Adversarially-Trained Tiny Autoencoders for Near-Sensor Continuous Structural Health Monitoring

Alessio Burrello
DEI, University of Bologna
Bologna, 40136, Italy
alessio.burrello@unibo.it

Giacomo Sintoni
DEI, University of Bologna
Bologna, 40136, Italy
giacomo.sintoni@unibo.it

Davide Brunelli
DII, University of Trento
Trento, 38123, Italy
davide.brunelli@unitn.it

Luca Benini
DEI, University of Bologna
IIS, ETH Zurich
lbenini@iis.ee.ethz.ch

***Abstract*—Structural Health Monitoring (SHM) systems are increasingly employed in many civil structures such as buildings, tunnels and viaducts. Typical installations consist of sensors that gather information and send it to a central computing unit, which then periodically analyzes the incoming data and produces an assessment of the structure conditions. To avoid the transmission of a huge amount of raw data and reduce latency in the detection of structural anomalies, recent works focus on moving computation on the sensor nodes. This work shows that a small autoencoder, which fits the tiny 2 MB memory of a typical microcontroller used for SHM sensor nodes can achieve very competitive accuracy in detecting structural anomalies as well as vehicle passage on bridges by leveraging adversarial training based on generative adversarial networks (GANs). We improve accuracy over state-of-the-art algorithms in two use-cases on real-standing buildings: i) predicting anomalies on a bridge $(+7.4\%)$ and ii) detecting vehicles on a viaduct $(2.30\times)$.**

I. Introduction & Related Works

The significant expansion of cities, highways and local roads has raised the importance of monitoring crucial road infrastructure such as bridges, viaducts, or tunnels, whose failure is potentially life-threatening for drivers. Structural Health Monitoring (SHM) focuses on collecting data to track with fine-grained spatial resolution the health status of civil structures, leveraging new technologies such as low-cost sensors and machine learning algorithms for sensor data analysis. A recent trend in SHM is the transition from periodic inspections to continuous monitoring. Streaming and analyzing data continuously is the new frontier to identify damages with low latency and with high precision. The ultimate goal of continuous SHM is thus the accurate and low-latency identification [1], [2], localization, and characterization of damage and deterioration, through installed sensors, such as accelerometers [3], ultrasonic sensors [4], or cameras [5].

Continuous SHM on extensive structures monitored by large arrays of sensors introduces a new "data deluge" challenge, i.e., performing anomaly detection on the enormous amount of gathered data. This challenge can be tackled by moving data analysis to the edge, i.e. detecting anomalies directly at the sensors. For instance, the authors of [2] use a simple principal component analysis (PCA) to identify anomalies for structural damage detection on an oil extraction platform. Similarly, in [6], the authors use fiber optic sensors and PCA coupled with a trained threshold on the reconstruction error to detect anomalies. In [7], a PCA, a fully-connected autoencoder, and a convolutional autoencoder are compared to predict the anomalies on an aged bridge.

All these edge detection methods suffer from very scarce data on real anomalies (which obviously are rare), that is the main limiting factor in achieving high accuracy. We propose employing generative adversarial networks (GANs) in the SHM field to tackle this problem. In [8], the authors have demonstrated that GANs can improve the generalization of autoencoders for anomaly detection without any increment in computing effort. We present the following contributions:

- We demonstrate that adversarial training[1] improves the performance of an autoencoder for two different benchmarks of structural health monitoring; noteworthy, accuracy boost compared to the baseline autoencoder comes without any additional cost for inference;
- We present two different real use-cases: the first consists of distinguishing between two different bridge states before and after a maintenance intervention, considered as aged and healthy states of the bridge (anomaly detection – AD). The second is the traffic estimation on a two-lanes viaduct (vehicle detection – VD).

We compare the results of an autoencoder trained via adversarial training with a classically trained autoencoder and state-of-the-art results published on these two use-cases. On the VD benchmark [9], we improve the unsupervised state-of-the-art results, reducing the mean absolute error (MAE) of real traffic prediction from 4.40 vehicles to 1.86 vehicles. We improve the state-of-the-art accuracy of a tuned PCA on the AD benchmark from 98.2% to 98.8%. Compared to three autoencoders trained without adversarial learning, we improved the accuracy of 48.3%, 7.4%, and 3.1%. Noteworthy, the proposed autoencoder fits the tiny memory of a microcontroller (MCU) such as the STM32H7 (2 MB FLASH), thus being suitable for processing directly on the sensor nodes of the SHM network.

II. Background & Real Scenario

A. Benchmarks: Civil structures

1) SS335 Viaduct: The AD benchmark, first introduced in [7], is built from data of an aged bridge located on the state highway SS335 in northern Italy. This viaduct underwent a maintenance intervention to strengthen the structure of a single section of the viaduct. The availability of continuous monitoring data before and after the intervention is a quite rare occurrence, and creates a unique opportunity to assess the capability of algorithms for continuous SHM to detect structural changes in presence of a clear positive ground truth. Figure 1.a) gives a high-level overview of the viaduct with its monitoring system.

[1]With adversarial training, we refer to the training based on a min-max game between a generator and a discriminator. We do not refer to the training with adversarial examples.

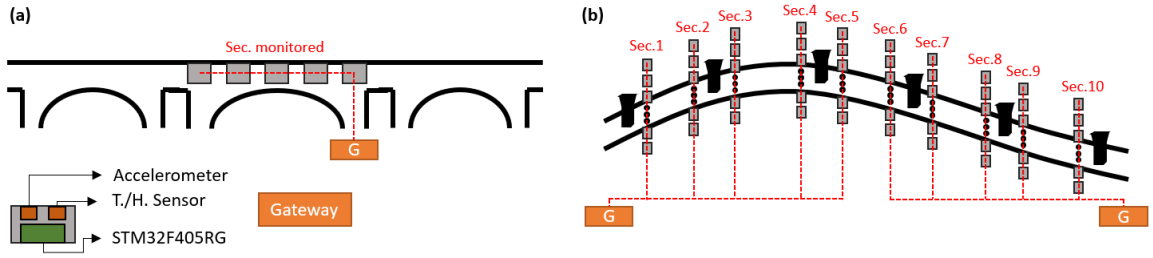

Fig. 1. *a)* Structure of the AD benchmark, with a single monitored section with 5 sensors. *b)* The Viaduct for the VD benchmark with 90 sensors installed.

2) Roccaprebalza Viaduct: This viaduct is located on a highway in the centre of Italy. It is a composite box girder with prestressed tendons used to strengthen the bridge. Given that the structural safety of the bridge is strongly dependent on the status of the tendons, monitoring their condition is a key feature for monitoring the bridge status. After experiencing the failure of one tendon, a continuous monitoring system was installed. Since 20 September 2017, the complete system has been continuously operational and gathers acceleration data from the viaduct, which are then stored in the cloud for subsequent analysis. Sub-figure *b)* of Fig. 1 depicts the structure of the viaduct and of its monitoring system. We use the data from the SHM installation to detect the passage of vehicles. In this use-case the focus is not on structural degradation anomalies, but on measuring the traffic on the bridge, which needs to be carefully tracked given the presence of a damaged tendon. The ground truth is provided by video data collected (and manually labeled) during a short term (30 minutes) expedition.

B. SHM Framework

Both the viaducts are equipped with similar chains of sensors. The SS335 viaduct is equipped with a chain of 5 sensors connected to a unique gateway for data streaming to the cloud and basic analysis. All the sensors are installed on the section which underwent the maintenance intervention. The Roccaprebalza viaduct has a chain of 90 nodes spread throughout the whole length of the viaduct in ten different sections, each with $\sim$ 9 sensors installed. The two gateways collect the data from half of the sensors each and send them to the cloud.

The sensors are produced specifically for viaduct monitoring, with three main key components: a LIS344ALH analog tri-axial accelerometer[2], a HTS221 temperature, and humidity sensor, and a STM32L476VGTx microcontroller (MCU).

C. Autoencoders and GANs

Autoencoders are neural networks whose scope is to reconstruct input data after compressing and unpacking them, i.e., not through trivial identity mapping. In an autoencoder, we can distinguish the encoder (E), which compresses the input data, and the decoder (D), whose aim is to reconstruct the input data starting from the output of E. Noteworthy, these models are trained to maximize the similarity of input x and output $\bar{x}$ without employing data labels, learning the optimal hidden space (output of E) that preserves the key features of the input data. During training, the loss function is constructed to reward a similarity metric between the original and the reconstructed signal. Autoencoders are exploited for either signal compression or anomaly detection. For the latter use case, reconstructed signals, similar to those encountered during training, lead to low reconstruction error. Conversely, receiving signals with different characteristics than those used for training will result in an imperfect reconstruction and are identified as anomalies if they surpass a threshold in the reconstruction error.

Generative adversarial networks (GANs) [10] are composed of two networks, the generator (G) and the discriminator (D). The scope of the generator is to generate new unseen synthetic data, similar to the training ones, starting from random noise $z \in \mathbb{R}^r$. Simultaneously, the discriminator is trained to distinguish real training data from synthetic generated ones. These two networks are trained with a min-max game. We maximize the similarity between generated synthetic data and training data while minimizing the loss in distinguishing the two classes.

The key idea in our work is to leverage the GAN to improve the autoencoder anomaly detection capability by generating adversarial (hard to detect) data and training the autoencoder to detect it correctly in a min-max game.

III. Material and Methods

A. Adversarial Training of the Autoencoder

Fig. 2 shows the four blocks of which the algorithm is composed. The first two blocks, the *Encoder* (E) and the *Generator* (G) are the ones that are learned during training and re-used in the inference step. The other two blocks, the *Discriminator 1* (D_1) and the *Discriminator 2* (D_2) are used for the adversarial training, and thus to improve the performance of E and G. In a nutshell, E and G blocks collaborate first to compress and then reconstruct the input signal. D_1 is trained to distinguish between random noise and the output of E, while D_2 can distinguish if the reconstructed signals are generated from real or synthetic data.

To train these blocks, we simultaneously minimize three losses associated with the output of G, D_1, and D_2:

- $\mathcal{L}_{\mathcal{R}}$, the reconstruction loss associated with G. As in normal training of autoencoders, we minimize the mean square error (MSE) between the input and the reconstructed signal, resulting in the optimization of E and G weights.
- $\mathcal{L}_{\mathcal{H}}$, the loss associated to D_1. In this case, a min-max loss is optimized: first, we want to maximize the similarity between the output distribution of the encoder and the distribution of a random variable z, forcing E to learn a good compressed representation of input signals; second, we want to optimize D_1 to distinguish the two distributions. In particular, we optimize the Wasserstein loss [11] as specified in [8]

$$\min \max \mathbb{E}_{z\sim\mathbb{P}_z}[D_1(z)] - \mathbb{E}_{x\sim\mathbb{P}_x}[D_1(E(x))]$$

Where $\mathbb{E}_{z\sim\mathbb{P}_z}$ point to the distribution of the random variable z. In this way, the two players, E and D_1, try to fool each other: D_1 maximizes the distinction of the two distributions,

[2]STM, LIS344ALH, https://www.st.com/resource/en/datasheet/lis344alh.pdf

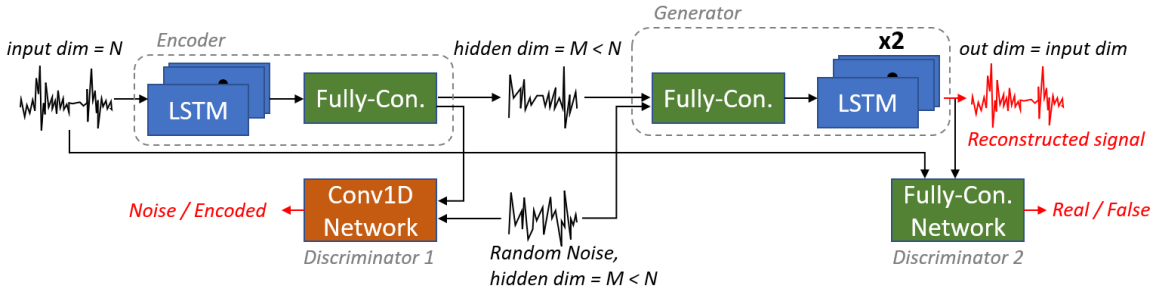

Fig. 2. High-level diagram of our GAN architecture. The first two blocks, the encoder and the generator, are the ones used also during the inference step. The other two, the discriminators 1 and 2, are only needed during the training to improve the quality of the results of the encoder and the generator, respectively.

E tries to generate similar distributions to the one of z that can not be distinguished.

- $\mathcal{L}_{\mathcal{D}}$, the loss associated with D_2, again formulated as a Wasserstein adversarial loss. $\mathcal{L}_{\mathcal{D}}$ is used for the min-max game between G and D_2. Similar to the previous loss, D_2 learns to distinguish reconstructed signals generated from synthetic or real data, while G tries to generate signals as similar as possible to either the random noise or the encoded signal. The formulation is the following

$$\min \max \mathbb{E}_{x\sim\mathbb{P}x}[D_2(x)] - \mathbb{E}_{z\sim\mathbb{P}z}[D_2(G(z))].$$

$\mathcal{L}_{\mathcal{R}}$, $\mathcal{L}_{\mathcal{H}}$ serves E to generate better-compressed signals, whereas $\mathcal{L}_{\mathcal{R}}$, and $\mathcal{L}_{\mathcal{D}}$ aid G to generate better reconstruction. Noteworthy, the two additional losses and the two additional Discriminators are used only during training, while the inference is identical. After the training, the autoencoder (E + G) learned to reconstruct input signals from the training set optimally.

B. Task adaptation of the Autoencoder

1) AD benchmark: The goal is to differentiate the two different states of the bridge, the healthy one and the aged one. During inference, we feed 5 seconds windows from the healthy state to teach the autoencoder to reconstruct this data type well. The two types of signals are distinguished based on a threshold applied on the MSE between input and reconstructed signals: the autoencoder indeed badly reconstructs (high MSE) anomalous signals that have not been seen during training. We set the threshold to the mean MSE achieved by the GAN on the windows of normal data of the validation set.

2) VD benchmark: In the second benchmark, the goal is the identification of the vehicles moving on the viaduct and the estimation of the traffic. The autoencoder is employed to clean 1 second input windows of the signal, therefore removing interfering noise to improve the distinction of very close peaks from nearby vehicles. Then, a trivial peak counting algorithm can be used to compare our cleaning strategy to the state-of-the-art [9]. We use the area-under-peak to distinguish between heavy and light vehicles.

C. Networks structure

E consists of a single LSTM layer with 100 units and a fully connected layer with 20 neurons in our work. G first decompresses the signal with a fully connected layer with 50 neurons and then applies two LSTM layers with 64 units and an up-sampling layer interposed between them. D_1 and D_2 are 1D convolutional networks. D_1 is composed of four identical blocks with a 1D convolutional layer with 64 filters of dimension 5×1, a LeakyRelu with $\alpha = 0.2$ as activation, and a Dropout of 0.25. A final fully-connected layer performs the classification random vs encoded signal. D_2 has two dense layers with 100 neurons each and a final one with a single neuron for classification. As in D_1, the LeakyRelu and the Dropout are used after each layer with the same parameters of D_1.

We use 800-epochs training with a batch size of 512, Adam optimizer a l_r of 0.001 and the Wasserstein loss to train our network. The deployed network for inference, i.e., E and G, only occupies 805 kB, fitting the small 2MB memory of a typical MCU used in SHM networks, such as an STM32H7.

As a baseline, we also trained the autoencoder (E + G) without adversarial training. In this case, we used the same parameters and only $\mathcal{L}_{\mathcal{R}}$ as network loss.

IV. Experimental Results

A. Datasets

1) SS335 Dataset: The first dataset is composed of 25 days of continuously streamed data from 5 sensors placed on a single section of the SS335 viaduct, as described in Sec. II-A. Five days out of 25 of data are recorded before the maintenance intervention. They are labelled as *anomalies* (i.e., aged bridge), while 20 days are recorded after the maintenance and labelled as *normal data*. We select five days from both anomalies and normal data to have a balanced test set. Noteworthy, no anomalies are included in our training set. They are only used during the testing phase to compute the accuracy. The remaining 20 days are divided into 15 days of training and 5 of validation.

2) Roccaprebalza Dataset: The second dataset consists of the sensors' acceleration data in section 10 of the viaduct described above. For labelling each passing vehicle, and to benchmark all the algorithms, we recorded a video of section 10 of the viaduct synchronized with the acceleration data to build the ground truth of the light vehicles (with 4 wheels) and heavy vehicles (with > 4 wheels) crossing it. We used 31 minutes of video recorded at high traffic, thus representing a challenge for traffic load estimation. The final dataset comprises $100 \times 1860s$ rows with 21 features (7 sensors with 3 accel. axis) and 1 label. The signal is windowed in windows of 1 second, with 10 ms of stride, to generate as much data as possible for the networks' training.

B. AD benchmark

Table I summarizes the results compared with the state-of-the-art algorithms in terms of accuracy, specificity and sensitivity on the ss335 dataset. Notice that the PCA model of [7] still reaches the best accuracy when compared with classically trained autoencoders. Nevertheless, our new autoencoder (C) outperforms the other two by a large margin of +4.3% accuracy. Furthermore,

TABLE I
PERFORMANCE COMPARISON FOR THE AD USE CASE.

Algorithm	Data type	Accuracy	Specificity	Sensitivity
PCA [7]	Time	98.2%	100%	96.0%
	Frequency	93.2%	84.9%	**100%**
Autoencoder A [7]	Time	91.5%	94.4%	89.1%
	Frequency	82.7%	60.7%	99.4%
Autoencoder B [7]	Time	50.6%	67.2%	37.1%
	Frequency	43.5%	38.2%	47.6%
Our Work				
Autoencoder C	Time	95.8%	100%	92.2%
AT Autoencoder C	Time	**98.9%**	**100%**	97.7%

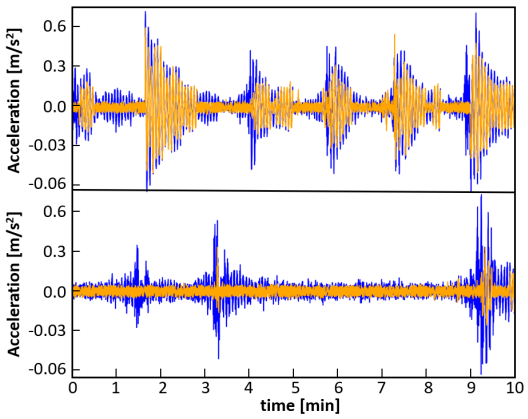

Fig. 3. Reconstruction of the AT autoencoder of normal and anomalous signals.

the adversarial trained (AT) autoencoder C surpasses all the other algorithms in terms of specificity and overall accuracy, reaching 100% specificity and 98.9% accuracy, improving the performance also over the baseline autoencoder (C) with a +3.1%.

Fig. 3 shows how our adversarial trained autoencoder reconstructs normal and anomalous signals. The algorithm learned to reconstruct the normal windows, while it does not recognize the peaks in the anomalous windows. Therefore, by computing the difference between input and reconstructed signal is easy to distinguish the two classes based on the autoencoder output.

C. VD benchmark

We compare the AT autoencoder, the baseline autoencoder, and the state-of-the-art cleaning strategy. We use a simple peak detector to compute the final traffic statistics. In [9], the authors employ a series of filtering steps (i.e., an L2 normalization of the 3 acceleration axis, a 4^{th} order Butterworth filtering and an exponential smoothing filter) before the detector.

Table II shows the comparison of the three solutions while computing the traffic on not overlapping windows of 20 seconds: our network trained with adversarial learning reaches the lowest mean absolute error (MAE, lower is better) of 1.86 V. on all vehicles.

TABLE II
PERFORMANCE COMPARISON FOR THE VD USE CASE.

	High Traffic [MAE]		
	Heavy Vehicles	Light Vehicles	All Vehicles
[9]	2.14 V.	4.60 V.	4.40 V.
Our Work			
Autoencoder	0.53 V.	2.16 V.	2.46 V.
AT Autoencoder	0.93 V.	1.25 V.	1.86 V.

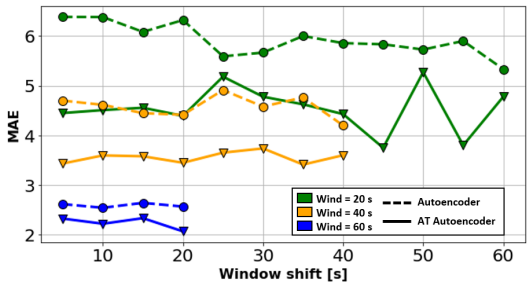

Fig. 4. MAE of the proposed approaches on the VD benchmark while predicting bot heavy and light vehicles. Both the variation of the window dimension and the window shift have been analyzed.

We also test the autoencoders trained with or without the adversarial training while changing both the input window size and the window shift of the data given to the network to estimate traffic. Interestingly, we find that the adversarially trained autoencoder consistently outperform the baseline one on all the different hyper-parameters variation. These results are shown in Fig. 4, where we sweep the window dimension for the TLE and its shift between 20s and 40s, and 5s to 60s, respectively.

V. CONCLUSIONS

Continuous structural health monitoring of transportation infrastructures is becoming increasingly important for people safety. Therefore, new algorithms to improve the analysis of the significant amount of structural data gathered from many different buildings are demanded. We introduce a new recurrent autoencoder trained using adversarial learning that fits the 2MB memory of a state-of-the-art MCU such as the STM32H7. It outperforms state-of-the-art algorithms on two real-world SHM benchmarks, an anomaly detection benchmark [7] and a traffic load estimation one [9]. On the AD benchmark, we outperform PCA-based methods by 0.7% and autoencoders by 7.4%. On the VD benchmark, we improve previous work results in detecting heavy and light vehicles by 2.30× and 2.36×.

REFERENCES

[1] F. Lamonaca *et al.*, "Internet of things for structural health monitoring," in *2018 Workshop on Metrology for Industry 4.0 and IoT*, 2018, pp. 95–100.

[2] A. Burrello *et al.*, "Embedded streaming principal components analysis for network load reduction in structural health monitoring," *IEEE Internet of Things Journal*, pp. 1–1, 2020.

[3] A. Girolami *et al.*, "Low-cost and distributed health monitoring system for critical buildings," in *2017 IEEE Workshop on Environmental, Energy, and Structural Monitoring Systems (EESMS)*, 2017, pp. 1–6.

[4] D. Dai *et al.*, "Structure damage localization with ultrasonic guided waves based on a time–frequency method," *Signal Processing*, vol. 96, pp. 21 – 28, 2014.

[5] A. Basharat *et al.*, "A framework for intelligent sensor network with video camera for structural health monitoring of bridges," in *Third IEEE International Conference on Pervasive Computing and Communications Workshops*, 2005, pp. 385–389.

[6] D. Garcia-Sanchez *et al.*, "Bearing assessment tool for longitudinal bridge performance," *Journal of Civil Structural Health Monitoring*, vol. 10, 2020.

[7] A. Moallemi *et al.*, "Model-based vs. data-driven approaches for anomaly detection in structural health monitoring: A case study," in *2021 IEEE International Instrumentation and Measurement Technology Conference (I2MTC)*. IEEE, 2021, pp. 1–6.

[8] A. Geiger *et al.*, "Tadgan: Time series anomaly detection using generative adversarial networks," 2020.

[9] A. Burrello *et al.*, "Enhancing structural health monitoring with vehicle identification and tracking," in *2020 IEEE International Instrumentation and Measurement Technology Conference (I2MTC)*, 2020, pp. 1–6.

[10] I. Goodfellow *et al.*, "Generative adversarial nets," *Advances in neural information processing systems*, vol. 27, 2014.

[11] M. Arjovsky *et al.*, "Wasserstein generative adversarial networks," in *International conference on machine learning*. PMLR, 2017, pp. 214–223.