Rolling Bearing Fault Diagnosis Using Perceptron Algorithm

M. Suphi Deniz Karanfil

***Abstract*—Roller bearings, as being widely used in variety of industries, are subjected to numerous studies for early fault detection. This study shows how a perceptron algorithm with three output classes can be implemented for such task. Vibration signal data was firstly processed with auto regressive (*AR*) filter and then features were extracted from the signal. For frequency domain features, signals were firstly analyzed with FFT, i.e., Hilbert spectral analysis, and then collected from the data. These features should be thought as characteristics representing vibration signals coming from a roller bearing with one of three conditions: inner race fault, outer race fault or healthy. Lastly, these features were given as input to perceptron algorithm, which resulted with weight vectors that can linearly classify and diagnose bearing's fault condition from its vibration signal.**

***Index Terms*—Bearing Fault Diagnosis, Perceptron Algorithm, Feature Extraction**

I. INTRODUCTION

Bearings are widely used as a part of modern machinery for their availability for efficient operation, transmission of power and to avoid mechanical breakdowns. Thus, bearing fault diagnosis in early stages is vital for systems to operate safely for longer period of time. That is why there a lot of studies about early fault detection methods in bearings. The most common approach for bearing fault detection is done by analyzing the vibration signals, since it has all the information about the bearing fault state. In this study, also vibration signals from a "nice" bearing test rig have been analyzed to detect one of the three following conditions: baseline, inner race fault and outer race fault. Given vibration signal data was consisted of signals coming from bearings with one of three conditions. Features were extracted mainly in consecutive two parts. Firstly, time domain features were extracted from the signal data without any preprocess. Secondly, for the frequency domain features, data was firstly analyzed via an AR filter and then envelope analysis was applied by Hilbert and FFT in order to find the maximum amplitudes corresponding to fault frequencies [1]. After all of the features were extracted accordingly with the fault type, they were split with a ratio of 5/6 into training and test data. Training data was used to learn parameters of representations of different fault types for the weight vectors of perceptron algorithm. Lastly, perceptron algorithm was tested with the test data, and results are displayed with a confusion matrix.

II. RELATED STATE OF THE ART STUDIES

Late studies mostly include examples where neural networks employed to accomplish fault diagnosis. One of the examples [2] shows use of topological networks by converting time series signal by using difference visibility graph (DVG) theory. Features were selected based on the most discriminative degree distribution in this graph. Selected features were subsequently fed as inputs to a deep learning model, i.e., a bidirectional long short-term memory network classifier, which delivered a very high fault detection accuracy.

Another application of fault detection by 1D-Convolutional Neural Network (CNN) was shown in [3] for electric motors. The key benefit of usage of CNN is the automatization of the feature extraction stage. 1D-CNN architecture performs the convolution and subsampling process in an alternating manner to automatically learn the filters that optimally extract features from the raw input signal as stated in [3].

III. FEATURE EXTRACTION

A. Time Domain Features

The raw vibration signals can be analyzed directly in the time domain without requiring any signal processing. These time domain features are called statistical parameters, also known as scalar indicators, can provide information about the cause and severity of defects in bearings [4]. All the time domain features used in this project and their formulas are given in Table I. More commonly used features can be briefly explained as following:

1) Mean is the average value of the signal
2) Standard deviation of the signal
3) Root mean square, which has a tendency to increase as the fault degree in the bearing increase [5]
4) The skewness, a measure of asymmetry of signal distribution
5) The kurtosis, a measure of sharpness of the signal distribution, the closer values to 3 mean closer to normal distribution [1]
6) The shape factor, parameter of object's shape, not dependent on its dimension
7) The crest factor, measure of how much impacting is occurred in a time waveform
8) The margin factor, calculation of the level of impact between rolling element and raceway.
9) The peak, maximum value of the absolute value of the signal

10) The peak to peak, a measure of the difference between the maximum and minimum values.
11) TALAF, relatively new parameter developed in [6], is a combination of kurtosis and the RMS values where RMS_0 is the RMS value of a nominal bearing. It provides a measure for defects in a wider range of fault state.
12) THIKAT, implication of how much severity can be tolerated is another parameter proposed in [6], also adds crest factor to the combination of kurtosis and RMS values.

TABLE I
TIME DOMAIN FEATURES AND THEIR FORMULAS

Feature Name	Formula
Mean	$\frac{\sum X}{N}$
Standard deviation	$\sqrt{\frac{\sum(X-\bar{\bar{X}})^2}{N-1}}$
Root mean square	$\sqrt{\frac{\sum(X)^2}{N}}$
Skewness	$\frac{\frac{1}{N}\sum(X-\bar{\bar{X}})^3}{STD^3}$
Kurtosis	$\frac{\frac{1}{N}\sum(X-\bar{\bar{X}})^4}{STD^4}$
Shape factor	$\frac{RMS}{\frac{1}{N}\sum\lvert X\rvert}$
Crest factor	$\frac{max\lvert X\rvert}{RMS}$
Impulse factor	$\frac{max\lvert X\rvert}{\frac{1}{N}\sum\lvert X\rvert}$
Margin factor	$\frac{max\lvert X\rvert}{\left(\frac{1}{N}\sum\sqrt{\lvert X\rvert}\right)^2}$
Peak	$max\lvert X\rvert$
Peak to peak	$max\lvert X\rvert - \min\lvert X\rvert$
TALAF	$log\left[KUR + \frac{RMS}{RMS_0}\right]$
THIKAT	$log\left[KUR^{CF} + \left(\frac{RMS}{RMS_0}\right)^{PEAK}\right]$

Since the given bearing dataset was consisting vibration signals under different load conditions, number of features to be extracted from the signal data was tried to be maximized. Also, features representing different stages of different faults such as TALAF, THIKAT were acquired in order to make a better representation of different types of faults in the dataset.

B. Frequency Domain Features

1) Signal Processing

In order to extract features in the frequency domain, signal data was firstly denoised to remove the unwanted part of the signal. For this project, auto regressive filter was applied to separate the signal from the unwanted noise. However, the expected results after AR filter were not satisfactory. For example, a signal with an inner race fault was filtered by AR filter and the results are shown in Fig. 1. Kurtosis values are explicitly shown to prove the effectiveness of the filter, since it provides information directly related to fault signal impulsiveness, which in this case shows it is not an optimal solution.

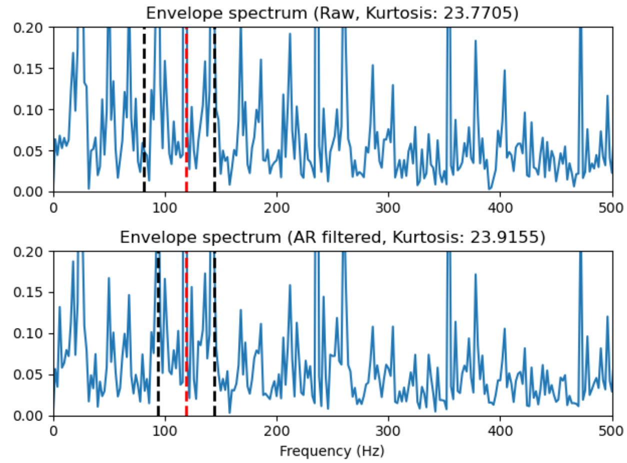

Fig. 1 Envelope spectrum before and after AR filter

The AR filter uses weighted sum of the past values and based on this, estimates the current value of time. It is stated as:

$$X_p(n) = -\sum_{k=1}^{p} a(k)x(n-k). \quad (1)$$

where x and x_p are the raw signal and the predicted signal based on the previous values of the signal, respectively. a_k coefficients of the AR filter can be obtained by using the Yule-Walker equation. After obtaining x_p, it is subtracted from the raw signal to get the residual signal only:

$$e(n) = x(n) - X_p(n)\,. \quad (2)$$

The order p is determined by the kurtosis value to maximize the degree of fault-related signal. Implementation of this filter was adapted with Python functions to estimate AR coefficients by YW equations, the algorithm of this filter was mainly based on the work in [1] and necessary commands are shown in Algorithm 1.

Once the residual signal is obtained with AR filter, analytic signal is estimated by series of Fast Fourier Transform, FFT. This is achieved by taking advantage of python built-in functions, fft and $hilbert$, whose equations are given in (3) and (4) respectively.

$$y[k] = \sum_{n}^{N} e^{-2\pi j\frac{kn}{N}}\ x[n] \quad (3)$$

The FFT of the signal is $y[k]$ for a signal with length N sequence signal, defined as $x[n]$.

$$x_a = F^{-1}(F\big((x)2U\big) = x + iy \quad (4)$$

The analytic signal, x_a, of a time domain signal, x, is estimated by series by Fourier transforms, represented by F, where U is the unit step function and y is the Hilbert transform of x.

2) Frequency Features

The faults in rolling element bearings can be diagnosed with the vibration signal spectrum analysis. Each fault type in roller bearings has its own characteristic frequency depending on the dimensions of the bearing and the shaft rate. Since, the goal of this project includes only two fault types, only Frequency Outer Race Defect, F_{ord} or $BPFO$, and Frequency Inner Race Defect, F_{ird} or $BPFI$ features were examined and used. Formulas to calculate these defect frequencies are given in Table II, where n is the number of elements; P_d is the pitch diameter; B_d is the ball (roller) diameter and ϕ is the contact angle. These dimensional parameters are also illustrated in Fig. 2 except the contact angle, which corresponds to the angle between the rolling element and the raceway. For given parameters, $B_d = 0.235$, $P_d = 1.245$, $n = 8$ and $\phi = 0$, related formulas can also be expressed with (3).

$$BPFO = 3.245\,(f_{shaft});\; BPFI = 4.755(f_{shaft})\,. \quad (5)$$

TABLE II
FREQUENCY DOMAIN FEATURES AND THEIR FORMULAS

Feature Name	Formula
$BPFO$	$\frac{n}{2} f_{shaft}\left(1 - \frac{B_d}{P_d}\cos\phi\right)$
$BPFI$	$\frac{n}{2} f_{shaft}\left(1 + \frac{B_d}{P_d}\cos\phi\right)$

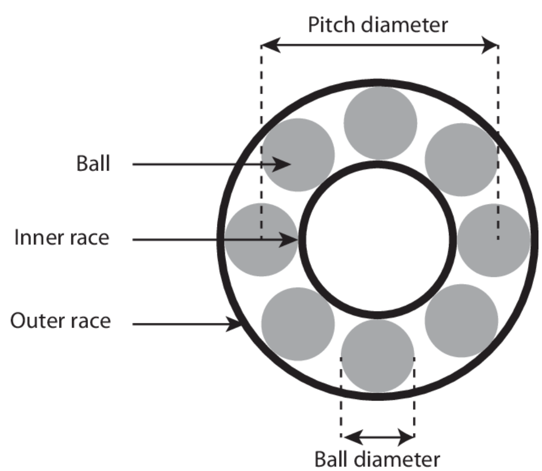

Fig. 2 Geometric characteristics of roller bearings

The amplitudes at the fault frequencies directly pose the fault related information. They can be calculated by extracting the envelope signal after AR filtering and bandpass filtering. One remark about the bandpass filter is that no function in python was found to estimate the bandwidth of the signal, which is why it had firstly been found with MATLAB's bandpass filter and then optimal bandwidth was selected and added to Python environment as a constant value. An illustration of this FFT is shown for each condition in the given dataset in fig. 3 and it is clearly seen how peak amplitudes are aligned with the fault frequencies, which are also added to features as fault frequency amplitudes.

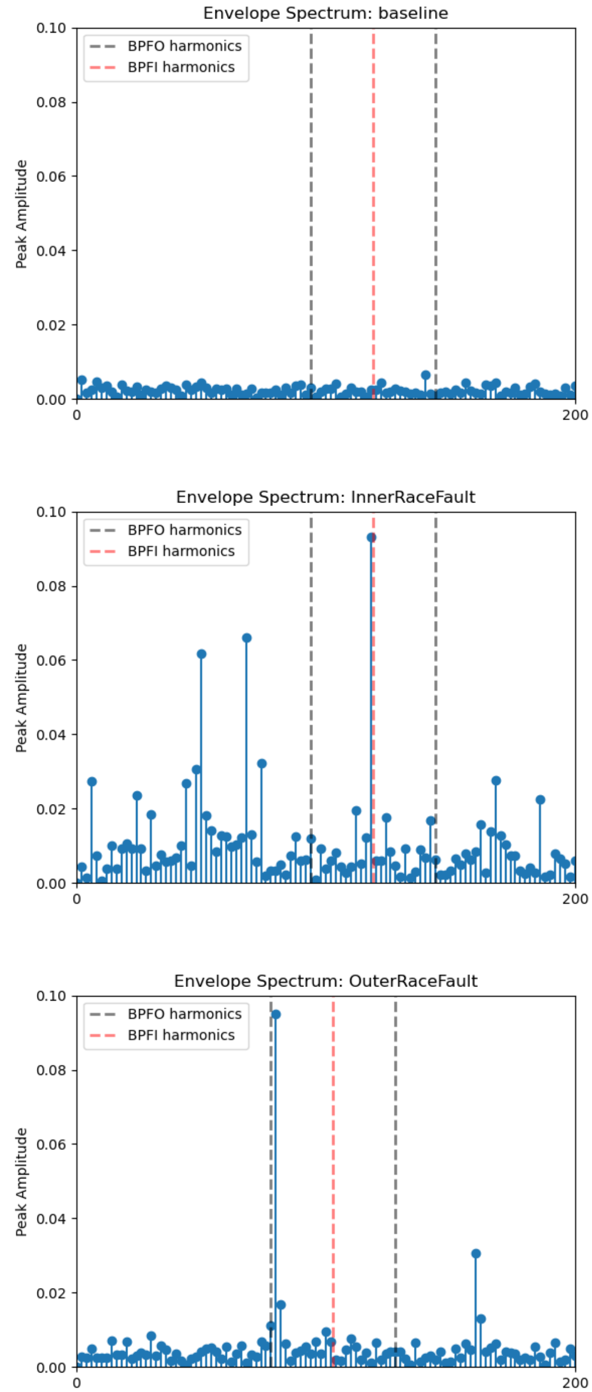

Fig. 3 Envelope spectrum of bearings

After receiving the amplitudes at the fault frequencies as well, all features have been collected and now data will require one last process step before going into algorithm.

C. Creating Feature Space and Data Separation

After features were collected for all the dataset there have been 15 features collected from each input signal with minimum length, which makes overall collected samples 84, with 36 of them representing normal conditions, 30 of them for outer race fault and the rest 18 samples are for inner race fault. These features' normalized values are shown in Fig. 4.

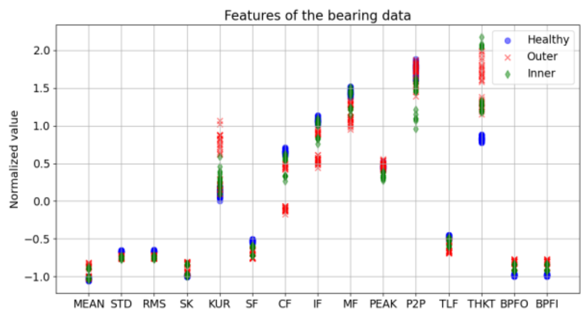

Fig. 4 All features extracted from the dataset

When the scatter plot for the dataset is examined thoroughly, some features can be seen overlapping but it should not be considered as they cannot provide any distinction from one class to another. Moreover, there are also distinguishing features without overlapping which will provide better differentiating, thus increase the chance of the algorithm to learn distinctive parameters for accurate classification.

Lastly before running the algorithm, collected features were split with a ratio of 5 to 6 to create training and test dataset, which corresponds to 70 samples for training data and 14 samples for test data, each consists of features from different cases. Then dataset was ready to given into the perceptron algorithm.

Algorithm 1 AR Filter command in Python.

$a1 = AR\_est\_YW(x,p)[0]$
$xp = lfilter(np.append(0,\ a1,\ [1],\ x)$
$rx = x - xp$
$k = kurtosis(rx,\ Fisher = True)$
$a1 = AR\_est\_YW(x,\ \max(k))[0]$
$xp = lfilter(np.append(0,\ a1,\ [1],\ x)$
$rx = x - xp$

IV. Perceptron Algorithm

The perceptron algorithm was firstly implemented by Frank Rosenblatt in 1958 and later developed with numerous studies because it was seen as a promising algorithm which is capable to solve binary or linear classification problems. The basic characteristics of the perceptron model is to map the original nonlinear data into a higher dimensional feature space where a hyperplane is constructed to bisect two classes' data and maximize the margin of separation between itself and those points lying nearest to the model parameters as described in [7].

It is mentioned in a lot of studies that feature selection is also an important part of bearing fault diagnosis task. Because the more features will imply more computational time with more sensors and data processing parts. Also, many features were stated to result with less efficient analysis and a decrease in the accuracy of the prediction model [7]. There are numerous suggestions on how to select these features. But, in this scope of this project different combinations of features were tried manually but there was no change in the results. So, it was decided to proceed with all the features obtained.

Accordingly, the classification of bearing faults can now be formulated like a linear classification problem as following:

- Inputs: the vectors representing the features of a sequence of signal with minimum lengths, $\boldsymbol{X} = (x_1, \dots, x_L)$, L is the number of samples.
- Labels: each input has a label of 0, 1 or 2 corresponding to fault states healthy, inner race fault or outer race fault respectively, $y = (y_1, \dots, y_L) \in \text{Ą}^L$, $\text{Ą} = \{0, 1, 2\}$
- Task: predict classes of each input feature vector for each sub-signal independently by a three-class linear classifier, where the parameters are weights, $\boldsymbol{w} = (\boldsymbol{w}, \dots, \boldsymbol{w}_{L|\text{Ą}|}) \in \mathbb{R}^{d.|\text{Ą}|}$ and biases $\boldsymbol{b} = (b_1, \dots, b_{|\text{Ą}|}) \in \mathbb{R}^{|\text{Ą}|}$, d is the number of features in each $\boldsymbol{x}$.

The perceptron algorithm is promised to converge to some degree of accuracy after some iteration which is independent of the number of samples. This can be explained by the loss function of perceptron algorithm, which is called as 0-1 loss. This means that the parameters weights and biases are updated for every wrongly classified data so that features of true label are added and wrongly classified ones are subtracted so they will be converging to better estimates for each wrongly classified data until there is none. The main optimizer function is given in (6), and loss function in (7)

$$\hat{y}_i \in Arg \max_{y \in \text{Ą}^L} \left(\left(\langle \boldsymbol{w}_{y_i}, \boldsymbol{x}_i \rangle + b_{y_i} \right) \right),\ i \in \{1, 2, \dots, L\}, \quad (6)$$

$$\widehat{\boldsymbol{w}}_{\boldsymbol{y}_i} = \boldsymbol{w}_{y_i} + (y_i == \hat{y}_i)\boldsymbol{x}_i - (\boldsymbol{w}_{\hat{y}_{i_l}} == \hat{y}_i)\boldsymbol{x}_i\,. \quad (7)$$

The main perceptron algorithm steps are also given in Algorithm 2. Because the dataset does not include so many samples, choice of number of iterations gains an importance for perceptron algorithm to converge. Given the training data, 70 of 84 samples, the parameters of the perceptron algorithm can learn the about the attributes of different fault states in a considerably short time. After training part is completed, the algorithm is tested with the rest of data.

Algorithm 2 Perceptron Algorithm

$error = 1000$
iterate until $error = 0$
  $\boldsymbol{w} \leftarrow 0$
  $b \leftarrow 1$
  find $\hat{y}_i = Arg \max_{y \in \text{Ą}^L} \left(\left(\langle \boldsymbol{w}_{y_i}, \boldsymbol{x}_i \rangle + b_{y_i} \right) \right)$
  if $(y_i == \hat{y}_i)$:
    $error - 1$
  else:
    $error + 1$
    $\boldsymbol{w}_{y_i} = \boldsymbol{w}_{y_i} + (y_i == \hat{y}_i)\boldsymbol{x}_i - (\boldsymbol{w}_{\hat{y}_{i_l}} == \hat{y}_i)\boldsymbol{x}_i$

V. RESULTS

The trained weight vectors could accurately classify 13 out of 14 samples. The results were also observed for the case in which the AR filter was left out. The resulting combinations are given in confusion matrices. The results shown in fig. 5 represents both the AR filter on and off case. This is understandable when all the other features considered. Weights are updated based on all the parameters, so a change in two features, fault frequency amplitudes, does not contribute overall outcome of the algorithm.

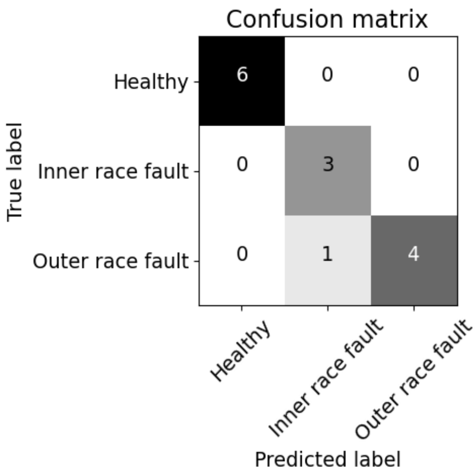

Fig. 5 AR Filter on and off, and all of the features

V. CONCLUSION

Bearing fault diagnosis is critical for most operating industry areas because it is one of the most commonly used parts. In this study, an FDD system has been implemented for the raw vibration signals coming from bearing test rig to classify signals with one of the following fault states, healthy, inner race fault and outer race fault. The process included feature extraction in two parts, time and frequency domain features and also a signal processing part for the latter. After all of the features were extracted accordingly with the fault type, data was split into two parts, training and test data. Lastly, after learning the parameters from the training data, the algorithm was tested with both AR filter on and off options and the results were the same.

One advantage of results being the same is that AR filter was taking so long to run for all the dataset, when it is commented out, all dataset can be examined in considerably shorter time.

In order contemplate on the wrongly classified outer race fault, it could be because of an effect of different loads in the signals with outer race fault, and further studies can be made to capture change of features with different loads.

In conclusion, it can be argued that perceptron algorithm is a good estimator for the problem at hand even though features has not been selected based on differentiation margin for different classes.

REFERENCES

[1] J. Sim, J. Min, D. Kim, S. H. Cho, S. Kim and J. H. Choi, "A python based tutorial on prognostics and health management using vibration signal: signal processing, feature extraction and feature selection," *Journal of Mechanical Science and Technology,* vol. 36, no. 8, p. 4083~4097, 2022.

[2] S. S. Roy et al., "Accurate Detection of Bearing Faults Using Difference Visibility Graph and Bi-Directional Long Short-Term Memory Network Classifier," *IEEE Transactions on Industry Applications,* vol. vol. 58, no. no:4, doi: 10., pp. 4542-4551, July-Aug. 2022.

[3] I. H. Ozcan, O. C. Devecıoglu and T. Ince, "Enhanced bearing fault detection using multichannel, multilevel 1D CNN classifier," *Electr Eng 104,* pp. 435-447, 2022.

[4] P. H. Jain and S. P. Bhosle, "Study of effects of radial load on vibration of bearing using time-Domain statistical parameters," in *IOP Conf. Ser.: Mater. Sci. Eng. 1070 012130*, 2021.

[5] Z. Shen, Z. He, X. Chen, C. Sun and Z. Liu, "A monotonic degradation assessment index of rolling bearings using fuzzy support vector data description and running time," *Sensors,* vol. 12, no. 8, pp. 10109-10135, 2012.

[6] S. S., B. B. and M. T., "Tracking surface degradation of ball bearings by means of new time domain scalar indicators," *International Journal of COMADEM,* vol. 11, pp. 36-45, 2008.

[7] D. D. Susilo, A. Widodo, T. Parahasto and e. al., "Fault diagnosis of roller bearing using parameter evaluation technique and multiclass support vector machine," in *AIP Conference Proceedings 1788, 030081 (2017),* 2017.