DEEP LEARNING-BASED GROSS VEHICLE WEIGHT ESTIMATION IN BRIDGE WEIGH-IN-MOTION BY USING SENSORS IN ONE CROSS-SECTION

Bence Szinyéri
PhD student, Department of Automation and Applied Informatics, Faculty of Electrical Engineering and Informatics, Budapest University of Technology and Economics
1111 Budapest, Műegyetem rkp. 3., e-mail: szinyeribence@edu.bme.hu

Bence Kővári
associate professor, Department of Automation and Applied Informatics, Faculty of Electrical Engineering and Informatics, Budapest University of Technology and Economics
1111 Budapest, Műegyetem rkp. 3., e-mail: kovari@aut.bme.hu

Dénes Kollár
associate professor, Department of Structural Engineering, Faculty of Civil Engineering, Budapest University of Technology and Economics
1111 Budapest, Műegyetem rkp. 3., e-mail: kollar.denes@emk.bme.hu

István Völgyi
associate professor, Department of Structural Engineering, Faculty of Civil Engineering, Budapest University of Technology and Economics
1111 Budapest, Műegyetem rkp. 3., e-mail: volgyi.istvan@emk.bme.hu

Attila László Joó
associate professor, Department of Structural Engineering, Faculty of Civil Engineering, Budapest University of Technology and Economics
1111 Budapest, Műegyetem rkp. 3., e-mail: joo.attila@emk.bme.hu

Abstract

Gross Vehicle Weight (GVW) estimation plays a crucial role in ensuring the safety, maintainability, and sustainability of road transportation by identifying and filtering out overloaded vehicles. Bridge Weigh-in-Motion (B-WIM) systems enable the determination of axle loads, vehicle speeds, axle spacings, and other vehicle parameters as they cross a bridge, using data from strain gauges installed beneath the bridge deck. This paper proposes a novel deep learning-based GVW estimation method designed for B-WIM systems equipped with sensors at a single cross-section. Unlike conventional axle load estimation-based GVW estimators, the proposed method does not rely on vehicle speed estimation or axle detection steps. The method is evaluated on an annotated dataset of 91 vehicles measured on the Monostori Bridge. Results demonstrate B+ accuracy with a Mean Absolute Percentage Error of 2.47% for GVW estimation in accordance with the COST 323 Weigh-in-Motion classification standard. Furthermore, the proposed solution can be integrated into standard B-WIM pipelines using ensemble models. Tests on the same dataset indicate that the ensemble approach may outperform existing B-WIM pipelines in GVW estimation accuracy by reducing the Mean Absolute Percentage Error by 0.1%.

Keywords: *Bridge Weigh-in-Motion, deep learning, gross vehicle weight, strain gauge*

1. Introduction

Overloaded vehicles that exceed legal weight limits contribute significantly to the deterioration of bridges and road networks, making effective traffic monitoring essential for infrastructure maintenance. Accurate estimation of axle loads and gross vehicle weight (GVW) is therefore a key concern in civil engineering (Paul et al., 2023). Weigh-in-Motion (WIM) systems allow these vehicle parameters to be measured without requiring vehicles to stop. There are two main types of WIM systems: Pavement Weigh-in-Motion (P-WIM) and Bridge Weigh-in-Motion (B-WIM). P-WIM systems are embedded in the pavement, whereas B-WIM systems rely on sensors installed under the bridge deck. B-WIM technology is non-invasive; it can be installed without disrupting traffic flow, and sensor maintenance or replacement is straightforward. B-WIM systems also tend to offer a more cost-effective solution compared to P-WIM systems. Moreover, the accuracy of P-WIM systems is more sensitive to pavement conditions (Burnos et al., 2017) such as surface roughness or deterioration, leading to a rapid decline in performance over time. In contrast, B-WIM systems may maintain consistent accuracy over the long term. These advantages highlight B-WIM systems as a promising alternative to P-WIM systems.

In most B-WIM systems, strain gauges are installed under the bridge deck (Carraro et al., 2019). When no additional sensors are embedded in or placed on the road surface, B-WIM systems are often referred to as Nothing-on-Road (NOR) systems (Yu et al., 2018). Vehicle parameters are calculated based solely on the data collected by these strain gauges. In some cases, additional sensors such as accelerometers (Lorenzen et al., 2022) are placed under the bridge deck to support the decision-making process of the B-WIM system. Recent research has also explored the integration of camera systems into B-WIM setups (Dan et al., 2019), as visual sensors are typically required in Traffic Surveillance Systems (TSS) (Kawakatsu et al., 2023) to identify overloaded vehicles. This paper presents a method that has no need for cameras or additional sensors; thus, relying only on strain gauges for GVW estimation.

B-WIM systems typically calculate not only the axle loads but also the speed and axle spacings of vehicles. Most axle load estimation algorithms (Carraro et al., 2019), including those based on the Moses algorithm (Moses, 1979), depend on either the axle spacings or, at minimum, the speed of the vehicle (He et al., 2019). The speed is usually determined using sensors placed at two or more cross-sections of the bridge (Cantero et al., 2024). Sensors placed within the same cross-section are referred to as a sensor array in this context. When a vehicle crosses the bridge, especially the section where the sensors are installed, at a near-constant speed, the strain signals recorded at different cross-sections will appear very similar, with the primary difference being a time shift. By analysing this time delay and knowing the distance between sensor arrays, the vehicle speed can be accurately calculated.

This paper proposes a Gross Vehicle Weight (GVW) estimation algorithm that requires only a single cross-section of sensors. Unlike conventional approaches, the algorithm does not rely on prior knowledge of the vehicle speed or axle spacings. The proposed method is suitable for use on bridges where strain gauges have been installed at only one cross-section, as well as in standard B-WIM systems as an alternative GVW estimation approach.

The proposed GVW estimation algorithm achieved an error of 2.47% on a measurement-based annotated dataset of captured on an orthotropic steel bridge deck. Integrating this approach with a previously developed B-WIM pipeline that requires two sensor arrays could further reduce the GVW estimation error to 2.03%.

The paper is organized as follows: Section 2 provides a literature review. The proposed deep learning-based GVW estimation algorithm and its predecessor vehicle time window detection module are presented in Section 3. The proposed method is evaluated on a measurement-based annotated dataset, and the achieved results are discussed in Section 4. Finally, conclusions and future research directions are summarized in Section 5.

2. Literature review

This section first provides a brief review of B-WIM pipelines, followed by a discussion of deep learning-based algorithms. Finally, the datasets used for training B-WIM methods are presented.

2.1. B-WIM pipeline

A standard B-WIM pipeline extracts the following features: the time window of each vehicle, vehicle speed, axle spacings, and axle load estimates. The vehicle time window (Kawakatsu et al., 2018) defines the time interval and lane in which the vehicle crosses the bridge above the sensor array. The vehicle speed is typically determined using the data of more sensor arrays (Cantero et al., 2024). Axles are typically determined by detecting the timestamps when each axle passes over the sensors (Lorenzen et al., 2022). Based on the speed and axle timestamps, the axle spacings and the vehicle trajectory can be calculated. Finally, using this trajectory, the axle loads are estimated (Carraro et al., 2019). Some algorithms also determine the transverse position (Yu et al., 2018) of the vehicle to improve axle load calculation. Other parameters, such as tire contact areas and axle widths, are generally ignored.

Moses’ algorithm (Moses, 1979) is a foundational B-WIM method for axle load estimation. Most modern axle load estimation algorithms in B-WIM systems (Carraro et al., 2019) are modifications or extensions of this original approach. Moses’ algorithm aims to calculate the axle loads of a vehicle, assuming its trajectory is known. It is based on the concept of influence lines or influence surfaces. The influence surface at a point on a structure describes how the shear force, bending moment, or typical strain components at that point change as a unit load moves along the structure (Szinyéri et al., 2023). In practice, it is often observed that when the same vehicle crosses the bridge in different transverse positions, the measured summed strain signals in each cross-section do not vary significantly with transverse position. This observation supports the use of influence lines (Obrien et al., 2006), which depend only on the longitudinal position of the unit load, and are typically defined for the sum of strain changes at a set of points on the structure.

Most B-WIM pipelines use at least two sensor arrays, primarily to enable vehicle speed estimation. When two cross-sections are placed close to each other, the strain signals induced by a vehicle crossing these sections are very similar. Assuming near-constant speed, the vehicle speed can be calculated by detecting local peaks in the strain signals at both cross-sections and determining the time delay between them (Kalhori et al., 2017). Dividing the known distance between the cross-sections by this time delay results in the vehicle speed.

Kawakatsu et al. (2019, 2021) proposed a method for estimating vehicle speed using only a single cross-section of sensors. Their solution, based on deep learning, achieved a mean absolute error of 3.29 km/h. This level of error may significantly affect the performance of axle load estimation modules that depend on accurate speed input. The approach of Kawakatsu et al. used a convolutional neural network (CNN) architecture combined with dense layers for feature extraction.

This paper aims to create a novel deep learning-based GVW estimator that uses the data of only one cross-section as well, but it will not depend on the speed estimation step. It will rely only on the vehicle time window detection step.

2.2. Dataset

Deep learning-based algorithms require large datasets for training; otherwise, there is a high risk of overfitting (Goodfellow et al., 2016). In deep learning-based B-WIM systems, the input consists of data collected by sensors installed on the bridge. While building a signal database is relatively straightforward by archiving measured data, the challenge arises when supervised training is needed, because proper labelling must be provided. For gross vehicle weight (GVW) estimation, one potential labelling method involves stopping vehicles crossing the bridge and measuring their axle loads using static axle weighing scales. However, this approach is highly resource-intensive. Kawakatsu et al. (2023) proposed a solution that leverages data from pre-installed axle weighing stations, though this is not generally applicable. Overall, labelling remains a key challenge that must be addressed for the widespread adoption of supervised deep learning-based B-WIM algorithms.

Synthetic datasets (Szinyéri et al., 2023) offer a solution for generating large datasets for deep learning applications in B-WIM systems. In this field, synthetic datasets can be created using the concept of influence surfaces or influence lines, as previously discussed. If the influence surfaces for the sensors are known and the parameters of the vehicles crossing the bridge are specified, the strain values at the strain gauge positions can be accurately simulated. Szinyéri et al. (2023) introduced a publicly available synthetic dataset, BME-Simulated I (BME-S1), based on this concept. The Monostori Bridge, a continuous, five-span cable-stayed bridge with orthotropic steel main girder with open cross-section has been investigated. The Monostori Bridge has a length of 600 m, connecting Hungary and Slovakia between Komárom and Komárno. BME-S1 contains simulated data for over 100,000 vehicles crossing the Monostori Bridge, covering both one-lane and multi-lane scenarios. The vehicles in the dataset have between 2 and 9 axles, and the data includes complex configurations such as convoys, tandem and tridem axles, adding complexity to the dataset.

In addition to this, a measurement-based annotated dataset was also created for the same bridge where the axle loads of the crossing vehicles were measured by certified axle weighing scales (Szinyéri et al., 2024). Dynamic effect of wind (Chen et al., 2010) and temperature (Xiao et al., 2023), vibration caused by vehicles and asphalt roughness (Ho et al., 2020) are not taken into account by direct simulation due to their high complexity. Quasi-static influence surfaces are used in this way. These additional effects are handled as some Gaussian noise added to the simulated signals (Wu et al., 2020). However, the deep learning-based solution can be trained on the synthetic dataset, it is also necessary to create a measurement-based annotated dataset to validate the method. This dataset may contain a smaller number of vehicles in accordance with the COST 323 WIM-standard (Jacob et al., 1998). A measurement-based dataset has been already captured for the Monostori Bridge that contains the gross vehicle weight parameters among other vehicle parameters of 91 vehicles.

3. Contribution

This section presents the proposed novel deep learning-based gross vehicle weight (GVW) estimator. The solution is built on a convolutional neural network (CNN) architecture that uses 1D kernels, as the task involves processing time-series signals. The proposed GVW estimator offers two main advantages: it does not depend on speed estimation or axle detection steps; thus, it depends only on vehicle time window detection; and it requires no additional sensor arrays, enabling GVW estimation using data from a single sensor array.

First, a window detection module is established. Then, the input, output and label representation, the architecture of the neural network and the training process for the proposed solution will be discussed in this section later.

3.1. *Vehicle time window detection*

A vehicle time window detection module can be established based on a previously introduced deep learning-based vehicle detection method (Kawakatsu et al., 2018) with some modifications. The module uses a Convolutional Neural Network (CNN) architecture that processes strain signals as a [B, Ch, T]-sized input, and the output of the network is a [B, L, T]-sized tensor, where B is the batch number, Ch is the number of signal channels, L is the number of lanes and T is the length of the signal, the number of time steps. The detection problem is a binary classification task, so the detection pipeline includes postprocessing steps such as sigmoid-based thresholding, majority voting using a mean averaging kernel, and merging of proximate vehicle windows to produce final detections as a mask. The mask shows if a vehicle was crossing above the cross-section of the sensor array in the corresponding lane and at the corresponding timestamp (1 if a vehicle is detected, 0 otherwise). Focal Loss is employed to address class imbalance during training for this binary classification task, which is an extended form of the well-known Binary Cross Entropy (BCE) loss function (Lorenzen et al., 2022).

3.2. *Deep learning-based GVW estimation*

The proposed GVW estimator solution depends on the window detection module. The output of the window detection module is the lane and the time interval of the vehicle, showing where and when the vehicle crossed the bridge. One input of the neural network is the mask derived from the lane and time interval information, the same mask as the label or output of the window detection module. The tensor created from this mask is [B; L; T]-sized, where B is the batch size, L is the lane count, and T is the number of time samples. The other input is the normalized strain signal measured by the strain gauges. Some preprocessing steps are applied to the raw strain signal in order to get the normalized signal. As a first step, the amplitude of the summed strain signals in the lane of the vehicle is calculated. The summed strain signal in each lane means the sum of the signals measured by the strain gauges placed under each lane. This amplitude is called the normalizer constant later. Some summed strain signals are derived from the raw strain signals to create the neural network input from them. The specific strain signals to be summed are configuration dependent; however, it is recommended that these derived strain signals should contain the sum of strain signals in each lane. A specific configuration of strain signals will be presented later in Section 4. These derived signals are then normalized with the previously calculated normalizer constant; it means a division by this constant. Additionally, a further normalization step should be also involved in the preprocessing steps. The strain signals should be dilated by a constant strain value so that the mean strain value in the time interval of the vehicle should be equal to 0. In this way, a tensor of size [B; Ch, T] is created, where Ch is the number of channels, which is the number of derived strain signals. The channel count is configuration dependent. The signal tensor and the mask tensor are then concatenated, so a [B; Ch+L; T]-sized tensor is the input of the convolutional neural network. *Figure 1* illustrates an example input of the neural network. In this case, Ch equals to two, since the sum of strain signals in each lane is given as an input. It can be seen where the mask equals to zero based on the background of the image. The vehicle is crossing the bridge in Lane 1. The red double-arrow shows that the amplitude of the summed strain signal in the lane of the vehicle equals to one. The proposed CNN is a Fully-Convolutional Neural Network (FCNN) architecture, which means that no dense layers are present at the end of the neural network. The convolutional blocks in the neural network work with the default stride 1 so the output dimension only differs in the channel number to the input dimension; the length of the output (T) is the same as before the neural network computation. The output of the CNN is a [B; L; T]-sized tensor. The next step is to calculate the GVWs from these tensors, one scalar for each vehicle. It can be created easily. The GVW is calculated by summing the output values in the window of each vehicle, and then it is multiplied by the previously calculated normalizer constant. In this way, a one-dimensional [B]-sized tensor is created where the *i*-th value is the GVW of the *i*-th vehicle. This process is summarized in *Equation (1)* where *f* denotes the CNN as a function, *M* is the mask tensor, and *S* is the signal tensor, and *cat* is the concatenation function through the channel dimension, and $\sum$ operator sums all values of a tensor. Batch size is not taken into consideration in this description for simplicity, multiplication means element-wise tensor multiplication in this formula.

$$GVW = \sum f(cat(S, M)) * M\,, \quad \text{where } S \in \mathbb{R}^{Ch \times T}, M \in \mathbb{R}^{L \times T}, f: \mathbb{R}^{(Ch+L) \times T} \to \mathbb{R}^{L \times T}, \sum: \mathbb{R}^{L \times T} \to \mathbb{R} \tag{1}$$

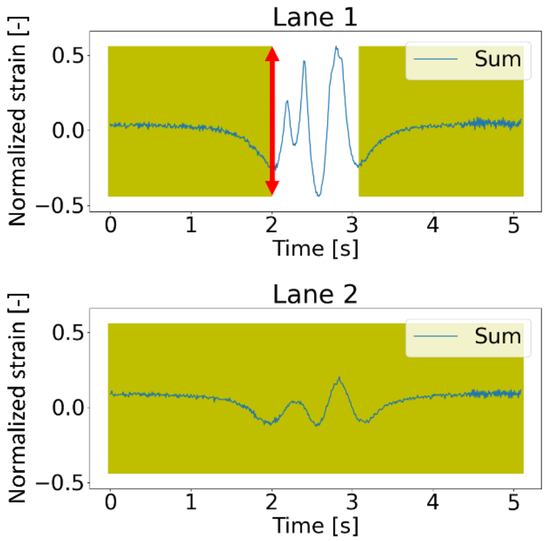

Figure 1. *Example of neural network input for a two-lane bridge where vehicle is crossing the bridge in Lane 1. Background colour defines the mask for the two lanes, white background color represents where the mask value equals to one.*

The used neural network architecture is a CNN as mentioned earlier. Leaky ReLU activation function (Goodfellow et al., 2016) is used in this CNN; thus, the gradients can backpropagate without the problem of shrinking or exploding gradients. The well-known residual blocks (He et al., 2016) are also used in the solution. During training, the L1-loss function is used. The Adam optimizer (Goodfellow et al., 2016) is used with a cosine-annealing learning rate scheduler (Loshchilov et al., 2016).

Data augmentation technique has also been used during training. Gaussian noise is added to the normalized strain signals in each case the following way: first, a level of error is sampled from a uniform distribution of range 0–5% (0–0.05), then a normally distributed noise is sampled with zero mean and the standard deviation of the sampled error level. Finally, this noise is added to the normalized signal.

4. Results

The proposed solution was implemented and trained for the Monostori Bridge that crosses the Danube between Komárom (Hungary) and Komárno (Slovakia). This bridge has already been investigated by the Authors (Szinyéri et al., 2023) as mentioned earlier in Section 2.

Strain gauges are installed in two cross-sections of the bridge. Sensors placed in the same cross-section are called a sensor array. Sixteen strain gauges are installed in the main sensor array of the Monostori Bridge, labelled from T1 to T16 as shown in Figure 2. The monitoring system also contains another secondary sensor array, which contains 10 strain gauges. The data of this secondary sensor array can be ignored by the usage of the proposed GVW estimator solution. The distance between the two cross-sections is 6 meters. The proper locations of these sensor arrays are illustrated in Figure 3. Previously, a full B-WIM pipeline has already been proposed by the Authors (Szinyéri et al., 2024). This solution is referred to as B-WIM pipeline using two cross-sections. It is considered as a baseline method later.

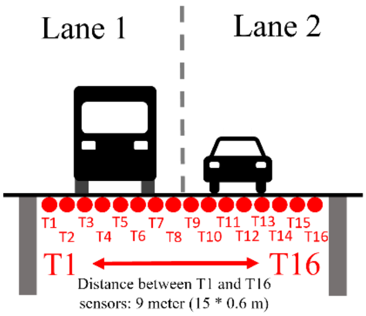

Figure 2. *Sixteen strain gauges in the main sensor array installed under the two-lane Monostori bridge*

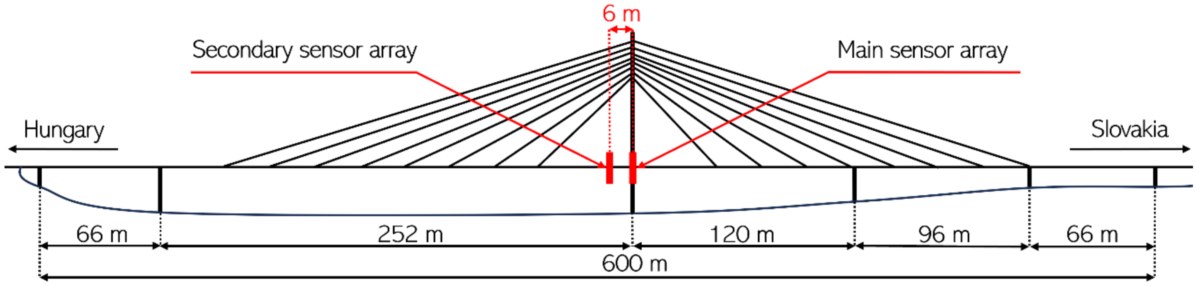

Figure 3. *Location of main and secondary sensor arrays*

It has already been mentioned that the channels of the neural network input are configuration dependent in the case of the proposed deep learning-based GVW estimator network. The so-called CNN-Edge architecture has been implemented. It means that the sum of strain signals in each lane and the sum of strain signals at the edge of each lane are given as the input of the GVW estimator. It means that the sum strain signals of T1-T8, T9-T16, T1-T5 and T12-T16 sensors are the input of the neural network. The CNN-Edge architecture is illustrated in Figure 4. It can be seen that the architecture awaits for an input signal with 6 channels; 4 channels are coming from the strain signal and 2 are coming from the mask concatenated to it. The architecture contains about 840k parameters in this way. Convolutional blocks on the image represent their parameters in *out channel number/kernel size/dilation/stride* format. As mentioned earlier, the stride is equal to one in every convolutional block. *Figure 5* shows a neural network input that can be used with the CNN-Edge configuration. Other properties of the training process are also given in *Table 1*.

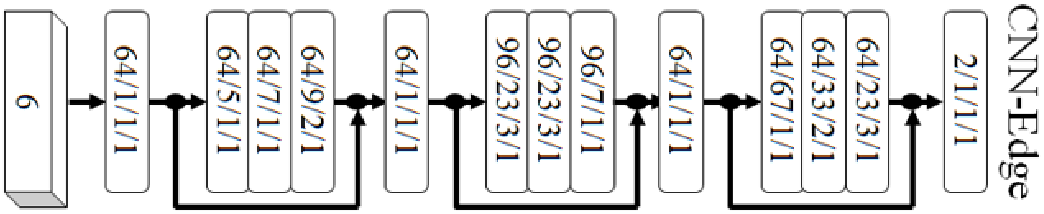

Figure 4. *CNN-Edge architecture*

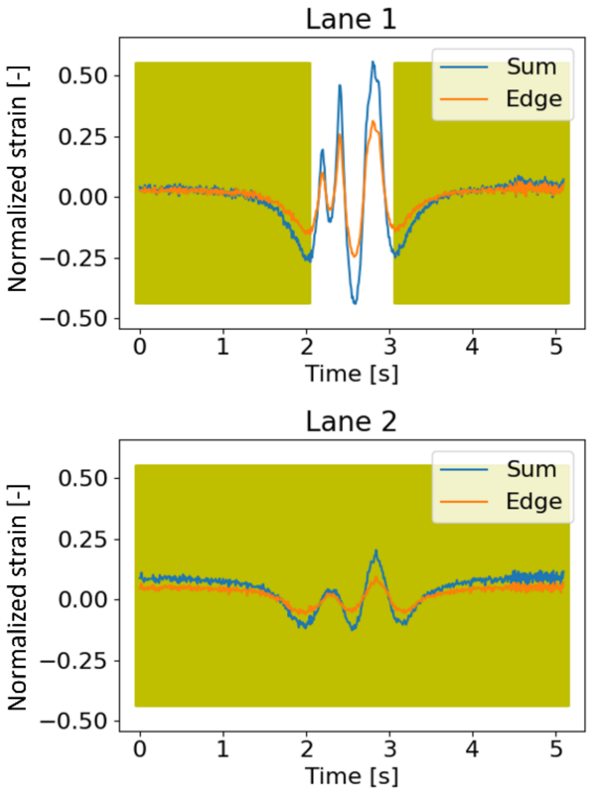

Figure 5. *Example of neural network input when the CNN-Edge architecture is used. Sum signals show the summed strain signals of T1-T8 and T9-T16 sensors, while Edge sensors show the summed strain signals of T1-T5 and T12-T16 sensors.*

Table 1
Most important properties of the training process are given for the Monostori Bridge

Aspect		**Description/value**
Trainable parameter count		840,000
Optimizer		Adam
Learning rate (at start)		0.001
Batch size		32
Train/validation split [%]		80/20
Learning rate scheduler	Type	Cosine annealing scheduler
	Min. learning rate	0.0001
	Annealing period	16,384 batch

A measurement-based annotated dataset has already been captured on the Monostori Bridge as mentioned earlier. This dataset contains the ground-truth axle loads of 91 vehicles. These measurements were captured in 2023 October and 2024 February. Vehicles were stopped and axle loads were measured statically by a certified axle weighing scale based on the OIML R 76-1 documentation (OIML, 2006) guaranteeing accurate measurement between certain limits. These vehicles vary in speed, axle loads, lateral position and axle count. The proposed deep learning-based GVW estimator has been evaluated on this dataset. The solution has been trained on the BME-S1 synthetic dataset. Table 2 summarizes the results achieved by the proposed GVW estimator solution and the B-WIM pipeline using two cross-sections. Mean Absolute Percentage Error (MAPE), standard deviation of percentage errors and COST 323 WIM classification (Jacob et al., 1998) of each pipeline can be read in that table. The COST 323 is a classification standard used for categorizing the proposed and the previously established two cross-section-based B-WIM pipeline. COST 323 divides B-WIM systems into classes from A to E, where A is the most accurate, E is the least accurate, and B+ is better than B. COST 323 also discusses how environmental conditions and the size of the dataset can be taken into consideration. The reproducibility, repeatability and environmental conditions are at the R2 (strictest) level according to the COST 323 standard in our case due to the variety of the vehicles and the environmental conditions (measurements in autumn and winter as well). It can be seen the two cross-section-based solution achieves lower errors as expected but the gap is only 0.34% in the case of MAPE metric; however, the proposed solution is only capable of B+ classification in this way.

An ensemble model can be created using the previously established B-WIM pipeline and the proposed GVW estimator. Ensemble model means the following in the case of deep learning models: the weighted average of the neural network outputs will be the output of the ensemble model. In our case, the weighted average is only a simple average. *Table 3* shows the results achieved by the established ensemble model. It can be seen that the proposed ensemble model achieved lower errors when compared to the original B-WIM pipeline using two cross-sections. It means that the proposed GVW estimator is worth to be built in the previously proposed B-WIM pipeline. It supports Gross Vehicle Weight estimation, and can reduce the MAPE metric even by 0.1%.

Table 2

Results (GVW estimation errors) achieved on the measurement-based annotated dataset of 91 vehicles by the proposed solution and the previously presented two cross-section-based B-WIM pipeline

Metric	Proposed solution	Using two cross-section B-WIM-pipeline
MAPE [%]	2.47	2.13
Standard deviation of percentage errors [%]	3.33	2.83
COST 323 [A-E]	B+	A

Table 3

Results (GVW estimator errors) of the ensemble model (proposed + two cross-section solution)

Metric	Ensemble model (Proposed + two cross-section)	Using two cross-section B-WIM-pipeline
MAPE [%]	2.03	2.13
Standard deviation of percentage errors [%]	2.64	2.83
COST 323 [A-E]	A	A

Table 4 shows how the multi-lane scenarios and Gaussian-noise added to signals are reducing the accuracy of the proposed GVW estimator on a synthetic dataset. Errors are also presented for the training dataset on which the CNN has been trained, and also for the test dataset. Reliable conclusions can be made based on the errors of the test dataset since it has not been used during the training process. The proposed GVW estimator is examined using two aspects. First, it is examined using traffic conditions. Three kinds of benchmark are available for this case. Multi-lane cases are where vehicles are crossing the bridge at both lanes at the same time period, while in the case of one-lane scenarios, the vehicles are crossing the bridge in a single lane, so the other lane is empty. Heaviest-based evaluation means that only the vehicle with the heaviest average axle load is benchmarked for each scenario. It is interesting in multi-lane scenarios because it shows how the GVW of the heaviest vehicle is predicted. Heaviest-based evaluation is relevant because in applications of WIM systems, the aim is to calculate the GVW of the heaviest vehicles in the most accurate way. The other aspect is the presence of noise; the same benchmarks are executed for the same scenarios by adding 5% Gaussian noise to the signal of each test case. The percentage of noise is defined the same way as in Section 3. Results show that the standard deviation of percentage errors become nearly 4% higher in multi-lane subset than in the one-lane subset so there are more outliers in those cases. The errors on the heaviest-based subsets are closer to the one-lane errors than to the multi-lane errors. It can be also seen that 5% noise causes an additional error up to 2.01% in all cases regarding all metrics.

The whole BME-S1 synthetic dataset and some measurement-based scenarios are publicly available (Szinyéri et al., 2023). The main parameters of the BME-S1 dataset are summarized in *Table 5*. It shows that the parameters of the vehicles show complex variety in speed, axle spacing, axle widths, tyre widths, etc.

Table 4

Errors achieved on different subsets of the BME-S1 dataset by the proposed GVW estimator. Achieved errors are given with (w/) and without (w/o) adding 5% Gaussian-noise to the signal.

Metric		One-lane		Multi-lane		Heaviest-based	
		train	test	train	test	train	test
MAPE [%]	w/o noise	1.07	1.20	2.87	3.63	1.80	2.50
	w/ noise	2.39	2.37	4.12	4.57	2.93	3.37
St. dev of percentage errors [%]	w/o noise	1.44	1.58	5.54	5.66	2.58	3.47
	w/ noise	3.11	3.03	7.02	6.78	3.89	4.51

Table 5

Main parameters of the BME-S1 synthetic dataset are summarized in the table

Parameters	BME-S1 synthetic dataset Train	Test
Vehicle speed [m/s]	10–30	
Number of vehicles in convoys	3	
Small axle spacing [m]	0.8–2.1	
Large axle spacing [m]	2.1–6.0	
Axle width [m]	1.3–3.0	
Delay between vehicles in convoys [s]	1–4	
Average axle load [kN]	4–200	10–100
Tyre width [m]	0.1–0.7	
Contact length of tyre [m]	0.05–0.40	

5. Conclusions and future work

The proposed GVW estimator solution depends on a time window detection module. It is unique in the field of B-WIM methods since other methods rely on at least the speed estimation method. The proposed GVW estimator method can also work based on the data proved by only a sensor array of one cross-section. Results show that the proposed solution achieves 2.47% average error on a measurement-based annotated dataset. It achieved B+ classification using COST 323 benchmark, which can be still acceptable for Structural Health Monitoring purposes, and it can still support filtering overloaded vehicles. If the proposed solution is used with the previously established two cross-section-based pipeline, then the ensemble performs lower errors than the B-WIM pipeline individually. It means that the established GVW estimator can be even useful in regular B-WIM systems.

Future work should focus on implementing the proposed method on other bridges as well. As mentioned earlier, the Monostori Bridge has an orthotropic steel bridge deck. The algorithm should be examined in the future on bridges with concrete box-girders as well. The process of the implementation for other bridges may be really similar to the process used in the case of this bridge: first, influence surfaces or influence lines should be identified. Then, the CNN should be trained using a synthetic dataset considering traffic conditions (number of lanes, direction of traffic, etc.). Finally, the trained GVW estimator module must be validated using a measurement-based annotated dataset.

6. Acknowledgements

The project supported by the Doctoral Excellence Fellowship Programme (DCEP) is funded by the National Research Development and Innovation Fund of the Ministry of Culture and Innovation and the Budapest University of Technology and Economics. We acknowledge the Digital Government Development and Project Management Ltd. for awarding us access to the Komondor HPC facility based in Hungary. We thank to the Road Network Protection Department of the Hungarian Public Roads for their support in creating the mentioned measurement-based annotated dataset.

References

[1] Paul, D., & Roy, K. (2023). Application of bridge weigh-in-motion system in bridge health monitoring: a state-of-the-art review. *Structural Health Monitoring*, 22 (6), 4194–4232. **https://doi.org/10.1177/14759217231154431**

[2] Burnos, P., & Rys, D. (2017). The effect of flexible pavement mechanics on the accuracy of axle load sensors in vehicle weigh-in-motion systems. *Sensors*, 17 (9), 2053. **https://doi.org/10.3390/s17092053**

[3] Carraro, F., Gonçalves, M. S., Lopez, R. H., Miguel, L. F. F., & Valente, A. M. (2019). Weight estimation on static B-WIM algorithms: A comparative study. *Engineering Structures*, 198, 109463. **https://doi.org/10.1016/j.engstruct.2019.109463**

[4] Yu, Y., Cai, C. S., & Deng, L. (2018). Nothing-on-road bridge weigh-in-motion considering the transverse position of the vehicle. *Structure and Infrastructure Engineering*, 14 (8), 1108–1122. **https://doi.org/10.1080/15732479.2017.1401095**

[5] Lorenzen, S. R., Riedel, H., Rupp, M. M., Schmeiser, L., Berthold, H., Firus, A., & Schneider, J. (2022). Virtual axle detector based on analysis of bridge acceleration measurements by fully convolutional network. *Sensors*, 22 (22), 8963. **https://doi.org/10.3390/s22228963**

[6] Dan, D., Ge, L., & Yan, X. (2019). Identification of moving loads based on the information fusion of weigh-in-motion system and multiple camera machine vision. *Measurement*, 144, 155–166. **https://doi.org/10.1016/j.measurement.2019.05.042**

[7] Moses, F. (1979). Weigh-in-motion system using instrumented bridges. *Transportation Engineering Journal of ASCE*, 105 (3), 233–249. **https://doi.org/10.1061/TPEJAN.0000783**

[8] He, W., Ling, T., OBrien, E. J., & Deng, L. (2019). Virtual axle method for bridge weigh-in-motion systems requiring no axle detector. *Journal of Bridge Engineering*, 24 (9), 04019086. **https://doi.org/10.1061/(ASCE)BE.1943-5592.0001474**

[9] Cantero, D., & Kim, C. W. (2024). Convoluted reciprocity and other methods for vehicle speed estimation in Bridge Weigh-in-Motion systems. *Journal of Bridge Engineering*, 29 (2), 04023114. **https://doi.org/10.1061/JBENF2.BEENG-6422**

[10] Kawakatsu, T., Aihara, K., Takasu, A., & Adachi, J. (2018). Deep sensing approach to single-sensor vehicle weighing system on bridges. *IEEE Sensors Journal*, 19 (1), 243–256. **https://doi.org/10.1109/JSEN.2018.2872839**

[11] Szinyéri, B., Kővári, B., Völgyi, I., Kollár, D., & Joó, A. L. (2023). A strain gauge-based Bridge Weigh-In-Motion system using deep learning. *Engineering Structures*, 277, 115472. **https://doi.org/10.1016/j.engstruct.2022.115472**

[12] OBrien, E. J., Quilligan, M. J., & Karoumi, R. (2006, March). Calculating an influence line from direct measurements. *Proceedings of the Institution of Civil Engineers-Bridge Engineering* (Vol. 159, No. 1, pp. 31-34). Thomas Telford Ltd. **https://doi.org/10.1680/bren.2006.159.1.31**

[13] Kawakatsu, T., Aihara, K., Takasu, A., Adachi, J., Wang, H., & Nagayama, T. (2021, June). Fully-neural approach to vehicle weighing and strain prediction on bridges using wireless accelerometers. *ICASSP 2021–2021 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)* IEEE (pp. 8027–8031). **https://doi.org/10.1109/ICASSP39728.2021.9414433**

[14] Kalhori, H., Alamdari, M. M., Zhu, X., Samali, B., & Mustapha, S. (2017). Non-intrusive schemes for speed and axle identification in bridge-weigh-in-motion systems. *Measurement Science and Technology*, 28 (2), 025102. **https://doi.org/10.1088/1361-6501/aa52ec**

[15] Goodfellow, I., Bengio, Y., Courville, A., & Bengio, Y. (2016). *Deep learning* (Vol. 1, No. 2). Cambridge: MIT Press.

[16] Szinyéri, B., Kovari, B., Völgyi, I., Kollár, D., & Joó, A. L. (2024). A deep learning-based BWIM system of a cable-stayed bridge. In *Bridge Maintenance, Safety, Management, Digitalization and Sustainability* CRC Press (pp. 1452–1459). **https://doi.org/10.1201/9781003483755-170**

[17] Chen, S. R., & Wu, J. (2010). Dynamic performance simulation of long-span bridge under combined loads of stochastic traffic and wind. *Journal of Bridge Engineering*, 15 (3), 219–230. **https://doi.org/10.1061/(ASCE)BE.1943-5592.0000078**

[18] Xiao, Q., Chen, C., Liu, Z., Zhou, L., Liu, Y., Jiang, Z., Yang B. & Tang, L. (2023, December). Temperature-induced response reconstruction for the dynamic reliability assessment of bridge girders. *Structures*, 58, 105374. **https://doi.org/10.1016/j.istruc.2023.105374**

[19] Ho, H., & Nishio, M. (2020). Evaluation of dynamic responses of bridges considering traffic flow and surface roughness. *Engineering Structures*, 225, 111256. **https://doi.org/10.1016/j.engstruct.2020.111256**

[20] Wu, Y., Deng, L., & He, W. (2020). BwimNet: A novel method for identifying moving vehicles utilizing a modified encoder-decoder architecture. *Sensors*, 20 (24), 7170. **https://doi.org/10.3390/s20247170**

[21] Jacob, B., & O'Brien, E. J. (1998, September). European specification on weigh-in-motion of road vehicles (COST323). In: *Proceedings of second European conference on weigh-in-motion of road vehicles, Held Lisbon, Portugal* (pp. 14–16).

[22] He, K., Zhang, X., Ren, S., & Sun, J. (2016). Deep residual learning for image recognition. In *Proceedings of the IEEE conference on computer vision and pattern recognition* (pp. 770–778). **https://doi.org/10.1109/CVPR.2016.90**

[23] Loshchilov, I., & Hutter, F. (2016). Sgdr: *Stochastic gradient descent with warm restarts*. arXiv preprint arXiv:1608.03983.

[24] International Organization of Legal Metrology (2006). OIML R 76-1: *Non-automatic weighing instruments – Part 1: Metrological and technical requirements – Tests*. Technical Report, OIML, https://www.oiml.org/en/files/pdf_r/r076-1-e06.pdf, accessed: 2025-07-31