Data Fusion for Deep Learning on Transport Mode Detection: A Case Study

Hugues Moreau, Andrea Vassilev, Liming Chen

Data Fusion for Deep Learning on Transport Mode Detection: A Case Study

Hugues Moreau[1,2][0000−0002−0569−4190], Andrea Vassilev[1], and Liming Chen[2]

[1] Commissariat à l'Energie Atomique {hugues.moreau, andrea.vassilev}@cea.fr
[2] Ecole Centrale de Lyon, {hugues.moreau,liming.chen}@ec-lyon.fr

Abstract. In Transport Mode Detection, a great diversity of methodologies exist according to the choice made on sensors, preprocessing, model used, etc. In this domain, the comparisons between each option are not always complete. Experiments on a public, real-life dataset are led here to evaluate carefully each of the choices that were made, with a specific emphasis on data fusion methods. Our most surprising finding is that none of the methods we implemented from the literature is better than a simple late fusion. Two important decisions are the choice of a sensor and the choice of a representation for the data: we found that using 2D convolutions on spectrograms with a logarithmic axis for the frequencies was better than 1-dimensional temporal representations. To foster the research on deep learning with embedded inertial sensors, we release our code along with our publication. [3]

Keywords: Transport Mode Detection · CNN · Deep learning · accelerometer · inertial sensors · spectrogram

1 Introduction

Transport mode detection is a classification problem aiming to design an algorithm that can infer the transport mode of a user given multimodal signals (GPS and/or inertial sensors). It has many applications, such as carbon footprint tracking, mobility behaviour analysis, or real-time door-to-door smart planning. The signals are collected from an embedded device (either the sensors of a mobile phone, or a dedicated device), and processed by a Transport Mode Detection Algorithm, in order to know the transport mode of the owner of the device. This algorithm has multiple steps that often include a preprocessing step, and classification in itself. The classification step is where algorithms differ the most from each other. All algorithms use Machine Learning *i.e.*, methods that use a certain amount of labeled data to learn how to predict the output transport mode, before making predictions on unseen samples.

Contrary to domains like Computer Vision, the approaches in this domain differ greatly from each other: preprocessing, model, chosen sensors, recording conditions, etc. This diversity of variables further prevent fair and efficient comparisons between publications. Recently, the Sussex-Huawei Locomotion team

[3] https://github.com/HuguesMoreau/TMD_fusion_benchmark

published a dataset containing data from inertial sensors (accelerometer, gyrometer, magnetometer, etc.), in order to organize a challenge: they published a certain amount of labeled and unlabeled data, and asked researchers to make predictions on the unlabeled data set. As they evaluated each prediction using unseen labels, they could establish clear comparisons between approaches. However, several many options typically differ between two publications, thus preventing comparisons. For instance, all the approaches [3,6,8,22] had different methods, preprocessing pipelines, architectures, etc. even though they worked with on same exact problem. We start from one of them, and evaluate carefully each of the choices that were made, with a specific emphasis on data fusion methods.

Our main contributions are the following:

- We measure the performance of each sensor: the single best sensor is the norm (magnitude) of the acceleration, and the second best real sensor is the gyrometer (section 5.1).
- We show the best preprocessing method is to resort to spectrograms (time-frequency diagrams), to compute the log of the power, with a log scale for the frequencies (section 5.2). We go further and measure numerically the importance of each of the parameters.
- We evaluate 13 different data fusion methods and show that they are, for the most part, equivalent. (section 5.3)

In a hope to help reproduce and foster research on this subject, we also publish the code that allowed us to generate the results.

The rest of the publication is organized as follows: section 2 reviews the publications that focus on Human Activity Recognition and data fusion. Sections 3 and 4 present the single-modality architecture, and the different options we will compare. Finally, we detail the results of the experiments and in section 5.

2 Related works

2.1 Transport Mode Detection

Transport Mode Detection (TMD) is a classification problem whose goal is to detect the transport mode of a human subject, *e.g.*, walking, cycling, using any sensor. With inertial sensors, multiple methods coexist: some use handcrafted features with traditional learning methods (see [6,9,1], for instance), while others use directly use Deep Neural networks (Convolutional [8] or Recurrent Neural Networks [15]). For inertial sensors, the most important dataset is the SHL dataset, which is used to organize a yearly challenge [20,18,19]. In the next section, we will present the dataset, the challenge that comes with it, and provide an outline of the best participations.

The SHL 2018 challenge The SHL 2018 challenge is a supervised classification problem. The participants were given a series of 16,310 consecutive annotated recordings of embedded sensors and had to classify some 5978 samples of

a test set. Each recording is 60-seconds long, and contains data from 7 sensors: accelerometer, magnetometer, gravity, linear acceleration (acceleration minus gravity), gyrometer, orientation quaternion, and barometric pressure. Each signal was recorded at $100Hz$, so that one sample to classify is a set of 20 vectors of size $60 \times 100 = 6000$ points. There are 8 classes available: Still, Walk, Run, Bike, Car, Bus, Train, Subway. Our protocol mimics the organisation of the challenge: we make our choices on a validation dataset, the annotations on the 5978 test samples are used only in section 3.2.

The most successful method is the submission from the Joseph-Stephan Institute [6]. This approach used an ensemble of methods (both Machine Learning and Deep Learning), with a hidden markov model as meta-classifier to return a prediction. They reached a F1-score of 93.86 % on the unseen test set, ranking first on the 2018 challenge.

The next best team [8] used spectrograms (time-frequency diagrams), along with specific preprocessing. The 'images' containing the log of the power were then given to a 2-dimensional CNN for classification. This pure-deep learning approach is the one we selected as our baseline (section 4.2). But the participation only used two sensors: the accelerometer with gyrometer. For each classification, the spectrograms from these two sensors were concatenated along their 'frequency' axis, to form a single image which was to be classified by a single network. In the challenge, this reached a 88.83 % F1-score.

Several other participants did submit a prediction to the challenge. Similarly to the Transport Mode Detection literature, some used traditional ML algorithms with handcrafted features, others relied on Recurrent Neural Networks, or a combination of CNN and RNN. Wang *et al.* [20] published a complete synthesis of the challenge participations along with the results. However, they did not conduct any experiments to assess the importance of each possible choice.

Since the end of the challenge, other publications worked on this dataset, sometimes using a different test set. The best one is from Gjoreski *et al.* [7], who improved the model that scored first in 2018 [6]. By adding another neural network to the prediction models, they managed to improve the F1-score on the SHL test set by one percentage point, up to 94.9%.

Most methods working with traditional Machine Learning models use a simple fusion: once all features are computed for all sensors, they are concatenated and given to a classifier ([6,9,1]). With deep learning models, the data fusion methods change much more between publications. In the next section, we will present the diverse fusion modes used in Deep Learning approaches.

2.2 Data Fusion modes in deep learning

Most approaches rely on simple fusion modes: Early fusion (concatenation of input signals [8,17]), intermediate fusion (concatenation of representation coming from different sensors [24]), or late fusion (average of predictions [16])

Some approaches are sensor-specific, either because they work on textual data [12] and rely on the specific structure of the medium; or because they create an explicit alignment between sensor data with different ranges (*eg* a RGB camera

looking forward and a LIDAR sensor gathering information from all directions [4]), which is not necessary in our case as the signals from different sensors are already synchronized. Others focus on pretraining by reproducing one modality with the other, in case one modality is suddenly missing [14].

But some methods still rely on relevant principles: for instance, Wang *et al.* [21] work with audiovisual videos, and create a model whose weights minimize the overfitting. Chen *et al.* [2] designed a network that attributes a weight to each sensor, similarly to attention [23]. Liu *et al.* [11] tried to conceive a network that does not require every sensor to be good everytime.

These methods are always better than the state of the art on the dataset each publication considered, but most work provide little comparisons with other fusion methods, if any. In general, most of the publications which deal with multiple sensors compare their architecture to a small subset of baselines, and those subsets do not always overlap between publications. Reviews exist to identify the different fusion modes [5,10], but they only report the performance each article provided. To our knowledge, none compare the performance of each method with the same exact sensors, architectures, and evaluation method. We provide a clear comparison of the different fusion modes, including the most basic ones.

3 Baseline

3.1 A network using a single modality

Before considering the fusion of different sensor data, we need to consider a network that succeeds at classifying signals from a single sensor. We rely on the architecture in [8]: the Convolutional Neural Network has three convolutional layers, a flatten step, and two fully-connected layers. The hyperparameters and training process are the same as in [8].

We will consider this network as our baseline to evaluate fusion methods: all fusion methods that use multiple sensors should at least be better than a network using only the best sensor (the accelerometer).

3.2 Training protocol

The 2018 SHL Challenge asked candidates to give one prediction per timestamp (that is, to output 6,000 predictions per sample) but, as only 4% of the samples of the database have more than one mode, we work on a simpler problem: each sample in the dataset is assigned a single transport mode, which is the mode at $t = 30s$. We split the train and validation splits similarly to [22]: to avoid contamination between training and validation samples, we sort the data chronologically, and we send the first 3,000 samples to the validation set, while the last 13,000 samples go into the train set (we use the first samples for the validation set because the end of the dataset is severely imbalanced).

To generate the results of table 2 to 3, the network is trained for 50 epochs on the training set, before being evaluated on the validation set. Each evaluation is

repeated 5 times (with a new random seed every time), the mean and standard deviation of the F1 score are given as a result. To test the best method against the state of the art, we train the model with the union of the train and validation set, and evaluate our results on the test set. The mean and standard deviation are given in section 5.4.

4 Experiments

We design three experiments where we evaluate a series of alternatives. As evaluating all the possible choice combinations is impossible, we evaluate each choice one by one. In most situations, when evaluating one parameter, we need to choose a value for the other parameters. In these cases, the other parameters are set to their *baselines*, which we will present in each case.

4.1 Sensor choice

To confirm our choice of sensors, we decide to evaluate each all the sensors individually. For each sensor available (accelerometer, gyrometer, etc.) we consider every possible axis (x, y, z, with the possible addition of w for the orientation quaternion), in addition to the norm of these axes (computed using the euclidean norm). The norm is hoped to represent an orientation-independent version of the signals. We compute a log-power of the spectrograms with a log axis, and evaluate each signal individually.

baseline The design of a heuristic is out of scope of this work. To make a choice, we simply selected the sensors using prior knowledge and related literature. We choose the norm of the accelerometer, for it is the single best signal available (*cf* table 1). As in [8], we add the y-axis of the gyrometer. But those two sensors only measure the inertial dynamics. To add a different kind of information, we choose to use the norm of the magnetometer. In exterior, this signal capture the Earth's magnetic field, which does not change much. It varies greatly around metal objects and strong electrical currents. We expect this means the sensor to help to distinguish the Still class from the motorized ones (Train, Subway). This can be checked by looking at the average power spectrum of each class (Fig. 1).

When considered alone, the norm of the magnetometer is worse than any of its three individual axes (x, y, z, table 1). This is because the axes can act as a compass, thus retaining information about the dynamics of the movement. Computing the norm destroys this information, which is not a problem in our setting as we always use the magnetometer along with other inertial sensors (accelerometer and gyrometer). As a sanity check, we add the w component of the orientation vector, which encodes the amount of rotation the phone detects. We expect this signal to carry similar information to the gyrometer, which means adding it to the triplet $(|Acc|, Gyr_y, |Mag|)$ should not improve results significantly.

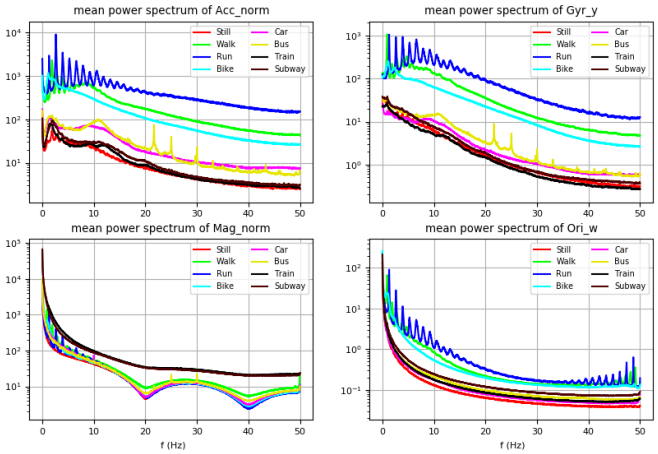


Fig. 1: The average power spectrum per class (only half of the spectrum is shown).

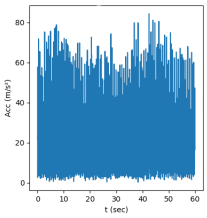
4.2 Preprocessing

For our comparisons, we consider the 1-dimensional temporal data, and different kinds of spectrograms. For each signal, we also compute the power spectrum using the Fast Fourier Transform (FFT). This is intended to be halfway between the 1D temporal and 2D spectrogram representations. For the results in the 'temporal' and 'FFT' categories (and for these results only), the network uses 1-dimensional convolutions, with filters of size 3. All the other parameters remain similar to [8].

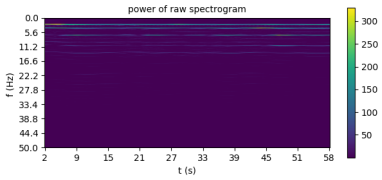
baseline We repeat the preprocessing protocol in [8]. Each temporal signal is first converted into a spectrogram using a moving STFT window. The samples are 6,000 points long segments (60 seconds at 100 Hz), and we use 5 seconds-long windows with 4.9s overlap. Then, the spectrograms are rescaled into 48×48 pixels (this exact resolution was chosen to fit the architecture of the network, see section 3). The time axis is rescaled linearly, while the frequency axis is rescaled using a logarithmic interpolation (similarly to the mel scale). This allows to give more importance to the lowest frequencies (Fig. 1 shows these frequencies are paramount). Lastly, we compute the log of the power, in order to make visible regions with different orders of magnitude (See fig. 2 for an illustration).

4.3 Fusion modes

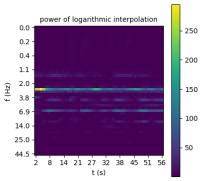
We selected several data fusion methods that are relevant in our setting for comparison. The names may vary, some methods might even not be named. One important note: most of these modes are equivalent to our baseline architecture



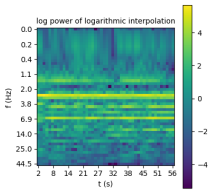
(a) Raw data



(b) full-size spectrogram



(c) using a log for the frequency interpolation



(d) using a log for the power scale

Fig. 2: An illustration of the preprocessing step with the norm of the accelerometer data from a running segment.

(section 3) when a single modality is used (if they leave the possibility to use a single sensor). The exceptions are the bottleneck filters and attention.

- *Early fusion* modes consist in giving all signals to a single neural network. The only difference between them is how the input signals are put together before they are processed by the neural network.
  - *Time concatenation*: the input spectrograms are concatenated along their temporal axis.
  - *Frequency concatenation*: the input spectrograms are concatenated along their frequency axis (after log interpolation).
  - *Depth concatenation*: the signals are put together like the channels of a RGB image: each convolution filter of the input layer has access to the same portion of all signals at the same time.

  Note that early fusion is not always possible for every problem. As Moya-Rueda *et al.* notice [13] using these modes require to have the sensors synchronized (which implies having the same frequency) in order for the fusion to be relevant.
- *Intermediate fusion* consists in merging together the features produced by different sensor-specific networks, so that a single classification network can process them. As it requires the classifier to have some kind of internal representation, it most adapted for deep architectures. *Feature concatenation*: The idea behind feature fusion is to let the convolutional layers compute features, before giving both features to the next layer. The rationale behind this method is to allow each convolution module to generate its own relevant features.

Table 1: The validation F1-score per signal. The highest result is in bold

sensor	Acc	LAcc	Gra	Gyr	Mag	Ori	Pressure
x	87.24 ± 0.53	83.97 ± 0.80	85.19 ± 0.26	81.31 ± 0.57	71.14 ± 0.67	73.82 ± 1.24	
y	87.22 ± 0.72	86.22 ± 0.84	84.44 ± 0.22	81.05 ± 1.25	73.63 ± 0.82	74.37 ± 1.25	
z	84.18 ± 0.77	85.36 ± 0.69	83.34 ± 0.54	79.32 ± 1.32	73.17 ± 1.03	75.46 ± 0.51	76.35 ± 0.67
$norm$	$\mathbf{89.14 \pm 0.65}$	81.01 ± 0.50	47.67 ± 3.27	76.52 ± 0.68	66.81 ± 0.47	42.05 ± 0.97	
w						78.54 ± 1.07	

– *Late fusion* methods rely on using only the predictions (scores or probability vector) of different sensor-specific models. As they allow to use any type of models (both Machine Learning and Deep learning), these methods are the most flexible category.
  - *Probability fusion*: Each network produces a probability vector (after the softmax), indicating the prediction of the network. With the probability fusion, we simply compute an average of probabilities.
  - *Scores fusion*: To merge the scores, we extract the vector before the softmax. This vector of scores (or logits) indicates whether the sample is likely to belong to each class. We compute the average, for each class, of the score each network assigns to each class, before using a softmax to obtain a single output probability.
  - *Weighted fusions*: With the two previous modes (probabilities and scores fusion), the average was unweigthed, which means a 'bad' sensor is given as much importance as a relevant one. To avoid it, we try letting the network learn the weight to assign to each sensor. With both methods, we compute a weighted sum of the predictions of the sensor-specific models.

We also included several algorithms from the literature whose motivations apply to our work. These works are the Bottleneck filters [17], Attention [23], Selective fusion [2], Learn to combine modalities in multimodal deep learning [11], and Gradient Blend [21]. In total, we evaluate 13 fusion modes. There is no need for a baseline in this case, because we choose to evaluate sensor choice and preprocessing method on single-sensor data.

5 Results

5.1 Evaluation of unimodal sensors

Table 1 shows the norm of the accelerometer is the single best signal available. If the accelerometer is the best sensor, the linear acceleration and gravity follow closely. Also, the pressure signal is surprisingly effective at distinguishing between transport modes.

For the accelerometer, the norm of the acceleration vector has a better performance than any of its individual axes, which shows the orientation-independent

Table 2: The validation F1-score (%) per preprocessing method. For each signal, the highest result is in bold, and the second highest result is underlined

mode	interpolation (frequency axis)	power scale	size (T, F)	$\|Acc\|$	Gyr_y	$\|Mag\|$	Ori_w
temporal			6000	70.20 ± 1.63	64.71 ± 2.74	7.49 ± 10.32	39.65 ± 2.26
FFT			6000	80.57 ± 1.30	74.30 ± 0.72	55.99 ± 1.53	64.27 ± 1.57
spectrogram	none	linear	550, 250	84.55 ± 1.03	71.81 ± 0.64	63.64 ± 0.96	2.29 ± 0.00
spectrogram	none	log	550, 250	$\underline{87.88 \pm 0.68}$	$\underline{79.89 \pm 1.30}$	$\underline{64.81 \pm 0.85}$	43.35 ± 33.56
spectrogram	linear	linear	48, 48	81.98 ± 0.75	58.42 ± 1.31	45.97 ± 1.88	2.29 ± 0.00
spectrogram	linear	log	48, 48	86.33 ± 1.00	77.06 ± 1.32	56.53 ± 0.68	$\underline{75.03 \pm 2.08}$
spectrogram	log-freq	linear	48, 48	84.46 ± 0.63	69.19 ± 0.89	53.36 ± 1.41	2.29 ± 0.00
spectrogram	log-freq	log	48, 48	$\mathbf{88.83 \pm 0.71}$	$\mathbf{82.64 \pm 0.68}$	$\mathbf{67.36 \pm 0.49}$	$\mathbf{78.39 \pm 1.79}$

signal is better than its components. For the norm of the magnetometer, however, the result is the opposite, as the norm of the magnetometer is worse than any of its individual axes.

One other surprisingly high performance is the norm of gravity and norm of the orientation vector. In theory, these two signals are constant (equal to 1 for the norm of the orientation vector, and $9.81 m.s^{-2}$ for the gravity), but in practice, these signals come from imperfect computations which yield indications about the dynamics of the sensor.

5.2 Evaluation of preprocessing methods

Table 2 gives the results the evaluation of each preprocessing method. Note that the results in table 2 are similar to those obtained by Richoz *et al.* [16] with a different neural network architecture: with three sensors (accelerometer, magnetometer, gyrometer), they obtained a F1-score of 79.4 %. For the record, we reproduced their approach (frequency concatenation of FFT segments), with our setting (architecture from [8], Fourier transforms of 60-seconds long segments instead of 5s), and obtained a validation F1-score of $81.44 \pm 1.06\%$.

Switching to the norm of the FFT is strictly better than using raw, temporal representations. This difference might be due to the fact that the power spectrum better separates patterns from noise (see fig. 1).

But using spectrograms seems better, in most cases, than using a power spectrum, or even a temporal segment. One notable exception, however, is the raw, full-size spectrogram, with the orientation vector. This method has an impressive standard deviation, because two of the five initializations were failure cases that did not learn efficiently and had a F1-score of 2.8% (which is the score of a classifier that predicts the most occurring mode). The others had a F1-score of $76.4 \pm 1.6\%$, which is closer to what one could expect given the results of the other sensors. We could not find any explanations to this behaviour, which is all the more surprising as neural networks are usually stable throughout different initializations (as our other experiment show).

Table 3: The mean and standard deviation of the validation F1-score of each method, for different fusion modes. For each sensor combination, we display the best result in bold and underline the results that are less than two standard deviations away from the best

method	early fusion				intermediate fusion			late fusion					
	time concat.	freq concat.	depth concat.	Bottleneck filters	features	Selective Fusion	attention	probas	scores	Weighted probas	Weighted scores	Gradient Blend	Learn to combine
$\|Acc\|, Gyr_y$	90.88 ±0.57	90.46 ±1.08	90.57 ±0.94	88.70 ±1.30	90.83 ±1.54	90.01 ±0.41	84.11 ±1.30	90.26 ±0.56	**90.95 ±0.37**	88.85 ±0.54	90.89 ±1.13	89.18 ±1.10	90.10 ±0.66
$\|Acc\|, \|Mag\|$	90.78 ±0.66	91.37 ±0.49	91.83 ±0.35	85.83 ±4.45	91.74 ±0.46	90.62 ±1.03	86.67 ±1.01	91.83 ±0.73	91.72 ±0.41	88.04 ±0.93	**92.17 ±0.59**	89.53 ±0.64	87.37 ±0.65
$\|Acc\|, Gyr_y, \|Mag\|$	91.36 ±0.74	92.13 ±0.90	91.91 ±0.88	87.01 ±2.16	91.87 ±0.64	92.39 ±0.87	87.85 ±1.05	92.33 ±0.61	92.55 ±1.08	89.40 ±0.55	**92.98 ±0.37**	89.47 ±0.92	89.56 ±0.98
$\|Acc\|, Gyr_y, \|Mag\|, Ori_w$	92.32 ±1.18	92.30 ±0.54	91.23 ±1.25	84.59 ±5.52	92.51 ±1.01	92.93 ±0.60	87.56 ±0.62	92.43 ±0.40	92.83 ±0.19	89.43 ±0.49	**93.01 ±0.36**	89.36 ±1.24	91.41 ±1.11

Table 4: A summary of the influence of the parameters in tables 1, 2, and 3. We purposely excluded the scenarios where the network did not learn anything (defined as F1-score $< 10\%$)

Average gain by switching:		
from	to	
Gyr_y	$\|Acc\|$	+12.55
$\|Mag\|$	$\|Acc\|$	+27.06
Ori_w	$\|Acc\|$	+20.06
spectrogram raw power	spectrogram log power	+8.66
250 × 550 spectrogram (log power)	48 × 48 spectrogram (linear interp., log power)	−4.22
250 × 550 spectrogram (log power)	48 × 48 spectrogram (log interp., log power)	+2.08
raw (temporal) data	48 × 48 spectrogram (log interp., log power)	+25.10
Median fusion method	Best fusion method	+1.12

Using the log of the power is strictly better than the raw power for the accelerometer, gyrometer, magnetometer: the average gain obtained by switching from the raw power to the log power is 7.66 percentage points.

Using a logarithmic interpolation for the frequency axis allows to effectively reduce the size of the data without altering the signal as a linear interpolation does. This might be due to the fact that interpolating linearly a (550, 250) spectrogram into a (48, 48) matrix erases the difference between the fundamental frequency of the Walk and Bike segments (fig. 1). A log scale preserves the distinction between these modes by giving more room to the lower frequencies.

5.3 A benchmark of fusion modes

Table 3 gives the results of each fusion method, applied to four sensor combinations. We notice that, given the standard deviation of the experiments, most fusion methods have statistically similar performance. In fact, data fusion may be the least determinant choice we evaluated (see table 4). As most fusion methods have the same performance, we recommend using the most simple ones.

5.4 Evaluations on the test set

In order to compare our approach against the state of the art, we select the best fusion method and sensor combination in table 3 (that is, a weighted score fusion of the four sensors), and train it 5 times on the union of train and validation sets, before evaluating the F1 score on the held-out test set of the challenge. If the F1-score is still significantly lower than it was on the validation set, the loss is acceptable ($89.96 \pm 0.07\%$ and 93.01 ± 0.36, respectively). However, the best model is barely superior to a single-sensor baseline ($87.44 \pm 0.98\%$ on test, 89.14 ± 0.65 on val), which consists in using the accelerometer alone. In practice, the use of additional sensors (and the energy consumption that comes with it) might not justify the performance gain.

6 Conclusion

We studied the application of Convolutional Neural Network on a Transport Mode Detection problem. By fixing all but one choice in each of our experiments, we could evaluate ech of the choices a practitioner can make. We found the more important choices to make are the sensor choice (the accelerometer being the best) and the preprocessing method (the use of spectrogram, with a logarithm axis for the frequencies), However, after evaluating 13 different data fusion methods, we found that no fusion method significantly outperforms the others. Future work might consist in adapting this benchmark to the newer datasets SHL 2019 and 2020, and the new research problems that come with them. Alternatively, we could extend our work to start from the approaches of [6,7], who used noticeably more complex methods to get better results on the challenge.

References

1. Alotaibi, B.: Transportation Mode Detection by Embedded Sensors Based on Ensemble Learning. IEEE Access **8**, 145552–145563 (2020)
2. Chen, C., Rosa, S., Miao, Y., Lu, C.X., Wu, W., Markham, A., Trigoni, N.: Selective Sensor Fusion for Neural Visual-Inertial Odometry. In: 2019 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR). pp. 10534–10543. IEEE (Jun 2019)
3. Choi, J.H., Lee, J.S.: Confidence-based Deep Multimodal Fusion for Activity Recognition. In: UbiComp '18. pp. 1548–1556. ACM Press (2018)
4. Feng, D., Cao, Y., Rosenbaum, L., Timm, F., Dietmayer, K.: Leveraging Uncertainties for Deep Multi-modal Object Detection in Autonomous Driving. arXiv:2002.00216 [cs] (Feb 2020)
5. Gao, J., Li, P., Chen, Z., Zhang, J.: A Survey on Deep Learning for Multimodal Data Fusion. Neural Computation **32**(5), 829–864 (May 2020)
6. Gjoreski, M., Gams, M., Janko, V., Reščič, N., Mlakar, M., Luštrek, M., Bizjak, J., Slapničar, G., Marinko, M., Drobnič, V.: Applying Multiple Knowledge to Sussex-Huawei Locomotion Challenge. In: UbiComp '18. pp. 1488–1496. ACM Press (2018)

7. Gjoreski, M., Janko, V., Slapničar, G., Mlakar, M., Reščič, N., Bizjak, J., Drobnič, V., Marinko, M., Mlakar, N., Luštrek, M., Gams, M.: Classical and deep learning methods for recognizing human activities and modes of transportation with smartphone sensors. Information Fusion **62**, 47–62 (Oct 2020)
8. Ito, C., Cao, X., Shuzo, M., Maeda, E.: Application of CNN for Human Activity Recognition with FFT Spectrogram of Acceleration and Gyro Sensors. In: UbiComp '18. pp. 1503–1510. ACM Press (2018)
9. Janko, V., Gjoreski, M., De Masi, C.M., Reščič, N., Luštrek, M., Gams, M.: Cross-location transfer learning for the sussex-huawei locomotion recognition challenge. In: UbiComp/ISWC '19. pp. 730–735. ACM Press (2019)
10. Liu, J., Li, T., Xie, P., Du, S., Teng, F., Yang, X.: Urban big data fusion based on deep learning: An overview. Information Fusion **53**, 123–133 (Jan 2020)
11. Liu, K., Li, Y., Xu, N., Natarajan, P.: Learn to Combine Modalities in Multimodal Deep Learning. arXiv:1805.11730 [cs, stat] (May 2018)
12. Ma, L., Lu, Z., Shang, L., Li, H.: Multimodal Convolutional Neural Networks for Matching Image and Sentence. In: Proceedings of the IEEE International Conference on Computer Vision. pp. 2623–2631 (2015)
13. Moya Rueda, F., Grzeszick, R., Fink, G.A., Feldhorst, S., Ten Hompel, M.: Convolutional Neural Networks for Human Activity Recognition Using Body-Worn Sensors. Informatics **5**(2), 26 (Jun 2018)
14. Ngiam, J., Khosla, A., Kim, M., Nam, J., Lee, H., Ng, A.Y.: Multimodal Deep Learning. In: ICML (Jan 2011)
15. Ordóñez, F.J., Roggen, D.: Deep Convolutional and LSTM Recurrent Neural Networks for Multimodal Wearable Activity Recognition. Sensors **16**(1), 115 (Jan 2016)
16. Richoz, S., Wang, L., Birch, P., Roggen, D.: Transportation Mode Recognition Fusing Wearable Motion, Sound, and Vision Sensors. IEEE Sensors Journal **20**(16), 9314–9328 (Aug 2020)
17. Wang, B., Lei, Y., Li, N., Yan, T.: Deep separable convolutional network for remaining useful life prediction of machinery. Mechanical Systems and Signal Processing **134**, 106330 (Dec 2019)
18. Wang, L., Gjoreski, H., Ciliberto, M., Lago, P., Murao, K., Okita, T., Roggen, D.: Summary of the Sussex-Huawei locomotion-transportation recognition challenge 2019. In: UbiComp/ISWC '19. pp. 849–856. ACM Press (2019)
19. Wang, L., Gjoreski, H., Ciliberto, M., Lago, P., Murao, K., Okita, T., Roggen, D.: Summary of the sussex-huawei locomotion-transportation recognition challenge 2020. In: Ubicomp'20. pp. 351–358. ACM (Sep 2020)
20. Wang, L., Gjoreskia, H., Murao, K., Okita, T., Roggen, D.: Summary of the Sussex-Huawei Locomotion-Transportation Recognition Challenge. In: UbiComp '18. pp. 1521–1530. ACM Press (2018)
21. Wang, W., Tran, D., Feiszli, M.: What Makes Training Multi-Modal Networks Hard? arXiv:1905.12681 [cs] (May 2019)
22. Widhalm, P., Leodolter, M., Brändle, N.: Top in the Lab, Flop in the Field?: Evaluation of a Sensor-based Travel Activity Classifier with the SHL Dataset. In: Ubicomp'18. pp. 1479–1487. UbiComp '18, ACM (2018)
23. Xu, K., Ba, J., Kiros, R., Cho, K., Courville, A., Salakhutdinov, R., Zemel, R., Bengio, Y.: Show, Attend and Tell: Neural Image Caption Generation with Visual Attention. arXiv:1502.03044 [cs] (Apr 2016)
24. Zeng, M., Nguyen, L.T., Yu, B., Mengshoel, O.J., Zhu, J., Wu, P., Zhang, J.: Convolutional Neural Networks for Human Activity Recognition using Mobile Sensors (Nov 2014)