Traffic Load Estimation from Structural Health Monitoring sensors using supervised learning

Traffic Load Estimation from Structural Health Monitoring Sensors using Supervised Learning

Alessio Burrello[a], Giovanni Zara[b], Luca Benini[a,c], Davide Brunelli[a,d], Enrico Macii[e], Massimo Poncino[b], Daniele Jahier Pagliari[b]

[a]*Department of Electrical, Electronic, and Information Engineering, University of Bologna, Bologna, Italy.*
[b]*Department of Control and Computer Engineering, Politecnico di Torino, Turin, Italy.*
[c]*Department of Information Technology and Electrical Engineering, ETH, Zurich, Switzerland.*
[d]*Department of Industrial Engineering, University of Trento, Trento, Italy.*
[e]*Interuniversity Department of Regional and Urban Studies and Planning, Politecnico di Torino, Turin, Italy.*

Abstract

Traffic Load Estimation (TLE) is increasingly adopted in public road infrastructures to regulate the access and limit heavy vehicles circulation. Standard approaches to TLE are based either on installing dedicated sensors such as intelligent cameras or infrared sensors or using existing smartphone sensors. However, both approaches have severe limitations, as often dedicated sensors are power-hungry and expensive to install and maintain, whereas smartphone-based approaches critically rely massively on users collaboration. More recently, researchers have started investigating TLE approaches using networks of accelerometers that are often already installed on critical road elements such as viaducts and bridges for Structural Health Monitoring (SHM) purposes. Specifically, in previous solutions, the detection and counting of vehicles was based on unsupervised anomaly detection and did not use any labelled data. While this simplifies the system's setup, it also makes full validation impossible.

In this work, we investigate the TLE problem using a supervised learning approach for SHM-sensor-based TLE for the first time. In particular, we use a relatively short recording session from a smart camera to label acceleration data with the corresponding number (and type) of passing vehicles. Labelled data are then fed to a Machine Learning (ML) model trained as a regressor to estimate the vehicle count corresponding to each input sample. We perform an extensive comparison among different types of ML models, both classic and deep. Our experiments find that the highest accuracy is achieved by a Support Vector Regressor (SVR) combined with simple feature extraction, which can reach a Mean Absolute Error (MAE) of 0.47 light vehicles and 0.21 heavy vehicles. This corresponds to a 9.8× and 8.1× error reduction compared to previous unsupervised solutions, respectively. Lastly, we show that our approach lends itself to an energy-efficient implementation on a real SHM gateway.

Keywords: Structural Health Monitoring, Traffic Load Estimation, Machine

1. Introduction

Automatic Traffic Load Estimation (TLE) consists of detecting, counting and (possibly) classifying the passing vehicles on essential nodes of the road infrastructure, such as highways or large junctions. The information extracted by automatic TLE systems can improve several aspects linked to the economic and environmental sustainability of urban and suburban transportation infrastructures, for instance, the scheduling of road maintenance work or the regulation of traffic to reduce air pollution [1, 2]. As for many other tasks, the Internet of Things (IoT) paradigm, which envisions billions of communicating sensors, has spurred the increasing diffusion of TLE systems worldwide. On the other hand, finding the IoT-based TLE implementation that maximizes the vehicle tracking accuracy while minimizing the sensors' energy consumption and installation costs, among the many variants proposed in literature, is still an open issue.

Sensor	**Accuracy**	**Energy**	**Instal. Cost**
Dedicated Sens.	High	High	High
Mobile Sens.	High	High	Low
SHM Sens.	Low	Low	Low

Table 1: Advantages and disadvatanges of different classes of TLE solutions. Abbreviations: Instal. Cost: installation cost, Sens. sensor,

In most instances, TLE is implemented with one of two main approaches. A first family of solutions relies on the installation of dedicated sensors on monitored road sections such as smart cameras or infrared sensors [3, 4, 5]. This typically yields high counting and classification accuracy but incurs high costs for installing and maintaining the sensor nodes. Moreover, many of these sensors, such as smart cameras, are often energy-hungry [6], thus complicating the implementation of TLE with battery-operated nodes for road portions not easily reached by the electrical network and causing concerns for data security and privacy.

Alternatively, many solutions use the information extracted from sensors already available in drivers' smartphones [7, 8]. While these approaches reduce the hardware cost of TLE to zero, they incur other limitations. Their effectiveness fundamentally relies on the collaboration of many users willing to share their data. Moreover, they only work well for use cases where the same group of drivers', who subscribe to the collaborative service, regularly travel in the same area (e.g., the citizens of a small town). They are much less effective, for instance, on suburban highways.

Recently, a third alternative has emerged, which consists of implementing TLE using the Sensor Networks (SNs) that are already present in critical portions of the road infrastructure for Structural Health Monitoring (SHM) purposes [9]. In general, SHM refers to the automatic monitoring of large structures such as masonry buildings [10], tunnels [11], or bridges [12], aimed at identifying unusual behaviour that might be a sign of wear out and prevent critical accidents. When applied to bridges and viaducts, SHM is typically based on measures of acceleration (vibration). Therefore, despite not being installed specifically for that reason, SHM sensors can also be used for TLE, exploiting the fact that bridges vibrate in response to crossing vehicles, proportionally to their mass and speed.

This form of TLE typically achieves lower accuracy, but it has the advantage of leveraging existing and always-on SNs, thus not incurring any additional hardware cost. Moreover, accelerometers are low-cost and energy-efficient devices, and, differently from a smartphone-based solution, this approach does not rely on drivers' collaboration. Finally, this approach eases confidentiality and data security, given the absence of visual recording and not relying on personal data. Table 1 summarizes benefits and drawbacks of each of the three main categories of TLE implementations.

In previous work [9], TLE based on SHM sensors was framed as an anomaly detection problem, where vibration patterns that deviated from a periodically updated baseline were associated with the passage of a vehicle. The main motivation for selecting an unsupervised approach was that it did not require an initial gathering of labelled data, i.e., an association between acceleration patterns and the ground truth number and type of passing vehicles. On the other hand, the overall performance of this solution is not high enough in high traffic conditions.

In this paper, we show that, at the cost of a (short) initial data labelling phase, which can be performed leveraging a temporary installation of a camera on the targeted section of the road, an approach based on *supervised learning* can lead to superior TLE accuracy, while maintaining all the advantages intrinsic to SHM sensors based TLE (i.e., low energy and low installation cost, see Table 1). Specifically, the following are the main contributions of this work:

- We collect a labelled dataset on a real viaduct in Italy, in conditions of typical daytime traffic. We do so using a single recording of 31 minutes, for a total of 297 vehicles recorded, performed with a smart camera, whose output is then used to automatically associate ground truth vehicle counts

to acceleration samples obtained from SHM-sensors.

- We feed these labelled data to supervised Machine Learning (ML) models and train them to perform regression on different vehicle categories (namely light and heavy vehicles). To the best of our knowledge, we are the first to frame the problem of TLE based on accelerometers for SHM in a supervised way. Furthermore, we perform an extensive comparison among different types of ML models while also exploring various hyper-parameters settings for each model.
- Through our experiments, we show that the additional (once-for-all) cost of data labelling is paid off by a significant increase in the estimation accuracy. Specifically, with a Support Vector Regressor (SVR), i.e., the best model found in our analysis, we obtain a Mean Absolute Error (MAE) of just 0.47 light vehicles and 0.21 heavy vehicles on a 60s window, that is the 7.45% and 6.71%, respectively, of the average traffic load. This corresponds to a 9.8× and 8.1× error reduction, respectively, compared to the application of a previous unsupervised pipeline [9] to the same data. Deployed on a SHM gateway, the model consumes just 44 mJ per inference with a latency of 2.53 ms. The fact that regression can be implemented at the edge, in turn, enables a continuous TLE, which would not be possible otherwise due to storage and transmission bandwidth limitations.

The rest of the paper is organized as follows. Section 2 provides an overview of the literature, describing in detail the different approaches to TLE. Section 3 describes the target viaduct and the interconnected network of accelerometers for SHM before describing the data acquisition and labelling process. Section 4 details the supervised learning pipeline used in this work and the different regressors considered. Section 5 describes our experimental results and Section 6 and Section 7 conclude the paper.

2. Related Works

TLE is beneficial for a broad spectrum of tasks associated with the improvement of sustainability in urban and suburban road infrastructures. Use cases reported in recent work include the scheduling of maintenance interventions [9], the monitoring and reduction of air pollution [2], and the prediction of the economic impact of travel delays due to traffic congestion [1].

Many different types of input data have been considered for TLE. This is due to the variability of the possible application scenarios and of the corresponding requirements, which include not only a high estimation accuracy but also concern non-functional requirements, such as low power consumption of the sensors. Table 2 summarizes some of the most relevant recent studies in this domain. Overall, five different data sources are employed for three categories of deployment scenarios, namely cities, bridges/viaducts, and freeway exits. The table also reports other factors that differentiate the previous works, namely the dataset size (in number of vehicles), the type of algorithm and the main results.

Work	Sensors	Task	Location	Algorithm	Vehicles	Main Results
Kamkar et al. [4], 2016	1 Camera	TLE	Bridge	Active Basis Model, Random Forest	1179	Sens.: 74.82-92.11%, FP: 0.004-0.3
Odat et al. [5], 2017	3 Ultrasonic Infrared	TLE	City	Bayesian Networks	187	Sens.: 99%
Wang et al. [13], 2017	3 Magnetometers	TLE	City	Adaptive Threshold	81	Sens.: 97.5%
Dong et al. [14], 2018	1 Magnetometer	TLE	Freeway Exit	Classification Tree	4507	Acc.: 99.8% (Single class) Acc.: 80.5% (Multi class)
Liu et al. [15], 2019	1 Microphone	TLE	City	Wavelet Denoising, Thresholding, SVM	106	MAE: 9.52-22.44%, Acc.: 71%
Ye et al. [16], 2020	20 Accelerometers	TLE	City	Cross Correlation, ANN, K-means	687	Sens.: 87.5%
Burrello et al [9], 2019	90 MEMS Accelerometers[1]	SHM	Viaduct	Adaptive Threshold	297	MAE: 72.8% (Light V.) * MAE: 53.2% (Heavy V.)
Our Work	7 MEMS Accelerometers[1]	SHM	Viaduct	Supervised ML models	297	MAE: 7.45% (Light V.) MAE: 6.71% (Heavy V.)

* Data produced in our work with the algorithm of [9]. [1] Sensors belonging to a pre-existing SHM installation.

Table 2: Recent works on TLE, categorized in terms of input data type, deployment scenario, algorithms, and results. The "Task" column highlights the initial goal of the sensor network installation. Abbreviations: ANN: Artificial Neural Network, SVM: Support Vector Machine, V. vehicles, MAE: Mean Absolute Erros, Sens.: sensitivity, FP: False Positives

Algorithms performance are measured differently depending on how the TLE problem is framed. Some works address it as a binary classification, where the goal is to detect, for each input sample, the presence/absence of a vehicle. Then, performance is measured in terms of *accuracy*, *sensitivity*, number of *false positives*, etc. In contrast, other works frame TLE as a regression, where the goal is to predict the *count* of vehicles (possibly of a particular class) crossing the monitored section in a time interval. In this case, performance is measured as the Mean Absolute Error (MAE) between the ground truth and predicted counts, often normalized to a percentage of the average true value. In the rest of the section, we overview the current state-of-the-art in urban and suburban TLE.

2.1. Urban TLE

Intelligent Transportation Systems (ITS) play a fundamental role in modern cities. TLE is a key component of ITSs, as it allows the monitoring and avoidance of traffic congestion in different city areas [13, 1]. Noteworthy, in most urban scenarios, TLE is implemented with *dedicated sensors*, which are therefore positioned optimally to maximize performance. As an example, Odat et al. [5], and Liu et al. [15] reach good performance combining ultrasonic waves, acoustic waves, and infrared sensors for urban TLE. On a small dataset of 187 vehicles, [5] reaches a sensitivity of 99% combining the prediction of three dual infrared and ultrasonic sensors. Liu et al. [15] employ wavelet denoising preprocessing together with a double threshold algorithm on a single sensor fiber optic strip. The system achieves 9.52%-22.44% mean absolute error (MAE) on the estimation of traffic with a total of 106 vehicles. The authors of [17] use the same sensors and improve the prediction performance using empirical mode decomposition (EMD) and STFT feature extraction. In [13], magnetic sensors

are employed together with a detection algorithm based on Adaptive thresholds, reaching up to 97.5% sensitivity, depending on the type of traffic analyzed. Ye et al [16] proposed the use of acceleration signals for TLE on a low traffic urban road. Noteworthy, they placed the sensor under the asphalt, collecting a very high-quality signal, which leads to a high detection sensitivity of 87.5%. Despite their very high performance, all aforementioned installations are explicitly dedicated to TLE, and require precise sensors positioning to obtain good predictions and a dedicated installation procedure.

Another research trend in TLE involves the detection of vehicles through the vibration data gathered from *smartphone* accelerometers [7, 8]. Although these methods can be highly accurate, they suffer from two key problems. First, they need a high consensus in population to share the personal smartphone data. Second, they are applicable only to urban regions, where it can be expected that the same group of subscribers to the collaborative service drive in the monitored area every day.

2.2. Suburban TLE

In suburban environments, one of the possible approaches for TLE is based on computer vision [4, 18]. For example, in [4], the authors exploit a smart camera installed on top of a bridge, coupled with an Active Basis Model for object detection and a random forest to perform vehicle detection and classification. While this algorithm reaches very high accuracy in high traffic situations (up to 92.11%), the performance is strongly influenced by the light conditions of the bridge, decreasing to as much as 74.82%. This behaviour shows one of the most critical drawbacks of employing images instead of ultrasounds, acceleration or magnetic fields, which are not influenced by weather-related factors or lighting conditions. Moreover, as anticipated in Section 1, intelligent cameras are expensive and power-hungry sensors, especially if compared to low-cost MEMS-accelerometers used in SHM installations, making them incompatible with energy-efficient TLE installations.

In [14], the authors apply a classification tree to data coming from a magnetic sensor to obtain a very high TLE accuracy (99.9%). However, this high performance is mainly attributable to the particular position monitored, namely a freeway exit, with a maximum of a single-vehicle crossing at low speed at any time.

Lastly, the work of [9] presents a framework to estimate traffic using accelerometers installed on pre-stressed cables of a viaduct for SHM purposes, coupled with anomaly detection techniques. This has the advantage of not requiring additional installations, as in most other approaches, nor collaboration from users, as in [19, 20]. One clear limitation of SHM-based TLE is that of being only applicable to specific structural road elements such as bridges, viaducts, overpasses etc. However, a national road system may include tens of thousands of such structures, an increasing percentage of which is equipped with SHM sensors, in an effort to improve their security and avoid tragic accidents. Furthermore, large viaducts and bridges are typically parts of critical highway backbones, making their TLE particularly critical. Compared to the work of [9],

which is based on the same installation and sensors as this one, but used unsupervised data analysis techniques, we propose a different approach leveraging supervised learning based on a short-term and inexpensive video data collection phase, which does not require infrastructure work. While several other literature works considered supervised models for *dedicated* TLE installations, we are the first, to the best of our knowledge, to assess their effectiveness for traffic monitoring based on pre-existing SHM installations.

As shown in Section 5, this approach allows us to achieve good performance even though our estimation is based on sensors that are not explicitly dedicated to TLE, and therefore are installed in sub-optimal locations, namely on the pre-stressed viaduct tendons, rather than directly on the road.

3. Structural Health Monitoring Installation

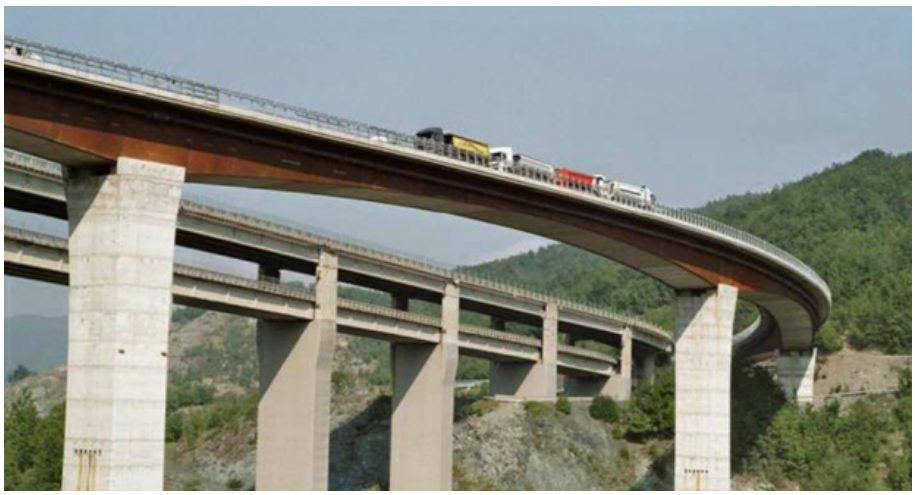

Figure 1: Aerial view of the viaduct under analysis.

In this study, we consider a concrete viaduct located in Italy, which has been monitored with a SHM sensor network since 2017. An aerial view of the viaduct is given in Fig. 1. The viaduct is a composite box girder, sustained by five concrete pillars; the total length of the viaduct ($\sim$580 m) is divided in five hyperstatic spans of length 67 m-112 m and one isostatic span ($\sim$43 m long) for a total of 6 spans. Each span is formed by a steel caisson inside which there are 12 prestressing cables. The height of the cross section of the main beam varies from 6.0 m (at the bearings) to 3.0 m (at the center-line of each span). Figure 2 shows the whole structure, together with the real-time SHM sensors system. After the construction, a series of unbound tendons (27 strands each) have been placed, prestressed, and anchored to the abutments. Several deviators are placed throughout the length of the viaduct. Their mechanical properties allow the structure to absorb greater stresses.

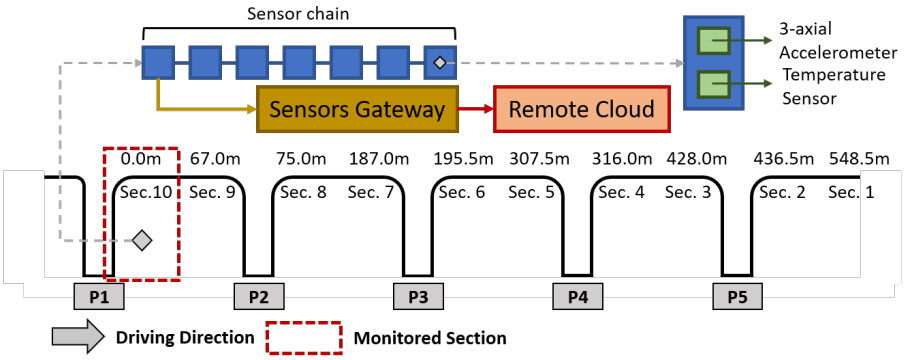

Figure 2: Synthetic overview of the viaduct. For TLE, we analysed the Section 10, where the video has been recorded. The input features are gathered from 7 wired sensors whose data are stored on a private cloud.

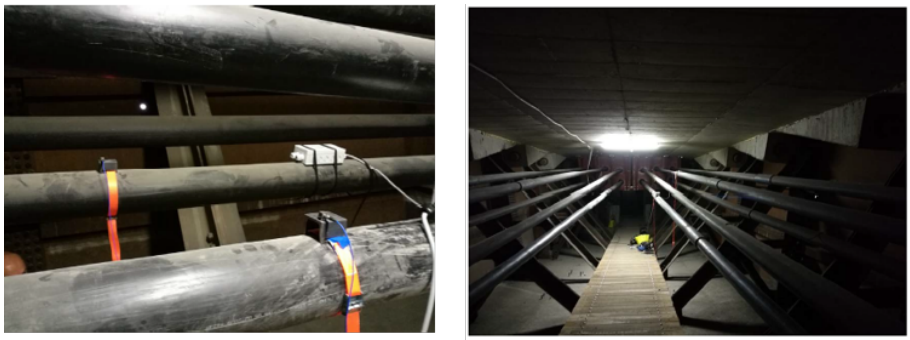

Figure 3: Internal installation of the sensors on the prestressed tendons.

The structural safety of a prestressed bridge is highly dependent on the durability of its cables. However, the steel tendons can breaks under the presence of corrosion by aggressive water-based agents. Indeed, in recent years, the structure under test suffered the failure of a tendon, highlighting the need to monitor the state of health of the prestressed elements. For this reason, a continuous monitoring system composed of 90 sensor-nodes has been installed between June and September 2017 for real-time data acquisition. Specifically, the sensors are installed on the tendons as shown in Figure 3, and monitor their natural vibration frequency, in response to external excitation. Drifting of this frequency from the ideal value computed for a cable without flexural stiffness, as well as unexpected high amplitude peaks, are considered signs of structural degradation, and trigger alarms sent to the viaduct maintainers.

3.1. Sensor Network

The 90 sensors which make up the SHM installation, fully active and gathering data since September 2017, are evenly distributed on the ten sections

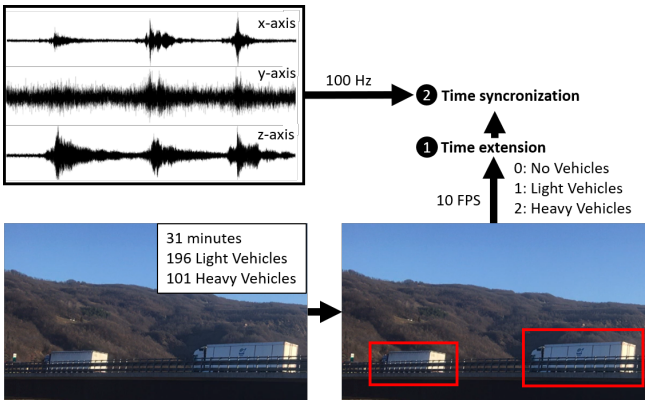

Figure 4: Dataset structure with both supervised and unsupervised data.

of the viaduct shown in Figure 2. At the time of installation, the sensors were positioned in the upper part of the cables, near the pillars, as shown in Figure 3.

In this study, we exploit the data of a single section, specifically section 10, which corresponds to the area of the road that was framed with the camera used for the labeling process, which provides the ground truth reference for TLE. This section comprises 7 sensors, mounted on different tendons.

Each node of the sensor network is an intelligent edge-platform, which comprises:

- A STM32F405RG microcontroller (MCU), which takes care of pipelining the data gathering, the filtering operations on the raw input data, and data transmission. The STM32F405RG includes an ARM 32-bit Cortex-M4 processor with FPU as main computational unit. Moreover, it is equipped with 1 MB of Flash for storing constants and code, and 192 kB of SRAM for temporary data. The maximum operating frequency is 168 MHz.
- A LIS344ALH [21] 3-axial Micro Electro-Mechanical Systems (MEMS) analog accelerometer, with a full-scale range of $\pm 2/\pm 6g$, to gather vibration data.
- A HTS221 [22] humidity and temperature sensor, whose outputs are not used in this work.

The LIS344ALH accelerometers gather linear acceleration in the three directions, x, y, z, with an angle of $90° \pm 2\%$ between each other. The data are initially sampled from the MCU at a frequency of 25.6 KHz. They are then filtered using a 6-state FIR filter, and downsampled by means of averaging to reach a frequency of 100 Hz, reducing each window of 256 values to a single measurement. This averaging increases the precision of measurements with respect to direct sampling at 100Hz, while producing a manageable amount of output data [23]. The sensor outputs are then sent to one of the two gateways installed on the viaduct (one for each side) through a CAN-BUS connection.

The gateway, which is based on a Raspberry Pi v3 module [24], stores the data from the sensors and forwards them to the cloud. Precisely, the data are sent through a 5GHz point-to-point Wi-Fi link to a "Ubiquity Nano M5" station located near pillar P2, approximately in the middle of the viaduct, which in turn uses a 4g cellular link to transmit to a cloud server.

At the time of installation, the scope of this sensor network was the monitoring of the prestressed cables to control deterioration and prevent further breakages of one or more tendons, which would impair the structural integrity of the bridge. Therefore, contrary to the sensor installations found in the literature analyzed in Section 2, we underline that in this work we exploit a pure SHM installation, which was never intended for TLE. Therefore the number and placement of sensors is not optimal for measuring vibrations due to passing vehicles, compared for instance to a solution based on sensors positioned directly at the road level, as in [16].

Another issue caused by the fact that the installation is aimed at SHM, and not TLE, is that the system parameters are not sized to *continuously* transmit data to the cloud. In contrast, due to transmission costs and storage limitations, only 15 minutes of data can be transmitted every 4 hours. This is not a problem for SHM, which monitors long-term degradation effects, but is critical for TLE, as the traffic load can vary significantly in a few hours. Therefore, our goal in this work is: i) to demonstrate that ML algorithms can still provide accurate traffic load estimates using indirect vibration data gathered from prestressed tendons, while eliminating all additional hardware costs for TLE and ii) that such algorithms can be executed directly on the edge gateway, thus allowing the direct transmission of traffic load estimates (a single value), rather than of the raw input data, which in turn enables real-time and continuous TLE.

3.2. Dataset

The dataset used in our study consists of 31 minutes of recording of acceleration data from the 7 sensors positioned in section 10 of the viaduct described above. In order to frame the problem as a supervised learning one, we also recorded (in sync with the acceleration) a video of the road portion corresponding to section 10 of the viaduct to obtain the ground truth traffic load. The data collection was performed in the morning, between 8.00 a.m. and 9.00 a.m, which represents a good example of a *difficult* timeslot for TLE, given the slightly higher traffic compared to the other day hours. The video was recorded at 10 FPS, and was then processed with an object detection algorithm, to have an initial estimation of the presence of vehicles. A human operator later checked the entire video to correct object detection errors. This semi-automatic labeling process significantly reduced the human effort for obtaining ground truth labels.

Based on the video, we generated a label variable relative to passing vehicles. Specifically, the label assumed value 1 for each frame corresponding to a light vehicle crossing an ideal vertical line placed above the pillar of section 10, and value 2 for each crossing by a heavy vehicle. Value 0 was associated to frames without crossing vehicles. Since the viaduct has two carriageways, two consecutive frames have been labeled as described above in case of two vehicles

crossing the vertical line almost simultaneously. Importantly, having multiple carriageways complicates SHM-based TLE since, for example, the vibration induced by two lightweight vehicles crossing the monitored section simultaneously might be confused by the ML algorithms with the vibration induced by a single heavy vehicle.

Acceleration data and labels were then synchronized in time, to form a single multi-variate time series with $100 \times 1860s$ rows and 22 columns ($7 \cdot 3$ accelerations + 1 label). Since the frame rate of the video and the acceleration sampling frequency are in a 1:10 relation (10 fps vs 100Hz), slices of 10 consecutive acceleration samples received the same label. As detailed in Section 4.1, these raw labels have been converted into the actual target variable for TLE, i.e., a vehicle count number, in a pre-processing step. Fig. 4 summarizes our dataset composition and construction.

Note that the dimension of the labelled dataset is quite small since we did not have the possibility to install a permanent camera on the viaduct. Thus, while our work provides a first important demonstration of the effectiveness of supervised ML for SHM-based TLE, larger labelled datasets are key to assess the generality of this approach in a variety of scenarios (e.g., different daytimes, traffic loads, drivers behaviours, etc). Among other goals, we hope that our results will showcase the importance of labeled data for this task, and spark new interest on road infrastructure stakeholders in collecting them. For the specific viaduct considered, a new data gathering session to expand the dataset is already planned in the future.

4. Methods

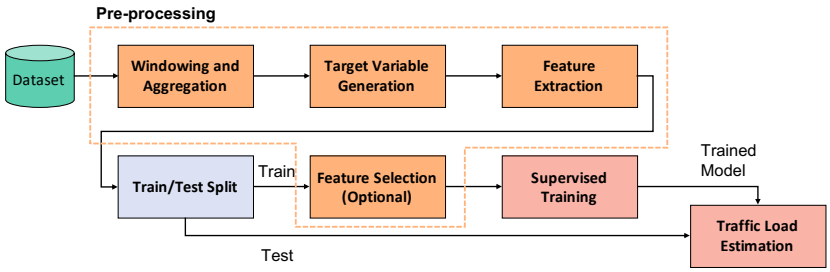

Figure 5: Proposed supervised learning pipeline for supervised learning-based traffic load estimation using SHM sensors.

Figure 5 shows our proposed supervised learning pipeline. The novelty of our work does not reside in any of the individual steps of the pipeline, which are standard for similar tasks [25, 26], but in applying them for the first time to the problem of traffic load estimation based on SHM sensors. In the rest of this section, we detail the techniques and algorithms that we selected for implementing each step of Figure 5. Specifically, Section 4.1 details the data

pre-processing (orange boxes in the figure), whereas Section 4.2 describes the ML algorithms that we trained and evaluated for TLE (red boxes).

We framed TLE as a regression problem, where the goal is to estimate the count of vehicles passing through the viaduct in a given time interval. However, we trained ML regressors to estimate *separtely* the count of light vehicles (e.g., cars) and heavy vehicles (e.g., trucks). This choice is motivated by the fact that heavy vehicles typically obey different laws and regulations compared to normal cars. Moreover, estimating the two types of traffic separately could also provide more useful and fine-grain indirect information, e.g., on commercial activity, air pollution, etc.

4.1. Data Preprocessing

This section describes the pre-processing pipeline that we implemented before feeding our data to the selected ML regressors. All pre-processing parameters have been set identically for training and validation data and for all considered classifiers, except where explicitly stated.

4.1.1. Windowing and Aggregation

We initially divided the data into sliding windows of variable length $T_{win} \in \{10, 20, 30, 40, 50, 60\}s$. The stride between two consecutive windows has been set to $2s$. As detailed in Section 5, we found that longer time windows generally yield superior TLE accuracy. However, we could not use windows longer than $60s$ due to the memory limitations of each SHM sensor. We then gathered and aligned sliding windows coming from each of the 7 3-axial SHM sensors, obtaining a final matrix of $(T, 21)$ raw acceleration samples at 100Hz, where $T = 100 \times T_{win}$ is the number of samples in each window.

4.1.2. Target Variable Generation

Since the automatic labeling process described in Section 3.2 was performed at 10fps, groups of 10 consecutive acceleration samples at 100Hz received the same label. Therefore, after forming the input windows, the corresponding target *traffic load* values for regression (y) have been set to:

$$y = \frac{\sum_{t=1}^{T}(l_t = k)}{10} \tag{1}$$

where $k = 1$ for light vehicles and $k = 2$ for heavy vehicles, and $\sum(l_t = k)$ indicates the number of samples that have label k in that window. Notice that y can also be a fractional number, when the beginning/end of the window includes only a subset of a group of 10 consecutive samples sharing the same label.

4.1.3. Feature Extraction

For each window, we extracted independent features relative to the accelerometer axis of each of the sensors. Specifically, we computed a set of 12

Name	Description
mean	Mean acceleration
std	Acceleration standard deviation
min	Minimum acceleration sample
max	Maximum acceleration sample
med	Acceleration median
kurt	Kurtosis coefficient
skew	Skewness index
rms	Root Mean Square value
sabs	Sum of the absolute values
eom	Count of elements larger than the mean
ener	Acceleration energy
mad	Median Absolute Deviation

Table 3: Acceleration features extracted for each axis of the 7 SHM sensors.

simple statistical features, reported in Table 3. These are well known and commonly used for other supervised machine learning tasks dealing with acceleration data. Therefore, a detailed description is out of the scope of this paper, and we refer the reader to the complete discussion of [27, 28, 25]. After feature extraction, each sliding window reduces to a vector of of $n = 21 \cdot 12 = 252$ variables.

4.1.4. Feature Selection

We further applied a feature selection step aimed at reducing the complexity of our models by considering only the important variables among those extracted in Section 4.1.3. For this, we used a correlation-based filter method based on the F-test [29, 25], which measures the correlation between each feature and the target variable, and then selects a user-defined number of most correlated features. While this method is less powerful than wrapper approaches, such as Recursive Feature Elimination (RFE) [30], it has the significant advantage of selecting features independently from the underlying regressor. For our study, this is beneficial because it allows to: i) apply feature selection also to regressors that do not have a natural way to measure feature importance, such as k-Nearest Neighbors and ii) use the exact same feature subsets for all regressors, in order to obtain a fair comparison among different models.

We considered 14 different settings in terms of number of features, with the following scheme: $n = 252, 225, 200, ..., 50, 25, 20, 15, 10, 5$. Feature selection has been performed based only on the training data. For each setting, we then repeated the training of all ML models from scratch.

4.2. Regression Algorithms

This section details the machine learning regressors that we considered as possible candidates for TLE based on SHM sensors. We do not aim at providing a complete theoretical discussion on each algorithm, but rather we mostly detail the settings we used for each type of regressor, and the hyper-parameters we

explored. For a more comprehensive discussion, we refer readers to machine and deep learning books such as [31, 32, 33].

As a general consideration valid for all algorithms, we found that training two separate regressors for estimating y_{light} and y_{heavy}, i.e., the traffic load of light and heavy vehicles respectively, achieved superior results with respect to training a single model to produce an output vector (y_{light}, y_{heavy}). Therefore, we trained all models listed below twice and independently as scalar regressors, once for each target output.

Linear Regressor (LR). The simplest algorithm that we considered is a linear regressor, which also serves as a baseline for more complex models. The LR estimates the traffic load as a linear combination of the input features, and its parameters have been trained to minimize the sum of squared errors between predicted and observed targets.

Random Forest (RF). We trained a Random Forest of Decision Tree (DT) regressors for TLE using the *variance reduction* criterion proposed in [34] to determine the best split variables and threshold values at each node. The forest has been trained using bootstrap aggregation, extracting random samples from the training dataset at each new tree. Predicted values $\hat{y}$ at each leaf node have been obtained as the average of the true y values of all samples assigned to that node during training. The final RF prediction is then the average of all trees' predictions in the forest. We experimented with different values of the DT depth and number of trees, i.e., the main hyper-parameters that affect the prediction performance and bias/variance trade-off of a RF. We considered trees' depths ranging in 10-400 and a number of trees between 5 and 50, and selected the combination that minimized the validation error.

K-Nearest Neighbors (k-NN). We considered a k-NN regressor as an example of instance-based ML algorithm [31]. Given that all input features of Table 3 are numerical, we measured similarity between samples as the inverse of the Euclidean distance. Moreover, we computed the prediction $\hat{y}$ simply as the average of the y values of the k neighbours. The main hyper-parameter that we explored for this algorithm is the number of neighbors k, which we varied from a minimum of $k = 1$ to a maximum of $k = 11$.

Support Vector Regressor (SVR). The last classical ML algorithm that we considered is a SVR, i.e., the equivalent of a Support Vector Machine (SVM) for regression problems. Similarly to SVMs for classification, the SVR can use both linear or non-linear kernels [31]. The hyper-parameters that we explored in our experiments are therefore the kernel function (Linear or Radial Basis Function - RBF) and the regularization parameter C, for which we considered values ranging from $C = 0.1$ to $C = 10$.

Multi-Layer Perceptron (MLP). As a first representative deep learning model [33], we experimented the MLP. We trained all MLP versions using the *Adam* gradient-based optimizer, a learning rate of 10^{-3} and a mini-batch size of 200. We used the Mean Squared Error (MSE) as a loss function and Rectified Linear Unit (ReLU) activation functions in all layers. In our experiments, we explored architectures including an input layer, an output layer (both with fixed sizes, imposed by the data) and either 2 or 3 hidden layers, all with identical

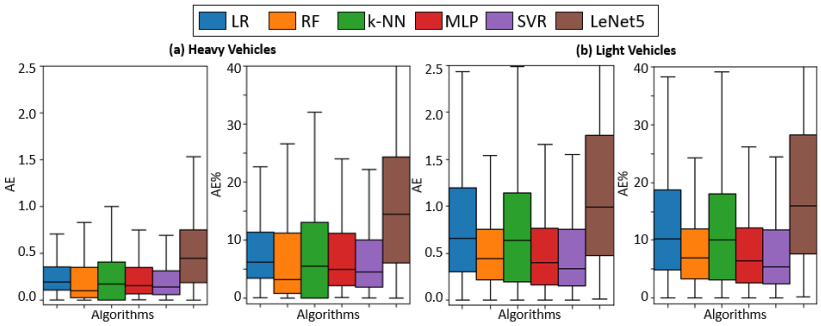

Figure 6: Performance of all the tested algorithms in terms of AE and AE% distribution for heavy (a) and light (b) vehicles. Middle black lines represent the 50^{th} percentile, i.e., the median, while colored boxes span the range from the 1st to the 3rd quartile.

hidden sizes; the latter have been varied from a minimum of 10 neurons to a maximum of 200, with an interval of 10. Forty architectures have been tested, the smallest with 2 layers and 10 neurons per layer and the largest with 3 layers and 200 neurons per layer. Although deep learning models can, in general, extract complex features starting from raw data, MLPs do not deal naturally with time-series. Therefore, in this work we fed MLPs with the same manually-extracted features used for classical algorithms (see Section 4.1.3).

Convolutional Neural Network (LeNet5). We also trained a LeNet5 [35] Convolutional Neural Network (CNN), as a representative of a more advanced deep learning model. The training parameters employed for this network are identical to the ones used for the MLP. Moreover, as for the MLP, we found that the best performances with LeNet5 are achieved when the network is fed with the same extracted features used for classic ML models. When using raw data directly, the obtained results were significantly worse due to over-fitting. Therefore, we organized the input data as a $12x21$ multi-channel 1D array, where each channel corresponds to one of the 3 axes of the 7 sensors, and is composed of the 12 extracted features for that axis. To deal with such input shape, we then transformed all 2D convolutional filters of the original network proposed in [35] into 1D convolutional filters.

Noteworthy, we also tested more complex deep neural networks such as Gated Recurrent Unit and Long-Short Term Memory Recurrent Neural Networks, as well as larger 1D CNNs. However, we do not report their performance in Section 5 since we found that, given their high complexity, these models over-fitted the small training dataset available for our task, resulting in poor generalization.

5. Results

In this section, we report the results that we obtained applying all the algorithms presented in Section 4 to our labelled TLE dataset based on SHM sensors

	Heavy Vehicles		Light Vehicles		Optimal Hyperparameters
	MAE	MAE%	MAE	MAE%	
Baseline					
Peak Detection [9]	1.7	53.2	4.6	72.8	-
Our work - ML					
LR	0.24	7.37	0.74	11.84	-
RF	0.27	8.56	0.58	9.20	Max Depth = 200, Trees = 30
k-NN	0.3	9.47	0.73	11.72	k = 7
MLP	0.26	7.89	0.59	9.92	Layers = 3, Neurons = 100
SVR	0.21	6.71	0.47	7.45	Kernel = RBF, C = 10.0
LeNet5	0.74	23.60	2.06	31.36	Default of [35]

Table 4: Comparison between the unsupervised baseline algorithm [9] and the supervised approaches proposed in our work. The comparison is done using all the extracted features and a window of 60 seconds.

measurements. We compare these algorithms with each other, as well as with the unsupervised solution introduced in [9]. Note that this is the only previous work that used SHM sensors for TLE; therefore, it serves as our primary comparison baseline. After that, we perform a detailed ablation study changing the input window dimension and the number of features passed as inputs to each model, using the feature selection procedure described in Section 4.1.4. Finally, we analyze the latency and energy consumption of all algorithms when deployed directly on the viaduct gateway, which is based on a Raspberry Pi v3 Model B (see Section 3). With this experiment, we show that an edge computing implementation of TLE is feasible and can provide benefits in terms of scalability by transmitting to the cloud only the aggregated traffic load prediction (a single number) instead of the raw data. This, in turn, could enable continuous TLE, surpassing the data storage and transmission limitations discussed in Section 3.

We compare the performance of different algorithms in terms of Absolute Error, i.e., of the difference between the predicted and true vehicle counts on each window:

$$AE = |y - \hat{y}|$$

Depending on the experiment, we report either the mean AE over all windows (i.e., the MAE) or the quartile values as aggregated error measures. Moreover, we also report the Percentage AE/MAE (denoted as AE% and MAE% respectively), which are obtained normalizing the absolute values by the *average* true traffic load over all windows ($\overline{y}$). Lastly, we also compute the R^2 score:

$$R^2 = \frac{\sum_{i=1}^{n}(\hat{y} - \overline{y})^2}{\sum_{i=1}^{n}(y - \overline{y})^2}$$

which is a standard a-dimensional coefficient representing the fraction of the variance explained by the model and varies between -1 and 1, where 1 is the value obtained by an ideal regressor.

We split our dataset randomly into train and test sets with a 70/30% proportion, and all results are reported on the test set. Model hyper-parameters have been optimized on a further 30% split of the training set left out and used

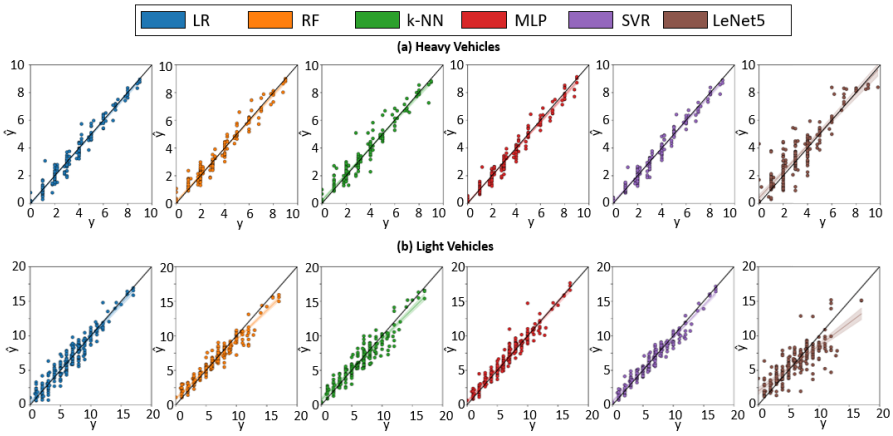

Figure 7: Real values versus predicted values on heavy and light vehicles. Colored areas represent 99% confidence intervals.

as validation set. All algorithms have been implemented using the scikit-learn Python package. All vehicles with four wheels (196 in total in the dataset) have been considered light vehicles for labelling, while all vehicles with 6 or more wheels (101) as heavy vehicles. The average number of heavy vehicles in each window of 60 seconds in our test set is 3.26 (min-max, 0.0-9.0). Light vehicles are 6.36 per window (0.0-17.0). Clearly, the average number of vehicles per window decreases when considering smaller windows.

5.1. Algorithm Comparison and Hyperparameters Exploration

Figure 6 shows the performance of all tested regressors for both light and heavy vehicles traffic estimation. The reported results refer to a window of 60s. As can be observed, the best models are the RF and the SVR. The best median AE and AE% on heavy vehicles are achieved by the RF, which reaches an impressive ~ 0.10 median AE on heavy vehicles, by inferring perfectly the vehicles count for almost 50% of the testing windows. On light vehicles, the same method achieves a 0.44 median AE and 6.9% median AE%. The SVR achieves the best *average* performance, with a MAE/MAE% as low as 0.21/6.71% for heavy vehicles and 0.47/7.45% for light ones (see Table 5).

On the other hand, as expected, the linear regressor yields the worse median AE and AE% among ML models. Nonetheless, its median percentage error is still acceptable ($< 10\%$), and its performance is interestingly comparable to those of much more complex algorithms, such as the MLP or LeNet5. This is due to the fact that the benefits of a deep learning model are typically obtained with large datasets. In contrast, the small amount of data available for our task tends to favour simpler models, less prone to over-fitting. This is proven by the significantly worse performance obtained by the tested CNN, LeNet5, compared to all other algorithms. LeNet5 achieves a MAE of 0.74 and 2.06 on heavy and

light vehicles, respectively, being more than 3× worse than the best regressor (i.e., the SVR).

Overall, all regressors except LeNet5 achieve less than 10% median AE% on heavy and light vehicle traffic estimation, thus confirming the effectiveness of a supervised learning approach for SHM-based TLE.

Figure 7 shows scatter plots of the real (y) and predicted ($\hat{y}$) traffic labels for all regressors. The black diagonal line is the ideal regressor ($y = \hat{y}$), whereas light-coloured areas represent the 99% confidence intervals of $\hat{y}$. These plots visually show that, with features extracted from a 60s window of acceleration data, all algorithms can estimate the traffic load well. Among the tested algorithms, the LeNet5 and the k-NN are the ones that experience a more significant number of outliers, which could be a problem if the goal of the TLE system is to trigger alarms above a certain traffic load threshold.

Table 5 summarizes the MAE and MAE% of all tested regressors and compares them to the unsupervised baseline of [9]. Moreover, the hyper-parameters of each algorithm that minimize the validation error are also reported.

For a fair comparison, we applied the baseline algorithm to the same data as our models, coming from the 7 sensors of section 10 of the viaduct. Note that these are different data compared to the ones considered in [9], which were taken solely during low traffic periods (i.e., during the night). In the baseline, traffic estimation is obtained by predicting the transit of a vehicle when at least $4/7$ of the sensors recognize an anomaly in a non-overlapping window of 100 ms (10 samples). The total traffic is then computed as the sum of the vehicles predicted in the same 60s windows used as inputs for the supervised methods. As we can observe, all machine learning algorithms strongly outperform the baseline. The most accurate model, the SVR, reaches a 8.1×-9.8× lower MAE% on light and heavy vehicles respectively. We can therefore conclude that, compared to an unsupervised method based on peak detection, using a labelled dataset to train a supervised classifier strongly improves the TLE performance, by learning not only the "anomalies" compared to the natural viaduct vibrations, but also the specific features of the signal generated in correspondence of heavy or light vehicles crossing the viaduct. Clearly, the reported results refer to a relatively small dataset. Consequently, the performance of the models could be negatively affected by unseen and anomalous data, e.g., caused by unconventional driver behaviours. Analyzing and possibly correcting these kinds of corner cases would require a larger and more comprehensive dataset, not currently available to us (nor publicly), but whose collection is already planned in our future work. Nonetheless, it is important to underline that the current results are already obtained in quite challenging conditions, given that we predict *daily* traffic on a two carriageways viaduct, where vehicles overcome each other, complicating the TLE.

5.2. Ablation Study

5.2.1. Window Size

This section explores the dependency between the algorithms' performance and the dimension of the data window used as input. Since changing the window

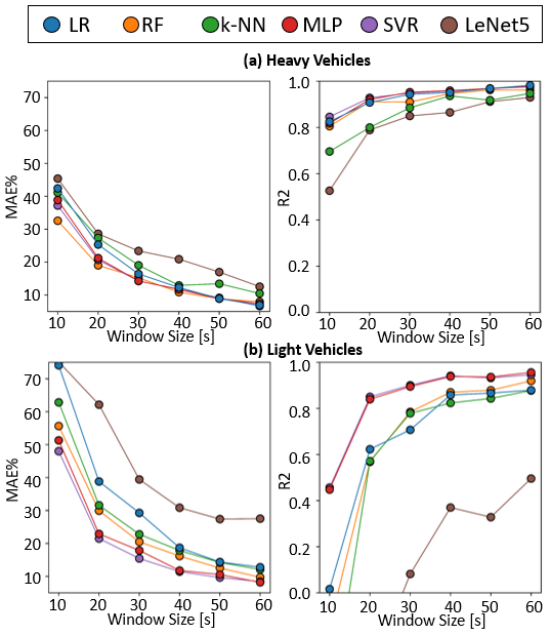

Figure 8: Window size versus performance for the tested regressors.

size also modifies the average traffic load, i.e., $\overline{y}$, we only use relative metrics (MAE% and R^2) for this experiment. Moreover, we maintain the same optimal hyperparameters settings found in Section 5.1 for each regressor. Figure 8 reports the results of the experiment. As anticipated in Section 4, since each sensor has to acquire its data window before sending them to the gateway, we did not test with windows longer than 60s due to the memory constraint of the sensor nodes.

As the figure shows, all ML algorithms' performance are positively correlated with the window size. We also observe that, as expected, the performance degradation caused by a smaller input window is more contained for the (easier) task of predicting heavy vehicles traffic. In this case, all algorithms except LeNet5 obtain a MAE% lower than 20% and a $R^2 > 0.8$ even using just a 30s window. On the other hand, since TLE is more challenging for light vehicles, due to the higher number of these vehicles, and to the weaker vibration that each of them induces on the viaduct, we need at least a window of 50s to achieve the same performance, with classic ML models.

The window size has a limited impact on the total inference time of each algorithm since it only impacts the feature extraction phase, and not the actual ML model execution. Furthermore, a latency of 60s in traffic prediction is acceptable for most applications based on TLE, such as queue monitoring or overall load estimation of the viaduct. Therefore, we conclude that a window of 60s, or in general the longest window that fits the tight memory constraints

of sensor nodes MCUs, is the best choice for this task.

5.2.2. Features Selection

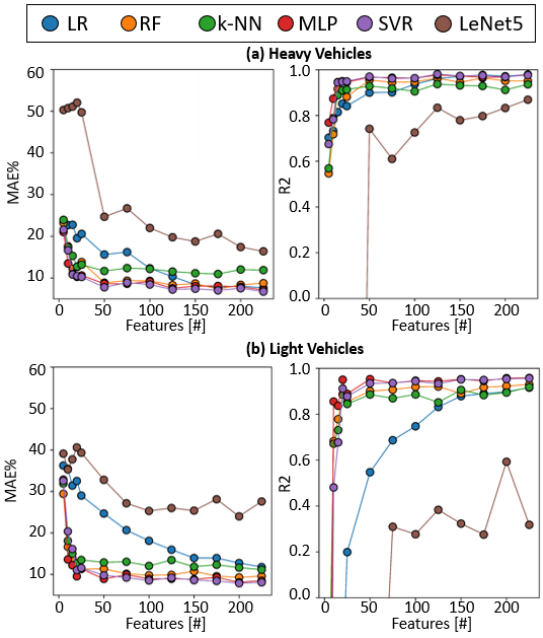

Figure 9: Performance variation with different number of selected features.

As anticipated in Section 4.1.4, we tested the ML algorithms with different numbers of input features to try to reduce their complexity without impairing the accuracy of TLE. Figure 9 reports the R^2 and the MAE% obtained in this experiment. As before, we keep the hyperparameters of each algorithm to the best values found in Section 5.1.

First, we notice that using 50 features, i.e. $1/5$ of the total 252, does not impair the results of most of the tested algorithms, allowing them to reach almost identical performance compared to the full features set. The Linear Regressor and the CNN are the only algorithms whose performance decreases significantly with the number of features, especially in the case of light vehicles TLE. Further, reducing the features to as low as 10-20 still results, for most of the classic ML methods in a MAE% lower than 15%, despite the relevant complexity reduction.

We also analyze the highest-ranked features obtained with the method described in Section 4.1.4. Among the first ten features ranked, seven belong to the *same sensor*. Since the signal quality depends on the position and on the sensor-node mounting, we conclude that this sensor receives the greatest solicitation when vehicles cross the viaduct, due to a better vibration transmission to the MEMS accelerometer. In terms of type, the most selected feature is the

	All Features		Feature Selection	
	Latency	Energy	Latency	Energy
LR	0.32 ms	4.0 mJ	0.31 ms	3.88 mJ
RF	12 ms	150 mJ	11.88 ms	146.3 mJ
k-NN	8.12 ms	101.5 mJ	3.17 ms	39.63 mJ
MLP	1.24 ms	15.5 mJ	0.88 ms	11.0 mJ
SVR	3.52 ms	44 mJ	0.95 ms	11.88 mJ
LeNet5	3.03 ms	37.98 mJ	-	-

Table 5: Latency and energy of all the algorithms measured on a Raspberry Pi v3 Model B.

median absolute deviation *mad* (see Table 3), appearing 7 times in the top-30 features. This is reasonable as a larger *mad* value correlates with wider vibration peaks caused by passing vehicles. However, $^{11}/_{12}$ of the features of Table 3 appear at least once in the top-30 list (the only one missing is *min*), confirming the goodness of our feature set.

5.3. Algorithm Deployment

Table 5.3 shows the latency and energy results obtained deploying all regressors on the Raspberry Pi v3 Model B used as viaduct gateway node. The gateways is equipped with a Broadcom BCM2837 chipset operating at 1.2GHz and 1GB of RAM, and runs a Linux operating system. Latency results are averaged over 10000 executions of each algorithm, whereas energy results are obtained with the conservative assumption that the gateway board is consuming its peak power (12.5W) throughout the execution. The table reports the results both for models trained on the full 252 features set and on a subset of highest-ranked features. Specifically, we selected the minimum number of features for each algorithm that yielded a MAE% $< 15\%$ for both light and heavy vehicles TLE. This number is 150 for LR, 10 for RF, 15 for k-NN, 20 for MLP, 15 for SVR, and 252 (i.e., *all*) for LeNet5. Note that LR and CNN require many more features to obtain the same performance of the other algorithms.

Globally, we demonstrate that implementing the TLE at the edge is feasible. As a consequence, we eliminate the limitations of a centralized cloud approach, described in Section 3, enabling continuous traffic monitoring. Among the different algorithms, k-NN and RF are by far the most costly to run. Overall, the choice among the tested regressors boils down to a cost/performance trade-off: if a 15% error is acceptable, using a Linear Regressor would lead to low latency (0.31ms) and energy consumption (3.8mJ). In contrast, if the goal is to achieve the lowest possible mean error, the SVR becomes the best choice, with a MAE% of just 6-7% using the complete feature set (see Table 5), at a still acceptable cost of 3.52ms/44mJ per inference.

6. Limitations

In this section, we summarize the limitations of our work which are individually underlined in the previous sections. First, although we are the first to demonstrate the feasibility of TLE exploiting only acceleration data gathered

from pre-installed SHM sensors, the benchmark dataset is limited to 31 minutes with a total of 196 light vehicles and 101 heavy vehicles. Therefore, further studies are demanded that extend this work to a more extended dataset or even to data coming from different types of roads (not only viaducts but also other types of bridges or suburban streets).

As an immediate consequence of having a reduced training dataset, we only explored classical machine learning solutions and shallow neural networks in this first study. While the error range resulting from some of these algorithms (such as the SVR) is acceptable for the viaduct used in our experiments, it must be underlined that, given more data, exploring state-of-the-art deep neural networks would probably allow for more precise traffic estimation.

7. Conclusions

We have presented the first application of supervised learning to the problem of traffic load estimation based on a native installation of accelerometers for SHM purposes. We have compared five different classifiers in terms of both performance and execution cost, analyzing the impact of feature extraction and selection, and deploying them on a real SHM gateway node.

From our experiments, we found out that the SVR achieves the best performance on both light and heavy traffic estimation, achieving a MAE of 0.47 light and 0.21 heavy vehicles. Compared to the state of the art [9], this corresponds to a performance improvement of 9.8× and 8.1× respectively. When deployed on the SHM gateway, the SVR consumes just 44 mJ with a latency of 3.52ms when fed with all the 252 extracted features.

Our future work will include the collection of a larger and more varied labeled dataset for SHM-based TLE, thanks to which we will assess the generality of our proposed approach. Moreover, we will also study the possibility of *combining* multiple TLE solutions to further improve the overall accuracy. For instance, the proposed approach could be complementary to a camera-based installation, forming a hybrid system that exploits the excellent accuracy of the latter in good visibility conditions, while falling back to the SHM-based strategy at night or when visibility is low. Lastly, other crucial aspects to be considered in future works on TLE are those related to security and reliability, both from the point of view of the whole system (i.e., devices and communication channels) and of the ML algorithms. Concerning the latter, important elements to be analyzed include the resilience of the different supervised ML algorithms proposed in this work to malicious/adversarial data alterations, as well as the use of dedicated techniques to implement trustworthiness checks [36], and consequently improve the reliability of TLE estimates.

Acknowledgment

This work was supported by the Italian Ministry for Education, University and Research (MIUR) under the program "Dipartimenti di Eccellenza (2018-2022)". Moreover the authors would like to thank Sacertis S.r.l. for the dataset.

References

[1] T. Afrin, N. Yodo, A survey of road traffic congestion measures towards a sustainable and resilient transportation system, Sustainability 12 (11) (2020). doi:10.3390/su12114660.

[2] Y. Guo, Q. Zhang, K. K. Lai, Y. Zhang, S. Wang, W. Zhang, The impact of urban transportation infrastructure on air quality, Sustainability 12 (14) (2020). doi:10.3390/su12145626.

[3] K. Song, J. Tai, Image-based traffic monitoring with shadow suppression, Proceedings of the IEEE 95 (2) (2007) 413–426. doi:10.1109/JPROC.2006.888403.

[4] S. Kamkar, R. Safabakhsh, Vehicle detection, counting and classification in various conditions, IET Intelligent Transport Systems 10 (6) (2016) 406–413.

[5] E. Odat, J. S. Shamma, C. Claudel, Vehicle classification and speed estimation using combined passive infrared/ultrasonic sensors, IEEE transactions on intelligent transportation systems 19 (5) (2017) 1593–1606.

[6] I. Y. Gu, M. Bolbat, Road traffic tracking and parameter estimation based on visual information analysis using self-calibrated camera views, in: 2013 Seventh International Conference on Distributed Smart Cameras (ICDSC), 2013, pp. 1–6. doi:10.1109/ICDSC.2013.6778231.

[7] P. Mohan, V. Padmanabhan, R. Ramjee, Nericell: Rich monitoring of road and traffic conditions using mobile smartphones, in: ACM Sensys, acm sensys Edition, Association for Computing Machinery, Inc., 2008, pp. 1–14, raleigh, NC, USA.

[8] R. Bhoraskar, N. Vankadhara, B. Raman, P. Kulkarni, Wolverine: Traffic and road condition estimation using smartphone sensors, in: 2012 Fourth International Conference on Communication Systems and Networks (COMSNETS 2012), 2012, pp. 1–6. doi:10.1109/COMSNETS.2012.6151382.

[9] A. Burrello, D. Brunelli, M. Malavisi, L. Benini, Enhancing structural health monitoring with vehicle identification and tracking, in: 2020 IEEE International Instrumentation and Measurement Technology Conference (I2MTC), IEEE, 2020, pp. 1–6.

[10] C. Ayyildiz, H. E. Erdem, T. Dirikgil, O. Dugenci, T. Kocak, F. Altun, V. C. Gungor, Structure health monitoring using wireless sensor networks on structural elements, Ad Hoc Networks 82 (2019) 68 – 76. doi:https://doi.org/10.1016/j.adhoc.2018.06.011.

[11] D. Porcarelli, D. Spenza, D. Brunelli, A. Cammarano, C. Petrioli, L. Benini, Adaptive rectifier driven by power intake predictors

for wind energy harvesting sensor networks, IEEE Journal of Emerging and Selected Topics in Power Electronics 3 (2) (2015) 471–482. doi:10.1109/JESTPE.2014.2316527.

[12] M. Reyer, S. Hurlebaus, J. Mander, O. E. Ozbulut, Design of a wireless sensor network for structural health monitoring of bridges, in: 2011 Fifth International Conference on Sensing Technology, 2011, pp. 515–520. doi:10.1109/ICSensT.2011.6137033.

[13] Q. Wang, J. Zheng, H. Xu, B. Xu, R. Chen, Roadside magnetic sensor system for vehicle detection in urban environments, IEEE Transactions on Intelligent Transportation Systems 19 (5) (2017) 1365–1374.

[14] H. Dong, X. Wang, C. Zhang, R. He, L. Jia, Y. Qin, Improved robust vehicle detection and identification based on single magnetic sensor, Ieee Access 6 (2018) 5247–5255.

[15] H. Liu, J. Ma, T. Xu, W. Yan, L. Ma, X. Zhang, Vehicle detection and classification using distributed fiber optic acoustic sensing, IEEE Transactions on Vehicular Technology 69 (2) (2019) 1363–1374.

[16] Z. Ye, H. Xiong, L. Wang, Collecting comprehensive traffic information using pavement vibration monitoring data, Computer-Aided Civil and Infrastructure Engineering 35 (2) (2020) 134–149.

[17] H. Zhao, D. Wu, M. Zeng, S. Zhong, A vibration-based vehicle classification system using distributed optical sensing technology, Transportation Research Record 2672 (43) (2018) 12–23.

[18] J. G. Chen, T. M. Adams, H. Sun, E. S. Bell, O. Büyüköztürk, Camera-based vibration measurement of the world war i memorial bridge in portsmouth, new hampshire, Journal of Structural Engineering 144 (11) (2018) 04018207.

[19] P. Harikrishnan, V. P. Gopi, Vehicle vibration signal processing for road surface monitoring, IEEE Sensors Journal 17 (16) (2017) 5192–5197.

[20] F. Seraj, B. J. van der Zwaag, A. Dilo, T. Luarasi, P. Havinga, Roads: A road pavement monitoring system for anomaly detection using smart phones, in: Big data analytics in the social and ubiquitous context, Springer, 2015, pp. 128–146.

[21] lis344alh sensor, STMicroelectronics.
URL `https://www.st.com/en/mems-and-sensors/lis344alh.html`

[22] Hts221 sensor, STMicroelectronics.
URL `https://www.st.com/resource/en/datasheet/hts221.pdf`

[23] A. Girolami, F. Zonzini, L. De Marchi, D. Brunelli, L. Benini, Modal analysis of structures with low-cost embedded systems, in: 2018 IEEE International Symposium on Circuits and Systems (ISCAS), IEEE, 2018, pp. 1–4.

[24] R. P. Foundation, Raspberry pi 3 module b (2016).
URL https://www.raspberrypi.org/documentation/hardware/raspberrypi/

[25] T. Cerquitelli, D. Jahier Pagliari, A. Calimera, L. Bottaccioli, E. Patti, A. Acquaviva, M. Poncino, Manufacturing as a data-driven practice: Methodologies, technologies, and tools, Proceedings of the IEEE 109 (4) (2021) 399–422. doi:10.1109/JPROC.2021.3056006.

[26] D. Cannizzaro, M. Zafiri, D. Jahier Pagliari, E. Patti, E. Macii, M. Poncino, A. Acquaviva, A Comparison Analysis of BLE-Based Algorithms for Localization in Industrial Environments, Electronics 9 (1) (2019) 44. doi:10.3390/electronics9010044.

[27] A. O. Salau, S. Jain, Feature extraction: A survey of the types, techniques, applications, in: 2019 International Conference on Signal Processing and Communication (ICSC), 2019, pp. 158–164. doi:10.1109/ICSC45622.2019.8938371.

[28] W. Caesarendra, T. Tjahjowidodo, A review of feature extraction methods in vibration-based condition monitoring and its application for degradation trend estimation of low-speed slew bearing, Machines 5 (4) (2017). doi:10.3390/machines5040021.
URL https://www.mdpi.com/2075-1702/5/4/21

[29] P. Bevington, D. Robinson, Data Reduction and Error Analysis for the Physical Sciences, McGraw-Hill Education, 2003.

[30] I. Guyon, J. Weston, S. Barnhill, V. Vapnik, Gene Selection for Cancer Classification using Support Vector Machines, Machine Learning 46 (1) (2002) 389–422. doi:10.1023/A:1012487302797.
URL https://doi.org/10.1023/A:1012487302797

[31] C. M. Bishop, Pattern recognition and machine learning, springer, 2006.

[32] C. C. Aggarwal, et al., Neural networks and deep learning, Springer 10 (2018) 978–3.

[33] I. Goodfellow, Y. Bengio, A. Courville, Deep Learning, The MIT Press, 2016.

[34] L. Breiman, J. H. Friedman, R. A. Olshen, C. J. Stone, Classification and regression trees, Wadsworth & Brooks/Cole Advanced Books & Software, 1984.

[35] Y. LeCun, B. Boser, J. S. Denker, D. Henderson, R. E. Howard, W. Hubbard, L. D. Jackel, Backpropagation applied to handwritten zip code recognition, Neural computation 1 (4) (1989) 541–551.

[36] M. Cheng, S. Nazarian, P. Bogdan, There is hope after all: quantifying opinion and trustworthiness in neural networks, Frontiers in artificial intelligence 3 (2020) 54.