25th IMEKO TC4 International Symposium
23rd International Workshop on ADC and DAC Modelling and Testing
IMEKO TC-4 2022
Brescia, Italy / September 12-14, 2022

Engine rotational speed estimation using audio recordings and machine learning algorithms

Cristian Fosalau[1], George Maties[1], Cristian Zet[1]

[1] *"Gheorghe Asachi" Technical University of Iasi, Romania, cfosalau@tuiasi.ro*

***Abstract* – There are many situations in practice when the value of the rotational speed of an engine is needed to be known and a direct instrument for measuring it is not available or, if present, it is decalibrated. One can use instead a non-contact method for estimating the speed. The paper presents such a non-contact method for estimating the rotational speed of a heat engine based on audio recordings and machine learning algorithms. The method principle, experimental validation and a discussion upon the optimal parameters and factors that lead to the best performances are presented.**

I. INTRODUCTION

In the automotive industry, measuring engine speed is essential both in operation and in troubleshooting and maintenance process. There are numerous methods of measuring engine speed in the automotive industry, presented both as a principle in the literature and as functional commercially available devices mounted on cars. The basic principles on which analog and digital tachometers operates, based on either electrical, magnetic, or optical methods, are well known [1-4]. These devices require coupling to the motor shaft and are most often mounted into the car gearbox. There are also non-contact solutions for measuring the engine speed, based on the analysis of engine vibration [5], on the electrical noise produced by the spark plug during ignition over the battery voltage [6], or on various digital image processing methods applied to video recordings over rotating elements [7]. An interesting method of analyzing the automotive engine sound is depicted in [8]. It uses a deterministic - stochastic signal decomposition approach through which the quality of the engine sound may be assessed and then synthesized according to the customer needs and claims. In this approach, the deterministic component of the audio signal is extracted using the synchronous discrete Fourier transform, whereas the stochastic one is modeled using a new suboptimal multipulse excitation approach. A similar approach is discussed in [9], in which sound and vibration measurements are utilized to estimate journal bearing performances of electric motors. In this research, 1/3 octave band analysis techniques are employed for training three unsupervised types of ML algorithms, namely Random Forest Classifier, k-Nearest Neighbours Classifier and Gradient Boosting Regressor. It was proved in the paper that the best results are obtained using sound and z acceleration sets of data for the KNN algorithm, thus obtaining an accuracy of 98 %.

The present paper aims to describe a new method of measuring / classifying the speed values of the heat engine of a car based on processing the sound waveform recordings of the engine using machine learning (ML) algorithms. The method is presented only as a principle in order to provide a very simple solution for estimating engine speed, being useful if the vehicle's tachometer is defective or decalibrated, but it can also be successfully used to determine the speed of any moving element in rotation which produces a characteristic sound whose features are related to speed.

II. METHOD PRESENTATION

In this section, the main steps of the method along with the operations to be performed in each step are presented. Fig. 1 shows schematically the steps to be accomplished for practically implementing the method.

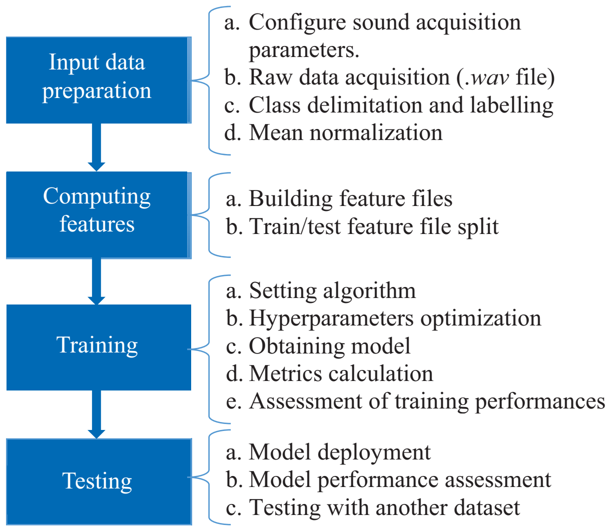

Fig. 1. Main steps of the method dataflow

A. Input data preparation

The preparation of the training set, as well as the test set, is a very important task because the success of the experiment essentially depends on this stage. The recording of the sound produced by the engine is carried out with the help of a mobile phone that is equipped with a built-in sound acquisition hardware system. Recording is done either with the phone's default software or with a sound recording application downloaded from specific application market. It is important to specify that the application be able to record sound continuously, as any delay in recording process causes desynchronization between the timeline and the sound track.

a) *Configure sound acquisition parameters*. This operation consists of setting the resolution, sample rate, number of channels and bitrate in the application chosen for sound recording. It is advisable that the recording is done with a high sampling rate, but not very high because it is directly related to delaying the processing time.

b) *Raw data acquisition*. The sound is recorded at a distance of 1 m from the engine in operation, in two situations: with the hood open and the hood closed. The records are saved in *.wav* format. There are accomplished N_1 records with the hood open and N_2 records with the hood closed, with a mobile phone, for a speed range between idle speed (800 rpm) and 3200 rpm when the engine speed is increased progressively between these two limits and then decreased. Speed recordings in the increasing sense are used for building the training dataset whereas those corresponding to the decreasing slope are for building the testing dataset.

The synchronization of the speed values with the engine sound can be done by video recording of the car's tachometer with a second phone and processing of the two tracks using a common video processing software for delimiting the classes (VideoPAD Editor for example). In this case, the car’s tachometer is the only measurement instrument in this approach, all other results being estimations of the measured quantity. A second more precise solution achieves synchronization by using a specialized device for reading the parameters of the car such as an OBDII, manufactured by ELM Electronics, which is plugged into the dedicated Data Link Connector of the car. Using this solution, the speed values are recorded at a sample rate of 10 samples / s, but precise synchronization of the OBDII start-up time and sound recording are still required. It must be noted here that the accuracy of angular speed measurement using an OBDII device is less than 2 %, as reported by the manufacturer, whereas the tachometer precision is not better than 5 %. However, this solution was chosen for simplicity as it does not require specialized devices.

c) *Class delimitation and labeling*. The delimitation of classes is achieved with the help of video graphics processing software, according to which the moments of time in which the tachometer indicates a certain speed is demarcated. It should be noted here that a very precise delimitation of the speed values is not necessarily to be done, the indications of the car's tachometer being sufficient with an accuracy up to 5%. The *.wav* files corresponding to the classes are thus converted to spreadsheet data files containing the sample values.

d) *Mean normalization*. In order to make the amplitudes of all the records comparable, the data strings are normalized according to the formula:

$$s_{in} = \frac{s_i - \mu}{S_{max} - S_{min}} \tag{1}$$

where s_{in} is the i-th normalized sample, μ is the mean of the class module and S_{max} and S_{min} are the maximum and minimum values of the record.

B. Computing features

The second step is to prepare the feature files for both training and testing. The feature file is built for each of the $N_1 + N_2$ waveform recordings, after which they are concatenated to form a compound feature dataset. It should be noted from the beginning that, due to numerous mechanical components of the motor the emitted sound depends on, such a sound waveform is very complex, containing a lot of time varying frequencies, as well as multiple influence factors that cannot be neglected. In the second section, a sample of such as waveform will be depicted and discussed.

Within a recording represented by a *.wav* file, for each feature class, a comma separated value (csv) file is built. This file will be used for training / testing the ML algorithm. The construction of the features is based on a series of statistical parameters calculated on batches that are cut from the spectrum of the signal corresponding to each class. Fig.2 schematically describes the process of feature computing algorithm.

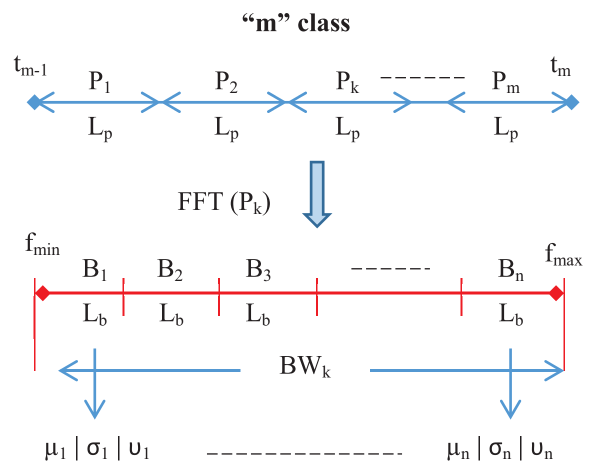

Fig.2. Schematic of feature computing algorithm

The file representing the "m" labeled class is divided into N_m packets of length L_P, which is one of the feature computing parameters and which is common to all classes. For each packet the signal spectrum is calculated by applying a windowed Fast Fourier Transform (FFT). The Flat-Top window was chosen for this due to its property to provide the best amplitude accuracy. A frequency band comprised between f_{min} and f_{max}, BW_k, is cut from the spectrum. Each BW_k is divided into a number of batches, N_{bk} of equal lengths, L_{bk}. Hence, each batch B_k represents a sub-band within BW_k so that

$$BW_k = N_{bk} L_{bk} \tag{2}$$

Next, the statistical parameters mean (μ), standard deviation (σ) and variance (υ) are calculated for each batch, obtaining the k-th feature line in the "m" class feature file. The files thus obtained for each record are concatenated to form the *general feature file*. The final operation in the feature computing stage consists in splitting the general feature file in training and testing datasets (usually the training is assigned to increasing speed whereas the second part of the feature file, corresponding to decreasing speed, is assigned for testing the speed estimator). From the training set, 20% is assigned to cross validation operation devoted to optimizing the hyperparameters of the training ML algorithm.

C. Model training

To obtain the ML model, three well-known algorithms were included for analysis, namely Support Vector Machine (SVM), Neural Network (NN) and Logistic Regression (LR). A comparison was made between the performances offered by them in various conditions and different values of their training parameters.

Table 1 lists the variables that accompany the training model operation. The hyperparameters specific to every algorithm are optimized using the *grid exhaustive search algorithm*, in which a series of manually set values for hyperparameters are scanned successively, and their performances are compared to minimize the training accuracy using cross validation method. The optimization algorithm uses for cross validation 20% of each training class.

Table 1. Feature parameters and hyperparameters for ML algorithms

Algorithm	Feature parameters	Hyperparameters
SVM	Building class: # of classes, f_{min}, f_{max}, BW, L_P, N_b Statistical: μ, σ, υ	SMT type, Kernel type, c, nu, degree, γ, C_0.
NN		# of hidden neurons, hidden layer type, output layer type, cost function type.
LR		tolerance, max iteration.

After optimizing the hyperparameters and training the algorithm, the model is obtained in .json format, for which the metrics are determined and the performance is evaluated on the testing dataset.

D. Model testing

The model testing is completed in two ways:

a) On the test datasets built together with the training datasets from the same *.wav* records, defined as supervised testing records with known class labels. In this case, the model deployment for testing is accomplished for every feature dataset built using initially acquired waveforms. For each dataset, the accuracy is calculated after which the average is assessed. A discussion is, however, performed according to special conditions such as open or closed hood.

b) Live testing with datasets acquired and processed on-line and in real time. For testing in live conditions, an appropriate software was built in LabVIEW which performs in real time the following chain of tasks: sound acquisition - building feature set - model deployment - class estimation - accuracy computing. This test is unsupervised as the class labels are not known a priori, but only the result indication.

III. EXPERIMENTAL VALIDATION

To experimentally validate the method, a number of 10 recordings in *.wav* files were acquired with a mobile phone placed 1 meter from the engine running, 5 with the hood open and other 5 with the hood closed. The recordings have been accomplished with a Samsung Galaxy A52 smart phone using the built-in sound recording application on the engine of an Opel Crossland car, 3 cylinders, 1.2 dm^3 volume, gasoline. The sound acquisition parameters were: sample rate 44100 sample / second, 2 channels, bit rate 256 kbit / s, resolution 16 bits. The recordings were made during approx. 8 minutes, 4 minutes in increasing speed between 800 and 3200 rpm, and 4 minutes when the speed decreases. The speed increase and decrease were approximately linear. Each *.wav* file was divided into 12 classes labeled 1, 2, 3,…, 12, for which the range 800 - 3200 rpm was divided into 12 segments, corresponding to a range of 200 rpm each segment.

Feature building, model training and model testing were carried out using the LabVIEW program and the Analytics and Machine Learning toolkit that is endowed with specific ML libraries and functions. In Fig. 3 is given a fragment of a sound recording containing raw data extracted from a *.wav* file within the range of a packet of 30,000 samples (0.68 s) and for the 3rd class, viz. for the interval of speeds from 1200 rpm to 1400 rpm. In Fig.4, the corresponding spectrum of the signal calculated for a bandwidth BW = 5000 Hz along with a detail up to 500 Hz is given, which represents the pattern for this packet. One may notice the richness of the spectral components, even in the field of high frequencies.

Unfortunately, in the low frequency range, the spectrum is limited to about 30 Hz by the phone's microphone. This may be a serious source of errors as it makes the task of determining features more difficult because the range of fundamental frequencies for this kind of recorded sound is located in this band. Finally, the pattern is segmented in N_b batchs using rectangular windows.

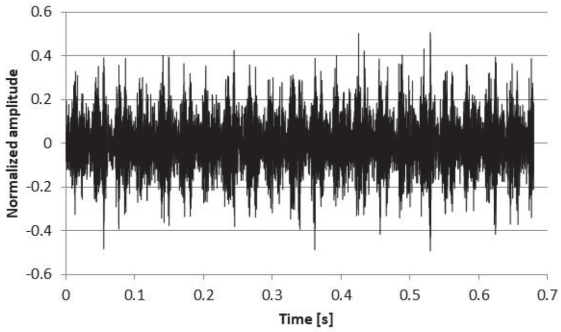

Fig. 3. A fragment of raw data for the 3rd class

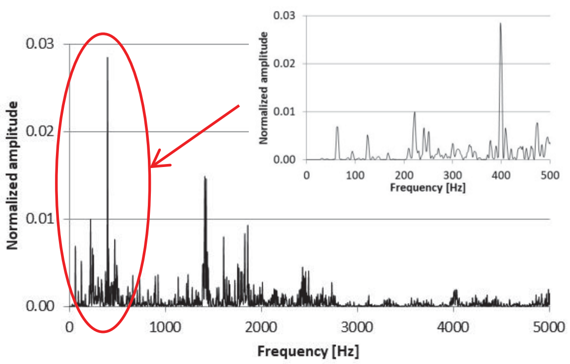

Fig.4. Example of a signal spectrum

It is known that in any ML application, setting the model features is the most important and difficult task because the success of the algorithm essentially depends on the ability to perform this action. It is, nevertheless, practically impossible to establish from the very beginning a set of features that provide maximum or optimal performance. The present paper aims to carry out a study upon the model performance when varying a series of parameters that define the training features.

These parameters are: ML algorithm type, L_P, N_b, f_{max} and the statistical parameters, μ, σ and υ employed, considering the hyperparameters for each algorithm already optimized using the grid exhaustive search method. The minimum frequency f_{min} was considered less than 30 Hz, having no influence over the bandwidth length, The metrics chosen for assessing the algorithm goodness is the accuracy, defined as:

$$Accuracy = \frac{TN + TP}{N} \qquad (3)$$

where TN and TP represent the true negative and the true positive results respectively, and N is the total number of trials for a certain class. We tested also other metrics for assessing the performances of the algorithm such as precision, recall and F1-score, but very similar results were obtained and we retained for our reports only the accuracy, which is the most comprehensive and relevant. The frequency resolution *df* for a packet *k* of length L_{Pk} is calculated as:

$$df = \frac{sampling\ frequency}{L_{pk}} \qquad (4)$$

IV. RESULTS AND DISCUSSION

There were performed a large number of trials (training and test) in order to draw a conclusion regarding the best set of feature parameters to be engaged for obtaining the best results.

Table 2 presents some of these trials underlying the most significant lines that may lead to a relevant conclusion. The effectiveness of the algorithm is assessed by the calculated accuracy as a metric score, but also a discussion is undergone regarding the computation time. An exact computation time has not been determined for a dataset, but estimation can be made according to the quantity of features to be calculated. By analyzing the lines of the tables, we can conclude the following:

- From lines 1, 4 and 7 one may observe that the increase of L_P improves the resolution in frequency and therefore the accuracy in both training and testing stages, because the features are more differentiated by their individual value of the spectral components, than by their statistical behavior in the range of a batch. Nevertheless, too high an L_P (lines 19 and 20), i.e. too low a resolution, mainly under 1 Hz, decreases the overall performance leading to overfitting, that is obtaining excellent accuracy in the training stage while testing with new data provides poor results.
- SVM and NN algorithms provide comparable results, but the benefits brought by LR are unsatisfactory.
- Comparing the lines 7 and 9 or 8 and 10, it is observed that increasing the batch number, i.e. narrowing the bandwidth of a batch on which the statistical parameters are calculated improves the selectivity of the features. The drawback is a bigger computing effort and hence a delay in obtaining the result.
- The best results are obtained by using the statistical parameters μ and σ. In this case, adding the variance υ does not significantly change the accuracy, but also increases the features computing time.
- By analyzing the lines 15 to 19, it is found that widening the frequency range (increasing f_{max}) leads to enlarging the dispersion of the features with maximum performance around 300 Hz, after which the accuracy drastically decrease. This also reduces the accuracy of new recordings that are not part of the test dataset.

Table 2. The most significant trials (training + testing) containing the significant parameters and their results

# trial	Algorithm type	L_p [samples | s]	df [Hz]	f_{max} [Hz]	N_b	L_b [Hz]	Statistical params	Training accuracy	Testing Accuracy
1.	SVM	10000 | 0.22	4.4	200	5	39.7	μ	0.88	0.58
2.	NN	10000 | 0.22	4.4	200	5	39.7	μ	0.79	0.63
3.	LR	10000 | 0.22	4.4	200	5	39.7	μ	0.65	0.55
4.	SVM	30000 | 0.68	1.4	200	5	39.7	μ	0.90	0.64
5.	NN	30000 | 0.68	1.4	200	5	39.7	μ	0.86	0.71
6.	LR	30000 | 0.68	1.4	200	5	39.7	μ	0.7	0.54
7.	SVM	50000 | 1.13	0.9	200	5	39.7	μ	0.93	0.71
8.	NN	50000 | 1.13	0.9	200	5	39.7	μ	0.89	0.75
9.	SVM	50000 | 1.13	0.9	200	10	20.2	μ	0.94	0.75
10.	NN	50000 | 1.13	0.9	200	10	20.2	μ	0.89	0.78
11.	NN	50000 | 1.13	0.9	200	10	20.2	μ, σ	0.99	0.87
12.	NN	50000 | 1.13	0.9	200	10	20.2	μ, σ	0.99	0.87
13.	NN	50000 | 1.13	0.9	200	10	20.2	μ, σ, υ	0.99	0.88
14.	NN	50000 | 1.13	0.9	200	30	7	μ, σ	1	0.90
15.	NN	50000 | 1.13	0.9	200	40	5.3	μ, σ	1	0.91
16.	NN	50000 | 1.13	0.9	300	40	5.3	μ, σ	1	0.93
17.	**SVM**	**50000 | 1.13**	**0.9**	**300**	**40**	**7.9**	**μ, σ**	**1**	**0.95**
18.	SVM	50000 | 1.13	0.9	400	40	9.7	μ, σ	1	0.89
19.	SVM	60000 | 1.36	0.7	300	40	7.9	μ, σ	1	0.86
20.	SVM	60000 | 1.36	0.7	300	30	10.3	μ, σ	0.97	0.83

- If reduced response times, that is sampling on periods below 0.2 s are desired, this can be done by reducing L_P, but at the cost of significantly diminishing the accuracy, even below 80%. This can be counterbalanced by reducing N_b and possibly adding a new statistical parameter (in this case υ may be efficient).
- From the example presented in Fig.5, it is observed that the reduction of score is mainly present during the transition zones between classes. It is therefore necessary to make this transition area as smooth as possible.

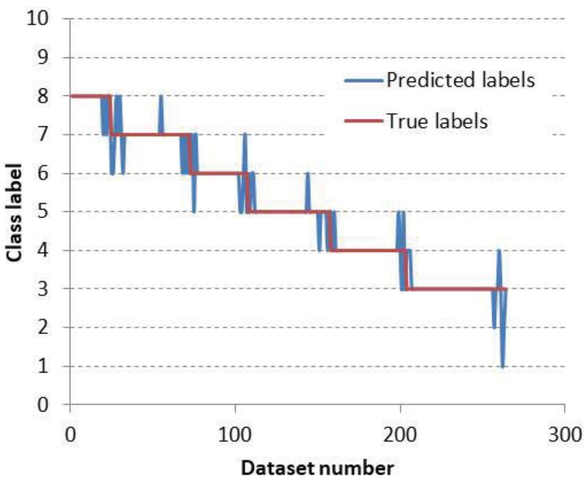

Fig.5. A sequence of testing results where inaccuracies at transition regions may be remarked

- Better results are obtained for longer records for each class, so widening the training records (minimum 30 seconds).
- There is an optimum of the feature parameters represented by the thicker line in the table (line 17) for which one obtained the best score.

As a general conclusion drawn over the above observations, a trade-off between the sampling period, the resources involved, the accuracy and the time computing must be done in order to optimize the feature parameters and to obtain a satisfactory result for further deploying the algorithm in practice.

V. CONCLUSIONS

In the paper, a method based on machine learning algorithms devoted to estimate de rotational speed of a heat engine based on recordings of the sound emitted by the engine was presented. The method was illustrated only as a working principle and has been implemented only on a portable computer for now, being planned to be deployed on smartphones to increase its usability and portability. The main benefits of the method are:

- it is simple, cheap and easy to implement as an application on a device equipped with a sound acquisition system (tablet, mobile phone, laptop, etc.);
- its performance expressed as accuracy in testing stage is satisfactory if a sufficiently large number of classes is considered, with the reverse that important resources for computing are needed;
- it can be used as a method of measuring speed for other rotating or cyclic objects, such as electric motors,

drive mechanisms, etc.;

The method has also some downsides:

- because of the large number of mechanical elements contributing to a specific sound of an engine, in the present study, training and deploying were performed on a single type of car. Training on an engine from one type of car and deploying on another car did not give satisfactory results using the presented feature parameters;
- the method is limited by the computing resources of the device used.

We plan for the future to implement the method as application for smartphones able to estimate speeds for any type of engine, even for electric motors. For this, new types of features are to be utilized using Short Time Fourier Transform and Wavelet Transform. The development of the method is also being considered for detecting malfunctions of an engine.

REFERENCES

[1] J. Marek, H.-P. Trah, Y. Suzuki, I. Yokomori (Editors.), “Sensors for Automotive Applications”, Wiley, 2003.

[2] F. Gustafsson, “Rotational speed sensors: limitations, pre-processing and automotive applications”, IEEE Instrum. & Meas.Magazine, vol. 13, no. 2, April 2010, pp. 16-23.

[3] Z. Shi et al., “Design and Development of a Tachometer Using Magnetoelectric Composite as Magnetic Field Sensor”, IEEE Trans. on Magnetics, vol. 54, no. 7, July 2018, pp. 1-4.

[4] G. H. Choi, W. S. Ra, T. S. Yoon and J. B. Park, “Low-cost tachometer based on the recursive frequency estimation for automotive applications”, SICE Annual Conference 2007, vol. 1, pp. 46-49.

[5] X. Shan, Lu Tang, He Wen, R. Martinek and J. Smulko, “Analysis of Vibration and Acoustic Signals for Noncontact Measurement of Engine Rotation Speed”, Sensors, vol.20, Jan. 2020, pp.1-10.

[6] J.M. Chicharro, A.L. Morales, R. Moreno, A.J. Nieto, P. Pintado, “Sensorless automotive engine speed measurement by noise analysis”, IEEE International Conference on Mechatronics, 2009. Malaga, Spain, pp.14-17.

[7] F.J.T.E. Ferreira, A.F.F.Duarte and F.J.P.Lopes, “Experimental Evaluation of a Novel Webcam-Based Tachometer for In-Situ Rotational Speed Measurement”, IEEE International Conference on Industrial Technology (ICIT), 2020, pp. 917-924.

[8] S.A. Amman and M. Das, “An Efficient Technique for Modeling and Synthesis of Automotive Engine Sounds”, IEEE Trans. on Ind. El., Vol. 48, No. 1, Feb. 2001.

[9] M. Moschopoulos, G.N.Rossopoulos, and C.I. Papadopoulos, "Journal Bearing Performance Prediction Using Machine Learning and Octave-Band Signal Analysis of Sound and Vibration Measurements" Polish Maritime Research, vol.28, no.3, 2021, pp.137-149.