Analysis of time series classification methods applied in Ford sensor measurements for anomalies detection

J. Galvis-Velandia, D. Yáñez-Oyarce, *Students, UPV/EHU*

***Abstract*—This paper presents an analysis of time series classification methods applied to Ford sensor measurements for anomaly detection in automotive subsystems. The study focuses on distinguishing between anomalous and normal time series using two classification approaches: k-Nearest Neighbors (kNN) with *Derivative Dynamic Time Warping* (DTW) as a distance measure and the *Time Series Forest* (TSF) classifier. Each time series represents 500 measurements of engine noise, collected under typical operating conditions with minimal noise contamination. The results provided by both techniques highlight the importance of emphasizing global patterns in time series to detect abrupt changes in engine noise and classify a time series as either anomalous or normal. Although both support the same hypothesis, TSF significantly outperforms kNN with DTW in terms of execution time, as well as in Precision, Recall, Accuracy, and F1-Score, by at least 11%, due to its ability to better capture data patterns and generalize.**

***Index Terms*—Time Series Classification, K-Nearest Neighbors, Derivative Transform Distance, Time Series Forest, Automotive System.**

I. Introduction

In the context of system performance analysis it is important to monitor the performance, reliability, and safety of both components and the entire process, especially in the automotive industry, where sensor signals are increasingly used to make decisions about preventive maintenance of equipment to prevent failures, leading to costly repairs or accidents [1]. In this type of problem, the inherent non-linearity and noise sensitivity of the signals generated by the subsystems within an automobile, the large number of signals and samples per signal and the frequency of the information makes identifying patterns and labelling information complex from a human and computational point of view, Therefore, the adoption of *machine learning* (ML) models that are able to discern patterns in the signals from information labelled as normal operation or faulty, presenting itself as an alternative capable of fulfilling these tasks [2].

Nowadays, traditional classification models are based on measuring distances between data in terms of their features, so the question arises about how to measure the distance among time series, which is not trivial due to temporal and frequency changes, as well as data length, noise and signal characteristics, so researchers have developed techniques to calculate distances of time series data and thus implement feature-based models. Considering all of the above, this paper aims to evaluate the performance of distance-based and interval-based classifiers using a *k-Nearest Neighbors* (kNN) classifier with Derivative Transform Distance (DTDC_C); and a *Time Series Forest classifier* (TSF) to determine the approach best suited to anomaly detection within a Ford vehicle subsystem.

The objective is to diagnose the presence or absence of anomalies in an automotive subsystem based on the analysis of acoustic signals generated by the engine. This can potentially be beneficial for companies and users, as it enables the timely detection of faults in engines or automotive subsystems, preventing major damages or catastrophic failures, improving predictive maintenance performed on vehicles based on their actual condition, among other advantages.

A. FordA dataset description

The dataset (available in [3]) to be analyzed consists of a total of 4921 time series obtained from a Ford vehicle subsystem under both normal and abnormal conditions. Each time series contains 500 samples ranging from -2 to 2 (which may represent the amplitude of the noise waves generated by the engine), with no clearly stable periodic behavior.

Figure 1 shows 3 signals illustrating the behavior of anomalous time series with noise contamination, anomalous time

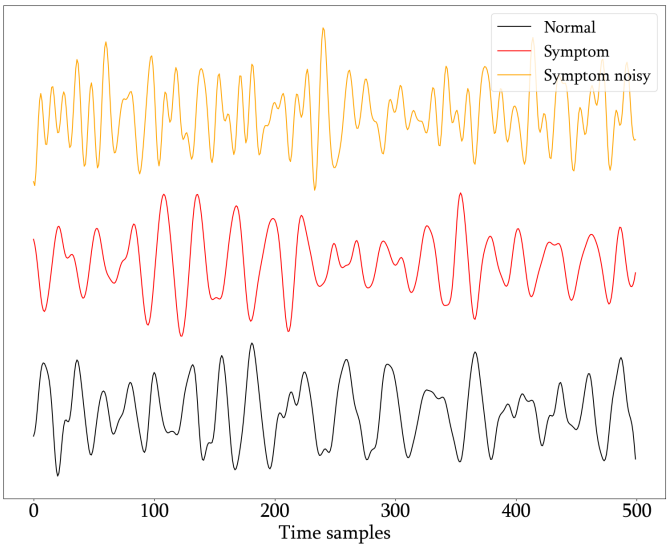

Fig. 1. Ford automotive subsystem time series dataset.

J. Galvis-Velandia is with the Universidad del País Vasco, San Sebastián, Basque Country, Spain (e-mail: jgalvis001@ikasle.ehu.eus).

D. Yáñez is with the Universidad del País Vasco, San Sebastián, Basque Country, Spain (e-mail: dyanez001@ikasle.ehu.eus).

series under conditions with noise contamination, and normal time series under conditions of minimal noise contamination. In this case, **we will work exclusively with time series under minimal noise contamination.**

The format of the dataset comes as a .ts file, which delivers a two-column dataset format, one with an array containing the 500 sensor samples and the other containing the label indicating whether a fault symptom exists in the analysed automotive subsystem.

II. Predictive Modelling

As mentioned in Section I, to make predictions about the anomaly of an automotive subsystem based on the time series containing 500 measurements of engine noise, it is proposed to classify using the *K-Nearest Neighbors* with Derivative Dynamic Time Warping (DTW) classifier and the *Time Series Forest* classifier. The procedure consists of:

- Loading and preparing the data.
- Training the models on the FordA training dataset with hyperparameter tuning to obtain the best model for the task.
- Testing the models on the FordA test dataset and obtaining the metrics Precision, Recall, Accuracy, and F1-Score to determine the best approach for this context.

A. Models

1) Derivative Transform Distance DTD_C: It is a metric designed by Gorécki and Łuczak [4] as an extension of DTW, to which mathematical differentiations and transformations are applied to compare how similar two time series are. It aims to address the signal alignment problem in the time domain by taking it to another space using derivatives and transformations and comparing the distance in the original space and the new space.

This algorithm extends the capability of *Dynamic Time Warping* (DTW) by considering both global differences, such as offset translation, amplitude scaling or linear trends, and local differences in signal amplitude, such that for the same amplitude value in two signals, one corresponds to a valley and the other corresponds to a peak [7]. For the cosine version, the transformation given in (1) is used.

By using *GridSearchCV* to adjust the hyperparameters *n_neighbors* $\in [10, 20, 50]$ and *weights* $\in$ [uniform, distance], the model described obtained has *n_neighbors* $= 20$ and *weights* $=$ distance, which is used to train and validate on the training data, to subsequently train with the entire train dataset and evaluate on the test dataset.

$$c_i = \sum_{j=1}^{m} \cos\left[\frac{\Pi}{2}\left(j - \frac{1}{2}\right)(i-1)\right] \forall i = 1 \dots m \qquad (1)$$

2) Time Series Forest (TSF): In practical terms, the *Time Series Forest* (TSF) classifier is an extension of *Random Forest* for time series, where multiple *Decision Trees* are trained, and each tree votes for a class. The final classification provided by TSF is determined based on the vote count assigned by each *Decision Tree* to the corresponding class. Starting from the definition of a classification tree that uses statistics calculated from the time series, such as mean, standard deviation, average value, Root Main Square (RMS), among others, the TSF algorithm overcomes the problem of the huge feature space that arises when using interval-based algorithms. It starts with the evaluation of all possible splits to find the best information gain, and then introduces a refined splitting criterion that chooses between features with equal information gain, calling this the distance between the splitting margin and the closest value.

Algorithm 1 DTDC(**a**, **b**). Source [5]

Parameters: Weights α and β, distance function *dist*, difference function *diff*
$c \leftarrow \text{diff}(\mathbf{a})$
$d \leftarrow \text{diff}(\mathbf{b})$
$e \leftarrow \cos(\mathbf{a})$
$f \leftarrow \cos(\mathbf{b})$
$x \leftarrow \text{dist}(\mathbf{a}, \mathbf{b})$
$y \leftarrow \text{dist}(c, d)$
$z \leftarrow \text{dist}(e, f)$
$d \leftarrow \alpha \cdot x + \beta \cdot y + (1 - \alpha - \beta) \cdot z$
return d

The intuition behind the idea is that if two splits have equal entropy gain, then the split that is further away from the closest case should be preferred. Considering this, [6] propose Algorithm 2 for training, which implies that for a single tree, they select $\sqrt{m}$ random intervals, generate the mean, standard deviation and slope of the random intervals and then create and train a tree on the resulting $\sqrt[3]{m}$ features. The hyperparameters tuned with *GridSearchCV* are *n_estimators* $\in [50, 100, 200]$ (the number of Decision Trees in Random Forest) and *min_interval* $\in [5, 10, 20]$ (the minimum length of the intervals extracted from the time series). A 10-fold cross-validation (*cv=10*) was used to reduce the risk of overfitting by evaluating the model across multiple subsets of the data.

Algorithm 2 buildClassifierTSF(A list of n cases of length m, $\mathcal{T} = \{X, y\}$). Source [5]

Parameters: The number of trees, r, and the minimum subseries length, p.
Let $\mathcal{F} = \langle F_1, \dots, F_r \rangle$ be the trees in the forest.
for $i \leftarrow 1$ to r **do**
    Let $\mathcal{S}$ be a list of n cases $\langle s_1, \dots, s_n \rangle$, each with $3\sqrt{m}$ attributes.
    for $j \leftarrow 1$ to $\lfloor \sqrt{m} \rfloor$ **do**
        $a \leftarrow \text{rand}(1, m - p)$
        $b \leftarrow \text{rand}(s + p, m)$
        for $k \leftarrow 1$ to n **do**
            $s_k[3(j-1)+1] \leftarrow \text{mean}(x_k, a, b)$
            $s_k[3(j-1)+2] \leftarrow \text{standardDeviation}(x_k, a, b)$
            $s_k[3(j-1)+3] \leftarrow \text{slope}(x_k, a, b)$
        end for
    end for
    F_i.buildClassifier($\{\mathcal{S}, y\}$)
end for

B. Performance

To evaluate TSF and DTD_C, metrics are used that relate the labels given by the models to both the hits and the types of failures committed in the inference. In this context, where the aim is to predict when a symptom of damage appears in the vehicle subsystem, the model that commits the highest number of False Negatives (FN) is penalised. Ignoring signals that indicate failures increases the probability of system malfunction, which can lead to irreparable damage, operational interruptions, or even safety hazards. In contrast, False Positives (FP) would result in unnecessary maintenance actions, which, while generating additional costs such as diagnostic and repair expenses or vehicle downtime, are less critical than addressing corrective maintenance after a failure occurs. Thus, the Recall metric calculated with (3), which considers True Positives (TP) as the most significant aspect, will be prioritised as the key metric. Precision, calculated with (2), is the next most relevant measure to ensure the model does not over-diagnose failures. Additionally, F1 Score, derived from (5) as the harmonic mean of Recall and Precision, provides a balanced perspective on the trade-off between these metrics. Accuracy, calculated with (4), is included to measure the overall correctness of the model, where $\hat{y}_i$ represents the label predicted by the model and y_i is the original label of the time series. To ensure robust evaluation, the metrics are calculated considering imbalanced classes, and cross-validation is employed to validate model performance across different subsets of the data.

The definitions of Precision, Recall, Accuracy, and F1-Score are shown in equations (2) to (5).

$$Precision = \frac{TP}{TP + FP} \tag{2}$$

$$Recall = \frac{TP}{TP + FN} \tag{3}$$

$$Accuracy = \frac{TP + TN}{TP + TN + FP + FN} \tag{4}$$

$$F1\ Score = 2 \cdot \frac{Precision \cdot Recall}{Precision + Recall} \tag{5}$$

III. Computational Experiments

All experiments in this study were conducted on a 13th generation Intel I7-13700H CPU with 16 GB of RAM. The operating system is Windows, and the programming language is Python 3.12.4, utilizing the following libraries: aeon, pandas, numpy, scikit-learn, sktime, matplotlib, time, and joblib.

A. Results

The hyperparameter tuning using *GridSearchCV* provides the following candidate configurations for each model:

- **kNN with DTW:** *n_neighbors*=10, *distance*=ddtw, *weights*=distance.
- **TSF:** *n_estimators*=200, *min_interval*=5.

It is worth nothing that the computational cost of each model is significantly different. In the case of *kNN with DTW*, the hyperparameter search was conducted over 6 possible combinations, taking 51,900.95 seconds. For *Time Series Forest*, 9 combinations were explored in 786.67 seconds.

Table I orders the results based on these parameters, showing that *Time Series Forest* performs better in Recall, Precision, Accuracy, and F1-Score.

TABLE I
PERFORMANCE COMPARISON OF METHODS ($window\_size$=5 70/30 TRAIN TEST)

Method	Time (s)	Recall	Precision	Accuracy	F1
DTD_C	51900.95	0.71987	0.72247	0.72257	0.7211
TSF	786.67	0.82273	0.82271	0.8226	0.82273

s = seconds

IV. Conclusion

For *kNN with DTW*, the hyperparameter tuning suggests that using 10 neighbors combined with the *ddtw* distance metric and distance-based weighting is the optimal strategy because the choice of 10 neighbors allows the model to focus primarily on local patterns (which is why *GridSearchCV* selected the lowest *n_neighbors*), as increasing the number of neighbors tends to produce smoother predictions by focusing more on global patterns.

On the other hand, the use of the *ddtw* distance metric highlights the importance of considering dynamic temporal distortions in the time series. This metric enables the model to capture meaningful variations in the sequence that are critical for anomaly detection in the Ford engine, based on the emitted sounds. Additionally, weighting neighbors by their distance ensures that closer neighbors have a greater influence on the prediction, which helps the model better identify between changes in the time series.

In the case of *TSF*, the hyperparameter tuning suggests that using numerous trees combined with a small interval of temporal point sequences is the best strategy. The choice of 200 trees allows the model to benefit from greater diversity in predictions, as a large ensemble of estimators tends to improve generalization. This increases the stability of the selected metrics, although it entails an additional cost in terms of training time.

Meanwhile, the selection of a minimum interval of 5 temporal points indicates that local patterns are particularly relevant for detecting anomalies in the Ford engine based on the sounds it emits. This small interval allows the model to identify abrupt changes in the time series that might go unnoticed with larger intervals, reinforcing the hypothesis that relationships between nearby points are more decisive than global patterns for this specific task.

As explained earlier, both models focus on better interpreting the local patterns of time series. This means that when detecting an anomaly, it is important to identify sudden changes that may occur in the sound of vehicle engines in the Ford subsystem.

Regarding the results of the algorithms, *Time Series Forest* (TSF) outperforms *kNN with DTW* in every aspect. One of the most notable differences is the training time, which is significantly higher for *kNN with DTW* due to the computational

cost of calculating distances using *Derivative Dynamic Time Warping* (DDTW). This process must be performed for every pair of available time series. Although the training phase for TSF can also be resource-intensive, once completed, inference is considerably faster because new instances only need to pass through the preconstructed trees. In contrast, with *kNN with DTW*, the DDTW distance must be calculated for each new time series in relation to all the series in the training set. This means each time sample is processed by DTW and (1) is calculated afterwards, doing this process for whole dataset, obtaining more than 2,500,000 iterations for FordA training-test process.

Regarding performance metrics (Precision, Recall, Accuracy, and F1-Score), *Time Series Forest* outperforms *kNN with DTW*, significantly (at least 11% in each metric), primarily because TSF is an ensemble model that combines multiple classifiers based on decision trees (*Decision Trees*). This structure allows for efficient exploration of diverse random features proposed by each tree model, resulting in better generalization capability. On the other hand, *kNN* relies more heavily on the quality and distribution of the data in the training set, which can negatively impact its performance if the data contains noise, is imbalanced, or insufficient.

Given the previous result, it is proposed as an extension to perform data augmentation using techniques such as *Time Series SMOTE* to improve the performance of the *Time Series Forest* classifier.

REFERENCES

[1] Liu, B., Xu, Z., Xie, M., & Kuo, W. (2014). A value-based preventive maintenance policy for multi-component system with continuously degrading components. Reliability Engineering & System Safety, 132, 83–89. doi.org/10.1016/j.ress.2014.06.012

[2] H. Xing, Z. Xiao, R. Qu, Z. Zhu, and B. Zhao, "An Efficient Federated Distillation Learning System for Multitask Time Series Classification," IEEE Transactions on Instrumentation and Measurement, vol. 71, pp. 1–12, 2022, doi: doi.org/10.1109/TIM.2022.3201203.

[3] A. Bagnall, "Dataset: FordA," IEEE World Congress on Computational Intelligence. Accessed: Jan. 13, 2025. [Online]. Available: https://timeseriesclassification.com/description.php?Dataset=FordA

[4] T. Górecki and M. Łuczak, "Non-isometric transforms in time series classification using DTW," Knowledge-Based Systems, vol. 61, pp. 98–108, May 2014, doi: 10.1016/j.knosys.2014.02.011.

[5] A. Bagnall, A. Bostrom, J. Large, and J. Lines, "The Great Time Series Classification Bake Off: An Experimental Evaluation of Recently Proposed Algorithms. Extended Version," Feb. 2016.

[6] H. Deng, G. Runger, E. Tuv, and M. Vladimir, "A time series forest for classification and feature extraction," Information Sciences, vol. 239, pp. 142–153, Aug. 2013, doi: 10.1016/j.ins.2013.02.030.

[7] H. Li, C. Guo, and W. Qiu, "Similarity measure based on piecewise linear approximation and derivative dynamic time warping for time series mining," Expert Systems with Applications, vol. 38, no. 12, pp. 14732–14743, Nov. 2011, doi: 10.1016/j.eswa.2011.05.007.