Sensor characterization for supporting de-noising and auto-calibration of sensor data

1nd Wout Van Hauwermeiren
WAVES, Ghent University, Belgium
orcid.org/0000-0001-9107-546X
Wout.VanHauwermeiren@ugent.be

2nd Yuanbo Hou
WAVES, Ghent University, Belgium
orcid.org/0000-0001-8469-5740
Yuanbo.Hou@ugent.be

3nd Karlo Filipan
Acoustics Lab, Catholic University of Croatia, Zagreb, Croatia
orcid.org/0000-0002-2966-5984
karlo.filipan@unicath.hr

4nd Dick Botteldooren
WAVES, Ghent University, Belgium
orcid.org/0000-0002-7756-7238
Dick.Botteldooren@ugent.be

***Abstract*—In earlier work, a mobile sensor network had been deployed in passenger cars to measure road quality using sound and vibration sensors. A self-calibration and confounder removal (SCCR) algorithm eliminated the effect of the measurement device in sound and vibration measurements. It relied on the hypothesis that two measurements of the same object should give the same result. However, once the deep learning model has been trained, the model cannot easily include more devices, because it relies on a one-hot encoded device identification. Therefore, additional characterization of both sensor and vehicle is proposed here using a batch of observations (a bag) from the same sensors. A set-encoder is trained with a device classification head. This set-encoder compresses the input bag into a latent representation, irrespective of the input order. This latent vector, which characterizes the sensor, is injected into the SCCR framework to steer the calibration. New devices are characterized by the pre-trained set-encoder. This relies on the hypothesis that closely related sensors with a similar response should be close to each other in the latent space. Therefore, SCCR is now possible with new devices, previously unseen at the training time of SCCR. However, an increase in reconstruction error is observed for the unseen devices.**

***Index Terms*—mobile sensors, road quality, representation learning, pooling, set2set**

I. Introduction

Sensor calibration is a pressing issue for various types of sensors (temperature, sound pressure meter, accelerometers, air pressure ...). Typically, sensors can be calibrated with a reference signal. For acoustic sensors, a sine-wave at a fixed sound pressure level could be emitted in a controlled way [1]. Then, the sensor readings can be rescaled to fit the reference signal. In [2], acoustic and vibration sensors have been deployed in multiple vehicles. The sensor devices are influenced by various internal and external factors such as vehicle construction, sensor installation, tire pressure, weight distribution of the vehicle, etc. Therefore, a more advanced calibration strategy is needed. To this end, a self calibration a self-calibration and confounder removal strategy (SCCR) for mobile road measurements is introduced [2]. SCCR is a deep learning auto-encoding neural network that reconstructs samples originating from one device, to another device under different circumstances. However, with SCCR it is not possible to add new sensors after training the model. Conceptually, sensor calibration can be achieved by measuring the same object with different sensors - in this case, the same stretch of road surface - or by comparing a set of measurements that are done on a comparable, but not necessarily the same set of objects. In the latter case, information on the measurement device needs to be extracted from a non-ordered set of measurements.

For some tasks such as translation into different languages, extracting an order-invariant representation of the input sentences might improve the translation [3]. However, designing a robust, order-invariant representation is a difficult problem. The amount of possible permutations of a sequence of length n is $n!$, which makes modeling the probability distribution of a set of observations intractable [3]. In the most simple case, a set of samples can be aggregated in an order-invariant way with statistical properties, such as the standard deviation, average, minimum and maximum. In convolutional neural networks, these are called pooling operators. Pooling operators most commonly reduce the dimension of the feature space, for instance, to downsize an image or to downsample time series data. Another way to obtain such a representation might be to sort the samples beforehand such as demonstrated in Ordered Weighted Averages, a fuzzy operator [4]. Another method models the set of observations with Sliced-Wasserstein embeddings [5]. Finally, Vinyals et al. [3] proposed a learnable attention mechanism with a Long Short Term Memory (LSTM) neural network.

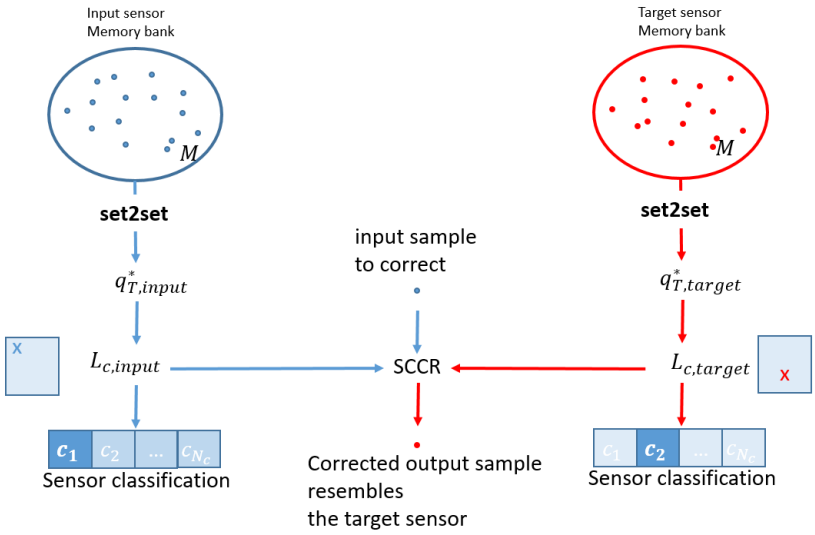


Fig. 1: Methodology: characterizing an input and a target sensor to a latent representation and consecutively transforming an input sample to a target sensor.

This paper characterizes sensors by their usual measurements: an unordered random bag of samples. To this end, set2set of Vinyals et al. [3] is chained to a classification neural network that distinguishes which sensor had made the measurements. As to our knowledge, no other research used this approach to characterize sensors in this way. Instead of using the one-hot encoded sensor identification, the latent representation in the bottleneck layer is injected into the SCCR framework (as shown in Fig. 1). Then, SCCR is evaluated for devices which are previously unseen for the model.

The paper is structured as follows. In Section II-A, the opportunistic road vehicle sensing dataset is introduced. Then in Section II-B, SCCR is presented; in Section II-D, the sensor characterization network is proposed and in Section II-E performed experiments are defined. In Section III results of the performed experiments are presented and discussed. Finally, conclusions and outlook are given in Section IV.

II. Methodology

A. Opportunistic sensing of roads

The source dataset has been thoroughly described in [2]. A brief introduction to the opportunistic roads vehicle dataset (ORVD2022): 41 vehicles of various types (sedans, light cargo vans, light passenger vans and light SUV's) collected data in various cities in Flanders from April 2020 until December 2021. The vehicles collected sound and vibration measurements during routine trips which were made on the underlying roads of different quality. The features extracted from the measurements are 1/3-octave band values in both the vibration and sound frequency domains. The observations are also geo-referenced on 20m road segments and laterally positioned using a lane detection algorithm designed by Trogh et al. [6]. Cars (devices) were driving in different locations - cities and regions, however, a partial overlap between the various measurement locations has been observed. This overlap is exploited in the self-supervised calibration and confounder removal (SCCR) algorithm.

B. Self-supervised calibration and confounder removal

In [2], a self-supervised calibration and confounder removal method (SCCR) had been proposed. An auto-encoding neural network is learned with a reconstruction loss term. It transforms high-dimensional samples gathered by one device to samples obtained by another device under different conditions. The overlapping measurement locations are seen as non-changing reference points within a time frame of 3 months. The model relies on a one-hot encoded label for a device. As a consequence, no new sensors can be added after the model has been trained. As an alternative, a first effort had been made in [2] to characterize a sensor using a set of their observations and this approach is further explored in this paper.

C. Set2set model

Set2set, introduced by Vinyals et al. in 2016, is a deep learning model which encodes an unordered set of observations X into a latent representation q^*_T [3], the observations are indexed x_i, however the model is invariant to permutations. The shape of X is (M, D). The set size is $|X| = M$ and $1 \leq i \leq M$. The whole model can be trained by gradient descent and chained to further layers. In this section, the set2set model is introduced.

At first, a function $m_i = g(x_i)$ encodes a memory bank m_i. A LSTM (long-short term memory) [7] calculates a hidden recurrent state q_t, which is further processed to obtain normalized weights a_i between 0 and 1 that focus on the memory elements m_i. The weighted sum of memory elements

is r_t. At the end of each recurrent step, $q_t^* = [r_t, q_t]$ is obtained. In detail, these are the equations that assemble the model:

$$q_t = LSTM(q_{t-1}^*) \tag{1}$$

$$e_{i,t} = f(m_i, q_t) \tag{2}$$

$$a_{i,t} = \frac{exp(e_{i,t})}{\sum_j^M exp(e_{j,t})} \tag{3}$$

$$r_t = \sum_i^M a_{i,t} * m_i \tag{4}$$

$$q_t^* = [q_t, r_t] \tag{5}$$

The function f computes the attention coefficients for each memory cell m_i. q_0^* is set to all zeros. Various parameters have to be chosen beforehand: the functions f and g, the number of time steps T, the memory size and the dimension of q_t. In [3], time steps is set to a maximum of 10. Significant computing resources are required to compile the model in Keras Tensorflow with a large T. In this publication, f is a simple dense layer with one hidden layer with D nodes and ReLU activation. The output dimension is 1. g is decided to be the identity function $g(x) = x$.

D. Sensor characterization

Sensors could be characterized by a random sample of their observations. The set2set pooling method [3], introduced in the previous section, is such a candidate.

The characterization neural network consists first of a set2set layer, second a fully connected layer with 30 nodes ReLU (rectified linear unit) activations, third a fully connected layer with no activation (which results in a latent representation L_C), fourth a fully connected layer with 30 nodes and ReLU activation and finally a fully connected layer with softmax activations (i.e. the classification head with N_c nodes). The classification head distinguishes which sensor has made the observations by a one-hot encoded vector. The cross-entropy loss is computed on the classification head. The key idea behind using this architecture is that the memory bank should characterize the sensor so that a latent vector L_C represents a sensor. Two points close to each other in the latent space should mean that the two sensors are very similar.

The basic model configuration takes in a memory bank of size $|M|$. Augmenting $|M|$ does not increase the amount of parameters of the model. However processing time increases with greater $|M|$. L_C has dimension 2 in this paper, for easy visualization.

The observations have been normalized by device beforehand to avoid learning obvious differences in sensors (e.g. a constant difference). The characterization model and set2set have been been implemented in Keras Tensorflow (2.6). Set2set has been implemented from scratch, based on the formulas in [3].

E. Experiments

In this section, several experiments are presented that evaluate (a) the capability of sensor characterization and (b) the effect of injecting the sensing device characterization in the SCCR-framework. To ensure the independence of the measurements between the training and the test set, the surveyed OpenStreetMap ways are assigned either to the training or the test set. OpenStreetMap ways are longer sections of roads, made up of multiple consecutive 20m segments. Different lanes of the same ways are also assigned to the same set, since the road construction of different lanes is often the same and therefore correlated. 80% of the locations are included in the train set and the remaining 20% in the test set.

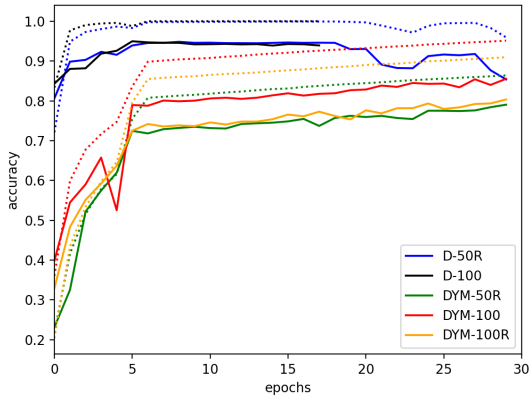


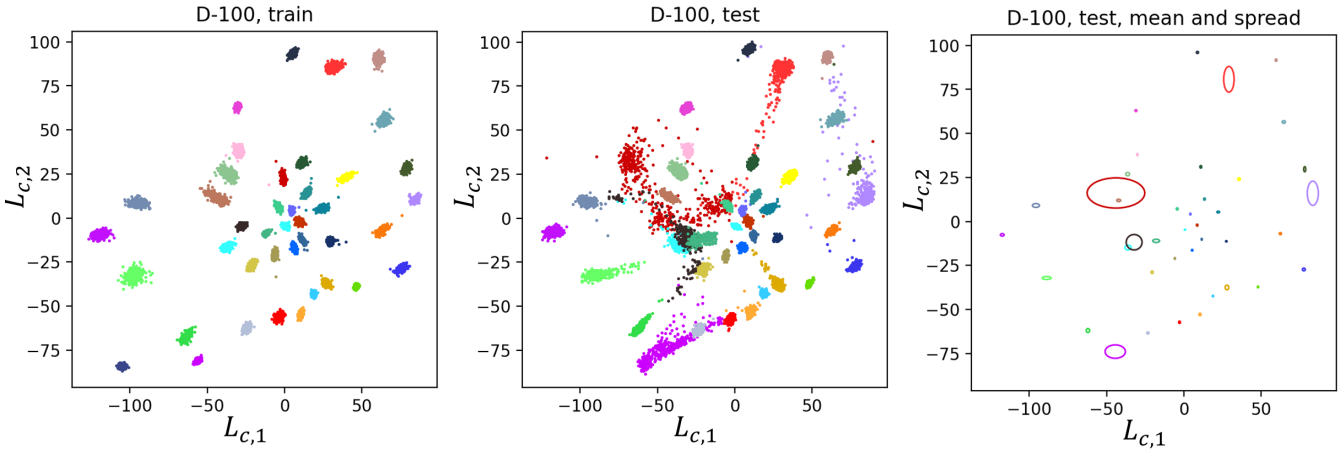
Fig. 2: Training curves of the different sensor characterization models. Solid lines indicate the test accuracy, dotted lines indicate the train accuracy.

	val acc device	train acc device	val acc DYM	train acc DYM
D-50R	91.5%	99.5%		
D-100	**93.9%**	100.0%		
DYM-50R			79.0%	86.4%
DYM-100	92.2%		**85.5%**	95.1%
DYM-100R			80.3%	90.9%

Fig. 3: Accuracy (acc) of the different models for both validation (val) and train set. The model performance is subdivided into device classification accuracy and device-year-month (DYM) accuracy.

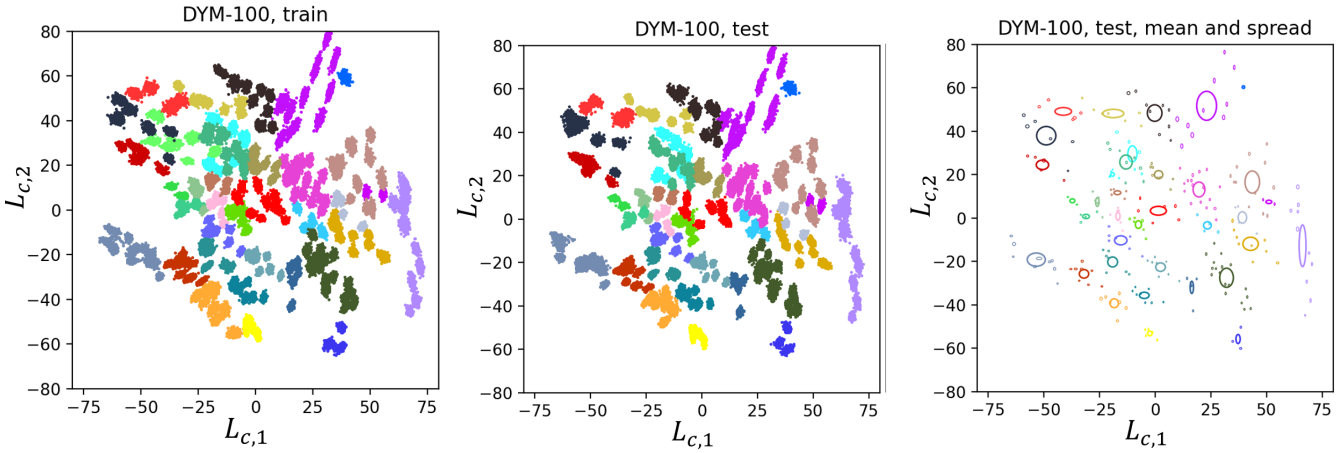
Two major classification tasks have been performed: (1) distinguishing the device, (2) distinguishing both the device and time period. Task 1 amounts to 41 classes for 41 devices. 5000 training bags and 1000 test bags are selected for every device. For task 2, the combination (device, year, month) yields 275 classes. 1250 training bags and 250 test bags are selected for each class. Time dependence is included to account for drift and seasonal dependence. Inherent to the ORVD2022 dataset, weather has an effect on noise emissions made by vehicles on roads. In addition, some parameters of the test vehicle fluctuate over time such as tire pressure or could change abruptly by changing winter and summer tires.

Next, a small parameter search is performed on the classification models. The amount of memory units is either



(a) (b) (c)

Fig. 4: Visualization of the latent space of model *D-100*. Every point on the graph corresponds to a bag of 100 observations of a certain device. Colors indicate the ground truth origin device. In (a) the training bags are plotted, in (b) the validation bags and (c) shows ellipses with radius equal to the standard deviation of each dimension and centered around the average of the latent variables.



(a) (b) (c)

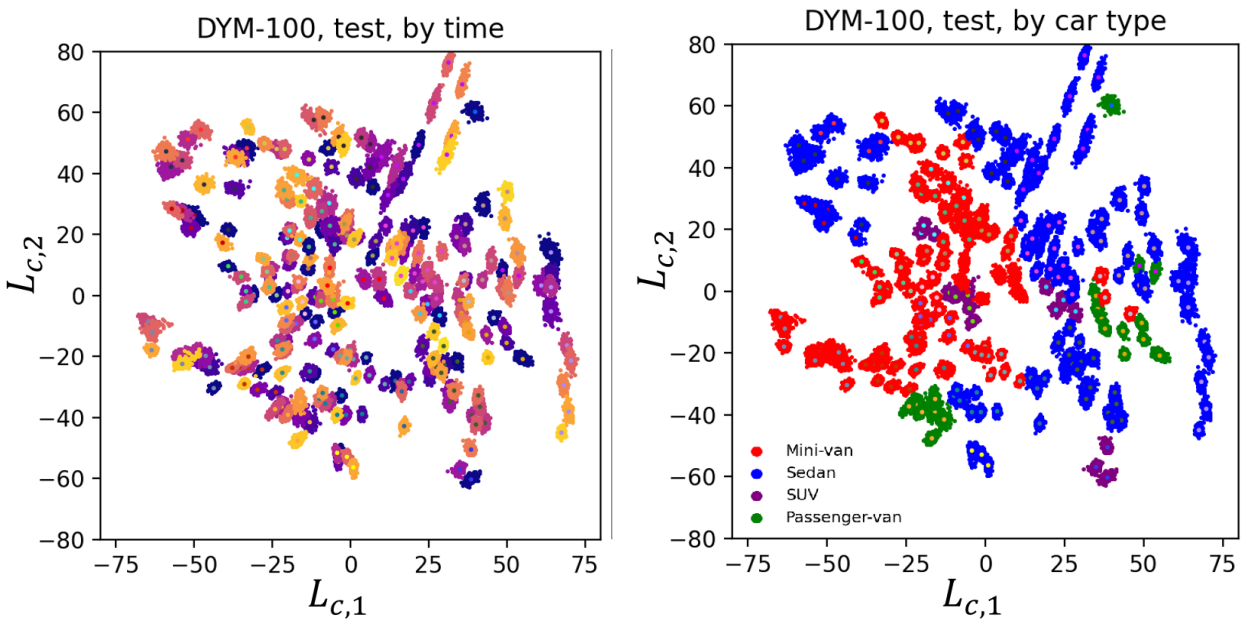
Fig. 5: Visualization of the latent space of model *DYM-100*. Every point on the graph corresponds to a bag of 100 observations of a certain (device, year, month) tuple. Colors indicate the ground truth origin device.In (a) the training bags are plotted, in (b) the validation bags and (c) shows ellipses with radius equal to the standard deviation of each dimension and centered around the average of the latent variables. The thickest lines are plotted by device, the thinner ones are plotted by (device, year, month).

50 or 100. Then, a reduced feature set is obtained by omitting features with frequency higher than 2.5kHz. These frequency bands are more related to sensor noise and less to the road quality.

In the final experiment, three devices are taken out of the set. The device classifier is trained on the remaining devices. For each device, For the average of the latent variables is taken as a continuous device identifier. Then, the latent representation of each device is injected into SCCR (instead of the one-hot encoded label for a device). SCCR is trained with the latent representation of the measurement device. At validation time, the three previously left-out devices are reintroduced. The reconstruction root mean squared error (RMSE) validates the experiment (lower is better). The reference SCCR model is trained with all devices.

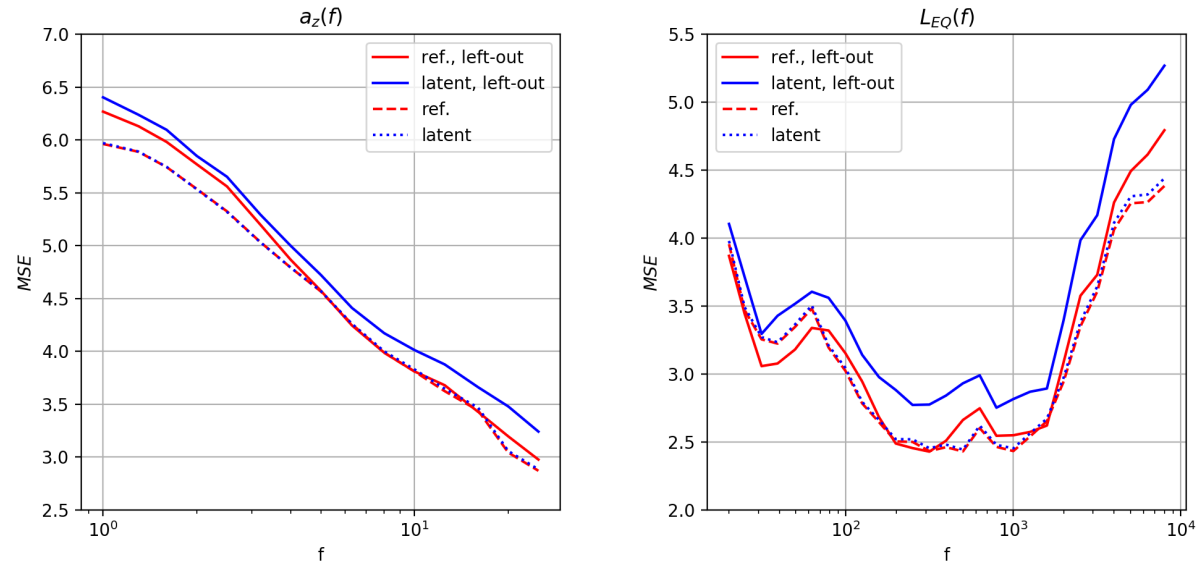
III. Results

The models have been trained on Tesla T4 GPU with Adam optimizer. The first 5 epochs are performed at $5{*}10^{-3}$ learning rate, the remaining 25 epochs at $5 * 10^{-5}$ to fine-tune the model. Training on the CPU (an AMD EPYC 7352 24-core processor), resulted in the same training time per batch.



(a) (b)

Fig. 6: Alternative colorings of the test latent space of model *DYM-100*. In the left panel, the bags are colored by time (dark blue is earliest, yellow is latest). In the right panel, the bags are colored according to vehicle type (mini-van, sedan, SUV, light passenger van). The cluster mean is colored according to the origin device.



(a) (b)

Fig. 7: SCCR performance for two scenario's: reference (with device as a one-hot encoded label) and latent (with device as a 2D-latent variable). The average root mean squared error (RMSE) for vibration ($a_z(f)$) and sound frequency bands ($L_EQ(f)$ is calculated for the group of left-out devices and the group of other devices.

The accuracy is used as a metric, since the classes are all balanced. The models are the following: **D-50R** (device classifier, 50 memory units, reduced feature set), **D-100** (device classifier, 100 memory units), **DYM-50R** (device, year, month classifier, 50 memory units, reduced feature set), **DYM-100** (device, year, month classifier, 100 memory units), **DYM-100R** (device, year, month classifier, 100 memory units, reduced feature set). The training curves are given in Fig.

2. The high learning rate of the first epoch gives faster convergence to higher accuracy, but at the risk of being unstable (see DYM-100). Final classification accuracy is given in Fig. 3. Model D-100 is trained until convergence and achieves a validation accuracy of 93.9% and a perfect train accuracy. Further training is therefore not useful, due to the overfitting on the train set. Model D-50R seemed to be stable for 20 epochs and scored similar to D-100, but then the performance degraded for both training and validation. Halving the size of the memory bank did not affect the accuracy. For the DYM models, DYM-100 achieves the best accuracy of 85.5%, which is less than the performance of the device classifiers. However, this is still a high accuracy in distinguishing both device and month of capture. This also means that there are inherent differences based on the time. DYM-50R and DYM-100R score approximately the same in validation. The train/test gap is the least for DYM-50R.

Model *D-100* classifies the training bags perfectly on the device label. For the test set, there is however confusion between some. Two devices have very low accuracy (respectively 3% and 20%). In Fig. 4 the latent space of model *D-100* is visualized for both the training and the test bags. Training and test bags are sampled from a separate set of locations, however the same devices are probed in both sets. The test bags are not seen during training. The purple cloud near $(L_{c,1}, L_{c,2}) = (-50, -75)$, the red cloud near $(0, 25)$ and the light red cloud near $(30, 90)$ are spread out most prominently and confuse with other devices. Although the training bags (see Fig. 4a) are clustered by device, the test bags (see Fig. 4b) are spread out more. Amongst which, five devices seem to have significant spread in the latent dimensions of the test set (as apparent on the right-most panel). However, the others have low spread. This means that the classification is robust and the procedure is stable, meaning that sampling a different subset of road segments still produces the same classification and thus ultimately the same calibration. However, there was also one outlier point for the test set (outside the range of Fig. 4). This indicates that multiple bags should be used to identify a vehicle properly. In addition, the dark blue device around $(-100, -90)$, is completely dislocated in the test set. Some bags end up near other devices (confusion between two devices). Although the classification performance is good, the device clusters are spread out.

The same analysis can be performed for model *DYM-100*, see Fig. 5. The classification target is a (device, year, month) tuple. For the same device there are now multiple clusters, and these lay in the vicinity of each other. This emerging characteristic is not induced by the procedure, since every (device, year, month) tuple is a one-hot encoded vector and no clustering objective on device is included in the loss function but stems from the dataset. In Fig. 5c, it can be observed that the test bags are clustered by separate (device, year, month) and that the clusters co-occur together by device. The accuracy increases to 92.2%, if the confusion is allowed to be across time and within device, which is on par with the *D-100* model. However, as an advantage, the model learns the trends and seasonal dependencies of the measurement device. In Fig. 6a, the monthly clusters mostly follow a smooth path through time. For instance, for the clusters around $(25, 60)$, identified as the "purple" device, the path is closed with yellow dots close to the dark blue ones as one could expect from a seasonal effect. For the cluster most to the right $(75, y)$, the trend is linear which would indicate a strong drift of the measurement device over time. A final example can be identified for the device around $(-50, -20)$ where the path is also periodic, but tend to make a jump between two periods of the year which could indicate a change of tires between summer and winter. In Fig. 6b, the latent space is also colored by car type. The most prominent clusters are mini-van (red) and sedan (blue). The other car types (SUV and passenger mini van) are too uncommon to make conclusions. However, it could also be that the SUV's may physically resemble the sedan vehicles. Mini-vans tend to cluster together which may be explained by the similar acoustic and vibration characteristics of these vehicles.

In Fig. 7, the performance of the reference SCCR model with one-hot encoded label is compared with the performance of the SCCR model with latent variables. Vibrations of the car body caused by the road surface, a_z, and sound levels caused by the road surface interacting with the tire, L_{EQ}, are shown. The root mean squared error (RMSE) is taken between the measurement of a 20m road surface by one devices and corrected by the SCCR and the measurement by another device of the same 20m road surface. Another trained model **D-100b** is introduced, similar to model *D-100*, however three devices are left out of the training data.

The performance differs between the left-out group and the rest of the devices for both SCCR models. For the devices included in the training set (non left-out) the performance remains the same for both the reference model and the latent model. This shows that the information extracted from the typical measurements is indeed sufficient to identify the devices in a way that allows calibration using the SCCR. The left-out devices are not seen on training time for the latent SCCR model. The most prominent difference in RMSE is in the L_{EQ} feature. Namely, the performance degrades for the L_{EQ} feature, compared to the reference model. a_z remains almost unaffected. For comparison, the figure also includes the results obtained using the unique car label for mapping to and from the devices that were selected as left-out for the training of the model proposed in this publication.

IV. Conclusion and outlook

In this manuscript, a method for characterizing sensors using a batch of their observations has been proposed. The characterization relies on the set2set algorithm [3] which handles the non-ordered bag of observations. A latent 2D representation has been learned through gradient descend. This enabled identifying the measurement device with high accuracy for most devices based on the test set (on average 93.9%). When the observation time and device (device, year,

month) were set as a classification target, the accuracy reduced slightly, but remained high (85.5%). The latter is explained by the known dependence of the signal generation (mainly noise) on temperature and thus seasonal and/or sensor drift. The latent representations follow gradual changes (due to time) and are clustered by the vehicle type. However in self-calibration and confounder removal (SCCR), when the one-hot encoded label had been replaced with the latent device representation, SCCR performed equally for known devices, but the calibration performance degraded for unknown devices. It is expected that this is due to the strong disentanglement of devices in the latent space. Points close together in the latent space should correspond to devices that bear a great resemblance to one another. A different kind of loss objective could remediate this. Another flaw of the classification task might be that several obvious differences in the observations such as a pure tone (originating from the engine) could provide the signature to identify a certain device while these features might be irrelevant for the calibration procedure. For future work, an end-to-end approach could be applied to jointly train the sensor characterization network and the SCCR framework. In this way, the in-between latent representations could have direct meaning in the SCCR framework. In addition, alternative loss functions (or training tasks) should be defined to improve the latent representations.

ACKNOWLEDGMENT

This research received funding from the Flemish Government under the ”Onderzoeksprogramma Artificiele Intelligentie (AI) Vlaanderen” program.

REFERENCES

[1] A. Tamil Chandran, M. P. Dhanya, P. Surendran, and J. Chandapillai, “Selection and calibration of acoustic sensors,” *International Journal of Engineering Research Technology (IJERT)*, vol. 05, June 2016. [Online]. Available: http://dx.doi.org/10.17577/IJERTV5IS060509

[2] W. Van Hauwermeiren, K. Filipan, D. Botteldooren, and B. De Coensel, “A scalable, self-supervised calibration and confounder removal model for opportunistic monitoring of road degradation.” *Computer-Aided Civil And Infrastructure Engineering*, pp. 1–18, 2022. [Online]. Available: https://doi.org/10.1111/mice.12821

[3] O. Vinyals, S. Bengio, and M. Kudlur, “Order matters: Sequence to sequence for sets,” in *Proceedings of the Tenth International Conference of Learning Representations*, 2016.

[4] S. Vluymans, N. Mac Parthaláin, C. Cornelis, and Y. Saeys, “Weight selection strategies for ordered weighted average based fuzzy rough sets,” *Information Sciences*, vol. 501, pp. 155–171, 2019. [Online]. Available: https://www.sciencedirect.com/science/article/pii/S0020025519305122

[5] N. Naderializadeh, J. Comer, R. Andrews, H. Hoffmann, and S. Kolouri, “Pooling by sliced-wasserstein embedding,” in *35th conference on Neural Information Processing (NeurIPS)*, 2021.

[6] J. Trogh, D. Botteldooren, B. De Coensel, L. Martens, W. Joseph, and D. Plets, “Map matching and lane detection based on markovian behavior, gis, and imu data,” *IEEE Transactions on Intelligent Transportation Systems*, pp. 1–15, 2020.

[7] S. Hochreiter and J. Schmidhuber, “Long Short-Term Memory,” *Neural Computation*, vol. 9, no. 8, pp. 1735–1780, 11 1997. [Online]. Available: https://doi.org/10.1162/neco.1997.9.8.1735