different fault classes in industrial environment data is more than 98 %.

Acknowledgments The data used in this paper were part of research supported by Technology Information, Forecasting and Assessment Council (TIFAC), Department of Science and Technology (DST), Government of India, under the project number TIFAC/EE/20070174.

References

- Bloch HP, Geitner FK (1997) Machinery failure analysis and trouble shooting. Houston, Gulf
- Yen GY, Lin K-C (1999) Wavelet packet feature extraction for vibration monitoring. *Neural Netw IJCNN* 5:3365–3370
- Wu J-D, Liu C-H (2008) Investigation of engine fault diagnosis using discrete wavelet transform and neural network. *Expert Syst Appl* 35
- Yadav SK, Kalra PK (2010) Automatic fault diagnosis of internal combustion engine based on spectrogram and artificial neural network. In: *Proceedings of the 10th WSEAS Int. conference on robotics, control and manufacturing technology*
- Raina R, Battle A, Lee H, Packer B, Ng AY (2007) Self-taught learning transfer learning from unlabeled data. In: *Proceedings of the 24th international conference on machine learning (ICML)*
- Vincent P, Larochelle H, Bengio Y, Manzagol P-A (2008) Extracting and composing robust features with denoising autoencoders. In: *Proceeding of, 25th international conference on Machine learning, ICML 08, ACM New York, NY, USA, pp 1096–1103*
- Hinton GE, Salakhutdinov RR (2006) Reducing the dimensionality of data with neural networks. *Science* 313:504–507
- Le QV, Ranzato MA, Devin M, Corrado GS, Ng AY (2012) Building high-level features using large scale unsupervised learning. In: *Proceedings of the 29th international conference on machine learning*
- Deng J, Zhang Z, Marchi E, Schuller B (2013) Sparse autoencoder based feature transfer learning for speech emotion recognition. *Humaine association conference on affective computing and intelligent interaction (ACII)*
- Lee H, Ekanadham C, Ng A (2008) Sparse deep belief net model for visual area v2. *Advances in neural information processing systems*
- Rifai S, Vincent P, Muller X, Glorot X, Bengio Y (2011) Contractive auto-encoders: explicit invariance during feature extraction. *International conference on machine learning*
- Shu M, Fyshe A (2013) Sparse autoencoders for word decoding from magnetoencephalography. In: *Proceedings of the 3rd NIPS Workshop on Machine Learning and Interpretation in NeuroImaging (MLINI)*. <http://www.cs.cmu.edu/~afyshe/papers/SparseAE.pdf>
- Bishop CM (2006) *Pattern recognition and machine learning*. Springer, Berlin, pp 205–213

14. http://ufldl.stanford.edu/wiki/index.php/UFLDL_Tutorial. Date 20/01/2015
15. Bhning D (1992) Multinomial logistic regression algorithm. *Ann Inst Stat Math* 44(1):197–200
16. Krishnapuram B, Carin L, Figueiredo MAT, Hartemink AJ (2005) Sparse multinomial logistic regression: fast algorithms and generalization bounds. *IEEE Trans Pattern Anal Mach Intell* 27(6):957–968
17. Zhang H, Zhu Q (2014) Gender classification in face images based on stacked-autoencoders method. The 2014 7th international congress on image and signal processing
18. Gao J, Yang J, Zhang J, Li M (2015) Natural scene recognition based on convolutional neural networks and deep Boltzmann machines. *IEEE international conference on mechatronics and automation (ICMA)*
19. Dong Z, Pei M, He Y, Liu T, Dong Y, Jia Y (2014) Vehicle type classification using unsupervised convolutional neural network. In: 22nd international conference on pattern recognition (ICPR), Stockholm, Sweden, 2014, p 172
20. Yadav SK, Tyagi K, Shah B, Kalra PK (2011) Audio signature-based condition monitoring of internal combustion engine using FFT and correlation approach. *IEEE transactions on instrumentation and measurement*, vol 60, no 4, April 2011
21. Sankar Nidadavolu SVP, Yadav SK, Dr Kalra PK (2009) Condition monitoring of internal combustion engines using empirical mode decomposition and Morlet wavelet. In: *Proceedings of 6th international symposium on image and signal processing and analysis*
22. Chouchane M, Ftoutou E (2011) Unsupervised fuzzy clustering of internal combustion diesel engine faults using vibration analysis. In: *Proceeding of 6th International Conference Acoustical and Vibratory Surveillance Methods and Diagnostical Techniques*, October 2011, Compiègne, France