Detecting industrial vehicles' duty levels using contrastive learning

Detecting industrial vehicles' duty levels using contrastive learning

Luca Cagliero, Silvia Buccafusco, Francesco Vaccarino
Politecnico di Torino
Turin, Italy
{name}.{surname}@polito.it

Lucia Salvatori, Riccardo Loti
Tierra SpA
Turin, Italy
{name}.{surname}@tierratelematics.com

Abstract*—Industrial vehicles equipped with CAN bus devices transmit large volumes of IoT signals. The analysis of CAN bus data can be helpful to monitor the current vehicles' workload, namely the vehicle *duty levels*, in an automated fashion. Despite the use of machine learning techniques to automatically detect vehicle duty levels is particularly appealing, existing approaches are challenged by the high cost of human data annotation and by the high heterogeneity of the analyzed vehicles types and models. In this paper, we present a self-supervised approach to automatically detect vehicles' duty levels based on contrastive learning. The multivariate CAN Bus signals are first divided into fixed-sized segments and then embedded into a vector space shared by all vehicles of the same model by leveraging a constrastive approach with a mixup augmentation strategy. The key idea is to embed similar segments in close proximity by self-learning a model-specific clustering, which allows automatic duty level assignment with minimal human supervision. We validate the proposed approach in a real industrial use case, analyzing CAN Bus data acquired from test heavy-duty vehicles. Data were provided by a multinational Internet-of-Things company specialized in telematics solution. The experiments show clustering performance superior to state-of-the-art models and as well as an higher ability to differentiate between *Moving* and *Working* duty levels.

***Index Terms*—Mining IoT Data, Contrastive Learning, Time Series**

I. INTRODUCTION

Managing large fleets of industrial vehicles requires advanced solutions for vehicle usage monitoring. To trace vehicles' activities, vehicle manufacturers install CAN bus data loggers on-board and connect them to the vehicles' OBD-II port [1]. These devices transmit signals, commonly denoted by Suspect Parameter Numbers (SPNs), such as engine speed, fuel level, and coolant temperature. IoT data transmission, storage, and analytics services are commonly supported by highly specialized telematics providers. Among the offered services, providers would like to also support end-users in monitoring vehicles' workload.

Detecting the current vehicles' workload from CAN Bus data is a known research task in the context of industrial vehicle management [2]. Vehicle's workloads arfe conventionally classified into discrete categories, namely the *duty levels*. They indicate whether a vehicle is *off*, *idle* (key on but not moving), *moving*, *working*, or *overloaded* (i.e., working hard). These categories are not standardized and thus can slightly change from one industrial context to another. Furthermore, the temporal sequences of observed duty levels are rather variable across different vehicle types and models. For example, refuse compactors are expected to be *moving* a relatively long time, whereas they are *working* for just short periods. Conversely, excavators likely show opposite trends.

The use of clustering techniques to profile vehicle usage patterns is established. For example, the work presented in [3] addresses multivariate clustering of CAN bus SPNs by using Gaussian Mixture Models. The main purpose is to optimize the maintenance activities of on-road vehicles. In [4], [5] the authors explore the use of both feature-based [6], [7] and shape-based clustering [8] approaches. However, all existing solutions require a prior knowledge of the characteristics of the input time series and of the features that most likely discriminate between different duty levels. Annotating CAN bus data with appropriate domain knowledge can be particularly expensive for domain experts. The increasing complexity of the current industrial scenario, where a large number of diverse vehicles need to profiled, calls for new, adaptive solutions to automatically detect vehicle duty patterns.

This paper presents a constrastive learning approach to automatically detect vehicle duty levels. It leverages a mixup augmentation strategy [9] combined with a Temporal Neighborhood Coding contrastive learning [10] to embed similar segments of the multivariate CAN Bus SPNs in close proximity by exploiting the invariances in the analyzed data. The key advantage is the use of *self-supervised* approach which does not require any prior knowledge about the data for clustering SPN segments. Based on a preliminary clustering in the embedding space, the automatic detection of vehicle's duty levels can be automated with a more limited (and hopefully affordable) human supervision.

We empirically verify the applicability of the proposed solution in real industrial case study. Specifically, we analyze SPNs of test heavy-duty vehicles provided by a multinational telematics company. The results show clustering performance superior to that of state-of-the-art approaches and an better precision/recall trade-off for the *Moving* and *Working* duty levels' assignments.

The remainder of the paper is organized as follows.

Section II describes the industrial scenario. Section III introduces fundamentals of contrastive learning. Section IV describes the proposed methodology. Section V reports the main experimental results, whereas Section VI draws conclusions and discusses future works.

II. The industrial scenario

Tierra Spa is a multinational Internet-of-Things company designing and providing telematics services to major industrial manufacturers in various sectors, among which agriculture, automotive, and construction. Industrial vehicles are monitored using onboard devices able to detect the geographical position and read data from the Controller Area Network (CAN bus). The devices are equipped with a SIM card and send vehicle data, via cellular network, to the cloud infrastructure and the web platforms for remote monitoring.

The Tierra devices are connected to the vehicle's OBD-II [1], gather messages required from the users and transmit the aggregated Suspect Parameter Numbers (SPNs, in short) measurements at different granularity levels, ranging from seconds to minutes. In this work, we will consider SPNs sampled every 30 seconds. The CAN messages are decoded using the SAE J1939 standards[1], the network communications protocol defined by the Society of Automotive Engineers (SAE) for real-time applications on heavy-duty and commercial vehicles. Currently, Tierra devices are spread over 170 countries. In the last three years, the number of monitored vehicles monitored by Tierra has quadrupled, reaching about 100K units. Tierra's infrastructure handles over 1 billion messages per month and processes them to give users real-time access through web platforms. The number of vehicles connected by Tierra devices is steadily on the rise, while customers are increasingly seeking for a broader range of data with enhanced performance of the telematic system.

The Big Data scenario in which telematics providers like Tierra Spa nowadays operate open to relevant challenges in both the management and analysis of CAN BUS data. First, efficient storage and retrieval of IoT data requires highly reliable and scalable infrastructures. Secondly, the development of smarter technologies for IoT networks further improves the connectivity and the volumes of exchanged vehicle data. Thirdly, the recent progress of Artificial Intelligence algorithms has offered more efficient and effective solutions to support decision makers. In this work, we present an application of Artificial Intelligence tailored to vehicle usage monitoring and, in particular, to duty level detection.

III. Contrastive learning from multivariate time series

a) Data modeling: We model the CAN Bus signals transmitted by a vehicle v as multivariate time series $\mathbf{X}_v$ = $[\mathbf{x}_v^{(1)}, \mathbf{x}_v^{(2)}, \ldots, \mathbf{x}_v^{(n)}]$. Each sample $\mathbf{x}_v^{(t)}$=$[x_{v1}^{(t)}, x_{v2}^{(t)}, ..., x_{vm}^{(t)}]$ $(1 \leq t \leq n)$ is m-dimensional ($\mathbf{x}_v^{(t)} \in \mathbb{R}^m$) as it is described by a combination of m univariate (potentially correlated) SPNs.

For the sake of simplicity, hereafter we will assume that consecutive measurements $x_j^{(t)}$ and $x_j^{(t+1)}$ are acquired at a fixed time step even if real transmissions can be partly incomplete or asynchronous.

We divide the series of v's SPNs into q fixed-size, consecutive segments $\mathbf{S}_v$=$[\mathbf{s}_v^{(1)}, \mathbf{s}_v^{(2)}, \ldots, \mathbf{s}_v^{(q)}]$, where $\mathbf{s}_v^{(i)}$ are the corresponding samples.

Each sample is labeled with dl indicating the corresponding vehicle duty category. In the industrial scenario we are considering, we can assume that within a short-lasting segments (of 30s each) the duty level of a vehicle is unlikely to change (i.e., we assign one duty level per segment).

b) Representation Learning: We leverage a self-supervised pretraining step for clustering and classifying time series segments. The idea behind it is to use neural networks to learn a feature function mapping $\mathcal{F} : \mathbb{R}^{mq} \rightarrow \mathbb{R}^{md}$ that maps the samples of a multivariate segment series to a d-dimensional latent space. The generated representation is instrumental for addressing segment clustering directly in the latent space and then classification as a downstream task [11].

c) Contrastive Learning: Contrastive Learning (CL) is a particular subclass of Representation Learning, where two samples representing the same pattern should lie close together in the latent space whereas samples corresponding to different patterns should be pushed away from each other [12].

In this work, we separately analyze the SPN series acquired from vehicles of different models. We build model-specific representations where similar SPN segments are in close proximity whereas dissimilar ones are kept relatively faraway. Our assumption is that the underlying trends and the data invariances are likely to be preserved within the set of time series acquired from the same model.

We apply a time series augmentation process to generate new, modified versions of an original samples and then couple pairs of augmented samples from the same origin in a self-supervised fashion (i.e., the positive pairs). The positive pairs are well separated from negative ones, which randomly couple (likely uncorrelated) samples. To this end, the encoder of a contrastive learning architecture is trained by passing different augmentations of the vehicle segments through the encoder and a projection head, before applying a contrastive loss.

As augmentation strategy we generate convex combinations of arbitrary pairs of segments s^i and s^j by adopting the mixup strategy proposed in [9]: $\hat{s} = \lambda s^i + (1 - \lambda s^j)$, where $\lambda \in$ [0,1] is a mixing parameter determining the contribution of each segment, such that $\lambda \sim$ Beta(α, α) and $\alpha \in$ [0, ∞).

To train the neural network we use the Temporal Neighbor Coding constrastive architecture [10], in which we integrate the the mixup augmentation strategy described above.

IV. Proposed methodology

We envisage an automated approach to detect the duty levels of industrial vehicle. It entails the following steps:

[1]https://www.sae.org/ latest access: August 2023

- *SPN preparation and annotation*: The input CAN bus data is first processed to make it suitable for the next analytics steps. Preparation encompasses time series synchronization, cleaning, correlation analysis, selection of representative vehicles per model, and segment annotation for the SPNs of the representative vehicle only.
- *Contrastive pretraining*: The preprocessed SPNs acquired from vehicles of the same model are encoded using a self-supervised mixup contrastive learning approach. Clusters in the latent space group similar SPN segments together.
- *Cluster labeling*: The available duty levels annotations are propagated by leveraging intra-cluster segment similarities.
- *Duty level classification*: On the top of the labeled segments, we train model-specific classifier to predict the duty levels of unseen SPN segments.

More details on each separate step are given below.

A. SPN preparation and annotation

a) Preprocessing: CAN Bus data is tailored to the data model described in Section III. All SPNs' series are aligned and synchronized. Since SPN measurements can be missing or noisy we apply a semi-automatic cleaning step to avoid data inconsistencies. For example, when a key off event is detected, the values corresponding to the SPN *Engine Speed* are set to zero. The SPNs whose values are constant or missing for more than half of the vehicles are early pruned. We also perform a preliminary correlation analysis, based on the Pearson correlation index [13], to remove repeated or highly similar SPNs. Based on the experts' indications, the SPNs that are most relevant to the industrial use case under analysis are enumerated below.

- SPN code: 100 (Engine Oil Pressure)
- SPN code: 190 (Engine Speed)
- SPN code: 180 (Engine fuel Rate)
- SPN code: Proprietary (custom code for the Engine Percent Load)
- SPN: Proprietary (custom code for the Engine Intake Manifold 1 Pressure)

b) Identification of vehicle representatives: Due to the high heterogeneity of the input SPNs, we propose to tailor the process of automatic duty level detection to each vehicle model. For this reason, we first analyze the SPN distributions over all the multivariate series relative to each model and select one representative vehicle per model.

Separately for each SPN, we rank the corresponding measurements and identify the value range comprising both the second and third quartiles (see the example reported in Figure 1). Then, given all the considered SPNs we shortlist the vehicle whose SPN measurements maximize the coverage of the selected ranges (or, alternatively, the vehicle whose SPN values are closest to that ranges).

c) Annotation of representative vehicles' SPNs: We ask domain experts to annotate the duty levels associated with the historical SPN segments acquired from the representative vehicles. Focusing on a representative vehicle per model limits

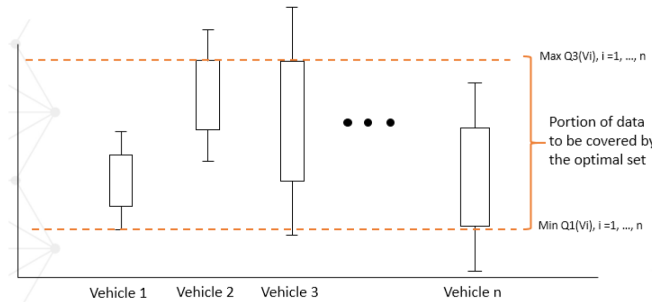


Fig. 1: Selection of per-model representative vehicle

the human effort and prevents the identification of irrelevant or misleading vehicle usage patterns.

As an example, Table I reports the rules recommended by the domain expert for two representative vehicles (A and B). Notice that the rules depend on the actual vehicle model and usage characteristics.

Defining ad hoc rules separately for each vehicle can be extremely expensive. For this reason, as discussed below, we leverage a self-supervised clustering strategy to find segments (of the same or other vechicles) with the same duty level.

B. Contrastive pretraining

Separately for each model we build a contrastive representation of the SPN segments according to the constrastive strategy described in Section III. Notice that the encoding phase is self-supervised, i.e., it does not require any human supervision.

C. Cluster labeling

Each cluster is labeled according to the duty levels assigned by the domain expert to the representative vehicles. The purpose is to propagate the information about the duty level associated with all the segments within the cluster. Hence, we assume that a cluster has at most one associated duty level. If this is not the case, we assign the most representative one by majority voting.

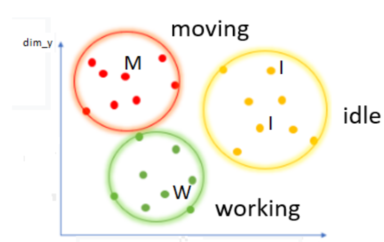


Fig. 2: Toy example of cluster labeling

Let us consider the toy example in Figure 2. It shows a 2-dimensional representation of the latent space generated on top of the SPN segments of an arbitrary vehicle model. Given the duty levels that were manually assigned by the domain experts to the SPN segments of the representative vehicle (respectively denoted by M=Moving, W=Working, I=Idle), we can automatically label, by majority voting, all the segments in

Test Vehicle	Type	Duty level	Rule
A	Excavator	Inactive	if the vehicle is characterized by engine speed equal to 0 rpm
		Idle	if its engine speed is between 0 and 1136 rpm, whereas the engine coolant temperature is higher than 63°C.
		Moving	if the engine speed is greater than 1136 rpm, while the engine intake manifold 1 pressure is lower than 37.6 bar.
		Digging	if the engine speed is higher than 1136 rpm
B	Tandem Roller	Inactive	if the engine speed is equal to 0 rpm.
		Idle	if the engine speed is between 0 and 902.5 rpm
		Moving	if the engine speed is is between 902.5 and 1772 rpm and the engine percent load is lower than 48%
		Working	if the engine speed is higher than 1772 rpm and the engine percent load is lower than 55.6%
		High workload	if the engine percent load is higher than 55.6%

TABLE I: Duty levels generated by domain experts for two representative vehicles (A and B)

the clusters. Here we assume that the data invariances captured in the latent space are reflected by the duty levels. Therefore, intra-cluster segment similarities are used to propagate the human annotations.

Whether a cluster does not contain any humanly labeled segment, it will be no longer considered for this automatic labeling stage.

D. Duty level classification

On top of the labeled segments we train a classifier $\mathcal{C}^m$ that automatically predicts the duty level of a new, unlabeled segments belonging to any vehicle of that particular model. Classifier training is performed separately for every model m present in the training data.

Given an arbitrary segment s of the multivariate series of vehicle v^m of model m, we define the following model-specific predictive function $\mathcal{C}^m : \mathbf{S}^m \rightarrow DL$ maximizing P($dl(s)| \ s, DL, \mathbf{T}$), where $\mathbf{T}$ is the training set consisting of annotated segments and DL is the set of duty levels in $\mathbf{T}$.

V. Results

A. Dataset

To simulate the application of the proposed approach in a real industrial scenario, Tierra Spa has provided an anynomized proprietary dataset consisting of Blend+ SPNs from 6 test vehicles equipped with AM53 CAN Buses. From each vehicle the dataset stores 19 SPNs among which engine speed, percent load, fuel rate, and coolant temperature, for a 3-month period. SPN measurements are aggregated over 30s time windows. The SPN segments are annotated by domain experts will be used as a ground truth. The full SPN list is given in Table II.

B. Metrics

a) Clustering: We analyze clustering performance in terms of *Silhouette coefficient* [13]. It measures how similar a sample is both to its own cluster (cohesion) and compared to other clusters (separation). The silhouette value vary from -1 to +1. Low or negative values are a clue of low-quality clustering, where high values (close to +1) indicate high cohesion and separation.

b) Classification: We evaluate classification performance in terms of precision and recall of the target duty level categories. They respectively indicate the proportion of SPN segments assigned to a given duty level that have been correctly predicted and the proportion of segments of a given duty level that have been detected.

Since the classification problem is inherently imbalanced in the training phase we apply undersampling of the majority class (Idle).

C. Algorithms

We compare the performance of the proposed approach, based on combination of mixup augmentation and TNC contrastive learning, with that of

- two established clustering algorithms previously adopted in [5] to detect duty levels from heavy-duty vehicle, i.e., the classical partitive *K-Means* algorithm [14], K-Medoids [6], and the established density-based *DBScan* algorithm [7].
- *Mixing Up*, a recently proposed contrastive framework based on mixup normalized temperature-scaled cross entropy loss [15].

For the classification task we consider as baseline methods the implementations of LSTM and GRU networks available in the PyTorch library [16].

D. Hardware

We run the experiments based on classical data science and machine learning models using an Intel(R) Core(TM) i7-8250U machine equipped with 16 GB of RAM and running Windows 10 64-bit. For training the Deep Learning models we also used a machine equipped with an NVIDIA® V100 GPU with 32 GB of VRAM.

E. Hyperparameters settings

For all clustering algorithms we varied the number of desired clusters between 2 and 10. For the Mixing Up and TNC approaches we performed a grid search by varying the encoding size, the number of epochs, the batch size, the learning rate, the decay, and the α parameter for the MixUp augmentation step.

SPN description	SPN number
Actual Engine - Percent Torque	513
Aftertreatment Diesel Exhaust Fluid Tank Level	Proprietary Code
Aftertreatment Diesel Exhaust Fluid Tank Temperature	Proprietary Code
Aftertreatment Diesel Oxidation Catalyst Intake Gas Temperature	Proprietary Code
Battery Potential - Power Input	Proprietary Code
Battery voltage	Proprietary Code
Engine Coolant Level	Proprietary Code
Engine Coolant Temperature	110
Engine Fuel Rate	180
Engine Fuel Temperature	Proprietary Code
Engine Hours	241
Engine hours (c1)	Proprietary Code
Engine Oil Level	98
Engine Oil Pressure	100
Engine Oil Temperature	175
Engine Percent Load	Proprietary Code
Engine Speed	190
Intake Manifold 1 Pressure	Proprietary Code
Intake Manifold 1 Temperature	105

TABLE II: Dataset SPNs.

F. Clustering results

Table III reports the results achieved on the test dataset for three representative cluster sizes K (3, 5, and 7). For all algorithm clustering performance are optimal for K=5. Notice that this number corresponds to the number of duty levels indicated by the domain experts during the annotation of the representative vehicles' SPNs. Contrastive approaches outperform the traditional clustering methods for most of the tested configurations. Combining MixUp augmentation with TNC constrastive loss yields the best performance, especially when $K > 2$.

Algorithm	K=3	K=5	K=7
K-Means [14]	0.394	0.466	0.461
K-Medoids [6]	0.401	0.475	0.461
DBScan [7]	0.418	0.451	0.448
Mixing Up [10]	0.491	0.646	0.598
TNC+MixUp (our)	**0.502**	**0.695**	**0.637**

TABLE III: Clustering results by varying the number k of desired clusters. For each k value the best score is written in boldface.

G. Classification results

Table IV compares the duty levels assignments made by our approach, the Mixing Up contrastive strategy, and by K-Means clustering combining a LSTM/GRU network. For each duty level we report the number of assigned SPN segments based on the predictions made separately for each method as well as the number of expected ones. We are interested in minimizing the gap (absolute difference) between the actual and predicted numbers of assignments per duty level.

All the tested approaches achieve high-quality results on *Off* due to the inherent simplicity of detecting the underlying SPN patterns. Conversely, *Idle* and *Moving* turn out to be the most challenging duty levels to detect. Most predictors tend to overestimate *Moving* duties (low precision) at the expense of *Idle* ones (low recall). Our approach, which performs best on 4 duty levels out of 5, achieves the best precision-recall trade-off for the most challenging duty levels.

H. Qualitative SPN series analysis

Figure 3 shows the sample distributions over clusters for 4 representative SPNs. The result has been achieved using the proposed approach with a number of cluster equal to 5. When the Engine Speed (SPN 190) is set to 0, the *Off* duty level is easy to detect (see Cluster 4). Conversely, the Moving (Cluster 1) and Working (Cluster 3) duty levels are not well separated according to the Engine Speed, Oil Pressure, and Percent Load. This likely causes the classification errors observed in the quantitative results (see Section V-G).

VI. Conclusions and future works

This paper explored the use of contrastive learning to automatically detect duty levels of industrial vehicles. A self-supervised constrastive pretraining stage allows us to capture the similarities among SPN segments of vehicles of the same model without requiring a large set of human annotations. We then propose to empirically identify a representative vehicle per model, provide experts with a limited subset of SPNs to analyze, and then propagate the humanly generated annotation within each cluster. The preliminary results achieved on the SPNs acquired from the test vehicles confirm the applicability of the proposed strategy in a real industrial scenario.

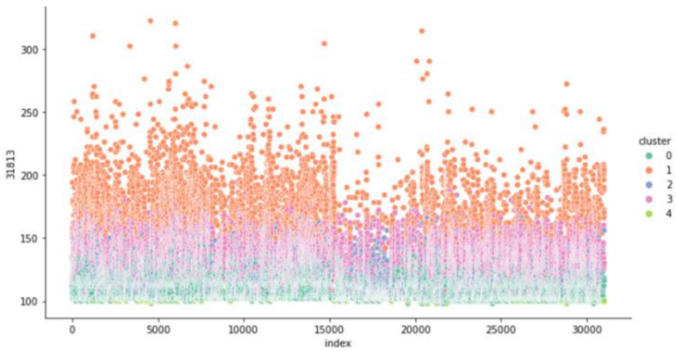
As a future work, we plan to design a pipeline to incrementally update vehicle usage patterns. Furthermore, we also aim at integrating contextual information about vehicles provided by geospatial databases and GPS devices.

Acknowledgment

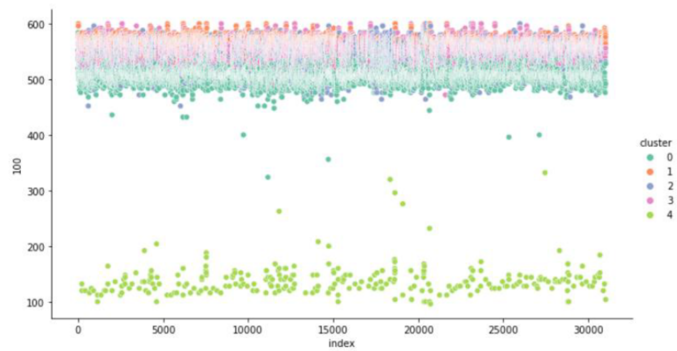
This study was carried out within the MICS (Made in Italy – Circular and Sustainable) Extended Partnership and received funding from Next-GenerationEU (Italian PNRR – M4 C2, Invest 1.3 – D.D. 1551.11-10-2022, PE00000004). This study

Duty Level	Ground Truth	K-Means+LSTM	K-Means+GRU	Mixing Up	TNC+MixUp (our)
OFF	297	305 (+8)	301 (+4)	299 (+2)	**298 (+1)**
IDLE	13146	9710 (-3436)	9952 (-3194)	**11809 (-1337)**	11008 (-2138)
MOVING	8232	12524 (+4292)	12842 (+4610)	4231 (-4001)	**11791 (+3559)**
WORKING	5636	9230 (+3594)	9235 (+3599)	10114 (+4508)	**4424 (-1212)**
OVERLOADED	3705	3450 (-255)	3390 (-315)	4563 (+1858)	**3495 (-210)**

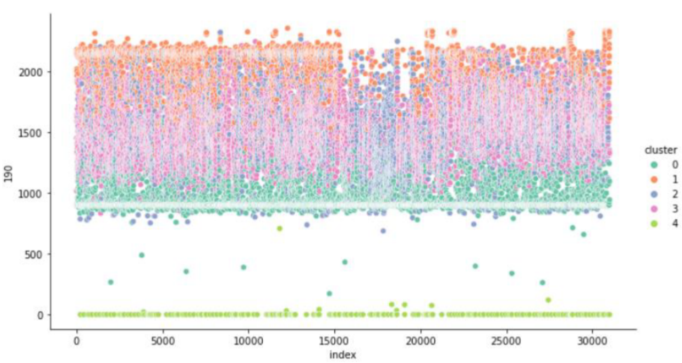
TABLE IV: Classification results. Number of SPN segments assigned to each duty level. The gap between the numbers of the actual and expected assignments (i.e., the Ground Truth) is reported in brackets. The best performance (i.e., the result with minimal absolute gap) is written in boldface.



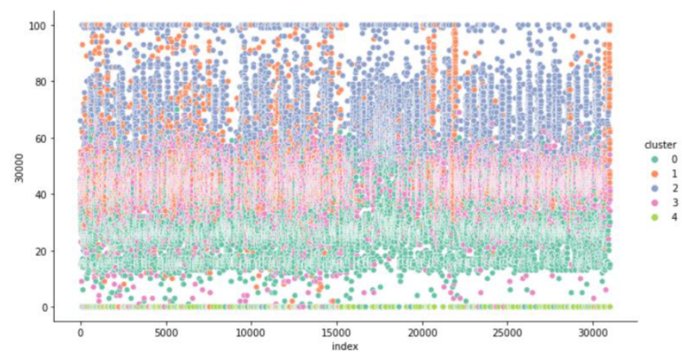
Intake Manifold Pressure



Engine Oil Pressure



Engine Speed



Engine Percent Load.

Fig. 3: SPN samples distribution over clusters (TNC+MixUp, K=5).

was also partially carried out within the FAIR (Future Artificial Intelligence Research) and received funding from Next-GenerationEU (Italian PNRR – M4 C2, Invest 1.3 – D.D. 1555.11-10-2022, PE00000013). This manuscript reflects only the authors' views and opinions, neither the European Union nor the European Commission can be considered responsible for them.

REFERENCES

[1] K. McCord, *Automotive Diagnostic Systems: Understanding OBD I and OBD II*. CarTech Inc, 2011.

[2] L. A. Schmidt and L. Riegger, "Automatic detection of machine status for fleet management," Oct. 4 2012, uS Patent App. 13/341,500.

[3] A. Holst, J. Bae, A. Karlsson, and M.-R. Bouguelia, "Interactive clustering for exploring multiple data streams at different time scales and granularity," in *Proceedings of the Workshop on Interactive Data Mining*, ser. WIDM'19. New York, NY, USA: Association for Computing Machinery, 2019. [Online]. Available: https://doi.org/10.1145/3304079.3310286

[4] S. Buccafusco, A. Megaro, L. Cagliero, F. Vaccarino, L. Salvatori, and R. Loti, "Profiling industrial vehicle duties using CAN bus signal segmentation and clustering," in *Proceedings of the Workshops of the EDBT/ICDT 2021 Joint Conference, Nicosia, Cyprus, March 23, 2021*, ser. CEUR Workshop Proceedings, C. Costa and E. Pitoura, Eds., vol. 2841. CEUR-WS.org, 2021. [Online]. Available: http://ceur-ws.org/Vol-2841/DARLI-AP_7.pdf

[5] S. Buccafusco, L. Cagliero, A. Megaro, F. Vaccarino, R. Loti, and L. Salvatori, "Learning industrial vehicles' duty patterns: A real case," *Comput. Ind.*, vol. 145, p. 103826, 2023. [Online]. Available: https://doi.org/10.1016/j.compind.2022.103826

[6] X. Jin and J. Han, *K-Medoids Clustering*. Boston, MA: Springer US, 2010, pp. 564–565. [Online]. Available: https://doi.org/10.1007/978-0-387-30164-8_426

[7] M. Ester, H.-P. Kriegel, J. Sander, and X. Xu, "A density-based algorithm for discovering clusters in large spatial databases with noise," in *Proceedings of the Second International Conference on Knowledge Discovery and Data Mining*, ser. KDD'96. AAAI Press, 1996, p. 226–231.

[8] J. Paparrizos and L. Gravano, "K-shape: Efficient and accurate clustering of time series," *SIGMOD Rec.*, vol. 45, no. 1, p. 69–76, jun 2016. [Online]. Available: https://doi.org/10.1145/2949741.2949758

[9] H. Zhang, M. Cissé, Y. N. Dauphin, and D. Lopez-Paz, “mixup: Beyond empirical risk minimization,” in *6th International Conference on Learning Representations, ICLR 2018, Vancouver, BC, Canada, April 30 - May 3, 2018, Conference Track Proceedings*. OpenReview.net, 2018. [Online]. Available: https://openreview.net/forum?id=r1Ddp1-Rb

[10] S. Tonekaboni, D. Eytan, and A. Goldenberg, “Unsupervised representation learning for time series with temporal neighborhood coding,” in *International Conference on Learning Representations*, 2021. [Online]. Available: https://openreview.net/forum?id=8qDwejCuCN

[11] J. PÃ¶ppelbaum, G. S. Chadha, and A. Schwung, “Contrastive learning based self-supervised time-series analysis,” *Applied Soft Computing*, vol. 117, p. 108397, 2022. [Online]. Available: https://www.sciencedirect.com/science/article/pii/S1568494621011558

[12] P. H. Le-Khac, G. Healy, and A. F. Smeaton, “Contrastive representation learning: A framework and review,” *IEEE Access*, vol. 8, pp. 193 907–193 934, 2020.

[13] P. Tan, M. S. Steinbach, A. Karpatne, and V. Kumar, *Introduction to Data Mining (Second Edition)*. Pearson, 2019. [Online]. Available: https://www-users.cse.umn.edu/\%7Ekumar001/dmbook/index.php

[14] J. MacQueen, “Some methods for classification and analysis of multivariate observations,” in *Proceedings of the 5th Berkeley Symposium on Mathematical Statistics and Probability - Vol. 1*, L. M. Le Cam and J. Neyman, Eds. University of California Press, Berkeley, CA, USA, 1967, pp. 281–297.

[15] K. Wickstrøm, M. Kampffmeyer, K. Ø. Mikalsen, and R. Jenssen, “Mixing up contrastive learning: Self-supervised representation learning for time series,” *Pattern Recognition Letters*, vol. 155, pp. 54–61, mar 2022.

[16] A. Paszke, S. Gross, F. Massa, A. Lerer, J. Bradbury, G. Chanan, T. Killeen, Z. Lin, N. Gimelshein, L. Antiga, A. Desmaison, A. Kopf, E. Yang, Z. DeVito, M. Raison, A. Tejani, S. Chilamkurthy, B. Steiner, L. Fang, J. Bai, and S. Chintala, “Pytorch: An imperative style, high-performance deep learning library,” in *Advances in Neural Information Processing Systems 32*, H. Wallach, H. Larochelle, A. Beygelzimer, F. d'Alché-Buc, E. Fox, and R. Garnett, Eds. Curran Associates, Inc., 2019, pp. 8024–8035. [Online]. Available: http://papers.neurips.cc/paper/9015-pytorch-an-imperative-style-high-performance-deep-learning-library.pdf