



HAL
open science

Unsupervised Representation Learning for Smart Transportation

Thabang Lebese, Cécile Mattrand, David Clair, Jean-Marc Bourinet, François Deheeger

► **To cite this version:**

Thabang Lebese, Cécile Mattrand, David Clair, Jean-Marc Bourinet, François Deheeger. Unsupervised Representation Learning for Smart Transportation. 22nd International Symposium on Intelligent Data Analysis (IDA 2024), Apr 2024, Stockholm, Sweden. pp.15-27, 10.1007/978-3-031-58553-1_2. hal-04644400

HAL Id: hal-04644400

<https://cnrs.hal.science/hal-04644400>

Submitted on 10 Sep 2024

HAL is a multi-disciplinary open access archive for the deposit and dissemination of scientific research documents, whether they are published or not. The documents may come from teaching and research institutions in France or abroad, or from public or private research centers.


L'archive ouverte pluridisciplinaire **HAL**, est destinée au dépôt et à la diffusion de documents scientifiques de niveau recherche, publiés ou non, émanant des établissements d'enseignement et de recherche français ou étrangers, des laboratoires publics ou privés.



Distributed under a Creative Commons Attribution 4.0 International License



Unsupervised Representation Learning for Smart Transportation

Thabang Lebese^{2,3}, Cécile Mattrand¹, David Clair¹,
Jean-Marc Bourinet², and François Deheeger³

¹ Université Clermont Auvergne, CNRS, Institut Pascal, Clermont-Ferrand, France

² Université Clermont Auvergne, CNRS, LIMOS, Clermont-Ferrand, France

³ Manufacture Française des Pneumatiques Michelin, Clermont-Ferrand, France

thabang.lebese@sigma-clermont.fr

Abstract. In the automotive industry, sensors collect data that contain valuable driving information. The collected datasets are in multivariate time series (MTS) format, which are noisy, non-stationary, lengthy, and unlabeled, making them difficult to analyze and model. To understand the driving behavior at specific times of operation, we employ an unsupervised representation learning method. We present Temporal Neighborhood Coding for Maneuvering (TNC4maneuvering), which aims to understand maneuverability in smart transportation data via a use-case of bivariate accelerations from three operation days out of 2.5 years of driving. Our method proves capable of extracting meaningful maneuver states as representations. We evaluate them in various downstream tasks, including time-series classification, clustering, and multilinear regression. Moreover, we propose methods for pruning the sizes of representations along with a window-size optimizing algorithm. Our results show that TNC4maneuvering has the capacity to generalize over longer temporal dependencies, although scalability and speedup present challenges.

Keywords: Multivariate Time-series · Representation learning · Classification · Clustering · Regression

1 Introduction

Modern transportation is now equipped with more sensors than ever before, making the term “smart transportation” more appropriate. This improves efficiency, security, and helps keep up with ever-changing environmental and government regulations. The sensors collect large amounts of data during operational hours from various parts of the vehicle, including but not limited to engine performance, external conditions, and tire states. These sensors measure different driving behaviors and states as a function of operational time or mileage. For example, the Global Positioning System (GPS) collects geographical data, while

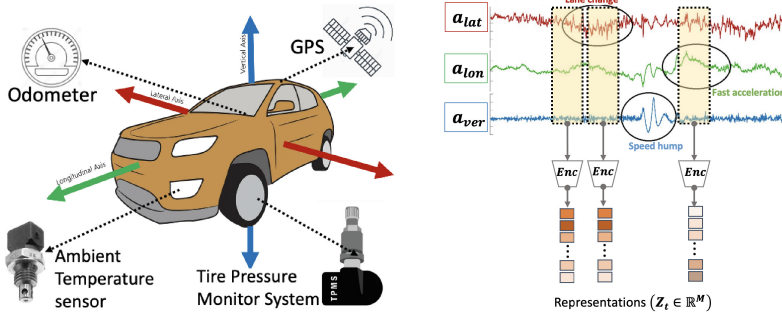


Fig. 1. Left: Smart vehicle with multiple sensors. Right: Encoding multivariate acceleration signals.

sensors inside the odometer read mileage coverage. Ambient temperature sensors measure external driving temperature conditions, and Tire Pressure Monitoring Systems (TPMS) measure the temperature and pressure inside the tires over time as depicted in Fig. 1 (left). The collected data is often high sampled, lengthy, noisy, and impractical to label. As a result, it is challenging to relate underlying behaviors/states to other datasets. This highlights the need for representation learning methods, which can output vectorial summaries from multi-sensory inputs of variables over a given time window. The resulting vectors are descriptors of latent behaviors of the physical system as illustrated with the three input accelerations in Fig. 1 (right). These accelerations are expected to describe different physical maneuvers of a vehicle, rendering them indirectly related. Vehicle maneuvers are rather directly related to driving behavior because driving generally involves three main actions: controlling the steering wheel, stepping on the accelerator, and pressing the brake pedal.

The three accelerations in Fig. 1 (right) are lateral acceleration (a_x), longitudinal acceleration (a_y), and vertical acceleration (a_z), which pertain to steering actions, accelerator and brake pedal usage, and up-and-down movements experienced by a vehicle, respectively. Following our work in [10] on simulated datasets, we here present Temporal Neighborhood Coding for Maneuvering (TNC4maneuvering), an unsupervised representation learning method to extract states for understanding maneuverability. TNC4maneuvering is robust enough to identify and locate temporal transitions between states without any prior knowledge about labels of the states. It employs contrastive learning for its ability to handle long, noisy, and non-linear MTS datasets without the need for reconstruction, significantly reducing computation costs. Furthermore, as an improvement we propose two offline pruning methods for optimizing the sizes of learned representations as well as a window-size selection algorithm. These are useful in the absence of expert knowledge. We evaluate the obtained latent representations by assessing key performance indicators (KPIs) of downstream tasks, namely clustering, classification, and multi-linear regression based on three different driving days to observe the generalization and scalability of our method. To sum up,

our contribution is three-fold: 1) TNC4maneuvering, an unsupervised representation method for understanding maneuverability in smart transportation, 2) an offline window-size selection and optimization method that avoids treating it as an additional hyper-parameter, and 3) two offline representation pruning strategies for optimizing dimensions of representations.

2 Related Work

Unsupervised representation learning has excelled in various MTS tasks, but its application to smart transportation MTS datasets is generally limited if any. Existing attempts, such as the application of Bag of Words (BoW) model in [1], led to a representation like output with focus on classifying aggressive driving maneuvers only. Such approaches do not generalize well making them incapable of other alternative subsequent tasks. Recent works explore contrastive learning for representation learning by contrast of similar and dissimilar instances. Examples include [3, 4, 6, 7, 9, 13, 15, 18, 19]. Notable exceptions are [17], which disentangles seasonal-trend features using time and frequency domains, and [2], which jointly learns contextual, temporal, and transformation consistencies, later applying them to classification, forecasting, and anomaly detection tasks. To the best of our knowledge this is the first work reporting the use of pure unsupervised representation learning on acceleration MTS, specifically for understanding vehicle maneuvering with capabilities of multitask downstream.

3 Method

In [10], we explored three state-of-the-art approaches [6, 13, 15] that use contrastive learning on simulated MTS datasets to extract underlying states. Building on these findings, we further enhanced TNC by incorporating offline window-size selection, latent space tuning by pruning, and an exponentially dilated convolutional neural network (CNN) encoder. In our extension, here dubbed TNC4maneuvering, introduces an unsupervised representation learning framework to extract underlying driver maneuver states from acceleration signals of a vehicle. Our encoder is specifically designed to efficiently extract maneuver states.

TNC4maneuvering: The backbone of our method is a non-linear composition function encoder (*Enc*), typically a deep neural network, taking a static window $W_t \in \mathbb{R}^{F \times \delta}$ centered at time t with sub-length δ and F number of features. A tuple of samples, an anchor (W_t), a positive (W_l) and negative (W_k) windows are sampled from input MTS where each window $W_{t,l,k} \in \mathbb{R}^{F \times \delta}$ generates a representation vector $Z_{t,l,k} \in \mathbb{R}^M$, where $M \ll F \times \delta$ is the size of the representation vector. W_l and $W_t \in N_t$ share the same neighborhood centered at t , while $W_k \in \bar{N}_t$ is at a distant non-neighbourhood. The semantic similarities and dis-similarities between windows is controlled by the temporal neighborhood around W_t . This region is defined as a region where acceleration signals

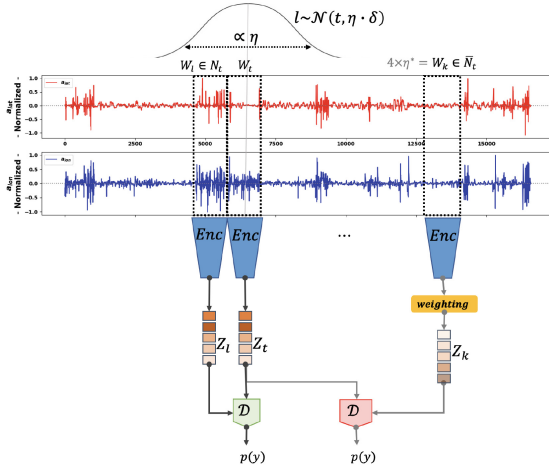


Fig. 2. Overall TNC4maneuvering framework: Encoder: $Enc(W_t) \in \mathbb{R}^{F \times \delta}$, outputs representations $Z_t \in \mathbb{R}^M$, with Discriminator: $\mathcal{D}(Z_t, Z_{l \vee k}) \in [0, 1]$.

are relatively stationary compared to their pre and post-windows, they are therefore assumed to be generated from the same underlying maneuvering state. The objective function (1) is a partial contrastive loss that learns signals via encoding and evaluates them using a Discriminator (\mathcal{D}) that identifies representations with similar underlying maneuverings.

$$\begin{aligned} \mathcal{L} = & -\mathbb{E}_{W_t \sim X} \left[\mathbb{E}_{W_l \sim N_t} \left[\log (\mathcal{D}(Z_t, Z_l)) \right] \right. \\ & \left. + \mathbb{E}_{W_k \sim \bar{N}_t} \left[w_t \log (\mathcal{D}(Z_t, Z_k)) + (1 - w_t) \log (1 - \mathcal{D}(Z_t, Z_k)) \right] \right] \quad (1) \end{aligned}$$

The unit root test, Augmented Dickey-Fuller (ADF)¹ is used for determining relative stationarity regions. Furthermore, the objective function is weighted with (w_t) and $(1 - w_t)$, an ideas from Positive-Unlabeled (PU) learning to counter potential sampling bias in the contrastive objective. This compensates for negative samples drawn from outside of the neighborhood which may in fact be similar to those of an anchor window. The overall framework is depicted in Fig. 2, details on this framework can be found in [10, 15].

4 Experiments

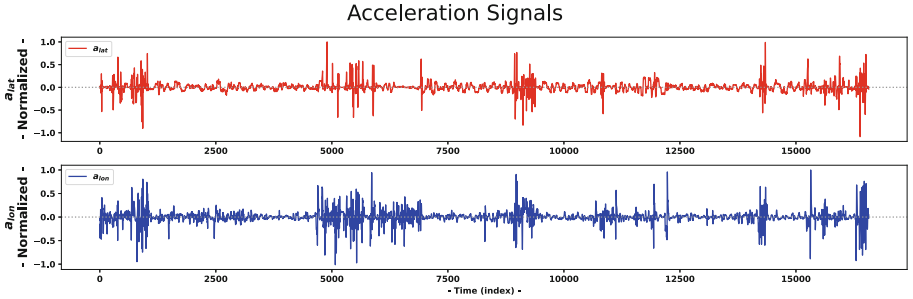
TNC4maneuvering extends [10], it is implemented in PyTorch framework (v1.12.1) and the source code is available on GitHub². All experiments are conducted using a single Nvidia Tesla P40 GPU with CUDA 11.2.152. All datasets

¹ arch.unitroot.ADF.

² <https://github.com/ThabangDLebesse/tnc4maneuvering>.

Table 1. Selected different driving days showing mileage coverage, time taken and corresponding length of observations.

Operational day	Mileage (Km)	Time (Mins)	No. observations
2018/10/23 (One D _s)	20	584	1957
2019/11/28 (One D _l)	665	932	16568
2018/10/24-31 (Eight D)	499	5260	19273

**Fig. 3.** Normalised bivariate (a_{lat}, a_{lon}) acceleration for One D_l operation day.

are normalized, and the evaluations include three downstream tasks: clustering, classification, and multi-linear regression.

4.1 Vehicle Acceleration Datasets

Vehicle maneuvering is an automotive problem that is central to understanding driving behavior from sensory signals. Our use-case vehicle is a Peugeot 208 model used as a fleet car, where operation time is accumulated as an amount of time where driving activities are collected by different sensors. In this particular work we focus only on the two accelerations, namely the lateral (a_{lat}) which is the effective measure of cornering (negative is for right turning, 0 is straight line or breaking and positive is left turning) and the longitudinal acceleration (a_{lon}) where the straight line acceleration (negative braking, 0 is constant speed and positive is accelerating).

Both accelerations are reported as a fraction of the gravitational acceleration (ms^{-2}). Analysing different vehicle acceleration behaviours on different driving days can help to understand different maneuvering behaviours of a vehicle over time. We consider three bivariate sample signals with different dates, signal lengths, total covered time and mileage covered as depicted in Table 1.

As a pre-processing stage, we perform a data normalization to avoid statistical biases that can lead to misinterpretation of the encoded results. Both input features are normalized such that each $X_i = x_i/x_{\max} \in [-1, 1]$, for $x_{\max} = \max|x_i|, i = \{1, 2\}$, preserving the zero values on each feature. Our bivariate acceleration datasets are extracted from different driving days from

the overall 2.5 years of driving. From these datasets, we extract one short day (One D_s), one long day (One D_l) and eight days (Eight D) that is inclusive of (One D_s). These days correspond to separate driving dates with different mileage coverage, time overall required time and total corresponding observation lengths. Figure 3 depicts the normalized accelerations of the One D_l operational day.

4.2 Encoder Details

From [10], we replace the Bidirectional Recurrent Neural Network (BiRNN) with an exponentially dilated Convolutional Neural Network (CNN) with causality as our backbone encoder. Exponentially dilated convolutions efficiently capture long-range dependencies without increasing network depth. Our CNN encoder is tailored for encoding time series data into a lower-dimensional vector space, particularly suited for datasets with extended temporal dependencies and characteristics like non-Gaussianity, intermittency, non-periodicity, and so on. It here comprises of three stacked convolutional layers, each using dilated convolutions to extract inter-temporal features. The dilation parameter exponentially increases (2^i for the i -th layer), while fixed-size filters ($f \in \mathbb{N}$) preserve temporal resolution and alignment. The output undergoes global max pooling, compressing temporal information into a fixed-size vector. This result is flattened and processed by a linear layer, further reducing the dimensionality to produce an encoding of size M , serving as a compressed representation based on a window size W_t .

Our encoder design offers flexibility by allowing customizable encoder sizes (M), this is to say it incorporates a classification component for compatibility with subsequent classification tasks. This design choice provides several advantages, including enhanced generalization for downstream tasks and easy pruning options. Each exponentially dilated convolution layer encodes data through a convolution operation with dilation defined as:

$$F(s) = (W_t \star_d f)(s) = \sum_{i=0}^{k-1} f(i)W_t^{s-d \cdot i}, \quad (2)$$

where $F(s)$ represents the computed output on each layer for samples $s \in W_t$ ($\in \mathbb{R}^{F \times \delta}$), with a dilation rate of d , filter size k , and $(s - d \cdot i)$ accounting for historical direction. Other hyperparameters include a batch size of 5, a learning rate of 1×10^{-5} , and a weight decay of 1×10^{-4} , using the Adam optimizer. We perform a train/test data split without validation, training epochs are limited to 30, 20, and 10 epochs for datasets One D_s , One D_l , and Eight D, respectively.

4.3 Hyperparameter Tuning

In [10], we recognized the need for further tuning two hyperparameters: the window size (W_t) and the latent space dimension (M), while keeping the PU learning parameter fixed at $w_t = 0.05$.

Window Selection: An appropriate window size should capture important information about maneuvering states without being too wide or too narrow.

Table 2. Cross data performances on multi-task downstream before pruning.

TNC4maneuvering (Before pruning)							
Operational day	W_t	Classification		Clustering		Regression	
		AUPCR	Accuracy	Silhouette	DBI	R^2	Loss
One D_s	250	0.988	98.82	0.715	0.492	-0.290	2.075
One D_l	250	0.936	84.86	0.372	1.014	0.326	1.023
Eight D	250	0.976	84.47	0.320	1.202	0.288	1.255

Determining a suitable window size can be achieved by relying on expert knowledge or by treating it as a hyperparameter. We here combine two offline methods for selecting a suitable window-size, 1) we examine numerical first order derivatives of acceleration signals of the window-size. If the derivative is constant, this indicates no state change; otherwise, a different state. This is simultaneously applied on both a_{lat} and a_{lon} , with windows non-overlapping. The numerical gradients within each window are approximated using `numpy.gradient`³ approximation, where interior⁴ and end⁵ points are approximated differently as the window size increases until a predefined necessary condition is satisfied. For sufficiency, 2) is employed using the Augmented Dicky-Fuller (ADF)⁶ test with a p -value threshold similar to that in TNC4maneuvering encoder. As a result, it was found that window sizes shorter than 250 do not contain enough non-stationarity, especially in One D_l and Eight D. Therefore, we determined that window size of $W_t = 250$ ($\equiv 4.2$ min of driving) is suitable. We use this window size in all experiments, for instances where the window size is larger than the sampled size, padding with zeros is applied. On the other hand, a downside of this window selection method is that gradients are prone to total samples evaluated compared to statistical variance.

Optimizing Representation Size: Determining the optimal size M for $Z_t \in \mathbb{R}^M$ is a challenging open question in representation learning. A larger encoding size captures more information but risks adding irrelevant details, affecting interpretability. Conversely, a smaller size may lead to insufficient encoding and reduced generalizability. Achieving the right balance is crucial. We propose two methods for selectively removing unnecessary details from representations, a technique referred to as latent space pruning. Initially setting $M \in \mathbb{N}$, we obtain the optimal pruned $m \leq M$, $m \in \mathbb{N}$ using two proposed methods: 1) Pearson Cor-

³ [numpy.gradient package](#).

⁴ Interior points: $(f(x+h) - f(x-h))/2h$, for evenly spaced ($h = 1$).

⁵ End points: $(f(x+h) - f(x))/h$ and $(f(x) - f(x-h))/h$, for evenly spaced ($h = 1$).

⁶ $ADF(W_t)$, if p -value > 0.01 signals is non-linear, else linear..

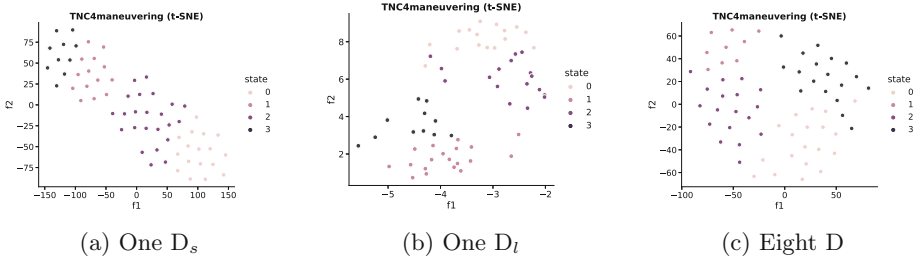


Fig. 4. t-SNE visualization of three representations of accelerations, before pruning.

relation Coefficient (PCC) [5], this method eliminates highly linearly correlated representations with a preset absolute correlation threshold of 0.7, resulting in representations of size m_1 and 2) Principal Component Analysis (PCA) [8], it utilizes a cumulative explained variance with preset threshold of 0.95 to determine m_2 , the number of components to retain. This threshold identifies the size (m_2) of the representations required to achieve it, and these representations are considered important.

4.4 Evaluation

In order to evaluate the performance of TNC4maneuvering, we evaluate three downstream tasks namely, time-series classification, clustering, and multi-linear regression across our datasets.

Classification: In this subsequent task, we employ a linear classifier due to its effectiveness in separating representations in high dimensions, assuming well-separated representations. In the TNC4maneuvering model, setting the parameter (*classify = True*) triggers the classification task. Encodings are input to a classifier comprising a dropout layer to prevent overfitting and a linear layer mapping the encodings to predefined maneuver output classes (n_{classes}) for classification. We evaluate using prediction accuracy and the area under the precision-recall curve (AUPRC) score, specifically suitable for imbalanced classification settings. The classification algorithm learns relationships between representations and predefined maneuver labels (defined in Sect. 5), facilitating accurate prediction and categorization of maneuvering states.

Clustering: Clustering of representations assesses their separability in the latent space using k-means [12], offering insights about resulting encoding properties with predefined maneuver labels (defined in Sect. 5). We employ two metrics for evaluation: the Silhouette score and Davies-Bouldin Index (DBI). The Silhouette score measures the similarity of an encoding within its assigned cluster

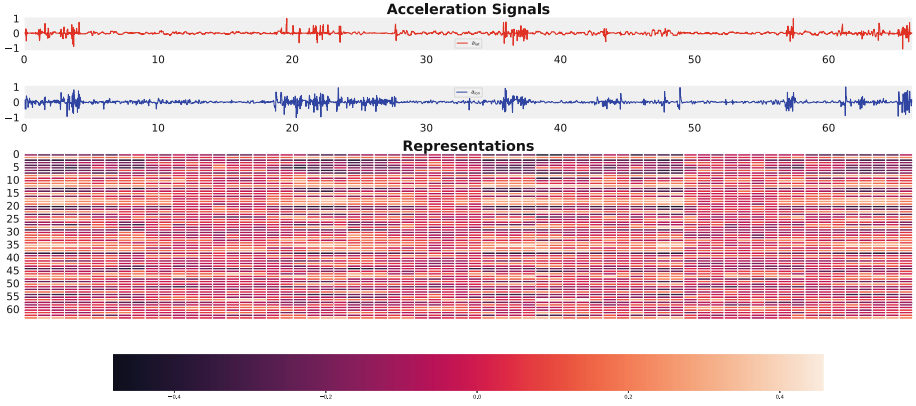


Fig. 5. One D_t accelerations in top and corresponding vector representations of size 64 encoded with a static window-size $W_t = 250$.

versus adjacent clusters, ranging from $[-1, 1]$. A higher score implies better cohesion. The DBI assesses both intra-cluster coherence and inter-cluster separation, with a lower score indicating better clusterability. Identified clusters in clustered representations are expected to reflect similar characteristics related to vehicle maneuver behavior.

Regression: In this subsequent task, peaks and valleys also known as turning points are collected. By taking consecutive differences between turning points and their square sums, quantifies their magnitudes in each window. This results to a vector $X_{man} \in \mathbb{R}^{M \times 1}$ as a summary. On the other hand, the resultant vector should offer insights into the intensity and characteristics of extrema fluctuations found in the datasets. We assume a linear mapping as a first trial where a vector $X_{man} \in \mathbb{R}^{M \times 1}$ is regressed by multivariate representations $Z \in \mathbb{R}^M$, although our perspective would be to propose a non-linear one. A train-test (70/30) data split is performed, as evaluation coefficient of determination (R^2) and learning loss are used.

Representations: Visualized representations against acceleration signals over time enhances the understanding and interpretation of extracted maneuver state and how they are modeling in the latent space ($Z \in \mathbb{R}^M$). This visual metric is crucial for comprehending vehicle maneuvering as it provides insights into maneuver behavior through visualization, facilitating the recognition of changes in maneuver states over time. Capturing these changes clearly enables deeper insights into the severity or gentleness of driver maneuvers.

5 Results and Discussion

Results Before Pruning. This section presents results of the subsequent downstream tasks before pruning. Table 2 shows task performances across all datasets before pruning. The three subsequent ML tasks on three different operation days exhibit variations. Linear regression performs the least consistently well, indicating that localized manually extracted maneuver behaviors are not linearly explained by representations. A perspective would be to resort to a non-linear mapping to better link the proposed representations with the quantity interest or further improve the quality of the representations. Overall, classification task perform rather well based on the AUPCR and accuracy scores. In Fig. 4 are the t-SNE [11] visualizations of representations of each dataset. In each visualization, each point in the plot is a 64 dimensional representation from a window-size of 250, where colors indicate different maneuvering states. With no prior domain knowledge on maneuver states, we propose a statistical approach serving as ground truth unlike in the works of authors [14]. We therefore label each dataset into four maneuvering activities, namely state 0: both a_{lat} and a_{lon} are stationary, state 1: only a_{lon} is stationary, state 2: only a_{lat} is stationary, and state 3: both a_{lat} and a_{lon} are non-stationary. Stationarity refers to cases where the ADF (p-values > 0.01) for each window-size of 250 of signals. We treat these states as a ground truth without loss of generality. In Figs. 4a and 4b, the two subgroups of states (1 and 3) and states (0 and 2) can be assumed to correspond to activities of a_{lon} and a_{lat} respectively. However, distinguishing patterns between One D_s

Table 3. Pruned representations: PCC vs. PCA on various operation days.

Operational day	Initial size (M)	PCC (m_1)	PCA (m_2)
One D_s	64	6	3
One D_l	64	4	7
Eight D	64	7	6

Table 4. Cross data performances on multi-tasks downstream after pruning.

Operational day	W_t	Classification		Clustering		Regression	
		AUPCR	Accuracy	Silhouette	DBI	R^2	Loss
TNC4maneuvering (After PCA pruning)							
One D_s	250	0.756	77.51	-	-	-0.407	2.262
One D_l	250	0.417	59.57	0.414	0.241	0.340	1.002
Eight D	250	0.450	48.15	0.497	0.202	-0.328	1.416
TNC4maneuvering (After PCC pruning)							
One D_s	250	0.956	97.02	0.404	0.758	-0.269	2.040
One D_l	250	0.936	84.86	0.247	1.211	0.330	1.017
Eight D	250	0.903	79.25	0.189	1.494	-0.313	1.400

and One D_l cluster patterns is challenging due to their difference of being short and long distance operations. In Fig. 4c, no clear-cut pattern emerges of state separability, as all states are present, reflecting that diverse driving behaviors are collected over multiple days. Figure 5 shows both accelerations and learned representations without additional for One D_l day. In this day of activity, we see that in cases where both accelerations (a_{lon}, a_{lat}) have simultaneous activity, it can also be observed with correspondence to the color code in the representation space, similar to when there is low activity. Overall, it appears that a_{lon} strongly influences the characteristics of the representations. This is due to the vehicle executing less full turns and rather accelerating and decelerating more on this particular operation day. A co-interpretation of Figs. 4b and 5 suggests that maneuverability is primarily governed by a_{lon} activity, corresponding to states 1 and 3 in Fig. 4b and the dark (negative) color codes in Fig. 5.

Results After Pruning. The two pruning methods yield different representation sizes, as shown in Table 3. These variations arise from the methods distinct selection criteria: PCC eliminates highly linear correlated representations, while PCA determines the required representation count based on cumulative explained variance. We apply pruning methods offline and subsequently evaluate model performance as post-pruned representations, as illustrated in Table 4. Our post-pruning methods further assume that representations are more disentangled since unnecessary components are removed. There was no further training to fine-tune and update model weights to recover some of the lost accuracy. Therefore, after pruning there is no major improvement on the three subsequent tasks. Linear regression performs the least further indicating that localized turning points are not linearly explained by representations. Overall, there is a decline in performance across each subsequent task. Consequently, our offline pruning methods have a reduced performance, as there is no post pruning model weights updates. In the work [16], the authors address this issue by implementing online pruning, which enhances efficiency, generalization, and interpretability without significant performance loss.

6 Conclusion

Our unsupervised representation learning method, TNC4maneuvering, effectively extracts maneuverability representations from complex MTS vehicle dataset. Its versatility is evidenced by performance in various downstream tasks, especially on a classification task. Although it allows one to capture longer temporal dependencies, scalability and speedup remain areas of challenge, which are our next points of focus. Another win to claim is that we have managed to get rid of two extra hyperparameters, the window-size and size of representations, reducing the number of hyperparameters to a bare minimum and hence reducing further complexity. TNC4maneuvering holds great promise for enhancing maneuverability analysis in smart transportation, laying a foundation for general future usage in other applications. In our future work, we will replace gradients

testing with a variance due to its insensitivity to total data samples and incorporate pruning within the training framework in order to update model weights after pruning. Regarding scaling and speedup, we plan to replace the ADF test with a pre-calculated stationarity matrix.

Acknowledgement. This work has received funding from the European Union’s Horizon 2020 Research and Innovation Programme, Grant Agreement n° 955393. Moreover, thanks to Manufacture Française des Pneumatiques Michelin for support and car dataset provision.

References

1. Carlos, M.R., González, L.C., Wahlström, J., Ramírez, G., Martínez, F., Runger, G.: How smartphone accelerometers reveal aggressive driving behavior?-the key is the representation. *IEEE Trans. Intell. Transp. Syst.* **21**(8), 3377–3387 (2019)
2. Choi, H., Kang, P.: Multi-task self-supervised time-series representation learning. [arXiv:2303.01034](https://arxiv.org/abs/2303.01034) (2023)
3. Eldele, E., et al.: Time-series representation learning via temporal and contextual contrasting. In: *International Joint Conference on Artificial Intelligence, IJCAI 2021*, pp. 2352–2359 (2021)
4. Eldele, E., et al.: Self-supervised contrastive representation learning for semi-supervised time-series classification. [arXiv:2208.06616](https://arxiv.org/abs/2208.06616) (2022)
5. Eslami, T., Awan, M.G., Saeed, F.: GPU-PCC: a GPU based technique to compute pairwise Pearson’s correlation coefficients for big FMRI data. In: *ACM International Conference on Bioinformatics, Computational Biology, and Health Informatics*, pp. 723–728 (2017)
6. Franceschi, J.Y., Dieuleveut, A., Jaggi, M.: Unsupervised scalable representation learning for multivariate time series. In: *Advances in Neural Information Processing Systems*, vol. 32 (2019)
7. Hyvarinen, A., Morioka, H.: Unsupervised feature extraction by time-contrastive learning and nonlinear ICA. In: *Advances in Neural Information Processing Systems*, vol. 29 (2016)
8. Kurita, T.: Principal component analysis (PCA). In: *Computer Vision: A Reference Guide*, pp. 1–4 (2019)
9. Lai, C.I.: Contrastive predictive coding based feature for automatic speaker verification. [arXiv:1904.01575](https://arxiv.org/abs/1904.01575) (2019)
10. Lebesse, T., Mattrand, C., Clair, D., Bourinet, J.M.: Unsupervised representation learning in multivariate time series with simulated data. In: *2023 Prognostics and Health Management Conference (PHM)*, pp. 217–225 (2023)
11. Van der Maaten, L., Hinton, G.: Visualizing data using t-SNE. *J. Mach. Learn. Res.* **9**(11) (2008)
12. MacQueen, J.: Classification and analysis of multivariate observations. In: *5th Berkeley Symposium on Mathematical Statistics and Probability*, pp. 281–297. University of California, Los Angeles, LA, USA (1967)
13. Oord, A.V.D., Li, Y., Vinyals, O.: Representation learning with contrastive predictive coding. [arXiv:1807.03748](https://arxiv.org/abs/1807.03748) (2018)
14. Sarker, S., Haque, M.M., Dewan, M.A.A.: Driving maneuver classification using domain specific knowledge and transfer learning. *IEEE Access* **9**, 86590–86606 (2021)

15. Tonekaboni, S., Eytan, D., Goldenberg, A.: Unsupervised representation learning for time series with temporal neighborhood coding. [arXiv:2106.00750](#) (2021)
16. Weatherhead, A., et al.: Learning unsupervised representations for ICU timeseries. In: Conference on Health, Inference, and Learning, pp. 152–168. PMLR (2022)
17. Woo, G., Liu, C., Sahoo, D., Kumar, A., Hoi, S.: Cost: contrastive learning of disentangled seasonal-trend representations for time series forecasting. [arXiv:2202.01575](#) (2022)
18. Yue, Z., et al.: Ts2vec: towards universal representation of time series. In: AAAI Conference on Artificial Intelligence, vol. 36, no. 8, pp. 8980–8987 (2022)
19. Zerveas, G., Jayaraman, S., Patel, D., Bhamidipaty, A., Eickhoff, C.: A transformer-based framework for multivariate time series representation learning. In: ACM SIGKDD Conference on Knowledge Discovery & Data Mining, pp. 2114–2124 (2021)