International Conference on Smart Infrastructure and Construction 2019 (ICSIC): Driving data-informed decision-making

ice Publishing

DeJong, Schooling and Viggiani
ISBN 978-0-7277-6466-9
https://doi.org/10.1680/icsic.64669.395

CSIC Cambridge Centre for Smart Infrastructure & Construction

UNSUPERVISED DEEP LEARNING FOR INSTRUMENTED INFRASTRUCTURE: A CASE STUDY

A. Mikhailova[1], N.M. Adams[1,2], C.A. Hallsworth[1], F.D.-H. Lau[1,3], and D.N. Jones[4]*

[1]*Department of Mathematics, Imperial College of Science, Technology and Medicine, London, United Kingdom*
[2]*Data Science Institute, Imperial College of Science, Technology and Medicine, London, United Kingdom*
[3]*Lloyd's Register Foundation Programme on Data-Centric Engineering, The Alan Turing Institute, London, United Kingdom*
[4]*Mathematical Institute, University of Oxford, Oxford, United Kingdom*
** Corresponding author*

ABSTRACT Deep learning methods have recently shown great success in numerous fields, including finance, healthcare, linguistics, robotics, and even cybersports. Modern computer hardware makes it possible to train very large deep neural networks that can handle datasets of high dimensionality and rich structure. The area of unsupervised deep learning addresses modelling underlying data distributions when no ground truth labels are present. Probabilistic deep learning studies models that provide a measure of uncertainty for a prediction. We present a case study where a state-of-the-art family of deep probabilistic unsupervised models called Variational Auto-Encoder (VAE) is applied to data accrued from a network of fibre-optic sensors installed within a composite steel-concrete half-through railway bridge. Our goals were to: 1) automatically characterise the response of the bridge under stimuli based on sensor measurements, and 2) based on this characterisation, determine when a train passes across a bridge. Based on the VAE model, we present an algorithm to automatically identify the "train event" points. Our case study illustrates how state-of-the-art deep learning methods can be applied to a civil infrastructure engineering problem without directly modelling the physics of the objects or using any labels.

1. Introduction

The purpose of this paper is to investigate the applicability of modern machine learning machinery to a real-world instrumented infrastructure problem. Smart sensors are being integrated into infrastructure: as an example, several bridges in the UK have recently been instrumented with fibre-optic sensors (Butler *et al.*, 2016). Analysis of measurements from such sensors can bring understanding of how bridges behave at rest, react to events, and recover afterwards, which is helpful for monitoring bridge health and detecting possible damage, an activity which is generically referred to as structural health monitoring (Chang *et al.*, 2003). These measurements record the response of multiple sensors over time, and reflect both the behaviour of the bridge and the sensor network itself. These spatio-temporal data are high-dimensional, high-frequency and hard to interpret. Statistical methods are being developed to model this sensor data (Lau *et al.*, 2018a). Our work presented in this paper demonstrates how a state-of-the-art unsupervised deep learning technique called Variational Auto-Encoder ("VAE") (Kingma and Welling, 2013) can be leveraged to analyse these sensor measurements.

Machine learning distinguishes supervised and unsupervised learning methods. Supervised methods require ground truth labels from which they learn. Examples of supervised learning tasks are classification and regression. Classification learns a discrete mapping from a data point to a class, based on the class labels present in the training set. The goal of regression is to learn a continuous mapping from a point to a real-valued scalar or vector. Unsupervised learning methods do not require any labels and at their essence map high-dimensional input data to a numerical space of a lower dimensionality.

With recent improvements in computational capacity, deep learning methods are becoming increasingly popular and are already being adopted in numerous fields. Such fields include speech recognition (Graves *et al.*, 2013), natural language processing (Sutskever *et al.*, 2014), computer vision (Krizhevsky *et al.*, 2012), finance (Heaton and Polson, 2016), robotics (Levine *et al.*, 2017), playing board games (Silver *et al.*, 2017), and even solving fundamental scientific problems such as modelling protein structure (Evans *et al.*, 2018). Deep learning excels at analysing high-dimensional data and discovering complex hidden dependencies.

VAE is a deep learning-based dimensionality reduction method. Unsupervised dimensionality reduction techniques are vital for numerous tasks, such as data compression and visualisation, generation of realistic synthetic data, feature extraction, and data density modelling. The most widely used techniques include PCA (Pearson, 1901) and its nonlinear generalisation Kernel PCA (Schölkopf *et al.*, 1998). The popular t-SNE method (van der Maaten and Hinton, 2008) projects data points onto a low-dimensional (2D, 3D) space and hence is well-suited for visualisation. In modern machine learning, neural network-based data mappings have been most popular in recent years. A notable example is the family of word2vec models (Mikolov *et al.*, 2013) that embed words into

numerical vector space so that words often appearing together in texts become vectors close to each other. Google's trained word2vec model was trained on about 100 billion words taken from the Google News dataset (Google Code Archive – word2vec, 2013), which illustrates the efficiency and capacity of neural networks.

Autoencoders are a family of techniques that learn both the embedding ("encoder") and the inverse mapping ("decoder", or "reconstruction") simultaneously (Goodfellow *et al.*, 2016, Chapter 14). VAE adds a probabilistic flavour to the autoencoder framework by operating with distributions over embeddings and reconstructions rather than single point estimates.

This paper discusses applicability of VAE to the task of determining when a train passes across a bridge instrumented with sensors. The original dataset does not contain any labels indicating when exactly there is a train on the bridge, hence manual "event" and "no event" labelling of the data is conducted so that standard metrics for estimating quality of classification could be used. This paper provides necessary deep learning background, presents the VAE model, and defines a VAE-based train event detection algorithm based on the work of An and Cho (2015). The raw data have pronounced temporal drifts partially due to environmental factors such as temperature. We propose a method for data transformation that removes temporal variability and ensures stability of the VAE training procedure. Finally, our VAE-based model is assessed using various classification metrics such as precision, recall, and F-measure, and the benefits of our approach are demonstrated, compared to a widely-used quality control method.

2. Data

This analysis considers three distinct datasets of sensor measurements collected from one bridge. The datasets consist of 80-dimensional vectors representing measurements of 80 sensors taken at 250Hz frequency. The sensor network consists of fibre-optic sensors with Bragg grating. There are two separate rail tracks on the bridge for trains travelling in opposite directions. There are two lines of 20 equidistant sensors located in the main girders of the bridge. The datasets contain 611108, 908238, and 803594 observations, accounting for runs of approximately 2440, 3630, and 3210 seconds, respectively. Such large volumes of data observed at high frequency pose challenges for analysis.

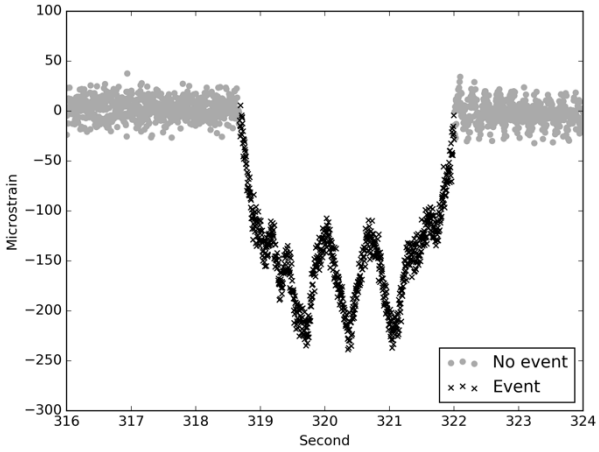
The original data records wavelength measurements in nanometres. Following Lau *et al.* (2018b), the sensor measurements are converted to relative strains via the photoelastic constant as shown in Equation (1). This conversion of the data transforms the data into interpretable physical units and plays an important role in practical use of neural networks.

$$x_t = \frac{1}{1-\rho}\left(\frac{y_t - y_1}{y_1}\right) \times 10^6 \qquad (1)$$

In Equation (1), y_t, x_t are wavelength and strain (expressed in microstrains) at time t and $\rho = 0.22$ is the photoelastic constant of the fibre-optic cable.

The original measurements do not have any labels indicating when a train is on the bridge. "Event" and "no event" labels needed for evaluation of our algorithms were constructed manually by considering the sum of strains of all sensors and hand-picking events. Figure 1 shows eight seconds of the sum of strains of all sensors marked by crosses and coloured in darker shade where, according to the constructed labels, an event happens.

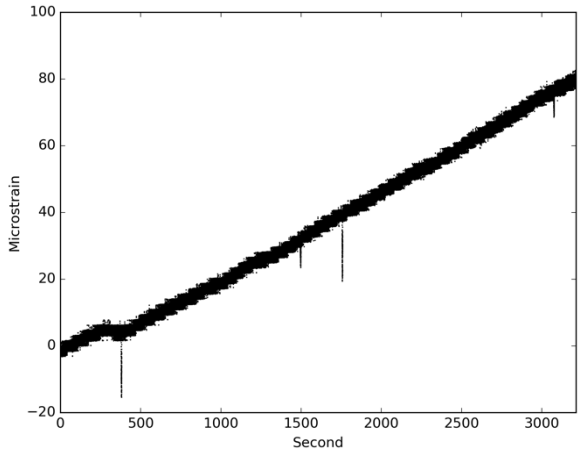
Figure 1 Sum of sensor strains around a train event



Per our manual labelling, the "event" and "no event" classes turned out to be heavily imbalanced: the "event" class accounts for only about 0.43% of all the datasets. This manual labelling procedure is laborious and infeasible for larger datasets, hence automated labelling methods are important.

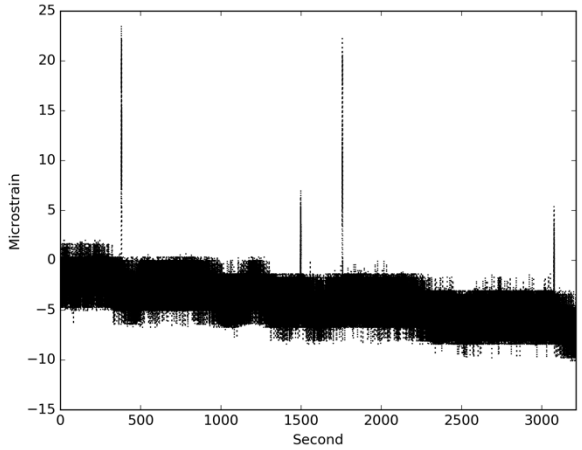
The sensor data exhibit unpredictable temporal variation partially due to environmental factors such as temperature. Figure 2 illustrates a drift occurring in our third dataset in the strains of a sensor. Such variability of the data tends to negatively affect machine learning methods that do not account for any sequential structure. The classic VAE model demonstrated in this paper does not assume any temporal structure, therefore in Section 4.1 we propose a pre-processing technique to remove such drifts from the data. The spikes in the data that can be seen in Figures 2 and 3 correspond to the train events that we aim to detect automatically.

Figure 2 Drift in the strains of an example sensor



It is worth noting that temporal variation and train events manifest different features in individual sensors, subject to where the sensor is positioned on the bridge. Figure 3 depicts strain values of a second example sensor from the same subset of data as that shown in Figure 2.

Figure 3 Drift in the strains from a second example sensor



Part of our interest in using VAEs is to automatically capture the rich structure present in this data.

3. Variational Auto-Encoder

This section provides the necessary background on deep learning, defines the VAE model, and presents the algorithm for VAE-based train event detection.

3.1 Background

As briefly mentioned in Section 1, an autoencoder learns its encoder and decoder mappings jointly. Modern autoencoder models, including VAE, typically use deep neural networks as their encoders and decoders. Deep neural networks are commonly used to construct complex non-linear representations suitable for challenging datasets.

Deep neural network ("DNN") is the core building block of deep learning algorithms. There are numerous DNN architectures, each tailored for certain tasks and types of data. This paper considers the architecture called multilayer perceptron ("MLP"). An MLP is a composition of at least three functions $f_n\left(f_{n-1}(\dots f_1(x))\right)$, where each component ("layer") $f_i(\cdot)$, except for the last, consists of a non-linear function, called an activation function, applied to a linear transformation of the input:

$$f_i(x) = g(W_i^T x + c_i) \quad (2)$$

(Goodfellow *et al.*, 2016, Chapter 6). Here, W_i and c_i are parameters to be determined which are typically estimated via gradient descent optimisation of some objective function. The layers $f_2(\cdot)$, …, $f_{n-1}(\cdot)$ are called hidden layers. DNNs are known to be extremely good and scalable approximations of complex functions with high-dimensional inputs.

Classic autoencoders use two separate MLPs as encoder and decoder. In training, their objective is to minimise the metric called reconstruction loss which is typically the mean squared error between the input and its reconstructed code. However, classic autoencoders are deterministic in nature and do not provide any measure of uncertainty either for the inferred code (that is, the latent representation) or for the reconstruction. For complex nonlinear models, it is usually necessary to perform approximate inference in order to quantify such uncertainty. VAE performs efficient variational inference, as discussed in the next subsection, under certain assumptions about the model while keeping its latent space close to a standard Gaussian distribution, and aiming at partitioning the latent space so that similar inputs are mapped to close codes.

3.2 Variational Auto-Encoder

Consider a generative latent variable model which consists of a random variable X driven by a latent random variable Z. The joint probability density function can be written as follows:

$$p_\theta(x,z) = p_\theta(x|z)\, p_\theta(z) \quad (3)$$

where θ denotes the true parameters of the model. Note that we are following the notation similar to that used by Kingma and Welling (2013). We would like to point out that various notations are used in the literature in this area, and they often differ from statistical conventions. Given a set of independent and identically distributed samples from X, the goal is to estimate the true parameters θ and to sample from the posterior distribution of $Z|X$. This generally implies computing the marginal likelihood $p_\theta(x)$ and the posterior $p_\theta(z|x)$ shown in Equations (4) and (5) respectively, which are often intractable or infeasible to calculate, hence require approximation.

$$p_\theta(x) = \int_z p_\theta(x,z)\, dz \quad (4)$$

$$p_\theta(z|x) = \frac{p_\theta(x,z)}{p_\theta(x)} \quad (5)$$

The two most common approaches to posterior approximation are MCMC (Gelfand and Smith, 1990; Hastings, 1970) and variational inference (Jordan, 1999; Wainwright and Jordan, 2008). According to Blei *et al.* (2016), the latter is often faster and scales better. VAE employs variational inference for its training procedure. In variational inference, the true posterior $p_\theta(z|x)$ is approximated by a simpler distribution $q_\varphi(z)$ whose parameters φ are learned jointly with the parameters θ of the generative model. The maximisation objective is Evidence Lower Bound ("ELBO") which is a lower bound of the true marginal likelihood $p_\theta(x)$:

$$\mathcal{L}(q;\theta,\varphi) = \mathbb{E}_{q_\varphi(z|x)}[\log p_\theta(x|z)] - D_{KL}(q_\varphi(z) \parallel p_\theta(z)) \quad (6)$$

where $D_{KL}(q_\varphi(z) \parallel p_\theta(z))$ denotes the Kullback-Leibler divergence ("KL-divergence"). The derivation of Equation (6) is given in (Blei *et al.*, 2016). Intuitively, Equation (6) optimises the likelihood of the decoder (i.e. converting codes into realistic data) while keeping the latent space of the codes close to the prior over Z.

Under certain assumptions made in the VAE framework, $\mathcal{L}(q;\theta,\varphi)$ is differentiable almost everywhere with respect to both θ and φ, and can be optimised via gradient ascent with the Stochastic Gradient Variational Bayes algorithm (Kingma and Welling, 2013). More specifically, we consider the case when $q_\varphi(z)$ and $p_\theta(x|z)$ are assumed to be Gaussians whose parameters are estimated by encoder and decoder DNNs. All covariance matrices are assumed to be diagonal for simplicity, rather than a valid modelling assumption in the problem context. It is possible to extend the model to non-diagonal covariance matrices. The prior over the latent variables Z is set to the standard normal distribution, $\mathcal{N}(0, I)$. With these assumptions, the probability density function $p_\theta(x|z)$ is differentiable almost everywhere with respect to θ, and the KL-divergence term in Equation (6) can be computed and differentiated in closed form as shown by Kingma and Welling (2013).

3.3 Model for anomaly detection

As "event" and "no event" classes are heavily unbalanced, the train event detection problem can be formulated as an anomaly detection task. We use the classic VAE model where both encoder and decoder are MLPs, and employ the reconstruction probability ("recproba") metric calculated via Algorithm 1 as per An and Cho (2015) to distinguish "event" from "no event". More details on the architecture of the networks are given in Section 4.4. We would like to note that this reconstruction probability metric is not a properly defined probability measure.

Algorithm 1 Calculation of reconstruction probability

> Let L be number of samples to compute recproba.
> Get $[\mu_z, \Sigma_z] = encoder(x)$.
> Draw $z^{(1)}, \dots, z^{(L)} \sim \mathcal{N}(\mu_z, \Sigma_z)$.
> Get $\left[\mu_x^{(l)}, \Sigma_x^{(l)}\right] = decoder\left(z^{(l)}\right)\ for\ l\ in\ 1 \dots L$.
> Finally: $recproba(x) = \frac{1}{L}\sum_{l=1}^{L} \log \mathcal{N}\left(x;\ \mu_x^{(l)}, \Sigma_x^{(l)}\right)$.

Here, $\mathcal{N}\left(x;\ \mu_x^{(l)}, \Sigma_x^{(l)}\right)$ is the probability density function for the normal distribution with the given parameters, evaluated at data point x. Essentially, the reconstruction probability is the MC estimate of $\mathbb{E}_{q_\varphi(z|x)}[\log p_\theta(x|z)]$, the left-hand term of the ELBO objective presented in Equation (6). For computational simplicity reasons, the number of samples L is set to 1 in this experiment.

To perform the "event" / "no event" classification, a threshold α is chosen for reconstruction probability. If the recproba value for a data point x is lower than this threshold, x is marked as an anomaly, i.e. a train event. In this case study, α is set to -130 based on empirical observation of a subset of the data. Generally, the choice of α controls the false positive and the false negative rates of such binary classifiers (see Section 4.2 for details on the evaluation methods).

There are two major simplifications in this model. First, for computational convenience, both covariance matrices Σ_z and $\Sigma_x^{(l)}$ are diagonal, hence any correlation between sensors is lost. Second, temporal dependencies in the data are not captured in this method.

4. Demonstration

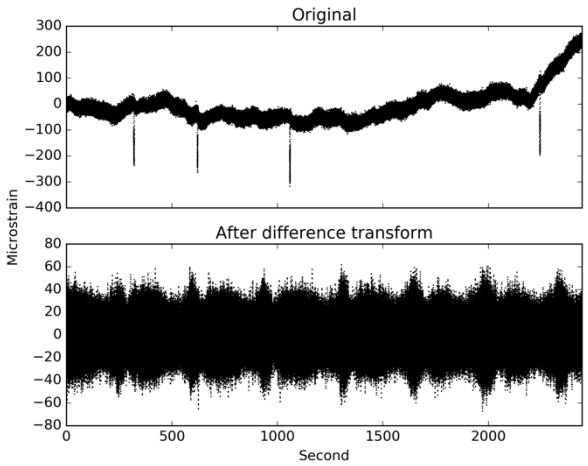
This section discusses data preparation for anomaly detection, describes the baseline used to compare the VAE model against, and defines metrics for evaluation of model quality. Finally, the evaluation of the baseline and the VAE is presented.

4.1 Data pre-processing

As illustrated in Figures 2 and 3 in Section 2, the individual sensor strain data are non-stationary and have unpredicted temporal variation. Such variation interferes with the process of training neural networks in the VAE model as the model assumes that there is no temporal correlation in the underlying process. The highly complex and non-convex VAE loss function is sensitive to variability in the input data, and when drifts become more pronounced (e.g. like that shown in Figure 2), the loss explodes, and the training procedure does not converge.

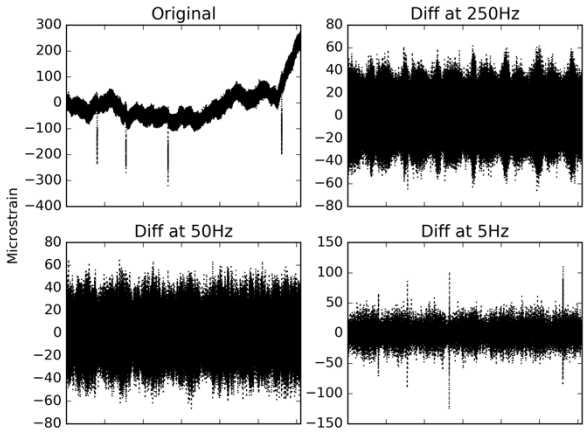
The difference transformation is a common method to remove temporal variability from data (Cowpertwait and Metcalfe, 2009: p. 93). It is performed by subtracting the data point at previous time step from the data point at the current time: $\Delta x_t = x_t - x_{t-1}$. However, as the sensor strain data were taken at a high frequency, differences between consecutive observations are small enough that the train events are no longer apparent in the data plots. This phenomenon is illustrated in Figure 4.

Figure 4 Sum of sensor strains before and after the difference transform



To resolve this issue, we first downsample the data by retaining only one observation at regular intervals. Intuitively, downsampling the data should make the differences larger when a train event starts to happen, compared to when the bridge is at rest. We show empirically that lowering the frequency from 250Hz to 5Hz is enough to visibly remove the temporal variability while preserving the signal of train events in the data. Figure 5 illustrates the same subset of data differenced at various frequencies: 250Hz (original frequency), 50Hz, and 5Hz.

Figure 5 Sum of sensor strains before and after difference transform with lowered frequency



4.2 Metrics and manual labelling

To evaluate our models, we use the standard metrics of the quality of binary classification: precision, recall, and F-measure (Manning *et al.*, 2008, pp. 154–157). The definitions of these metrics are built on the concept of a confusion matrix for binary classification. A binary confusion matrix is a table that characterises the performance of a classification algorithm based on the known ground truth labels. Henceforth, the "event" class is referred to as "Positive", and the "no event" class as "Negative". The confusion matrix counts the number of times the algorithm correctly identified instances of both classes ("True Positive" and "True Negative"), and the number of times it mistook an ordinary data point for an anomalous event ("False Positive") and vice versa ("False Negative"):

Table 1 Layout of the binary confusion matrix

	Predicted Yes	Predicted No
True Class Yes	True Positive (TP)	False Negative (FN)
True Class No	False Positive (FP)	True Negative (TN)

In this context, precision is the ratio of correctly detected events to the total number of data points labelled as events, which measures the accuracy of an anomaly detector:

$$Precision = \frac{TP}{TP+FP} \quad (7)$$

Precision, Equation (7), alone is not enough to evaluate the quality of a classifier. Suppose that the classifier is tuned to be very conservative to the false positive labels, and only classifies an entry as positive in very few cases, hence missing a large amount of actual positive samples. In this setting, precision is going to be very high, but such a classifier is practically useless. Therefore, the number of samples labelled as positive needs to be assessed as well.

Recall, Equation (8), is the ratio between the number of items correctly labelled as positive by the classifier and the number of all items that have ground truth positive labels. Recall essentially characterises how many positive items the classifier misses, the quantity which is particularly important to access when the classes are unbalanced (as in our case):

$$Recall = \frac{TP}{TP+FN} \quad (8)$$

Similarly, recall can be high even if the classifier is poor: consider the classifier that labels every data point as positive.

Finally, the F-measure ("F1") balances precision and recall and summarises them in one number. It is defined as the harmonic mean of precision and recall:

$$F1 = 2 \times \frac{Precision \times Recall}{Precision+Recall} \quad (9)$$

These three metrics require ground truth "event" and "no event" labels to be assigned to the data, hence we performed manual labelling of the sensor data as mentioned in Section 2. Out of 2322940 data samples in total, 9934 were labelled as "event".

4.3 Baseline – CUSUM

We apply the cumulative sum control chart ("CUSUM") method initially proposed by Page (1954) to our event detection problem, and compare our VAE model against it. As

the sensor data are multidimensional and classic CUSUM accepts only one-dimensional inputs, we input the sum of strains of all 80 sensors at each time step to the algorithm. Using the sum of strain measurements within a CUSUM is a simple and arbitrary choice. Other one-dimensional summaries can be used within the CUSUM approach. The sensor strain values are pre-processed before they are summed, as described in Section 4.1.

The CUSUM method is used to detect points in a time series when the process goes out of control. It monitors changes in the mean of the process over time, and signals when it detects a deviation from the expected mean that is outside of the allowed limits. This paper considers two standard CUSUM charts for train event detection, where events are flagged using the same threshold on both charts.

The method uses the following parameters: 1) expected mean of the process, $\hat{\mu}$; 2) magnitude of a change in the process that is to be labelled as an anomaly, k; 3) control limit, H. The parameters k and H are set to $\hat{\sigma}$ and $3\hat{\sigma}$ respectively, where $\hat{\sigma}$ is the estimated standard deviation of the process. In this case study, $\hat{\mu}$ and $\hat{\sigma}$ are set to 0 and 15 respectively based on empirical observation of a subset of the data. For consistency, the data subset used here is the same as that chosen to compute the VAE reconstruction probability threshold mentioned in Section 3.3. The full CUSUM-based anomaly detection method is presented below in Algorithm 2.

Algorithm 2 CUSUM-based event detection

> Compute $\hat{\mu}$ and $\hat{\sigma}$.
> Set $k := \hat{\sigma}$ and $H := 3\hat{\sigma}$.
> Set $S_1^{low} := 0$ and $S_1^{high} := 0$.
> For each data point (sum of sensor strains) x_t:
>> Set $S_t^{low} := \min(0, S_{t-1}^{low} + x_t - \mu + k)$.
>> Set $S_t^{high} := \max\left(0, S_{t-1}^{high} + x_t - \mu - k\right)$.
>> If $S_t^{low} \leq -H$ or $S_t^{high} \geq H$, label x_t as event.

4.4 Technical specification of the VAE model

The input to VAE is multi-dimensional, hence vectors of sensor strains are passed to it directly after the pre-processing procedure described in Section 4.1 is performed.

Both encoder and decoder are multilayer perceptron networks with one hidden layer (see Section 3.1). The hidden layer size is set to 40, ReLU activation (Nair and Hinton, 2010) is chosen as the non-linearity, and L2 weight regularisation is used in both encoder and decoder. The latent space (code) dimensionality is set to 2. All the neuron weights are initialised randomly from a Gaussian distribution with mean 0 and variance 0.01. The model is trained on the whole dataset with mini-batches of size 64 for 50 epochs via stochastic gradient descent using the Adam optimiser (Kingma and Ba, 2014).

The presented choice of the hyperparameters and the DNN properties is a reasonable standard choice for MLP networks and the VAE framework. No special hyperparameter tuning was performed for this case study.

The data are normalised by computing the Z-score before being passed to the VAE model as such standardisation of inputs aids the convergence of the training of deep neural networks.

4.5 Evaluation

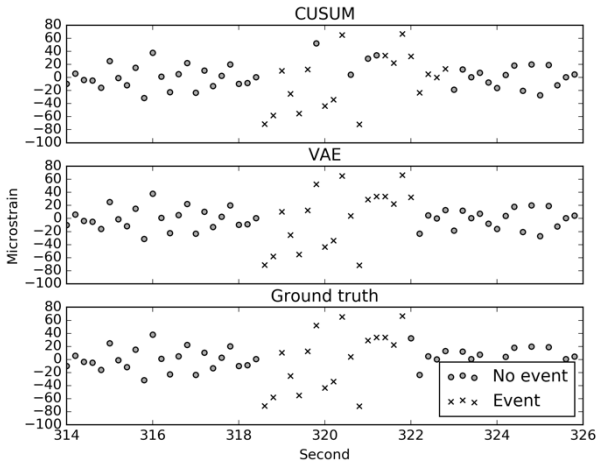
Table 2 gives the results of the evaluation of CUSUM and VAE. Both algorithms are sensitive to the choice of hyperparameters, mainly to the thresholds controlling the size of anomalies to be detected. As mentioned earlier, we did not perform any special hyperparameter tuning, adopting reasonable standard choices instead. With the given choice of parameters, VAE noticeably outperforms CUSUM:

Table 2 Evaluation results

	CUSUM	VAE
Precision	0.579	0.840
Recall	0.905	1.000
F-measure	0.706	0.913

Figure 6 illustrates one of the events labelled by CUSUM and VAE, compared to the ground truth manual labels. Notably, CUSUM produces a few false positives right after the end of the event, and misclassifies several anomalous points in the middle of the event. Note that as the frequency of the data was lowered during pre-processing, the number of points input to the algorithms and shown in Figure 6 is smaller than the original number of entries in this time interval. To bring the predicted labels to the original data, one can set all the original observations falling between the two labelled points to "event" or "no event" based on these two predicted labels.

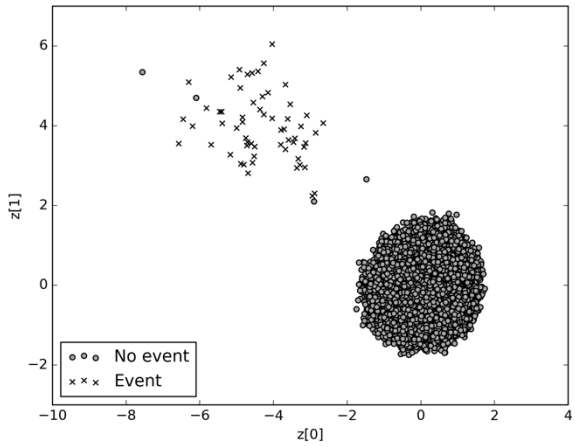
Figure 6 Predicted vs. ground truth labels



Apart from event detection, another interesting property of the VAE is the representation of the sensor strain data in the latent space. Figure 7 depicts the two-dimensional latent space of the VAE on a subset of the data. The data points are marked based on the ground truth manual labels. One can see that the "event" and the "no event" data are clearly separated in the latent space,

with the "no event" data distribution being close to a standard Gaussian.

Figure 7 Latent space of the VAE



5. Conclusion

The purpose of this case study is to show that state-of-the-art deep learning methods have potential for engineering problems and are feasible to adopt. VAE is a fully unsupervised probabilistic deep learning technique that is commonly used to model data distributions. This paper demonstrates how a standard VAE model can be applied to a real-world anomaly detection problem for instrumented infrastructure. The performance of our VAE model is reasonable compared to the widely used CUSUM benchmark, while VAE, unlike CUSUM, can process vector data and also map inputs to a low-dimensional space, separating the anomalous points there. The VAE approach does not involve any mathematical modelling of the bridge structure and behaviour. As this case study reveals, VAE can be treated as a black box as long as the input data do not have drifts and are standardised to expedite the convergence of the training procedure.

The full source code of our case study is available online at https://github.com/red-cheese/icsic-vae-for-bridges.

6. Future work

As stated earlier, the standard VAE model is not sequential in nature, i.e. it does not consider temporal dependencies between subsequent data points. When training on or predicting for x_t, VAE does not take earlier observations $x_1, \dots, x_{t-1}$ into account. Due to this, the pre-processing of our data presented in Section 4.1 was required, as VAE assumes the input process to be stationary. As a solution to this limitation, several methods have been introduced in the recent literature. For example, variational RNN ("VRNN") (Chung *et al.*, 2015) employs a VAE machinery on each time step within a fixed time window, and takes previous data points as inputs as well. It appears to us that this or other similar sequential approaches that allow for richer model specifications can be beneficial for instrumented infrastructure tasks such as ours, and they might help to get rid of the additional step of data pre-processing altogether.

Table 2 and Figure 7 show that when the "event" class accounts for a small portion of the dataset, VAE is able to separate the dominating ("no event") class and detect train events with high quality. Another interesting research task is to assess the performance and the latent space structure of VAE in cases when the proportion of events in the data is increasing and eventually becomes the same as the proportion of "no event" entries.

7. Acknowledgements

The authors would like to acknowledge the Cambridge Centre for Smart Infrastructure and Construction (CSIC), The Laing O'Rourke Centre at Cambridge and the Staffordshire Alliance (Network Rail, Volker Rail, Atkins and Laing O'Rourke) for providing the dataset used in the paper.

F. D.-H. Lau's work was supported by The Alan Turing Institute under the EPSRC grant EP/N510129/1 and the Turing-Lloyd's Register Foundation Programme for Data-Centric Engineering.

8. References

Butler LJ *et al.* (2016) Integrated fibre-optic sensor networks as tools for monitoring strain development in bridges during construction. *The 19th Congress of IABSE Proceedings, Stockholm, Sweden* (Elfgren L and Jonsson J (eds)). Curran Associates, Inc., Red Hook, NY, USA, pp. 1767–1775.

Chang PC *et al.* (2003) Review Paper: Health Monitoring of Civil Infrastructure. *Structural Health Monitoring* 2(3): 257–267, 10.1177/1475921703036169.

Kingma DP and Welling M (2013) Auto-Encoding Variational Bayes. *Proceedings of the 2nd International Conference on Learning Representations (ICLR 2014), Banff, AB, Canada.*

Lau FDH *et al.* (2018a) The role of statistics in data-centric engineering. *Statistics & Probability Letters* 136:58–62, 10.1016/j.spl.2018.02.035.

Graves A *et al.* (2013) Speech recognition with deep recurrent neural networks. *Proceedings of the 2013 IEEE International Conference on Acoustics, Speech and Signal Processing, Vancouver, BC, Canada.* IEEE, pp. 6645–6649, 10.1109/ICASSP.2013.6638947.

Sutskever I *et al.* (2014) Sequence to sequence learning with neural networks. *Proceedings of the 27th Advances in Neural Information Processing Systems Conference (NIPS 2014), Montreal, QC, Canada* (Ghahramani Z, Welling M, Cortes C, Lawrence ND and Weinberger KQ (eds)).

Krizhevsky A *et al.* (2012) ImageNet classification with deep convolutional neural networks. *Proceedings of the 25th Advances in Neural Information Processing Systems Conference (NIPS 2012), Lake Tahoe, NV, USA* (Pereira F, Burges CJC, Bottou L and Weinberger KQ (eds)).

Heaton JB and Polson NG (2016) Deep learning for finance: deep portfolios. *Applied Stochastic Models in Business and Industry* 33(1): 3–12, 10.1002/asmb.2209.

Levine S *et al.* (2018) Learning hand-eye coordination for robotic grasping with deep learning and large-scale data collection. *The International Journal of Robotics Research* 37(4–5): 421–436, 10.1177/0278364917710318.

Silver D *et al.* (2017) Mastering the game of Go without human knowledge. *Nature* 550: 354–359.

Evans R *et al.* (2018) De novo structure prediction with deep-learning based scoring. *The 13th Critical Assessment of Techniques for Protein Structure Prediction (Abstracts)*.

Pearson K (1901) On lines and planes of closest fit to systems of points in space. *Philosophical Magazine* 2: 559–572, 10.1080/14786440109462720.

Schölkopf B *et al.* (1998) Nonlinear component analysis as a kernel eigenvalue problem. *Neural Computation* 10(5): 1299–1319, 10.1162/089976698300017467.

van der Maaten L and Hinton G (2008) Visualizing data using t-SNE. *Journal of Machine Learning Research* 9: 2579–2605.

Mikolov T *et al.* (2013) Efficient estimation of word representations in vector space. *Proceedings of the International Conference on Learning Representations (ICLR 2013), Scottsdale, AZ, USA*.

Google Code Archive – word2vec (2013) https://code.google.com/archive/p/word2vec/ (accessed 28/12/2018).

Goodfellow I *et al.* (2016) *Deep learning*. MIT Press, Cambridge, MA, USA.

An J and Cho S (2015) Variational autoencoder based anomaly detection using reconstruction probability. *Technical Report.* SNU Data Mining Center, Seoul, Korea. pp. 1–18.

Lau FDH *et al.* (2018b) Real-time statistical modelling of data generated from self-sensing bridges. *Proceedings of the Institution of Civil Engineers – Smart Infrastructure and Construction* 171(1): 3–13, 10.1680/jsmic.17.00023.

Hastings WK (1970) Monte Carlo sampling methods using Markov chains and their applications. Biometrika 57(1): 97–109, 10.2307/2334940.

Gelfand AE and Smith AFM (1990) Sampling-based approaches to calculating marginal densities. *Journal of the American Statistical Association* 85(410): 398–409, 10.2307/2289776.

Jordan MI *et al.* (1999) An introduction to variational methods for graphical models. *Machine Learning* 37(2): 183–233, 10.1023/A:1007665907178.

Wainwright MJ and Jordan MI (2008) Graphical models, exponential families, and variational inference. Foundations and Trends in Machine Learning 1(1–2): 1–305, 10.1561/2200000001.

Blei D *et al.* (2016) Variational inference: a review for statisticians. *Journal of the American Statistical Association* 112(518): 859–877, 10.1080/01621459.2017.1285773.

Cowpertwait PSP and Metcalfe AV (2009) *Introductory Time Series with R*. Springer Science+Business Media, New York City, NY, USA.

Manning CD *et al.* (2009) *Introduction to information retrieval*. Cambridge University Press, Cambridge, UK. Online edition.

Page ES (1954) Continuous inspection schemes. *Biometrika* 41(1–2): 100–115, 10.1093/biomet/41.1-2.100.

Nair V and Hinton GE (2010) Rectified linear units improve restricted Boltzmann machines. *Proceedings of the 27th International Conference on Machine Learning (ICML 2010), Haifa, Israel* (Fürnkranz J and Joachims T (eds)).

Kingma DP and Ba J (2014) Adam: a method for stochastic optimization. *Proceedings of the 3rd International Conference on Learning Representations (ICLR 2015), San Diego, CA, USA*.

Chung J *et al.* (2015) A recurrent latent variable model for sequential data. *Proceedings of the 28th Advances in Neural Information Processing Systems Conference (NIPS 2015), Montreal, QC, Canada* (Cortes C, Lawrence ND, Lee DD, Sugiyama M and Garnett R (eds)).