Applied Soft Computing 112 (2021) 107836

Contents lists available at ScienceDirect

Applied Soft Computing

journal homepage: www.elsevier.com/locate/asoc

Deep learning with small datasets: using autoencoders to address limited datasets in construction management

Juan Manuel Davila Delgado *, Lukumon Oyedele

Big Data Enterprise and Artificial Intelligence Laboratory, Faculty of Business and Law, University of West of England (UWE) Bristol, Coldharbour Lane, BS16, 1QY Bristol, UK

ARTICLE INFO

Article history:
Received 10 June 2019
Received in revised form 27 July 2021
Accepted 16 August 2021
Available online 25 August 2021

Keywords:
Autoencoders
Variational autoencoders
Deep learning
Machine learning
Predictive analytics

ABSTRACT

Large datasets are necessary for deep learning as the performance of the algorithms used increases as the size of the dataset increases. Poor data management practices and the low level of digitisation of the construction industry represent a big hurdle to compiling big datasets; which in many cases can be prohibitively expensive. In other fields, such as computer vision, data augmentation techniques and synthetic data have been used successfully to address issues with limited datasets. In this study, undercomplete, sparse, deep and variational autoencoders are investigated as methods for data augmentation and generation of synthetic data. Two financial datasets of underground and overhead power transmission projects are used as case studies. The datasets were augmented using the autoencoders, and the project cost was predicted using a deep neural network regressor. All the augmented datasets yielded better results than the original dataset. On average the autoencoders provide a model score improvement of 7.2% and 11.5% for the underground and overhead datasets, respectively. MAE and RMSE are lower for all autoencoders as well. The average error improvement for the underground and overhead datasets is 22.9% and 56.5%, respectively. Variational autoencoders provided more robust results and represented better the non-linear correlations among the attributes in both datasets. The novelty of this study is that presents an approach to improve existing datasets and thus improve the generalisation of deep learning models when other approaches are not feasible. Moreover, this study provides practitioners with methods to address the limited access to big datasets, a visualisation method to extract insights from non-linear correlations in data, and a way to improve data privacy and to enable sharing sensitive data using analogous synthetic data. The main contribution to knowledge of this study is that it presents a data augmentation technique for transformation variant data. Many techniques have been developed for transformation invariant data that contributed to improving the performance of deep learning models. This study showed that autoencoders are a good option for data augmentation for transformation variant data.

1. Introduction

Construction management is a very complex and challenging task. Compared with other engineering sectors such as the automotive and aerospace, construction is labour intensive, has a vast and diverse supply chain, and is dominated by "one-off" solutions, in which every project is different and delivered to a different client [1]. The industry is placated with cost and time overruns, which are partly caused by poor estimation, planning, and risk management [2,3]. Construction is a high-risk and low-profit sector. For example, in the US, only 36.4% of construction companies continue in operations after 5 years of being funded [4]. In the UK, the average profit margin of the top 100 construction companies was just 1.5% in 2017 [5]. Construction management is, therefore, key to the success and survivability of construction companies. Big datasets that enable deep learning can potentially provide many benefits to the construction sector in general, ranging from document classification (e.g. [6]) and predicting construction costs (e.g. [7]) to identifying and coordinating spatial conflicts (e.g. [8]) and detecting structural damages (e.g. [9]). The benefits of data-driven solutions for construction management can potentially revolutionise current practice. This potential is highlighted by the large technology companies, e.g. IBM [10] and Oracle [11], venturing into the very large global construction market; which is expected to reach $10 trillion by 2020 [12]. However, there are many challenges for the implementation of big data analytics and deep learning for construction management (e.g. [13]). The most prominent challenge is the limited availability and accessibility of data in

* Corresponding author.
E-mail addresses: manuel.daviladelgado@uwe.ac.uk (J.M. Davila Delgado), l.oyedele@uwe.ac.uk (L. Oyedele).

https://doi.org/10.1016/j.asoc.2021.107836
1568-4946/

the construction sector. Many of the data recording is still paper-based, database systems are not always used resulting in a lack of consolidated and structured data sources, which is a significant limitation for compiling sufficiently large datasets required for deep learning. Algorithms used for deep learning require large datasets to produce good results, and they perform poorly with limited datasets.

The challenge of limited datasets affects many fields. For example, in computer vision tasks, e.g. object recognition, there is evidence that the most crucial factor in ensuring good results is the size of the dataset [14]. But compiling a very large dataset is an onerous task and can be prohibitively expensive. The challenge of small datasets has been addressed in various manners including improving the datasets used for training, i.e. data augmentation (e.g. [15]); generating new datasets from simulations i.e. synthetic data (e.g. [16]); using pre-trained models from other datasets i.e. transfer learning [17]; and using generative models to generate new data based on the original dataset (e.g. [18]). However, most of these approaches work for transformation invariant data (e.g. images) only; and they are not suitable for transformation variant data (e.g. financial data). This study addresses this gap by exploring the potential use of autoencoders (unsupervised machine learning models) and variational autoencoders (generative models) as tools for dealing with limited data variant datasets in construction management. In other domains, autoencoders have significantly improved data analytics performance especially when the training data is limited (e.g. [19]). Autoencoders can be used to augment variant datasets as they can preserve non-linear similarities in the input data and reproduce them in the generated data [20]. The objectives of this study are: (i) to define a methodology to use autoencoders to augment variant datasets for construction management activities; and (ii) to compare the performance of different types of autoencoders for data augmentation.

In the next section, a brief overview of the data challenges for deep learning in construction management and the approaches used to address these challenges are presented. After that, a review of the autoencoders that can be used for data augmentation is presented. Then, the method used in this study is presented, which compares the performance of four types of autoencoders using two datasets of power transmission construction projects. Deep Neural Network regressors, Random Forest Regressors, and Support Vector Regressors are used to test the data augmentation method proposed. Lastly, discussion, implications for practice and conclusions are presented.

2. Background

2.1. Deep learning data challenges for construction management

Success in the implementation of deep learning depends largely on access to large amounts of data. However, little research has been focused on addressing the existing challenges for compiling sufficiently large datasets. Research efforts on big data for construction have been focused mainly on identifying potential challenges with data management and potential architectures. For example, big data challenges have been investigated for: (i) storing and visualising very large BIM models [21,22]; (ii) handling geospatial data [23], (iii) handling earth observation data [24], and (iv) for managing big visual data and its interaction with Building Information Modelling (BIM) [25]. This type of research is needed, but it does not address the challenges of compiling all the required data. In this respect, Martínez-Rojas et al. [26] have identified that novel information technologies could facilitate data compilation of construction management related activities such as cost estimation, planning, risk identification, progress monitoring and quality control; which are critical aspects for successful project delivery [27]. Nevertheless, construction is an industry with very low levels of digitisation [28], in which information technologies are not fully leveraged to facilitate the compilation of large datasets. In the construction sector, data sources are usually decentralised, unstructured and unlabelled. For example, none of the traditional data management and exchange methods provide a way to integrate different types of data [29–31], support dynamic visualisations [32,33], or provide real-time links with big data repositories [34]. In practice, compiling a sufficiently big construction management dataset would be very labour intensive and, in many cases, could become prohibitively expensive. When compiling a big dataset is not a viable option, data analytics can be carried out with limited datasets. Using experience from other sectors, e.g. computer vision [14], the best option is to seek a trade-off between using the minimum amount of data and obtaining acceptable results.

Lack of access to big datasets is a challenge in many domains and research efforts have been reported in the literature to address them. For example, habitat suitability models were developed using limited data [35]. In computer vision, the lack of big labelled datasets is a common problem that has been addressed using different approaches (e.g. [36]). In general, there are three main approaches to address the limited data challenges: (1) data augmentation, (2) synthetic data, and (3) transfer learning. Data augmentation and synthetic data use methods to improve the existing dataset or to generate new datasets based on the original data; while transfer learning uses codified knowledge from other domains to improve generalisation of data analytics models. This study focuses on data-based approaches only, but for completeness, a brief description of transfer learning is also presented in this section.

2.2. Data augmentation

Data augmentation is an approach to augment or improve datasets to increase the performance of predictive models used in data analytics. Three main methods exist, i.e. linear transformations, interpolations and distortion methods, and probabilistic methods. The underlying principle of data augmentation is to increase the dataset used for training by creating duplicates of instances in the original dataset and transforming or distorting them to create slightly different instances. Linear transformations to generate additional data can be applied if the features of the dataset are not affected when alterations are introduced. Deep learning algorithms can infer the transformation invariance of the dataset as the invariance is embedded in the parameters. Linear transformations have been proved successful to improve performance if the training data is limited, and the dataset has transformation invariance properties. For example, linear transformations are commonly used for image classification tasks as images are transformation invariant. Simple transformations on images such as rotation, reflection, scaling, and squeezing can be generated by applying displacement fields to the images and changing the original location of every pixel to a new position. For example, Krizhevsky et al. [15] used image translations and reflections to increase the original training set by a factor of 2048. The authors note that the training examples are highly interdependent, but that without the augmented training dataset the model suffered from substantial overfitting.

Interpolations and distortion methods have been used in computer vision applications to increase the size of the training sample and improve the effectiveness of the models. Conversely to linear transformation methods, these methods can introduce non-linear distortions and randomness to the newly generated datasets For example, Simard et al. [14] presented an approach, in which new data is generated by applying random distortion

fields on the initial dataset as follows: $\Delta x(x, y) = \text{rand}(-1, +1)$; $\Delta x(x, y) = \text{rand}(-1, +1)$ In which, rand $(-1, +1)$ is a random number between -1 and +1 drawn from a uniform distribution. Then, the fields are convolved with a Gaussian standard deviation of pixels. The authors note that the introduction of additional distorted data have a considerable impact for shallow and deep neural networks. Also, the authors indicate that the original training dataset is too small for most algorithms to infer generalisation adequately, and that the additional distorted data provides additional and relevant a-priori knowledge [14].

Probabilistic methods for data augmentation seek to generate missing data in the training dataset. These approaches generate a distribution of the variables in the training set instances in a probabilistic manner [37]. Then, the augmented dataset is usually used in combination with deep learning models with probabilistic characteristics such as Restricted Boltzmann Machines (e.g. [38]) and Deep Belief Networks [39]. For example, Gan et al. [40] present a probabilistic data augmentation method in which the logistic likelihood of each instance in the training dataset is reformulated. The augmented data was used to train a Sigmoid Belief Network (SBN), and it was tested on common image recognition tasks.

2.3. Synthetic data

The other approach is to increase the original training dataset with data generated by simulations from analogous mediums. For example, Aubry et al. [41] presented a method that uses 3D models of chairs to generate a dataset and train machine learning models to recognise chairs in 2D images. The machine learning model was able to recognise chairs in 2D images without using 2D images for training. Aubry and Russell [42] later showed that the codified representations, also called embeddings, generated while training deep neural networks using 3D models, are similar to the ones trained using 2D images. Dosovitskiy et al. [43] used random background images downloaded from the internet and overlaid images of chairs on top of them to create an entirely new synthetic dataset for training. The results obtained using this unrealistic dataset, known as "flying chairs" due to the unconventional position of the chairs in relation to the background images, generalises very well in comparison to other existing traditional datasets [43]. Using 3D models to generate a synthetic dataset to improve the detection of objects in 2D images is a common approach for computer vision applications [44,45]. For example, Handa et al. [16] presented a method to generate synthetic 3D-indoor-scenes (e.g. 3D models of bedrooms, kitchens, dining rooms, etc.) to improve the segmentation of the images into distinct objects. Using already existing labelled 3D models (e.g. beds, chairs, tables, windows, etc.), their approach creates new scenes following the underlying structure of the existing ones. Their approach demonstrated that adding synthetic data improves the performance of image segmentation algorithms. Inverse approaches have been investigated as well, in which 3D information is inferred from 2D images. For example, Wu et al. [46] presented a system to predict physical events in dynamic events and to infer the physical properties of objects based only on static images. For example, to determine the weight of various objects based on how they look on 2D images. The system leverages a 3D physics engine and deep learning generative models.

One characteristic of the synthetic data approaches is that they add distortions and noise into the generated data to make the synthetic data similar to real-life data. For example, Krizhevsky et al. [15] alter the intensities of RGB channels in training images to simulate the varying distribution of intensities in real-life images. Handa et al. [16] also introduced noise to the generated 3D models to improve the performance of the image segmentation. In this case, the depth maps of the generated 3D scenes were altered to simulate a noise distribution of real-world datasets.

2.4. Transfer learning

Transfer learning is the approach that transfers codified knowledge learned by a machine learning model for one task into other models for other tasks. In other words, what has been learned for specific tasks is exploited to improve generalisation in other tasks. The primary motivation for transfer learning is that compiling sufficiently big sets of labelled data for supervised models is a very demanding job. Therefore, compiling big datasets for every specific task is, in practice, impossible. The potential benefits of this approach have been identified many years ago [47]; and terms such as learning to learn, knowledge consolidation, and inductive transfer refer to similar approaches. Transfer learning has been used for various tasks such as to improve image classification models [48], and to improve machine translations models for languages for which not enough data is available [17].

2.5. Examples of data augmentation and synthetic data for construction applications

Most of the augmentation and synthetic data approaches for construction applications focuses on developing 3D environments from which labelled data can be produced in a quick manner. For example, Neuhausen et al. [49] presented an approach for generating synthetic labelled images of workers in construction sites rendered in a 3D environment, which then were used to train computer vision models that support machine operators to detect pedestrians walking near the machine. Hong et al. [50] presented an approach that generates synthetic images from a 3D environment and then uses a generative adversarial network to make the synthetic images look more realistic by transferring the image style from real-life images to the synthetic ones.

Other data augmentation approaches focus on time-series data, for instance to detect activities on image sequences [51,52]. Synthetic 3D representations have been generated as well. For instance, Wu et al. [53] presented a method that generates 3D seismic images to train a seismic fault detector. Ma et al. [54] presented an approach to generate synthetic point clouds from 3D models and then train a detection model for semantic segmentation. Chokwitthaya et al. [55] present a similar approach to the presented in this paper; but it uses a Gaussian mixture model to generate independent samples of data from a small dataset to improve model performance. However, this approach only uses transformation invariant data, as all the approaches referred before.

In sum, research efforts in data augmentation and synthetic data in construction, and other fields, have been focused mostly on approaches for transformation invariant data, while this study focuses on approaches for transformation variant data. This study addresses this knowledge gap by investigating autoencoders as an alternative approach to address transformation variant data.

3. Autoencoders for data augmentation

This paper explores the potential use of autoencoders to augment financial datasets of construction projects. Conversely to images, financial data is not transformation invariant. Therefore traditional linear and non-linear transformation and distortion methods cannot be used; thus, autoencoders can potentially be a good alternative. Autoencoders are unsupervised neural networks that try to reproduce its input as its output. The typical architecture of an autoencoder usually consists of two parts, i.e. the encoder and the decoder (see Fig. 1 - left). The encoder compresses the original input (x) into a new representation called "latent vector" or "compressed representation". The decoder decompresses the representation into a new reconstructed input (x')

based on the correlations among input features. The network can be trained by minimising a loss function between the original input and the reconstructed input $L(x, x')$. Non-linear activation functions must be used in the nodes (e.g., a_1^1 in Fig. 1 - left) and the weights must be initialised randomly so that the network does not simply reproduce the exact input.

Autoencoders are used for many machine learning applications including dimensionality reduction (e.g. [56]), semantic hashing (e.g. [57]), data denoising (e.g. [58]), clustering and classification (e.g. [59,60]), change detection [61], and demand forecasting [62]. Also, autoencoders have been used to improve model performance by pretraining the models, that is by learning features in the underlying structure of the data to provide good domain generalisation (e.g. [63,64]). In this paper, autoencoders are used to improve model performance by augmenting the original training data instead of pretraining the model. Both approaches are similar because both intend to extract the essential structure of the training data to improve the performance of the model. The pretraining approach adds the learned structure to the model itself, while the data augmentation approach generates a more robust dataset based on the learned structure. This latter approach has the advantage to contribute to data privacy issues as the generated data preserves the same structure of the original training data, but it is not exactly the same. For example, an augmented dataset can be generated and shared with experts to carry out data analytics without sharing the original dataset. This is of extreme importance, especially when dealing with confidential personal or financial data, as is the case in this study.

3.1. Undercomplete autoencoders

Undercomplete autoencoders constraint the number of nodes in the hidden layers to limit the amount of information that flows through the network. In this way, during training, the network will learn to reconstruct the essential features from the original input from the compressed representation. Undercomplete autoencoders use a "bottleneck" in their architecture to restrict the flow of information (Fig. 1 - left). The main drawback of undercomplete autoencoders is that when using deep autoencoders – that is with many hidden layers – the model tends to overfit the data. Overfitting occurs when random fluctuations in the training data are learned as the main features. Then, the model is not generalisable as its performance reduces significantly when new data, not seen during training, is used.

3.2. Denoising autoencoders

Denoising autoencoders are undercomplete autoencoders that intentionally add noise randomly to the original input before feeding it to the autoencoder. Adding noise avoids that the autoencoder simply copies the input to the output without learning the structure of the training data. Adding noise is usually done by randomly converting some input values to null values (see Fig. 1 - right). Denoising autoencoders learn to ignore the noise in the input data and reconstruct the input based on the structure of the original data. The main application of denoising autoencoders is to remove noise in data; for example, denoising autoencoders have been used in medical image analysis [65]. But, they have been used for pretraining [66] and fault detection of rotary machinery as well [67].

3.3. Sparse autoencoders

Sparse autoencoders randomly limit the number of active nodes during training to create the bottleneck in the information flow, instead of reducing the number of nodes in the hidden layers. Fig. 2 – left shows a sample architecture of a sparse autoencoder, in which only the highlighted nodes will be activated, and the greyed-out nodes will not be used. This forced sparsity in the hidden layers prevents the autoencoder from learning the noise in the data, from reproducing the input identically, and it promotes that the autoencoder learns the intrinsic features in the input data. A way to force this sparsity is by including a penalisation term in the loss function that randomly yields an activation value very close to zero. The loss function is then $L\left(x, x'\right) + \lambda \sum \left|a_i^h\right|$, in which the activation a in the layer h for observation i is scaled by a parameter λ. Then, during the optimisation process, only the highest activation values are used to prevent the autoencoder from using all the hidden nodes at the same time. Sparse autoencoders have been used for medical image analysis [68], hyperspectral imagery [19], and fault detection [69].

3.4. Variational autoencoders

Variational autoencoders [70] are generative models that estimate the probability density function of the original training data. They are composed of an encoder and a decoder as well. The main difference with other types of autoencoders is that variational autoencoders learn the parameters of a probability distribution that represents the original data. Variational autoencoders force a continuous latent space from which data samples can be generated. This latent space is defined by computing a vector of means (μ) and a vector of standard deviations (σ) of the corresponding variable (see Fig. 2 - right). The vector of means defines where in the latent space the encoding of an input should be centred, while the vector of standard deviations defines a limited area in which the mean encoding can vary. The Kullback–Leibler (KL) divergence is introduced in the loss function to limit the values that (μ) and (σ) can take and to ensure that the compressed representations are close to each other, but that they are distinct. The loss function will minimise the KL divergence, which measures how much the two probabilities diverge from each other. Then, the loss function is $L\left(x, x'\right) + \sum KL(q\left(z|x\right) \parallel p(z))$, in which the first term penalises the reconstruction error and the second term, the sum of all KL divergences, induces the learned distribution $q\left(z|x\right)$ to be similar to the original distribution $p(z)$. The main limitation of regular autoencoders is that the latent space to which the input is encoded and where the encoded vectors lie may not be continuous. If the latent space is discontinuous, the decoder could reconstruct an unrealistic input. Variational autoencoders address this issue by defining a latent space based on the probability distribution of the original data. In this sense, variational autoencoders have generative features and can be regarded as a method to generate synthetic data rather than a data augmentation method. Variational autoencoders have shown very effective in generating many types of complex data including handwritten digits, faces, physical models [71]. Also, variational autoencoders have been used for a wide range of generative tasks including pixel anticipation in computer vision [72], generative text modelling [73], and generation of novel molecular structures [74].

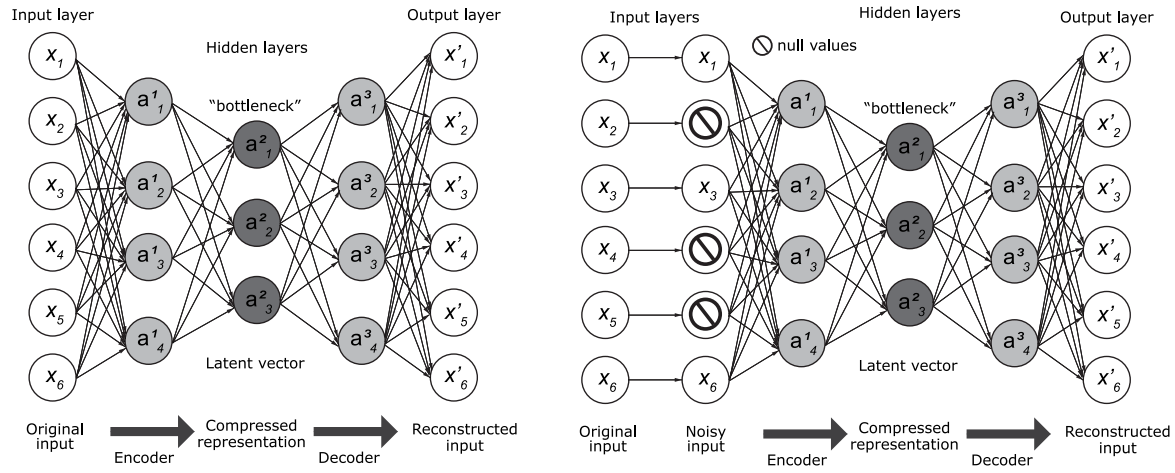

Fig. 1. Generic architectures of an undercomplete autoencoder (left) and a denoising autoencoder (right).

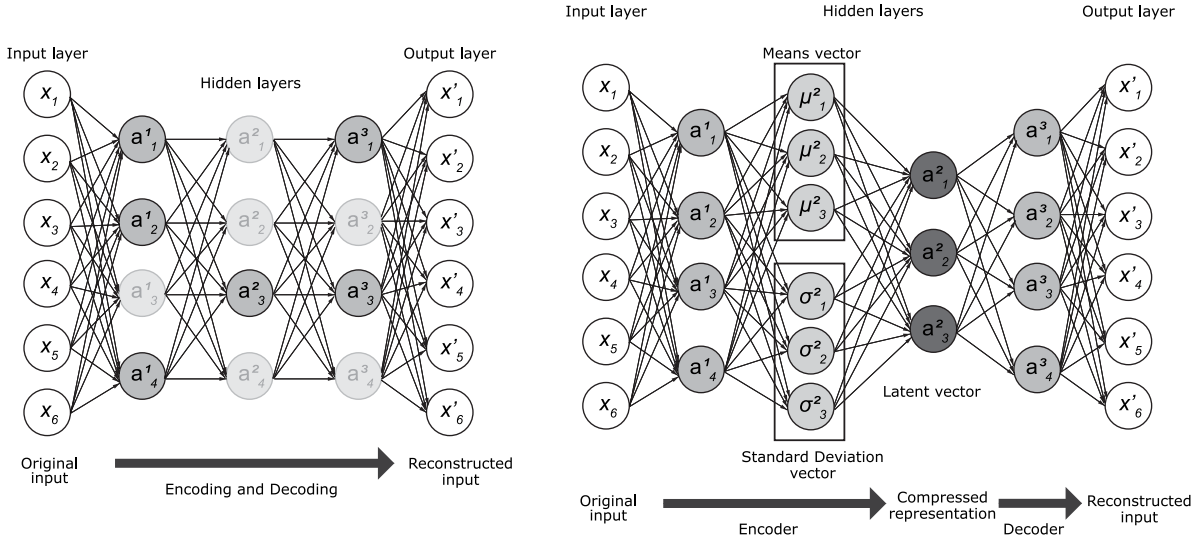

Fig. 2. Generic architectures of a sparse autoencoder (left) and a variational autoencoder (right).

4. Method and materials

The objective of this paper is to show how autoencoders can contribute to addressing the limited data challenge typical in construction management. Accurate cost estimation is one of the key activities that defines whether a project is profitable or not. Estimating the cost during tendering and preconstruction activities is a critical aspect for the successful delivery of construction projects. Poor estimations often lead to profit erosion and even economic losses. Estimating the cost of a construction project accurately is a difficult task that still has not been addressed satisfactorily [3,75]. This paper presents an investigation on how autoencoders can augment relatively small datasets and improve the performance of deep learning models to predict the cost of construction projects.

The method employed for this investigation is depicted in Fig. 3. Two sets of financial data from power transmission construction projects delivered in the last 12 years in the United Kingdom were used as input. One dataset relates to overhead transmission lines (*overhead* in Fig. 3) and the other one to *underground* cabling lines (*underground* in Fig. 3). Table 1 presents the data available in both datasets. These datasets represent the data that is traditionally used to come up with cost estimation during preconstruction. Note that for this study, only the continuous variables present in both datasets were used. The underground dataset consists of 27 projects, while the overhead dataset consists of 41 projects. The datasets were split into a training set and a test set. The training and test data split used are 21/6 and 31/10 for the underground and overhead datasets respectively. Note that all variables were normalised into a 0–1 range.

Then, both training sets were augmented using four types of autoencoders, i.e.: (1) an undercomplete autoencoder, (2) a sparse autoencoder, (3) a deep autoencoder, and (4) a variational autoencoder (Fig. 3). Table 2 presents the architectures and hyper-parameters of the four autoencoders used in this study. The architecture of the autoencoders defines the number and type of hidden layers and the activation function used in the nodes. The rectifier linear unit (reLU) function $g(x) = \max_{(0,x)}$ was used as an activation function in all the nodes.

4.1. Regression methods

For each dataset, underground and overhead, five deep neural network regression models (DNN regressors in Fig. 3) were trained with the four augmented training sets and with a training set without augmentation. Traditional non-deep learning methods were implemented as well. A Random Forest Regressor (RFR) [76] and Support Vector Regression (SVR) [77,78] were

implemented to provide additional baselines and compare the presented approach with other types of regression methods. These regression methods were selected because they are commonly used for these types of estimation tasks [79,80]. The implementation details of the three regression methods are as follows:

(1) DNN Regressor. The regression loss function used was $L\left(\hat{y}, y\right) = \frac{1}{2}\left(\hat{y} - y\right)^2$, in which y and $\hat{y}$ are the target and estimated values, respectively The vectorised forward propagation is: $\boldsymbol{Z}^l = \boldsymbol{W}^l\boldsymbol{X} + b^l$, $\boldsymbol{A}^l = g\left(Z^l\right)$, where $\boldsymbol{X}$ is the vector of input parameters, $\boldsymbol{W}$ is the vector of weights, $\boldsymbol{A}$ is the vector of activation functions, and l is the number of layers of the model. The vectorised backward propagation is: $\{\delta\boldsymbol{Z}^l = \delta\boldsymbol{A}^{l-1} * g^l\left(\boldsymbol{Z}^l\right); \delta\boldsymbol{W}^l = (1/2)\delta\boldsymbol{Z}^l \cdot \boldsymbol{A}^{(l-1)T}; \delta b^l = (1/2)\sum \delta\boldsymbol{Z}^l; \delta\boldsymbol{A}^{l-1} = \boldsymbol{W}^{(l)T} \cdot \delta\boldsymbol{Z}^l\}$.

(2) RFR. The basic idea behind RFR is combining multiple decision trees. In a decision tree, given the training set X and the test set Y, the dataset is partitioned so that the instances with the same corresponding y values are grouped together. The function used to measure the quality of the partitions is the mean square error $\text{MSE} = 1/n\sum\left(y - \hat{y}\right)^2$, in which n is the number of training samples. In this implementation, each tree is built from a bootstrap sample drawn with replacement from the training set.

(3) SVR. The SVR function used is: $\hat{\boldsymbol{y}} = (\boldsymbol{W}, (\Phi\boldsymbol{X})) + \text{b}$. Variables ζ_i and ξ_i^* are introduced to measure the deviation of samples; thus the SVR optimisation problem is expressed as $\min \frac{1}{2}\|\boldsymbol{W}\|^2 + C\sum\left(\zeta_i + \xi_i^*\right)$ subject to: $\{f\left(x^i\right) - y_i \leq \varepsilon + \zeta_i; y_i - f(x^i) \leq \varepsilon + \xi_i^*; \zeta_i, \xi_i^* \geq 0\}$. In which, C is the parameter that regulates the trade-off between the margin and the prediction error denoted by the variables ζ_i and ξ_i^*. The final regression function is $f(x) = \sum\left(a_i - a_i^*\right)K(x, y) + b$, where a_i, a_i^* are the Lagrange multipliers and $K(x, y)$ is the kernel function. In this case, a linear kernel function was used ($K(x, y) = x^T y + \text{b}$), $\text{C} = 1.0$, and $\varepsilon = 0.1$.

Table 2 presents the architecture and hyperparameters of the three regression methods used to calculate all the cost estimations. Also, the adadelta optimiser [81] and the adam optimiser [82] were used for training the autoencoders and the DNN regressors, respectively.

4.2. Increasing the size of the augmented dataset

The process above was repeated 20 times for each type of autoencoder. Each time, the training set was expanded by a higher data increment factor. That is, in the first augmentation iteration, the training set was doubled in size; in the second iteration, it was tripled, and so on until 20 iterations. The results of the increasing training sets using the four types of autoencoders were compared. The results of the training set without augmentation were used as a baseline using the DNN regressor, DTR, and RFR.

4.3. Performance metrics

Three metrics were used for comparing performances: (1) the coefficient of determination (r^2), which provides an indication of the goodness of fit by providing a measure of the variance between predicted and actual values. In this study, it is called the model score because is computed using the augmented data generated by the autoencoders and provides an indication of how closely the model represents the training data. It is a positively oriented metric and ranges from 0–1. The model score is computed as follows $r^2 = 1 - \sum(y_i - \widehat{y_i})^2 / \sum(y_i - \overline{y})^2$. Where $\sum(y_i - \widehat{y_i})^2$ is the sum of squares of the difference between the actual values (y_i) and the predicted values ($\widehat{y_i}$); and $\sum(y_i - \overline{y})^2$ is the sum of squares between the difference of the actual values (y_i) and their mean ($\overline{y}$). (2) The mean absolute error (MAE), which measures the average magnitude of the errors in absolute terms. It is calculated using: $\frac{1}{n}\sum|y_i - \widehat{y_i}|$, where n is the number of errors and $|y_i - \widehat{y_i}|$ are the absolute errors. (3) The root-mean-square-error (RMSE), which is the square root of the averaged squared error. It is computed as follows: $\sqrt[2]{1 - r^2}SD_y$, where SD_y is the standard deviation of Y. Both MAE and RMSE range from 0 to ∞ in the same units as the predicted variable. They are negatively-oriented metrics for which lower values indicate better performance. In this study, the reported MAE and RMSE are normalised into a 0–1 range, instead of their actual units, i.e., British pounds (£). This normalisation has been carried out due to the sensitive commercial nature of the data; however, the metrics still provide a clear indication of the performance of the models.

5. Results

Fig. 4 shows the different model scores of the underground (left) and overhead (right) datasets for predicting the total cost of the construction project and increments factors ranging from 1 to 20 using the DNN model. The increment factor defines the amount of additional data included in the training dataset. For both datasets, the autoencoders yield better results than the initial model. An increasing trend in performance can be observed as the increment factor increases. However, this increase in performance can potentially be attributed to the model overfitting the data, because the MAE and RMSE increase with higher increment factors, as shown in Fig. 5 and Fig. 6, respectively. For both underground and overhead datasets, the MAE and RMSE decrease initially as the increment factor increases and start to increase afterwards until they reach similar values as obtained in the initial model without augmentation. This is an indication that, in this case, data augmentation provides benefits up to a point and after that, the additional data can cause overfitting.

Regarding differences between datasets, the overhead datasets reach better results overall than the underground. This can be attributed to a stronger correlation between the variables in the overhead dataset. In other words, it is easier to estimate the cost of overhead transmission projects because the distance of the project correlates more closely to the cost; which is not the case with underground projects, in which the underground conditions are unknown. The variational autoencoder is the best performing for the underground dataset; while the deep architecture is the best performing for the overhead. This could be explained because the correlations between variables in the overhead dataset are more explicit than in the underground dataset. More substantial variations can be observed for the variational autoencoder that can be attributed to its generative nature.

Table 3 compares the performance metrics (model score, MAE and RMSE) for the DNN regressors trained with the original data and trained with data augmented with four types of autoencoders. Note that the performance metrics that correspond to increment factor equal to 5 have been used; because there are indications, in the graphs above, that for higher increment factors overfitting can be occurring. All autoencoders provide better model scores and lower MAE and RMSE. On average the autoencoders provide a model score improvement of 7.2% and 11.5% for the underground and overhead datasets, respectively. No autoencoder provides the best results for all metrics, but the deep and the variational autoencoder architectures provide the best results on average. Summarising, using any of the autoencoders presented here will improve all the performance metrics, which is an indication that data augmentation using autoencoders is beneficial for regression problems when using DNN models.

Non-traditional deep learning regression methods were investigated as well. Figs. 7 and 8 present the results using the

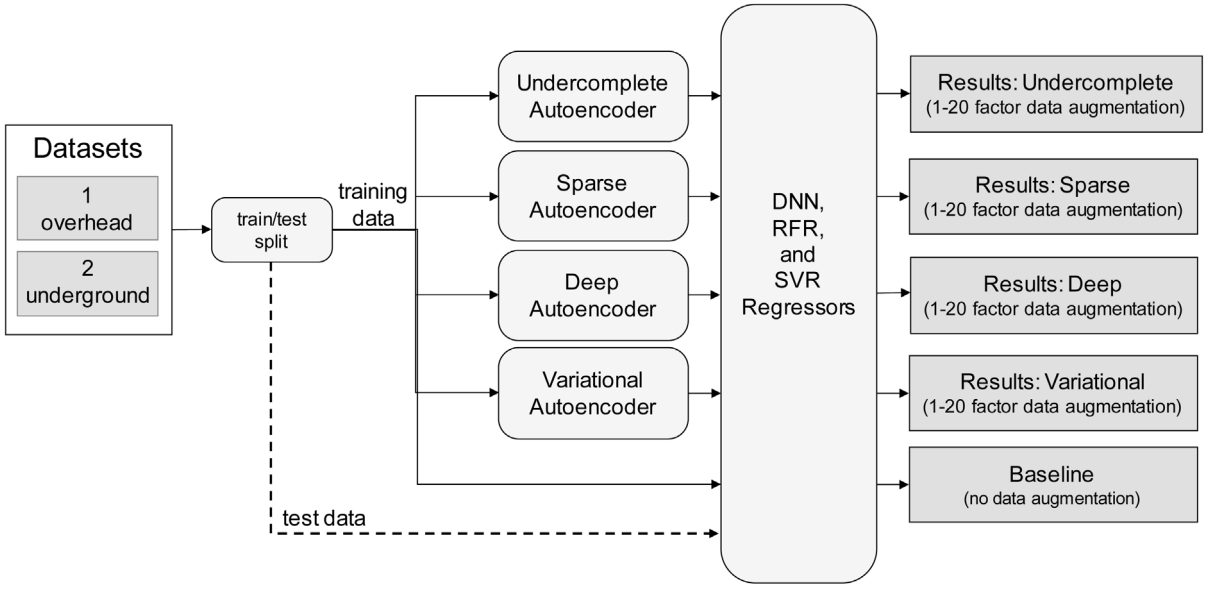

Fig. 3. Diagram of the method used.

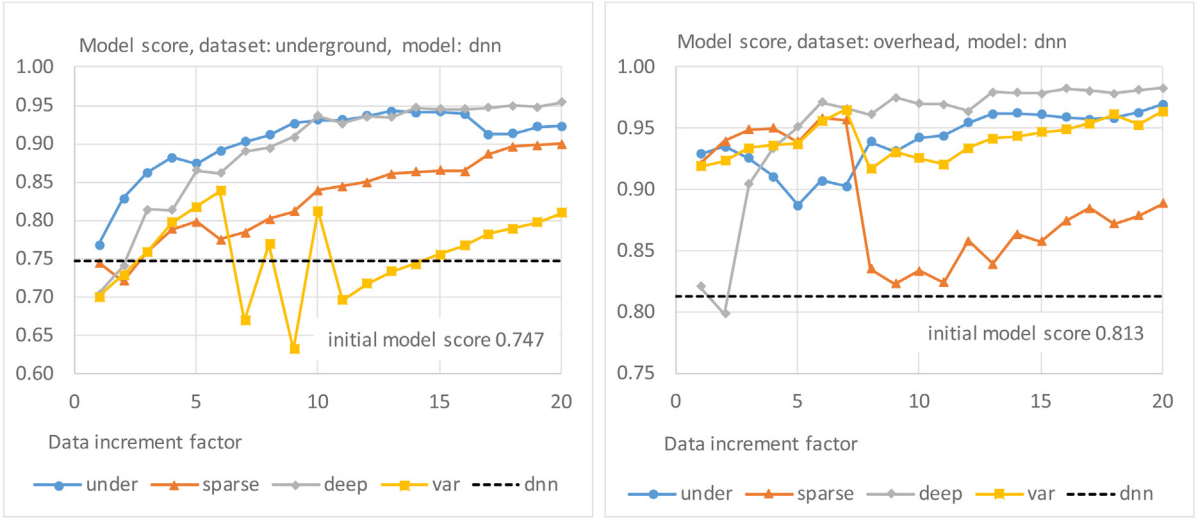

Fig. 4. Graphs showing the varying model scores of the underground dataset (left) and the overhead dataset (right) for predicting cost at various increment factors.

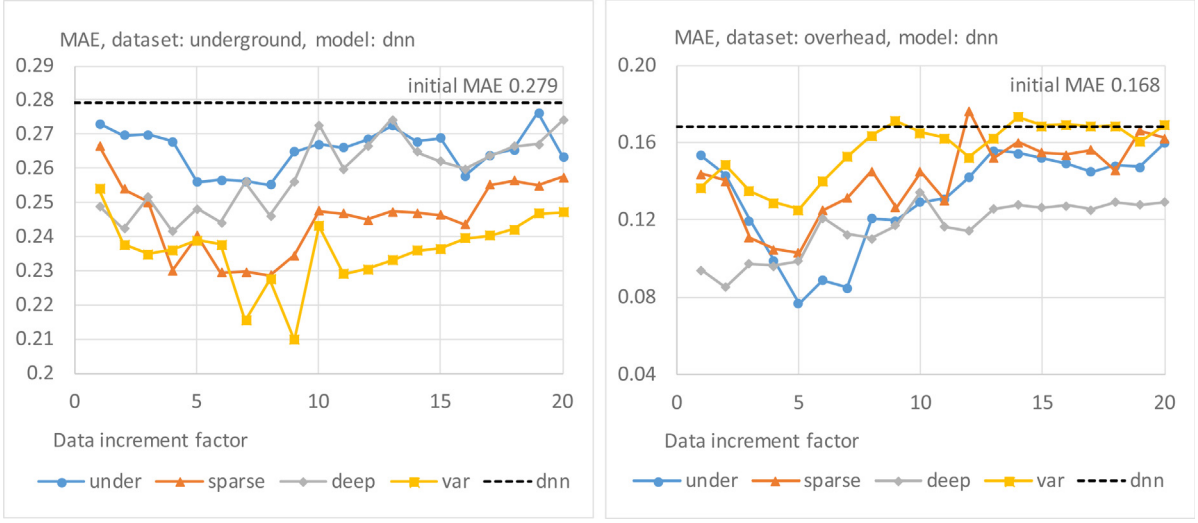

Fig. 5. Graphs showing the varying MAEs of the underground dataset (left) and the overhead dataset (right) for predicting cost at various increment factors.

Table 1
Variables from the overhead and underground datasets.

Variable	Description	Underground	Overhead
Distance	Distance in kilometres of the project.	✓	✓
Duration	Duration in weeks of the project.	✓	✓
Tender Value	Value in British pounds (£) for the tender proposal.	✓	
Sales Value	Final agreed value with the client in British pounds (£).	✓	
Cost	Total cost of the project in British pounds (£).	✓	✓
Profit	The difference between the Sales Value and the Cost in British pounds (£).	✓	✓
Region	Geographical region in which the project is located.	✓	✓
Client	The client that is requesting the project.		✓
Contract type	The type of contract of the project (e.g. lump sum, schedule of rates, framework agreement, quantity-based contract, etc.).	✓	
Voltage	Voltage in kV of the transmission line (33 kV, 66 kV, 133 kV, 275 kV, 400 kV).		✓

Table 2
Autoencoders used in this study.

Type	ID	Architecture	Hyper-parameters
Undercomplete	under	– Input data – Dense layer (initial vector size, reLU) – Encoded representation (latent vector size) – Dense layer (initial vector size) – Decoded representation (initial vector size) – Reconstructed data	Latent vector = 2 Optimiser = adadelta Loss = mse
Sparse	sparse	– Input data – Regularised layer (initial vector size, reLU) – Regularised layer (latent vector size*2, reLU) – Encoded representation (latent vector size) – Regularised layer (latent vector size*2, r reLU) – Regularised layer (initial vector size, reLU) – Decoded representation (initial vector size) – Reconstructed data	Latent vector = 2 Optimiser = adadelta Loss = mse
Sparse/Deep	deep	– Input data – Regularised layer (initial vector size, reLU) – Regularised layer (latent vector size*6, reLU) – Regularised layer (latent vector size*4, reLU) – Regularised layer (latent vector size*2, reLU) – Encoded representation (latent vector size) – Regularised layer (latent vector size*2, reLU) – Regularised layer (latent vector size*4, reLU) – Regularised layer (latent vector size*6, reLU) – Regularised layer (initial vector size, reLU) – Decoded representation (initial vector size) – Reconstructed data	Latent vector = 2 Optimiser = adadelta Loss = mse
Variational	var	– Input data – Dense layer (initial vector size, reLU) – Dense layer (initial vector size, mean) – Dense layer (initial vector size, variance) – Encoded representation (latent vector size) – Dense layer (latent vector size, reLU) – Dense layer (initial vector size, reLU) – Dense layer (initial vector size) – Decoded representation (initial vector size) – Reconstructed data	Latent vector = 2 Optimiser = adam Loss = mean (xent, kl)
DNN regressor		Optimiser = adam Activation function = reLU Hidden layers = 10	
Random Forest Regressor (RFR)		Quality of partition function = mse Number of trees = 10 Max depth = 3	
Singular Vector Regressor (SVR)		C = 1.0 $\varepsilon = 0.1$.	

See Section 3 for details regarding the autoencoders loss functions.

RFR and SVR models, respectively. Regarding RFR and the underground data set, only the deep autoencoder provided a better model score, while the MAE and RMSE yielded lower errors and remained somewhat constant through all increment factors. Regarding RFR and the overhead data set, all autoencoders yielded better results except for the undercomplete autoencoder. The MAE and RMSE remained at similar levels as the initial value throughout all the increment factors. Table 4 presents a comparison of the performance metrics using the RFR model for the underground and overhead datasets. For both datasets, the deep sparse autoencoder yielded the best results; while the undercomplete autoencoder yielded the worst results. Regarding MAE and RMSE no single autoencoder yielded the lowest errors, and no augmentation provided the largest errors.

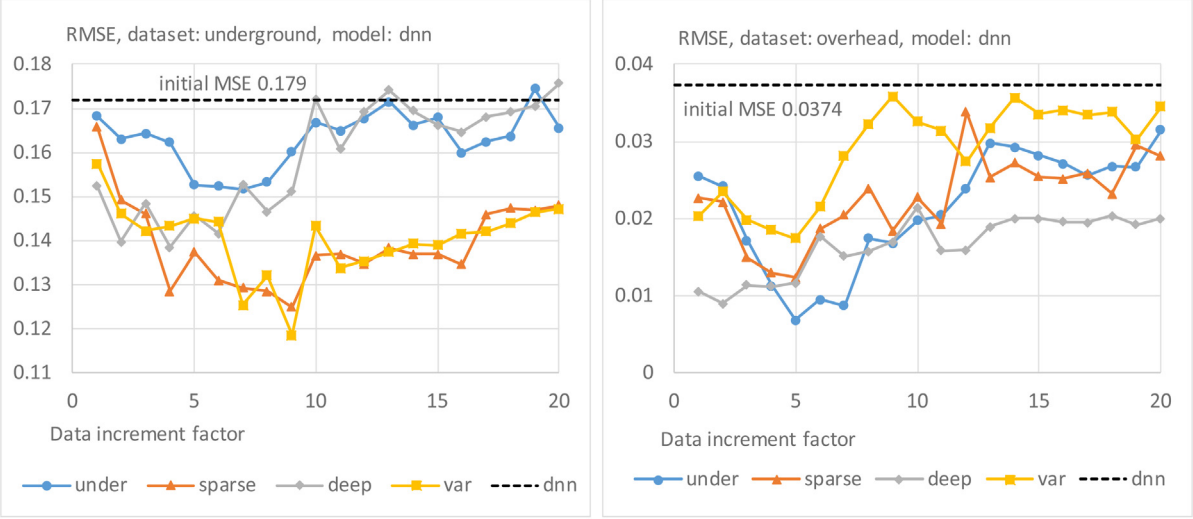

Fig. 6. Graphs showing the varying RMSEs of the underground dataset (left) and the overhead dataset (right) for predicting cost at various increment factors.

Table 3
Comparison of the performance metrics for the original data and autoencoders using the DNN model. Bold indicates the best results. **Bold** indicates the best result and underlined the worst.

DNN	Model score	MAE	RMSE
Underground			
Original data			
Deep Neural Network Regressor (DNN)	0.747	0.279	0.179
Augmented data			
Undercomplete autoencoder	**0.881**	0.267	0.162
Sparse autoencoder	0.788	**0.230**	**0.128**
Deep sparse autoencoder	0.813	0.241	0.138
Variational autoencoder	0.797	0.236	0.143
Overhead			
Original data			
Deep Neural Network Regressor (DNN)	0.813	0.168	0.037
Augmented data			
Undercomplete autoencoder	0.896	0.098	**0.011**
Sparse autoencoder	**0.950**	0.104	0.012
Deep sparse autoencoder	0.933	**0.096**	**0.011**
Variational autoencoder	0.936	0.128	0.018

Results for augmented data correspond to an increment factor equal to 5.

Table 4
Comparison of the performance metrics for the original data and autoencoders using the RFR model. Bold indicates the best results. **Bold** indicates the best result and underlined the worst.

RFR	Model score	MAE	RMSE
Underground			
Original data			
Random Forest Regressor (RFR)	0.891	0.746	0.647
Augmented data			
Undercomplete autoencoder	0.772	0.256	**0.129**
Sparse autoencoder	0.839	0.257	0.173
Deep sparse autoencoder	**0.921**	0.254	0.160
Variational autoencoder	0.889	**0.253**	0.167
Overhead			
Original data			
Random Forest Regressor (RFR)	0.843	0.102	0.025
Augmented data			
Undercomplete autoencoder	0.831	**0.063**	**0.009**
Sparse autoencoder	0.930	0.102	0.023
Deep sparse autoencoder	**0.945**	0.090	0.020
Variational autoencoder	0.932	0.080	0.017

Results for augmented data correspond to an increment factor equal to 5.

Regarding the SVR model, results are presented in Fig. 8 and Table 5. For the underground dataset, the model score improved using all autoencoders, while for the overhead dataset the model score remained somewhat constant at a slightly lower level than without augmentation, except for the undercomplete autoencoder which yielded larger errors on larger increment factors. For MAE and RMSE in the underground dataset, only the sparse autoencoder yielded better results without data augmentation. For the overhead dataset, none of the autoencoders provided lower MAE and RMSE. The sparse autoencoder yielded significantly higher errors for larger increment factors. Table 5 presents the results using the SVR model. For the underground dataset, the sparse and deep sparse yielded the best metrics, while the undercomplete yielded the worst. For the overhead dataset, no augmentation yielded the best model score, MAE and RMSE.

In summary, autoencoder-based data augmentation using RFR did not provide significant improvements, while using SVR decreased performance. The performances between the underground and overhead datasets were more dissimilar using the RFR and SVR than using DNN.

Table 5
Comparison of the performance metrics for the original data and autoencoders using the SVR model. Bold indicates the best results. **Bold** indicates the best result and underlined the worst.

SVR	Model score	MAE	RMSE
Underground			
Original data			
Support Vector Regressor (SVR)	0.788	0.358	0.213
Augmented data			
Undercomplete autoencoder	0.730	0.537	0.424
Sparse autoencoder	0.811	**0.250**	**0.183**
Deep sparse autoencoder	**0.885**	0.388	0.341
Variational autoencoder	0.858	0.294	0.212
Overhead			
Original data			
Support Vector Regressor (SVR)	**0.928**	**0.072**	**0.007**
Augmented data			
Undercomplete autoencoder	0.885	0.513	0.368
Sparse autoencoder	0.918	0.247	0.081
Deep sparse autoencoder	0.893	0.172	0.035
Variational autoencoder	0.882	0.201	0.048

Results for augmented data correspond to an increment factor equal to 5.

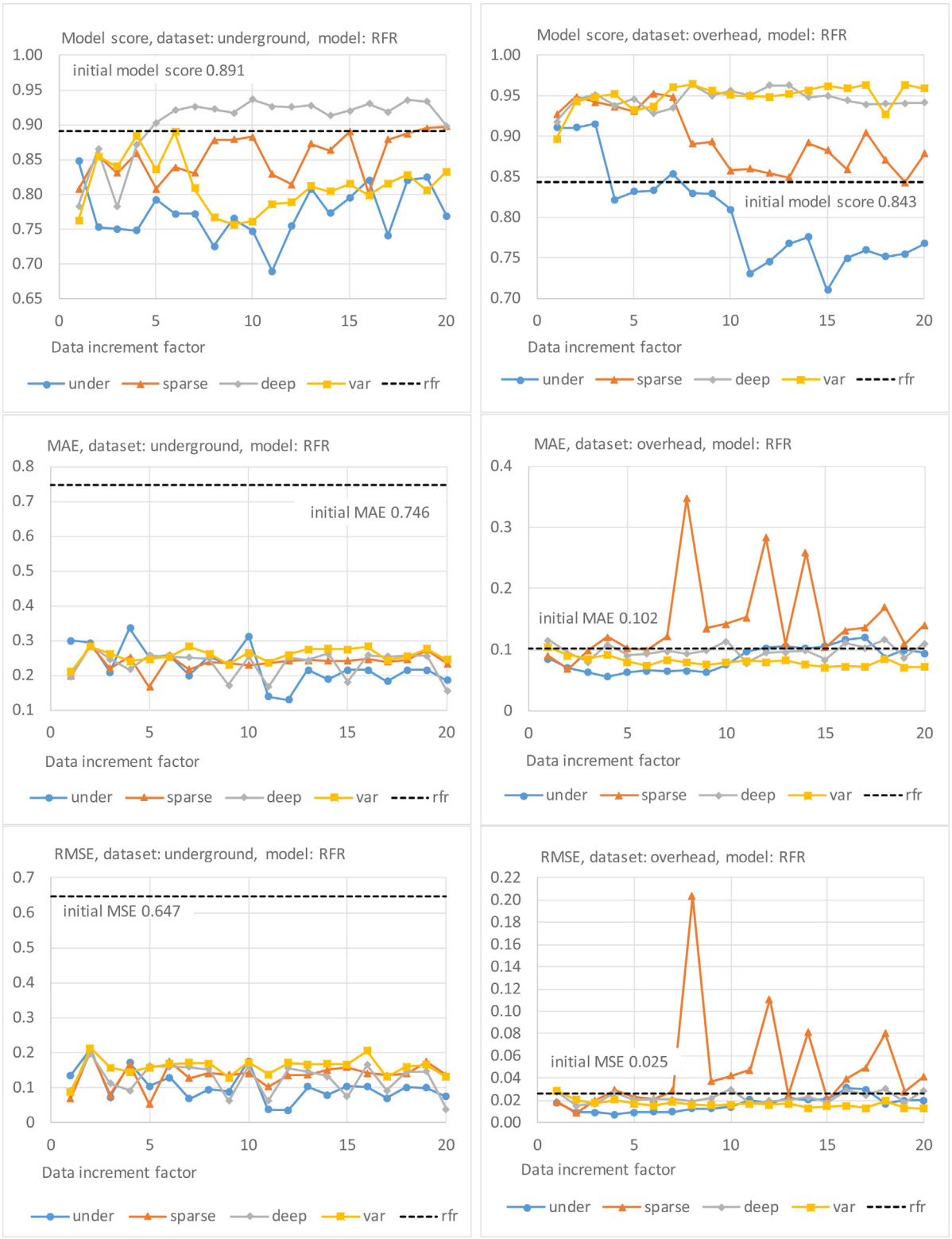

Fig. 7. Graphs showing the results (model score, MAE and RMSE) of the underground dataset (left) and the overhead dataset (right) for predicting cost at various increment factors using the RFR model.

5.1. Analysing the compressed representations

Another manner to analyse the behaviour and to study the underlying structure of the datasets is to plot the compressed representation generated by the autoencoders. Note that, in this case, the compressed representations are two-dimensional representations compressed from a multi-dimensional dataset. Fig. 9 presents four graphs that plot the compressed representations generated by the unsupervised autoencoders (undercomplete, sparse, and deep) and the generative autoencoder (variational) for the underground dataset. The total cost of each project is

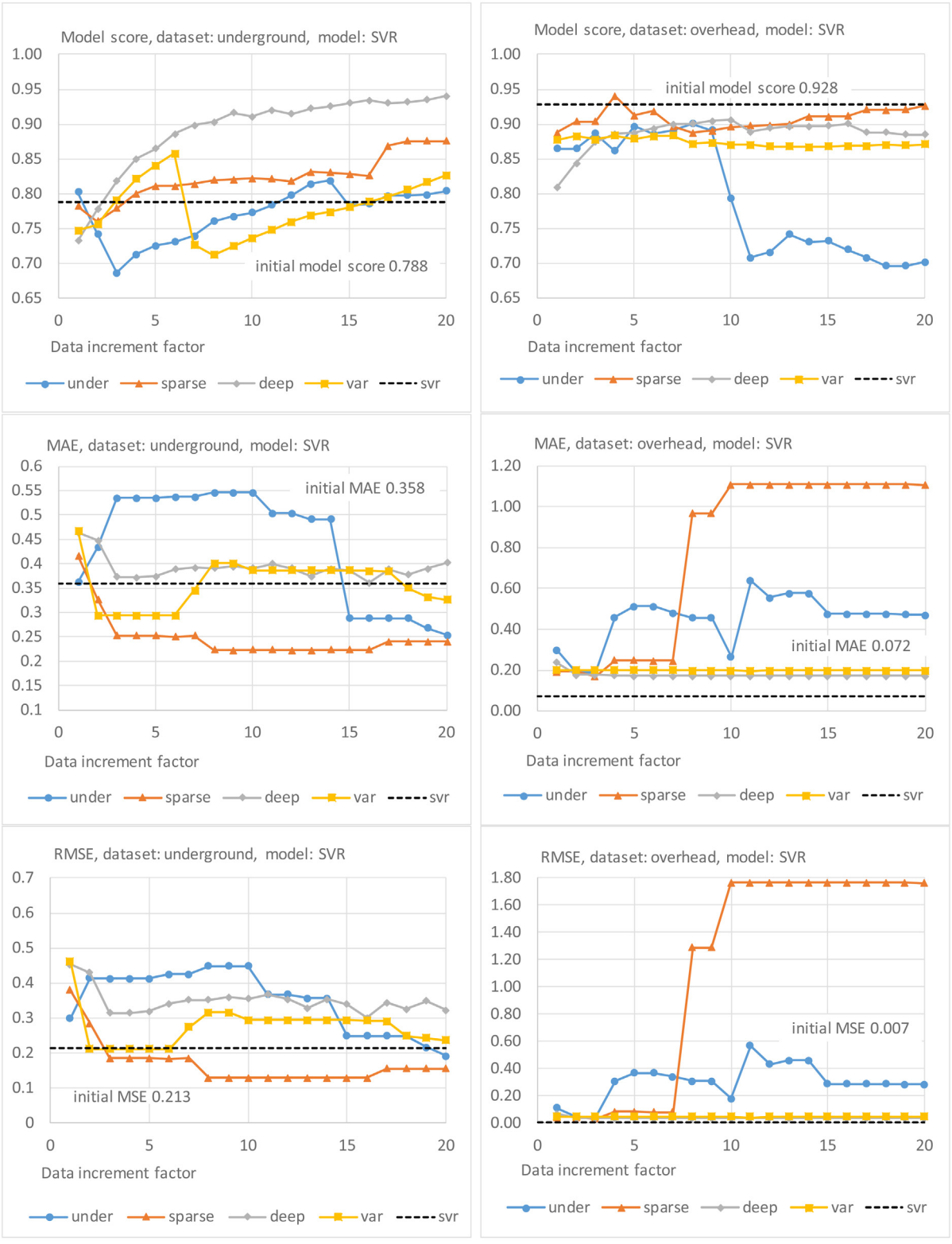

Fig. 8. Graphs showing the results (model score, MAE and RMSE) of the underground dataset (left) and the overhead dataset (right) for predicting cost at various increment factors using the SVR model.

indicated by different shades of colour. Different diameters of the circular markers indicate varying costs. Small diameters indicate small costs and larger diameters larger costs. For example, small dark blue circles represent projects with low costs, while large bright circles represent projects with large costs. The colour scale is different in each graph because a completely different dataset was generated by each autoencoder. The different nature of the unsupervised and the generative autoencoders can be seen in the different configurations of the compressed representations. All the unsupervised autoencoders present linear configurations along the main axis of the plots; while in the variational autoencoder well-defined clusters can be observed. In

general, for all cases, a positive relationship between the values of the compressed representation and cost is present.

Fig. 10 presents similar graphs for the overhead dataset, in which similar configurations can be observed. That is, the unsupervised autoencoders present strong linear configurations along the main axis of the plot, and the variational autoencoder presents well-defined clusters. Note that for the overhead dataset the linear nature of the configurations is more defined, which can be attributed to stronger linear correlations between variables in the overhead dataset. This strong linear configuration is present even for the variational autoencoder, in which the observable clusters are arranged in a linear manner across the graph. The undercomplete and sparse autoencoders present very similar configurations in which the instances are arranged in an increasing order based on cost; while in the deep autoencoder this arrangement is not as clear. The other characteristic of the configuration present in all cases is that the instances with lower costs are grouped together very closely; while the instances with higher costs are distributed sparsely across the latent space.

In all graphs, defined differences between instances based on cost can be observed. These differences can be leveraged to define a categorisation of projects in relation to cost and complexity. For example, in the variational graph in Fig. 7, three groups could be defined as follows: (1) low-cost/low-complexity, (2) mid-cost/mid-complexity and (3) high-cost/high-complexity. For the variational graph in Fig. 8, a similar categorisation could be made as well but defining four groups instead of three. These types of groupings based on the configurations of compressed representations can be very useful, as they can be used by other machine learning algorithms to automatically classify projects. For example, during the planning stages of a new project, an autoencoder can use the existing information of the project to classify it based on the previously generated latent categories. Then adequate resources can be allocated that correspond to the identified category to ensure successful project delivery. Note that there are no implications of having high or low-cost projects as long as the cost estimations are accurate. The compressed representations generated by the variational autoencoder indicate that the underlying relations among variables are very similar for low-cost projects, mid-cost projects and high-cost projects. That is why clusters can be identified in Figs. 9 and 10. Note as well that the relation among variables seems stronger for low-cost projects as the clusters are denser than for mid-cost or high cost, as the clusters are more spread out.

Fig. 11 presents the compressed representations using the variational autoencoder for the underground and overhead datasets, in which the colour scale represents the profit margin instead of the total cost, as in Figs. 9 and 10. It can be seen that there is no clear relation between profit margin and the clustering of the compressed representations; as was the case with Figs. 9 and 10, in which the clusters correspond to different levels of costs. This is an indication that the profit margin is not strongly related to the variables included in the compressed representation. This is the opposite case as the total cost, in which a clear relation between the compressed representations and the total cost can be observed particularly in the compressed representations generated by the variational autoencoders. In other words, low-cost and high-cost projects are not directly related to a specific level of profit margins.

6. Discussion

The main findings of this study can be summarised as follows: (1) using autoencoders for data augmentation is a potentially good approach when using neural network regressors. All the autoencoders investigated provided significantly better model scores and lower MAE and RMSE errors. (2) However, using autoencoders for data augmentation when using other non-deep learning methods does not provide clear improvements. For example, in the presented experiments for RFR data, augmentation did not provide significant improvements, while for SVR decrease in performance was observed. This indicates that data augmentation using autoencoders is only beneficial for neural network regressors. Note that the objective of this study is not to compare the performance of different regression methods, but to investigate if autoencoders can be used for data augmentation. (3) The compressed representations created by the autoencoders can provide useful insights regarding the underlying non-linear relations among variables in the datasets. In particular, the variational autoencoder was able to highlight the relations among variables by generating distinguishable clusters (Figs. 9 and 10). (4) Furthermore, the compressed representation can potentially be used as well to investigate whether variables are correlated or not. In this case, it was shown that the total cost is highly correlated to the other variables in the compressed representation, while the profit margin was not (Figs. 9, 10, and 11). (5) Lastly, this study presents evidence that using autoencoders for data augmentation can improve the generalisation of deep neural network regression models. This point is addressed in more detail in the following subsection.

6.1. Improving generalisation

The main issue with models trained using small datasets is that the models do not generalise well. That is when the model is confronted with data slightly different than the one used during training it performs poorly. Models trained with very big datasets do not suffer from this issue as increasing the size of datasets increases model generalisation. There are three approaches to improve generalisation (i) collect more data, (ii) improve your dataset, and (iii) transfer learning. Collect more data can be in many cases, impractical or impossible, and to use transfer learning existing trained models are required. For construction management, improving datasets is the alternative with the best trade-offs. This paper proposes autoencoders as good tools for data augmentation for transformation variant datasets. The idea is to generate larger datasets based on the original dataset to improve generalisation. Existing methods for data augmentation only work with data invariant datasets such as images. Financial data is a transformation variant so different approaches are required. This paper provides empirical evidence that autoencoders are a good alternative for data augmentation to improve generalisation in models that use construction financial data. Note that improving current datasets should be considered as an interim approach and that efforts to collect larger datasets must continue. Note as well, that data augmentation approaches have a limit on the increase in performance. As can be seen in Figs. 5 and 6, the MAE and RMSE start decreasing as more augmented data is used but increase again with more data. This is an indication that overfitting is occurring. Overfitting ensues when a machine learning model learns features that occur a lot in the training dataset and discards small variations. Overfitting reduces generalisation. Therefore, a balance should be found so that the amount of augmented data improves generalisation, but it does not cause overfitting. In Table 3, the results of the metrics for an increment in the dataset with a factor of 5 is presented, because, for this case, it seems that overfitting may start occurring with larger increment factors. The potential overfitting behaviour observed in this study should be further investigated in other datasets.

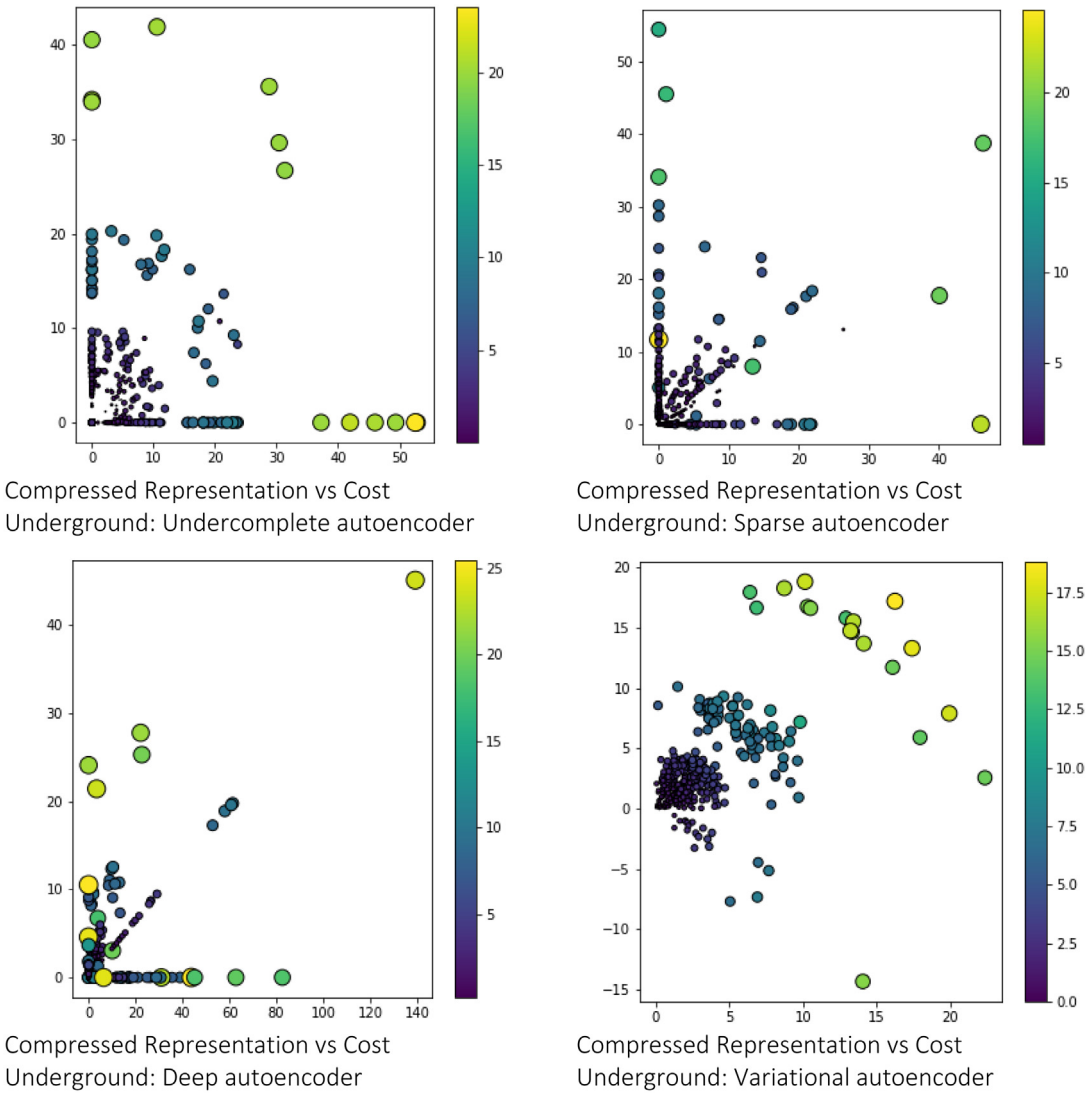

Fig. 9. Scatter graphs of the underground dataset project data encoded in two dimensions using the four autoencoders studied. The vertical and horizontal axes are unitless. The total cost of each project is indicated by different shades of colour. The colour scale is also unitless. Different diameters of the circular markers indicate varying costs. Small diameters indicate small costs and larger diameters larger costs.

6.2. Data privacy

Data privacy is a major challenge for implementing deep learning. Lack of data privacy is a significant limitation for companies to adopt deep learning. Financial data is particularly sensitive, and construction companies are usually reluctant to share this information with experts for data analytics. Data used in construction management activities are very sensitive as well and in it resides the competitive advantage of construction companies. The approach presented in this paper contributes to protecting sensitive data as the augmented datasets preserve the same correlations of the original dataset but with different values. Using the approach presented in this paper, a construction company could generate an augmented dataset based on its original data, and then share the augmented dataset with data analysts without the need to share the real dataset.

6.3. Limitations and further research

The main limitation of this study is that it uses only two similar types of datasets to evaluate the proposed approach, i.e.: overhead and underground transmission lines, which limits the generalisation assessment of the approach. This is a complex limitation to address because the lack of existing datasets is the main motivation of this study, but it is also the main limitation for its validation. Nevertheless, the approach has been presented in a detailed manner so that it can be easily implemented by other researchers. Consequently, further research efforts should focus on testing the approaches presented in this paper with other types of datasets and deep learning models. This will confirm that autoencoders are suitable models for data augmentations and provide more information related to the overfitting behaviours observed in the study. Also, a comparison of latent representations of different types of datasets could provide valuable insights regarding the underlying correlation among data attributes. Further research should also focus on how to handle and interpret categorical variables. A large part of the construction management data consists of categorical variables; which is not the case with other datasets on which data augmentation has been used, e.g. images. Generative adversarial neural networks [83], another prominent generative model, should be investigated as an alternative to variational autoencoders. Further studies should also be focused on identifying the specific characteristics of datasets used in construction management and fine-tune data augmentation techniques to fit those characteristics.

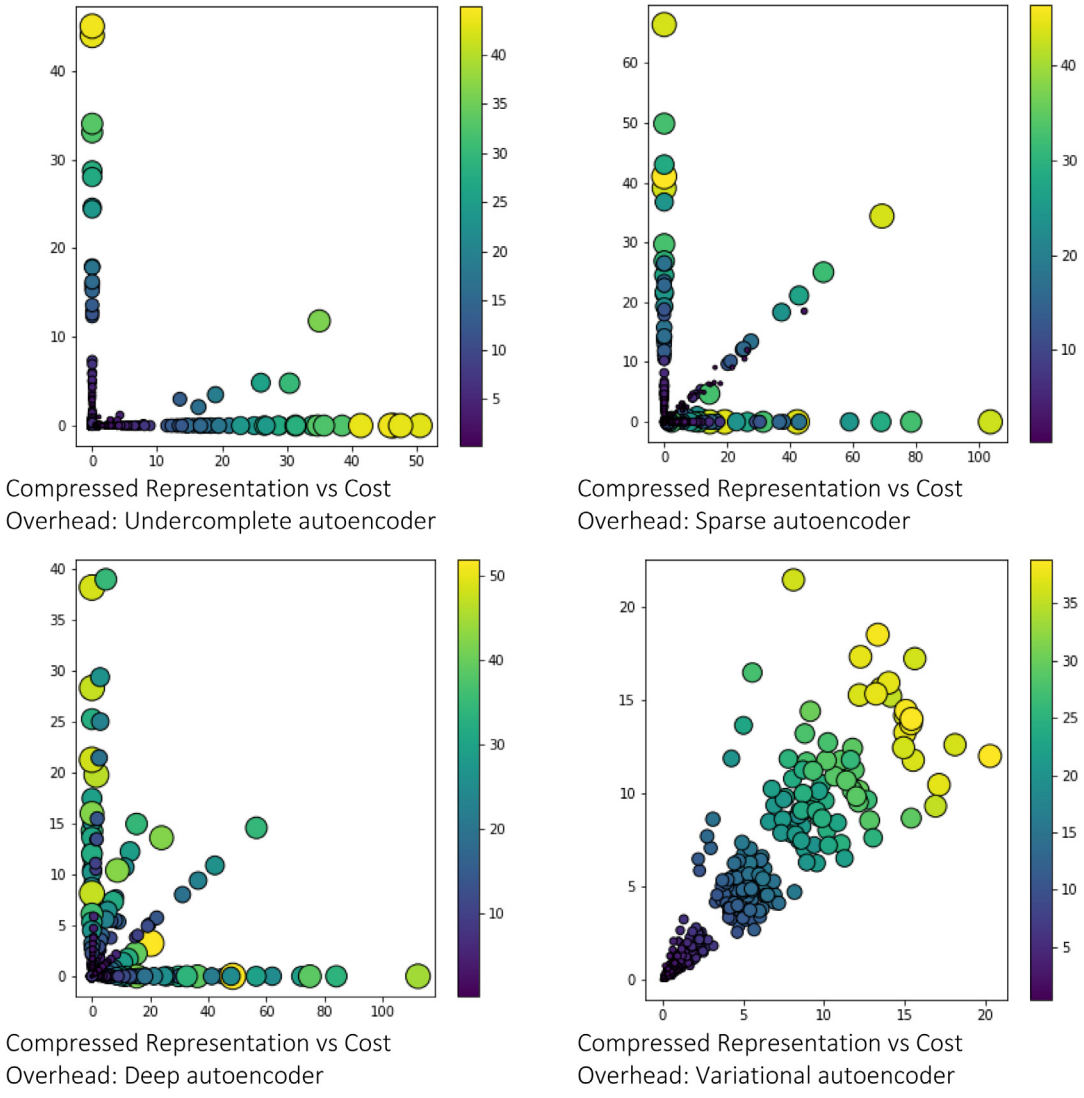

Fig. 10. Scatter graphs of the overhead dataset project data encoded in two dimensions using the four autoencoders studied. The vertical and horizontal axes are unitless. The total cost of each project is indicated by different shades of colour. The colour scale is also unitless. Different diameters of the circular markers indicate varying costs. Small diameters indicate small costs and larger diameters larger costs.

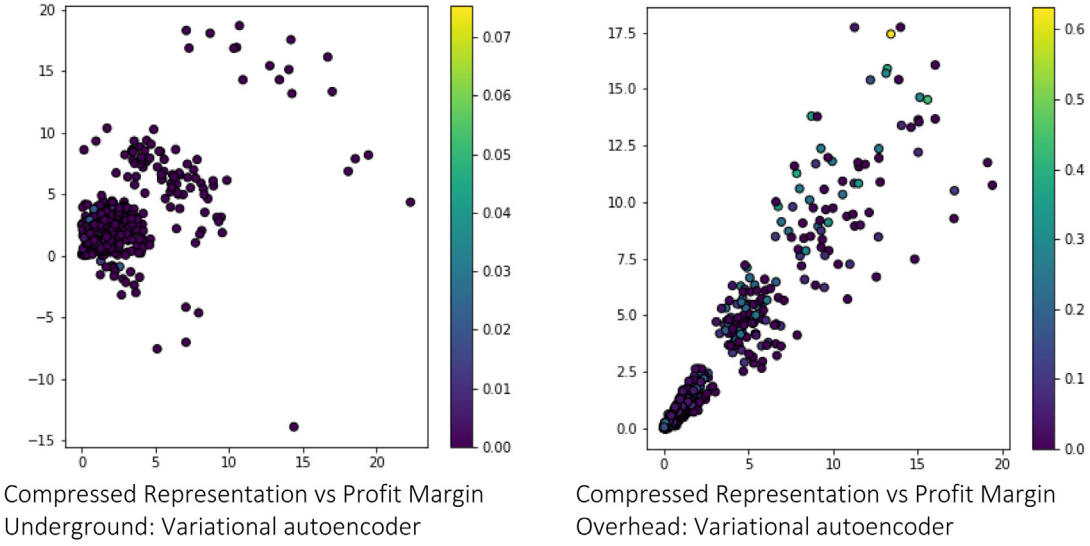

Fig. 11. Scatter graphs of the compressed representations. The vertical and horizontal axes are unitless. The profit margin of each project is indicated by different shades of colour. The colour scale is also unitless.

6.4. Contribution to knowledge

The contribution to knowledge of this study are (1) it presented a data augmentation technique for transformation variant data. Many techniques have been developed for transformation invariant data that have contributed to improving the performance of deep learning models (e.g. [15]). This study showed that autoencoders could be a good option for data augmentation for transformation variant data. (2) It presented empirical evidence showing that autoencoders can help with generalisation problems when large datasets are not available. (3) It provided evidence that data augmentation using autoencoders is useful for deep learning regression models such as DNN, but not for other regression models such as RFR and SVR. Lastly, (4) it showed that compressed representations generated by variational autoencoders could provide valuable insights that can unearth underlying relations among variables in the datasets.

7. Implications for practice

The main implication for practice of this study is that it provides stakeholders with an effective interim solution to the lack of accessibility to big datasets. In many cases, compiling a sufficiently large dataset is not possible. This study provides a practical approach to stakeholders to improve existing datasets so that advanced deep learning techniques can be used. These types of solutions have been applied in other sectors successfully. This study shows that data augmentation and synthetic data are valid approaches to address the challenge of limited data and can be used by practitioners in construction management as well.

This study also presents practitioners with a method to extract insights from data by plotting the data compressed representations. These graphical visualisations represent the underlying non-linear correlations of the data. These visualisations can be used by practitioners to extract hidden insights from project data. For example, as the study showed, stakeholders can use latent visual representations to group projects in different categories. Projects in different categories can then be treated differently according to their similarities to ensure successful project delivery. For example, additional resources or additional oversight can be allocated to more complex projects.

Lastly, this study also provides practitioners with a method to improve data privacy. Financial data on construction projects contain sensitive information that must not be publicly available. The approach presented in this paper enables practitioners to generate an analogous dataset based on the original. The analogous dataset preserves the underlying correlations among attributes in the dataset, but the individual instances are not the same. Using the analogous dataset for data analytics provides an extra layer of security. For example, construction companies have relatively limited access to in-house data specialists. For many construction companies is impossible to hire experts dedicated to deep learning. The approach presented here enables construction companies to generate analogous datasets that can be shared with external data analysts without sharing the real data.

8. Conclusions

This paper investigated autoencoders as potential tools to address the challenges with limited data availability in construction management. Autoencoders can be an effective alternative when compiling sufficiently big datasets becomes a prohibitively expensive task. Three types of unsupervised autoencoders (undercomplete, sparse, and deep) and one generative autoencoder (variational) were compared. The autoencoders were used to augment two financial datasets of underground and overhead power transmission projects. Better results were obtained when using the augmented datasets generated by all autoencoders. On average the autoencoders provide a model score improvement of 7.2% and 11.5% for the underground and overhead datasets, respectively. MAE and RMSE are lower for all autoencoders as well. The average error improvement for the underground and overhead datasets is 22.9% and 56.5%, respectively. Overall, the deep and variational autoencoders were the best performing models. The generative nature of the variational autoencoder can provide added benefits as the latent representations generated can be used for project classification, as discussed in Section 5.1. The augmentation method proposed here was tested using other non-deep-learning regression methods, i.e. RFR and SVR. Contrary to the DNN, significant improvements were not achieved when using RFR and SVR. This can be explained due to the completely different nature among the regression models.

The novelty of this study is that presents an approach to improve existing datasets and thus improve the generalisation of deep learning models when other approaches are not feasible. For instance, even though the size of construction datasets is increasing by the hour, compiling big enough datasets is onerous, and in many cases, is an impossible task; presenting an unavoidable hurdle that limits the adoption of deep learning models for construction management tasks. This paper provides empirical evidence that autoencoders are a good alternative to facilitate the implementation of deep learning models, while larger datasets are available.

CRediT authorship contribution statement

Juan Manuel Davila Delgado: Conceptualization, Methodology, Software, Validation, Formal analysis, Investigation, Resources, Data curation, Writing – original draft, Writing – review & editing, Visualization, Project administration. **Lukumon Oyedele:** Funding acquisition, Supervision.

Declaration of competing interest

The authors declare that they have no known competing financial interests or personal relationships that could have appeared to influence the work reported in this paper.

Acknowledgements

The authors would like to gratefully acknowledge the EPSRC and Innovate UK for funding this research under grant 102061, application number: 44746-322224.

Research data

The data used in this study is confidential due to its sensitive commercial nature.

References

[1] R. Vrijhoef, L. Koskela, A critical review of construction as a project-based industry: identifying paths towards a project-independent approach to construction, Proc. CIB Comb. Forces (2005) 1–10.
[2] O.J. Olaniran, P. Love, D. Edwards, A. Olatunji, J. Matthews, Cost overruns in hydrocarbon megaprojects: A critical review and implications for research, Proj. Manag. J. 46 (6) (2015) 126–138.
[3] J.S. Shane, K.R. Molenaar, S. Anderson, C. Schexnayder, Construction project cost escalation factors, J. Manage. Eng. 25 (4) (2009) 221–229.
[4] US Census Bureu, Business Dynamic Statistics, 2014.
[5] TCI, Top 100 Construction Companies 2017, 2018.
[6] C.H. Caldas, L. Soibelman, Automating hierarchical document classification for construction management information systems, Autom. Constr. 12 (4) (2003) 395–406.

[7] J.M. Davila Delgado, L. Oyedele, M. Bilal, A. Ajayi, L. Akanbi, O. Akinade, Big data analytics system for costing power transmission projects, J. Constr. Eng. Manag. 146 (1) (2020) 05019017.
[8] L. Wang, F. Leite, Knowledge discovery of spatial conflict resolution philosophies in BIM-enabled MEP design coordination using data mining techniques: A proof-of-concept, Computing in Civil Engineering - Proceedings of the 2013 ASCE International Workshop on Computing in Civil Engineering, 2013.
[9] H.-B. Liu, Y.-B. Jiao, Application of genetic algorithm-support vector machine (GA-SVM) for damage identification of bridge, Int. J. Comput. Intell. Appl. 10 (04) (2011) 383–397.
[10] E. Murchu, D. Platt, G. Webb, The Performance Advantages of Digitizing the Built Environment, 2016.
[11] B. Ali, The Silicon Valley giant that wants to reinvent construction, 2018, Construction News, available at: https://www.constructionnews.co.uk/best-practice/technology/the-silicon-valley-giant-that-wants-to-reinvent-construction/10034264.article (accessed 8 2019).
[12] D. Farnham, Global construction industry expected to reach $10 trillion by 2020, in: Case Study Research: Design & Methods: Applied Social Research Methods Series, 2018.
[13] J. Whyte, A. Stasis, C. Lindkvist, Managing change in the delivery of complex projects: Configuration management, asset information and 'big data', Int. J. Proj. Manage. 34 (2) (2016) 339–351.
[14] P.Y. Simard, D. Steinkraus, J.C. Platt, Best practices for convolutional neural networks applied to visual document analysis, in: Seventh International Conference on Document Analysis and Recognition, 2003, Proceedings (2003) 958–962.
[15] A. Krizhevsky, I. Sutskever, G.E. Hinton, ImageNet Classification with Deep Convolutional Neural Networks, 2012.
[16] A. Handa, V. Patraucean, V. Badrinarayanan, S. Stent, R. Cipolla, Understanding Real World Indoor Scenes With Synthetic Data, The IEEE Conference on Computer Vision and Pattern Recognition (CVPR), 2016, pp. 4077–4085.
[17] B. Zoph, D. Yuret, J. May, K. Knight, Transfer Learning for Low-Resource Neural Machine Translation, 2016.
[18] S. Gurumurthy, R. Kiran Sarvadevabhatla, R. Venkatesh Babu, DeLiGAN : Generative adversarial networks for diverse and limited data, in: CVPR 2017, 2017, pp. 166–174.
[19] Chao Tao, Hongbo Pan, Yansheng Li, Zhengrou Zou, Unsupervised spectral–spatial feature learning with stacked sparse autoencoder for hyperspectral imagery classification, IEEE Geosci. Remote Sens. Lett. 12 (12) (2015) 2438–2442.
[20] M. Kampffmeyer, S. Lø kse, F.M. Bianchi, R. Jenssen, L. Livi, The deep kernelized autoencoder, Appl. Soft Comput. 71 (2018) 816–825.
[21] H.-M. Chen, K.-C. Chang, T.-H. Lin, A cloud-based system framework for performing online viewing, storage, and analysis on big data of massive BIMs, Autom. Constr. 71 (2016) 34–48.
[22] S. Gao, L. Li, W. Li, K. Janowicz, Y. Zhang, Constructing gazetteers from volunteered big geo-data based on Hadoop, Comput. Environ. Urban Syst. 61 (2017) 172–186.
[23] C. Yang, M. Yu, F. Hu, Y. Jiang, Y. Li, Utilizing cloud computing to address big geospatial data challenges, Comput. Environ. Urban Syst. 61 (2017) 120–128.
[24] J. Xia, C. Yang, Q. Li, Using spatiotemporal patterns to optimize earth observation big data access: Novel approaches of indexing, service modeling and cloud computing, Comput. Environ. Urban Syst. (2018) available at:https://doi.org/10.1016/J.COMPENVURBSYS.2018.06.010.
[25] K.K. Han, M. Golparvar-Fard, Potential of big visual data and building information modeling for construction performance analytics: An exploratory study, Autom. Constr. 73 (2017) 184–198.
[26] M. Martínez-Rojas, N. Marín, M.A. Vila, The role of information technologies to address data handling in construction project management, J. Comput. Civ. Eng. 30 (4) (2016) 04015064.
[27] D.W. Halpin, G. Lucko, B.A. Senior, Construction Management, Wiley, 2011.
[28] J. Manyika, S. Ramaswamy, K. Somesh, H. Sarrazin, G. Pinkus, G. Sethupathy, A. Yaffe, Digital America: A Tale of Haves and Have-Mores, 2015.
[29] J.M. Davila Delgado, I. Brilakis, C.R. Middleton, Open data model standards for structural performance monitoring of infrastructure assets, in: J. Beetz (Ed.), CIB W78 Conference 2015, TU Eindhoven, Eindhoven, The Netherlands, 2015, pp. 1–10.
[30] J.M. Davila Delgado, L.J. Butler, N. Gibbons, I. Brilakis, M.Z.E.B Elshafie, C. Middleton, Management of structural monitoring data of bridges using BIM, Proc. Inst. Civ. Eng. 170 (3) (2017) 204–218.
[31] T. Gerrish, K. Ruikar, M.J. Cook, M. Johnson, M. Phillip, Attributing in-use building performance data to an as-built building information model for lifecycle building performance management, in: J. Beetz (Ed.), Proceedings of the 32nd CIB W78 Conference, CIB, 2015, pp. 1–11.
[32] J.M. Davila Delgado, L.J. Butler, I. Brilakis, M.Z.E.B Elshafie, C.R. Middleton, Structural performance monitoring using a dynamic data-driven BIM environment, J. Comput. Civ. Eng. 32 (3) (2018) 04018009.
[33] M. Mousa, X. Luo, B. McCabe, Utilizing BIM and carbon estimating methods for meaningful data representation, Procedia Eng. 145 (2016) 1242–1249.
[34] M. Bilal, L.O. Oyedele, J. Qadir, K. Munir, S.O. Ajayi, O.O. Akinade, H.A. Owolabi, et al., Big data in the construction industry: A review of present status, opportunities, and future trends, Adv. Eng. Inf. 30 (3) (2016) 500–521.
[35] S.H. Hamilton, C.A. Pollino, A.J. Jakeman, Habitat suitability modelling of rare species using Bayesian networks: Model evaluation under limited data, Ecol. Model. 299 (2015) 64–78.
[36] A.G. Roy, S. Conjeti, D. Sheet, A. Katouzian, N. Navab, C. Wachinger, Error corrective boosting for learning fully convolutional networks with limited data, in: International Conference on Medical Image Computing and Computer-Assisted Intervention, Springer, Cham, 2017, pp. 231–239.
[37] N.G. Polson, J.G. Scott, J. Windle, BayesIan inference for logistic models using polya-gamma latent variables, 2012.
[38] N. Hazrati, B. Shams, S. Haratizadeh, Entity representation for pairwise collaborative ranking using restricted Boltzmann machine, Expert Systems with Applications, Pergamon 116 (2019) 161–171.
[39] L. Zhao, Y. Zhou, H. Lu, H. Fujita, Parallel computing method of deep belief networks and its application to traffic flow prediction, Knowl.-Based Syst. 163 (2019) 972–987.
[40] Z. Gan, R. Henao, D. Carlson, L. Carin, Learning deep sigmoid belief networks with data augmentation, Artif. Intell. Stat. (February) (2015) 268–276.
[41] M. Aubry, D. Maturana, A. Efros, B. Russell, J. Sivic, Seeing 3d chairs: exemplar part-based 2d-3d alignment using a large dataset of cad models, Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, 2014.
[42] M. Aubry, B. Russell, Understanding deep features with computer-generated imagery, Proceedings of the IEEE International Conference on Computer Vision, 2015.
[43] A. Dosovitskiy, P. Fischer, E. Ilg, P. Hausser, C. Hazirbas, Flownet: Learning optical flow with convolutional networks, in: Proceedings of the IEEE International Conference on Computer Vision, 2015.
[44] S. Gupta, R. Girshick, P. Arbeláez, J. Malik, Learning rich features from RGB-d images for object detection and segmentation, in: Computer Vision ECCV 2012, Springer, Cham, 2014, pp. 345–360.
[45] S. Gupta, P. Arbeláez, R. Girshick, J. Malik, Aligning 3D models to RGB-D images of cluttered scenes, in: Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, 2015.
[46] J. Wu, I. Yildirim, J.J. Lim, B. Freeman, J. Tenenbaum, Galileo: Perceiving physical object properties by integrating a physics engine with deep learning, in: (NIPS 2015), Adv. Neural Inf. Process. Syst. 28 (2015) 127–135.
[47] S. Thrun, L. Pratt, Learning to learn: Introduction and overview, in: Learning To Learn, Springer US, Boston, MA, 1998, pp. 3–17.
[48] J. Yosinski, J. Clune, Y. Bengio, H. Lipson, How transferable are features in deep neural networks?, 2014.
[49] M. Neuhausen, P. Herbers, M. König, Synthetic data for evaluating the visual tracking of construction workers, in: Construction Research Congress 2020: Computer Applications - Selected Papers from the Construction Research Congress 2020, American Society of Civil Engineers (ASCE), 2020, pp. 354–361.
[50] Y. Hong, S. Park, H. Kim, Synthetic data generation for indoor scene understanding using BIM, in: Proceedings of the 37th International Symposium on Automation and Robotics in Construction (ISARC), International Association for Automation and Robotics in Construction (IAARC), 2020, available at:https://doi.org/10.22260/isarc2020/0048.
[51] K.M. Rashid, J. Louis, Times-series data augmentation and deep learning for construction equipment activity recognition, Adv. Eng. Inform. 42 (2019a) 100944.
[52] K.M. Rashid, J. Louis, Window-warping: A time series data augmentation of IMU data for construction equipment activity identification, in: International Symposium on Automation and Robotics in Construction (ISARC 2019), IAARC Publication, 2019b, pp. 651–657.
[53] X. Wu, L. Liang, Y. Shi, S. Fomel, FaultSeg3D: Using synthetic data sets to train an end-to-end convolutional neural network for 3D seismic fault segmentation, Geophysics 84 (3) (2019) IM35–IM45.
[54] J.W. Ma, T. Czerniawski, F. Leite, Semantic segmentation of point clouds of building interiors with deep learning: Augmenting training datasets with synthetic BIM-based point clouds, Autom. Constr. 113 (2020) 103144.
[55] C. Chokwitthaya, Y. Zhu, S. Mukhopadhyay, A. Jafari, Applying the Gaussian mixture model to generate large synthetic data from a small data set, in: Construction Research Congress 2020: Computer Applications - Selected Papers from the Construction Research Congress 2020, American Society of Civil Engineers (ASCE), 2020, pp. 1251–1260.
[56] G.E. Hinton, R.R. Salakhutdinov, Reducing the dimensionality of data with neural networks, Science 313 (5786) (2006) 504–507.
[57] M.A. Carreira-Perpinan, R. Raziperchikolaei, Hashing with binary autoencoders, in: CVPR 2015, 2015, pp. 557–566.

[58] B. Xia, C. Bao, Wiener filtering based speech enhancement with weighted denoising auto-encoder and noise classification, Speech Commun. 60 (2014) 13–29.
[59] R. Lan, Z. Li, Z. Liu, T. Gu, X. Luo, Hyperspectral image classification using k-sparse denoising autoencoder and spectral–restricted spatial characteristics, Appl. Soft Comput. 74 (2019) 693–708.
[60] A. Majumdar, Graph structured autoencoder, Neural Netw. 106 (2018) 271–280.
[61] H. Li, M. Gong, C. Wang, Q. Miao, Self-paced stacked denoising autoencoders based on differential evolution for change detection, Appl. Soft Comput. 71 (2018) 698–714.
[62] S.-X. Lv, L. Peng, L. Wang, Stacked autoencoder with echo-state regression for tourism demand forecasting using search query data, Appl. Soft Comput. 73 (2018) 119–133.
[63] M. Ghifary, W.Bastiaan. Kleijn, M. Zhang, D. Balduzzi, Domain generalization for object recognition with multi-task autoencoders, in: ICCV 2015, 2015, pp. 2551–2559.
[64] C. Zhou, J.G. Chase, G.W. Rodgers, Degradation evaluation of lateral story stiffness using HLA-based deep learning networks, Advanced Engineering Informatics 39 (2019) 259–268.
[65] L. Gondara, Medical image denoising using convolutional denoising autoencoders, in: 2016 IEEE 16th International Conference on Data Mining Workshops (ICDMW), IEEE, 2016, pp. 241–246.
[66] P. Vincent, H. Larochelle, Y. Bengio, P.-A. Manzagol, Extracting and composing robust features with denoising autoencoders, in: Proceedings of the 25th International Conference on Machine Learning - ICML '08, ACM Press, New York, New York, USA, 2008, pp. 1096–1103.
[67] C. Lu, Z.-Y. Wang, W.-L. Qin, J. Ma, Fault diagnosis of rotary machinery components using a stacked denoising autoencoder-based health state identification, Signal Process. 130 (2017) 377–388.
[68] J. Xu, L. Xiang, Q. Liu, H. Gilmore, J. Wu, J. Tang, A. Madabhushi, Stacked sparse autoencoder (SSAE) for nuclei detection on breast cancer histopathology images, IEEE Trans. Med. Imaging 35 (1) (2016) 119–130.
[69] Z. Chen, W. Li, Multisensor feature fusion for bearing fault diagnosis using sparse autoencoder and deep belief network, IEEE Trans. Instrum. Meas. 66 (7) (2017) 1693–1702.
[70] D.P. Kingma, M. Welling, Auto-Encoding Variational Bayes, 2013.
[71] C. Doersch, Tutorial on Variational Autoencoders, 2016.
[72] J. Walker, C. Doersch, A. Gupta, M. Hebert, An uncertain future: Forecasting from static images using variational autoencoders, in: European Conference on Computer Vision, Springer, Cham, 2016, pp. 835–851.
[73] Z. Yang, Z. Hu, R. Salakhutdinov, T. Berg-Kirkpatrick, Improved variational autoencoders for text modeling using dilated convolutions, in: Proceedings of the 34th International Conference on Machine Learning - Vol. 70, JMLR.org, 2017, pp. 3881–3890.
[74] T. Blaschke, M. Olivecrona, O. Engkvist, J. Bajorath, H. Chen, Application of generative autoencoder in De Novo molecular design, Mol. Inform. 37 (1–2) (2018) 1700123.
[75] S.M. Trost, G.D. Oberlender, Predicting accuracy of early cost estimates using factor analysis and multivariate regression, J. Constr. Eng. Manag. 129 (2) (2003) 198–204.
[76] L. Breiman, Random forests, Mach. Learn. 45 (1) (2001) 5–32.
[77] H. Drucker, C. Burges, L. Kaufman, A. Smola, V. Vapnik, Support vector regression machines, Adv. Neural Inf. Process. Syst. (1997) 155–161.
[78] A.J. Smola, B. Schölkopf, A tutorial on support vector regression, Stat. Comput. 14 (3) (2004) 199–222.
[79] S. Deng, T.-H. Yeh, Using least squares support vector machines for the airframe structures manufacturing cost estimation, Int. J. Prod. Econ. 131 (2) (2011) 701–708.
[80] G.K.F. Tso, K.K.W. Yau, Predicting electricity energy consumption: A comparison of regression analysis, decision tree and neural networks, Energy 32 (9) (2007) 1761–1768.
[81] M.D. Zeiler, ADADELTA: An Adaptive Learning Rate Method, 2012.
[82] D.P. Kingma, J.L. Ba, Adam: A method for stochastic optimization, in: ICLR 2015, 2015, pp. 1–15.
[83] I. Goodfellow, J. Pouget-Abadie, M. Mirza, B. Xu, D. Warde-Farley, S. Ozair, A. Courville, et al., Generative Adversarial Nets, 2014.