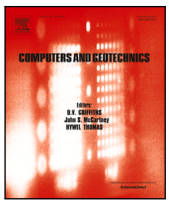
Computers and Geotechnics 152 (2022) 105049

Contents lists available at ScienceDirect

Computers and Geotechnics

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



Research paper

Data-driven strain–stress modelling of granular materials via temporal convolution neural network

Mengqi Wang [a], Tongming Qu [a], Shaoheng Guan [a,c], Tingting Zhao [b], Biao Liu [a,c], Y.T. Feng [a,*]

[a] Zienkiewicz Centre for Computational Engineering, Faculty of Science and Engineering, Swansea University, Swansea, Wales, SA1 8EP, UK
[b] College of Mechanical and Vehicle Engineering, Taiyuan University of Technology, Taiyuan, 030024, Shanxi, China
[c] School of Water Resources and Hydropower Engineering, Wuhan University, Wuhan, 430072, Hubei, China

ARTICLE INFO

Keywords:
Machine learning
Stress–strain relations
Temporal convolution neural (TCN) networks
Bayesian optimisation
Transfer learning
Discrete element modelling

ABSTRACT

Machine learning offers a new approach to predicting the path-dependent stress–strain response of granular materials. Recent studies show that temporal convolution neural (TCN) networks, a mutation of the 1D convolution neural network (CNN), have a powerful capability of addressing time-related prediction tasks. In this work, TCN networks are constructed to explore their potential in capturing the constitutive law of granular materials. To train and test the TCN network, three types of numerical experiments are implemented to generate datasets via discrete element modelling. The Bayesian optimisation method is employed to find the optimum architecture of the network. Furthermore, to improve the training accuracy and efficiency, a transfer learning (TL) scheme is innovatively leveraged, which utilises the trained network parameters from a set of shorter time steps and/or coarser data points of the training strain–stress loading curves, as the initial values, to train the network for a longer time step. The prediction ability of the trained TCN network is assessed and compared with a recurrent neural network which has been proved to perform well in predicting constitutive laws of the granular materials. In addition, training datasets with artificially added noise are also used to test and analyse the robustness of TCN networks.

1. Introduction

Granular materials, consisting of diverse sizes, shapes, and porosity of grains, present complex constitutive characteristics. When subjected to external loads, the granular media tends to form inhomogeneous and discontinuous force chain networks to transfer the internal and external forces and thus exhibits remarkable anisotropy (Qu et al., 2019b; Anandarajah, 2008; Ueda and Iai, 2019) and strain localisation (Qu et al., 2019a; Desrues and Andò, 2015; Bréchet and Louchet, 1988) in the macroscopic scale. Additionally, the stress–strain response of the granular media is also state and rate-dependent (Das and Das, 2019; Alipour and Lashkari, 2018) and highly depends on the path of deformation. All these complicated physical and mechanical features make it very challenging to predict the elasto-plastic response of granular materials.

Many traditional methods have been leveraged to explore and describe particle behaviours. Physical experiments (Wang et al., 2018; Ezzein and Bathurst, 2011; Wei and Yang, 2014) have been widely used to capture the constitutive characters of granular materials but can only provide a qualitative analysis rather than a quantitative description. Theoretical and phenomenological constitutive models are also employed to formulate the stress–strain relations of particle assemblies (Zhu et al., 2010; He et al., 2019; Yang et al., 2020b) but they are limited by involving not only sophisticated mathematical formulations but also numerous free parameters (Zhang and Yin, 2021), and cannot precisely predict the large deformation of granular material (Qu et al., 2021c). As a complement to the theoretical method, numerical methods, such as hierarchical multiscale analysis (Liu et al., 2016; Jeong et al., 2018; Guo and Zhao, 2014), are introduced (Liu et al., 2016; Jeong et al., 2018) to address some of the problems mentioned, but still confront the problem of prohibitive computational costs. Therefore, there are still many obstacles that have prevented the generation of universal models to characterise the constitutive features of granular materials.

As a potential methodology to resolve the above challenge, artificial neural networks (ANNs) have been applied to represent constitutive laws of particle materials since the 1990s (Ghaboussi and Sidarta, 1998). Resorting to multi-layer affine transformation and nonlinear squashing functions, the ANN is regarded as a data-driven surrogate model to approximate any complex mapping relation. Recently, many reliable deep neural networks (DNNs) for time-sequential problems

* Corresponding author.
E-mail address: y.feng@swansea.ac.uk (Y.T. Feng).

https://doi.org/10.1016/j.compgeo.2022.105049
Received 17 March 2022; Received in revised form 21 September 2022; Accepted 22 September 2022
Available online 5 October 2022
0266-352X/© 2022 The Author(s). Published by Elsevier Ltd. This is an open access article under the CC BY-NC-ND license (http://creativecommons.org/licenses/by-nc-nd/4.0/).

have been exploited because of the advancement of computational science (Ali et al., 2019; Fernández et al., 2020). Particularly, recurrent neural networks (RNNs) (Qu et al., 2021a,b; Ma et al., 2022) are generally applied in describing constitutive laws for granular materials and can acquire good prediction results (Gorji et al., 2020; Mozaffar et al., 2019). The trained model has also been adopted to simulate macroscopic problems within the coupled finite element and machine learning modelling framework (Shaoheng Guan et al., 2022).

More recently, a variant of convolution neural networks (CNNs) (Wang et al., 2022; Abueidda et al., 2019; Yang et al., 2020a; Gu et al., 2018), termed the temporal convolutional neural (TCN) network, has also been adopted to address time sequence prediction problems with a more flexible architecture. The TCN network is firstly proposed in computer graphics to detect the action segment from video recordings (Lea et al., 2017) and subsequently employed in other sequential tasks, such as word prediction, digital recognition (Bai et al., 2018), weather and wind speed prediction (Hewage et al., 2020; Fukuoka et al., 2018), and energy-related time series forecasting (Lara-Benítez et al., 2020). Compared with CNNs, the TCN network can be fed with time sequence data directly rather than image information and therefore can be trained with variable-length inputs, and at the same time keeps all advantages of CNNs, such as parallelism and stability (Bai et al., 2018).

In solid mechanics, both RNNs and TCN networks have demonstrated excellent performance on time series problems involving path-dependent plasticity and thermo-viscoplasticity (Abueidda et al., 2021), but the TCN network has higher computational efficiency on GPU than RNNs (Abueidda et al., 2021). However, to the best of our knowledge, there is currently very limited work on exploring the potential of the TCN network to predict constitutive relations of granular materials.

In this study, the capability of the TCN network in capturing the stress–strain response of granular materials is investigated and compared with one of the RNNs, the Gated Recurrent Unit (GRU) network (Qu et al., 2021a), which is recognised as a powerful network for the problem concerned. Three types of numerical experiments are performed via discrete element modelling (DEM) to generate the dataset used to train and test the TCN network. The first type is triaxial compression experiments including the conventional triaxial compression (CTC), and the true triaxial compression (TTC) testing based on the same granular sample and confining stress. The second type is the TTC experiment but with different granular samples, intermediate principal stress coefficients (e.g. b value), and initial stress states. The final one is triaxial experiments with random strain loading paths.

Different from 1D-CNN, both dilated casual layers and the residual layer are introduced to the TCN architecture. To fully explore the capability of the TCN network, the difference between the TCN network and the 1D-CNN network is first investigated, and then the Bayesian optimisation method is leveraged to optimise the hyper-parameter combination of the TCN network in the certain hyper-parameter space identified by experiments.

Furthermore, a transfer learning scheme is proposed to accelerate the training stage and improve the training accuracy of the TCN network. In this scheme, the trained network parameters from a set of shorter time steps and/or coarser datapoints of the training strain–stress loading curves are used as the initial values to train the network for a longer time step. The effectiveness of this transfer learning scheme is validated by performing three comparison experiments with or without transfer learning.

The trained TCN network is tested by loading cases that are never involved in the training stage. To further demonstrate the prediction capability of the TCN network, the test result of each case is compared with the prediction of a GRU network trained on the same dataset. By recognising the fact that training datasets may often contain errors from different sources, the robustness of TCN networks is also investigated with the dataset contaminated by artificially generated noise data.

The remaining of the paper is structured as follows: Section 2 introduces the architecture of the TCN network and the general training strategy to be used. Section 3 describes the three types of numerical experiments based on discrete element modelling for training data generation. Section 4 provides a detailed description of the training procedure used, particularly the transfer learning scheme proposed. Some representative prediction results of the TCN and GRU networks under different loading conditions are presented and compared in Section 5. The robustness of the TCN network is also demonstrated by the training dataset that is contaminated by the artificially added noise. Concluding remarks are made in Section 6.

2. Methodology

In this section, a brief introduction to deep learning (DL) theory is first provided, then the general training strategy using the Bayesian optimisation and the transfer learning scheme is introduced for obtaining a well-trained TCN network with higher efficiency.

2.1. The basic principle of using deep learning to train stress–strain relations

Under the quasi-static condition, the constitutive law of granular materials can be presented as follows:

$$\boldsymbol{\sigma} = f^{DL}(\boldsymbol{\epsilon}) \tag{1}$$

where $\boldsymbol{\sigma}$ and $\boldsymbol{\epsilon}$ are stress and strain tensors, respectively; f^{DL} denotes the relation between the $\boldsymbol{\sigma}$ and $\boldsymbol{\epsilon}$. However, when subjected to complicated loading paths, such as recycling loading, the path-dependent stress–strain response of granular materials will no longer be a one-to-one mapping, and history variables should be taken into account. Consequently, the constitutive relationship of granular materials can be modified as follows:

$$\boldsymbol{\sigma} = f^{DL}(\boldsymbol{\epsilon}, \boldsymbol{\alpha}) \tag{2}$$

where $\boldsymbol{\alpha}$ represents a set of history variables. Including these historical states significantly increase the complexity of establishing the stress–strain relation of granular materials with traditional methods.

According to the theorem of universal approximation, the deep neural network, with the combination of linear matrix multiplications and nonlinear activation functions, can approximate any continuous function to any precision (Hornik et al., 1989). The relationship between the input and output of one neural network layer can be generally expressed as:

$$\begin{cases} \boldsymbol{a}^{(l+1)} = \boldsymbol{W}^{(l+1)} f_l(\boldsymbol{z}^{(l)}) + \boldsymbol{b}^{(l+1)} \\ \boldsymbol{z}^{(l)} = \boldsymbol{a}^{(l)} \end{cases} \quad l = (0, 1, \ldots, Q) \tag{3}$$

where l represents the l^{th} layer in the neural network; Q refers to the number of hidden layers; $l = 0$ and $l = Q$ denote the input and output layer in the neural network, respectively; $\boldsymbol{W}$ and $\boldsymbol{b}$ represent the weight and bias matrices; f_l denotes the nonlinear squashing or activation function; and $\boldsymbol{a}$ is the activation of the activation function; $\boldsymbol{z}$ is the output of one neuron in the neural network layer.

Constructing an AI-based constitutive model for granular materials is to train a neural network to approximate the stress–strain relation based on a (large) set of strain-stress data. The whole process can be formulated as:

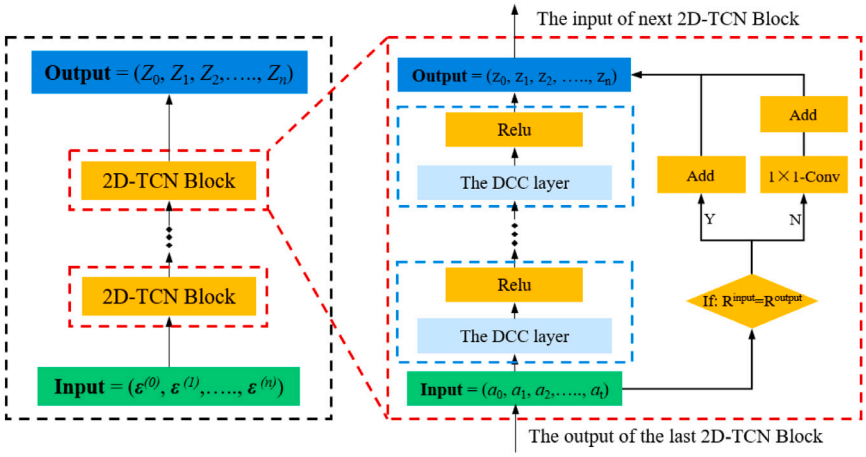
$$\hat{\boldsymbol{\sigma}}(\boldsymbol{\epsilon}, \boldsymbol{W}^{(K+1)}, \boldsymbol{b}^{(K+1)}) = f_{K+1}(\boldsymbol{a}^{(K+1)}) \tag{4}$$

where $\hat{\boldsymbol{\sigma}}$ represents the predicted stress via the neural network, and Eq. (3) is used to obtain $\boldsymbol{a}^{(l+1)}$, $\boldsymbol{W}^{(K+1)}$, $\boldsymbol{b}^{(K+1)}$ when $l = Q$ with $\boldsymbol{a}^{(0)} = \boldsymbol{\epsilon}$. In the train stage, a loss function $L(\hat{\boldsymbol{\sigma}}, \boldsymbol{\sigma})$ is defined to update weight and bias metrics of every layer until the error between the output of the neural network $\hat{\boldsymbol{\sigma}}$ and the actual stress $\boldsymbol{\sigma}$ reduces to an acceptable level.

The history dependency of the stress–strain curve is accounted for by using a discrete strain–stress data sequence as the input for the network training and prediction:

$$\hat{\boldsymbol{\sigma}^m} = \hat{\boldsymbol{\sigma}}(\{\boldsymbol{\epsilon}_{m-n+1}, \ldots, \boldsymbol{\epsilon}_{m-2}, \boldsymbol{\epsilon}_{m-1}, \boldsymbol{\epsilon}_m\}) \tag{5}$$

where n is called the time step.



(a) The architecture of the TC part (b) One 2D-TCN block in the TCN network

Fig. 1. Temporal convolution part of a TCN network.

2.2. The architecture of the temporary convolution network

The temporary convolution neural network is a mutation of the CNN and is composed of two parts: the temporary convolution (FC) part, and the fully-connected (FC) part. As shown in Fig. 1, the FC part consists of several stacked 2D-TCN blocks, while each block comprises of many dilated casual convolution (DCC) layers. The detailed description of the DCC layer can be found in Appendix A.

For an objective function $\boldsymbol{h}(\boldsymbol{a})$ with input argument $\boldsymbol{a}$, it can be equivalently expressed as:

$$\boldsymbol{h}(\boldsymbol{a}) = \boldsymbol{a} + \boldsymbol{f}(\boldsymbol{a}) \tag{6}$$

where $\boldsymbol{a}$ is called the identity function, and $\boldsymbol{f}(\boldsymbol{a}) = \boldsymbol{h}(\boldsymbol{a}) - \boldsymbol{a}$ is defined as the residual function. It has been demonstrated (Cybenko, 1989) that approximating the residual function $\boldsymbol{f}(\boldsymbol{a})$ may be more effective than directly approximating $\boldsymbol{h}(\boldsymbol{a})$ via neural networks. Consequently, as shown in Fig. 1(b), a residual layer, which is a branch combining the input data and the output of the last DCC layer in one 2D-TCN block, is included in each 2D-TCN block. However, unlike the standard ResNet whose input and output have an identical dimension and can be added directly, a 1×1 convolution operation has to be adopted to offset the shape difference between the input and output of the TCN block. Further details about the 2D-TCN blocks used can be found in Appendix A.

In the TC part, as illustrated in Fig. 2, the number of 2D-TCN blocks is set to be M. The input strain sequent data with a time step of n is processed by the first TCN block, and the output of each TCN block is used as the input of the next 2D-TCN block. The output of the final M^{th} 2D-TCN block with depth K has the shape of a 2D array ($n \times K$), where K represents the number of filters of the last DCC layer of the M^{th} TCN block.

After obtaining the outputs of the TC part, they are stretched into a 1D vector that can be accepted by the FC part via the flatten layer and then fed into the FC part. Eventually, the TCN network acquires the prediction of the three principal stresses corresponding to the nth principal strain of the input data.

2.3. Network training strategy

As the architecture of the TCN network is determined by the TCN blocks, DCC layers, filters, and kernel size, there are many hyper-parameters involved that influence the prediction ability of the TCN network. To fully explore the potential of the TCN network, the optimal combination of the hyperparameters should be searched. In this study, the Bayesian optimisation method is employed to obtain the optimal hyperparameters. The detail of the optimisation will be given in Section 4.1.

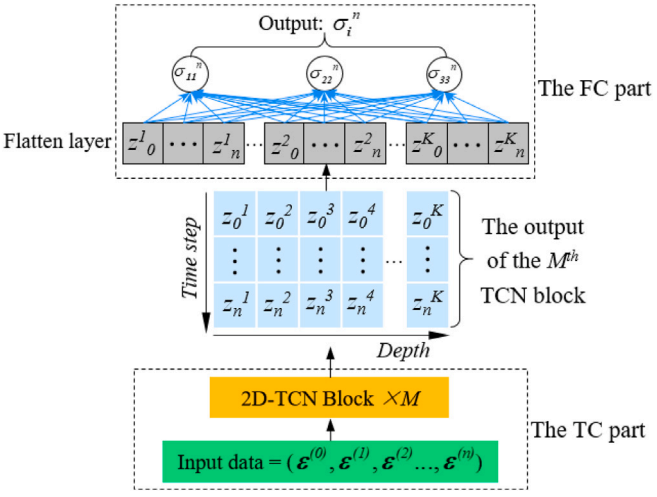


Fig. 2. The architecture of the TCN network.

In addition, the training of a time-sequential neural network, such as RNN and TCN, can be time-consuming. To reduce the training costs, and also improve the training accuracy, the transfer learning scheme mentioned in the introduction will be used. The detail of this scheme will be described in Section 4.2.

2.4. Evaluation of the trained data-driven models

To evaluate the prediction accuracy of a deep learning network, the mean absolute error, MAE, is always used as the loss function during the training and computed as:

$$MAE = \frac{1}{M}\frac{1}{N}\sum_{m=1}^{M}\sum_{n=1}^{N}\left|\hat{\sigma}_{ii} - \sigma_{ii}\right| \tag{7}$$

where M and N represent the number of stress–strain curves in the dataset and the data points of each curve, respectively; m and n refer to the n^{th} point on the m^{th} stress–strain curve; and $\hat{\sigma}_{ii}$ and σ_{ii} denote the actual and prediction main principal stresses at the n^{th} data point of the m^{th} stress–strain curve.

In addition to the MAE, many other metrics, such as Euler angles and Frobenius norm (Heider et al., 2020), can also be used to estimate

the performance of the neural network. However, these metrics may not provide an explicit indication of how well the prediction for a whole stress–strain loading path is. Therefore we adopt a score A_{score}, first introduced in Wang and Sun (2019) and also used in Qu et al. (2021a), as an alternative index to describe the prediction results over test datasets directly.

Before defining A_{score}, the scaled squared error (SSE) of every single point i on the j^{th} stress–strain curve is calculated as:

$$\mathrm{SSE} = (\hat{y}_{ij} - y_{ij})^2 = (\hat{y}_{ij} - y_{ij})^2 \tag{8}$$

where $\hat{y}_{ij}$ and y_{ij} are the scaled prediction value and the ground truth at the i^{th} point of the j^{th} stress–strain curve, respectively. After calculating all SSE values of the j^{th} stress–strain curve, they will be ranked in ascending order, and then the empirical cumulative distribution function (eCDF) F_j of the jth stress–strain curve can be defined as:

$$F_j(\mathrm{SSE}) = \frac{r}{N_j}(r = 1, 2, \ldots, N_j) \tag{9}$$

where N_j is the total number of data points in the j^{th} stress–strain curve.

Based on the obtained SSE values and the F_j of the j^{th} stress–strain curve, A_{score} can be defined as:

$$A_{score} = max\left(\frac{log\left[max(\varepsilon_{p\%}, \varepsilon_{crit})\right]}{log(\varepsilon_{crit})}, 0\right) \tag{10}$$

where $\varepsilon_{p\%}$ refers to the SSE value when r/N_j is designated to be $p\%$ in the F_j; ε_{crit} is regarded as the critical accuracy to judge if a prediction is good enough to obtain a score of one. In this work, 90% is selected as $p\%$ and ε_{crit} is set to be 0.001.

3. Data preparation for the data-driven constitutive model

In this section, three types of numerical experiments are performed using discrete element modelling of representative volume elements (RVE) of particle systems to generate training and test datasets for TCN networks. The first type includes two triaxial compression experiments under the same particle size distribution (PSD) and initial stress state: the conventional triaxial compression (CTC) and true triaxial compression (TTC). The second type is also a TTC experiment but with particle systems which have different PSDs, initial stress states and b values. The third type involves experiments with random strain loading paths.

3.1. CTC and TTC compression experiments

In the CTC testing, multiple loading–unloading–reloading conditions will be considered, while the TTC testing will incorporate both the isobaric axisymmetric triaxial loading, i.e., constant-p (CP), where $p = -(\sigma_{11}+\sigma_{22}+\sigma_{33})/3$, and the intermediate principal stress coefficient tri-axial loading, i.e., constant-b (CB), where $b = (\sigma_{22} - \sigma_{33})/(\sigma_{11} - \sigma_{33})$.

All the experimental data is obtained from the discrete element modelling of a granular system, also used in Qu et al. (2021a). In the granular system, as shown in Fig. 3, a total of 4037 spherical particles, whose radii are uniformly allocated between 2 mm and 4 mm and with a density of 2600 kg/m^3, are used to generate the dataset. Both the inter-particle frictional coefficient and the local contact damping ratio are set to be 0.5. The initial particle packing is consolidated to a hydrostatic confining stress of 200 kPa. The maximum axial strain is limited to 12% for all numerical experiments.

Based on the observation that the reloading strain usually has a smaller value than its preceding unloading strain, a set of unloading–reloading points is employed to generate stress–strain curves for CTC and CP conditions. Only the monotonic loading path is considered in the CB condition, and different b values, increasing from 0 to 1.0 at an interval of 0.05, are used to generate CB samples.

To provide an overview of the distribution of the specimens, all one-cycle loading paths of CTC and CP specimens are plotted in the lower right part of Fig. 4. Meanwhile, partial CTC test specimens with multi-cycle loading paths are presented in the radar map in the upper left part of Fig. 4 via the combination of scatter points and different colours of dashed lines, where the values in the radial and circular axes represent the unloading and reloading strain, respectively.

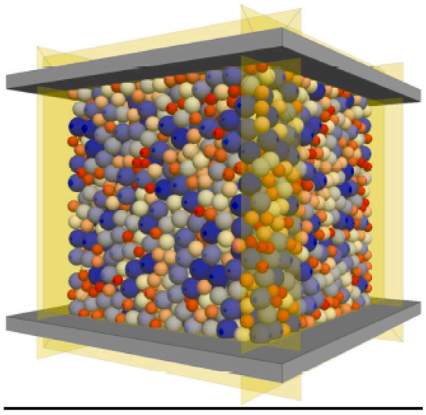


Fig. 3. The granular system used to generate the data for CTC and TTC experiments.

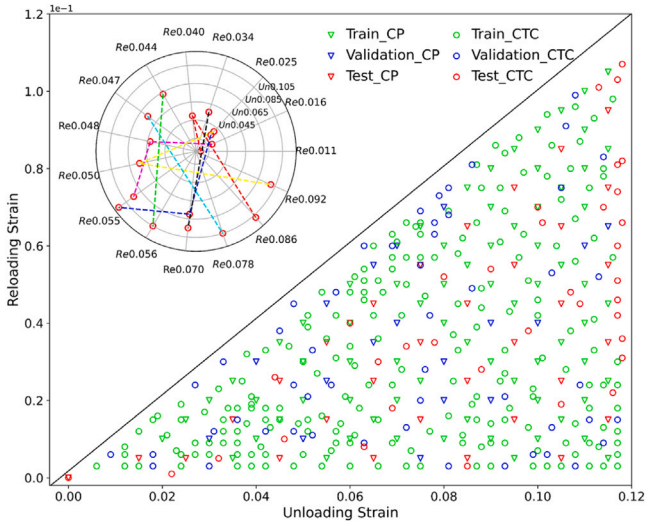


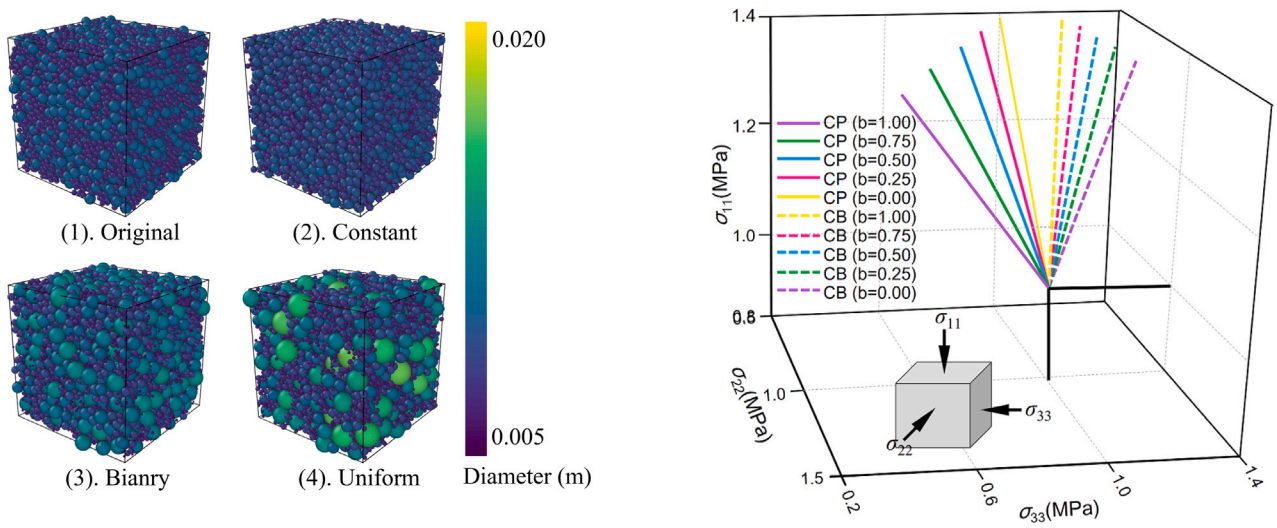
Fig. 4. The distribution of train, validation, and test samples under CTC and CP loading conditions.

Eventually, 297 CTC tests and 143 TTC tests are conducted. To test the prediction ability of TCN networks, 80 samples, including CTC, CP, and CB experiments, are extracted from all samples in advance, and the remaining samples are used to train and validate the TCN network.

3.2. The TTC experiments with different granular samples and loading paths

Considering the diversity of microstructure in grain media, the RVEs with different PSDs are generated, and the TTC testing, including CB and CP loading, is conducted again under different initial stress states via discrete element modelling.

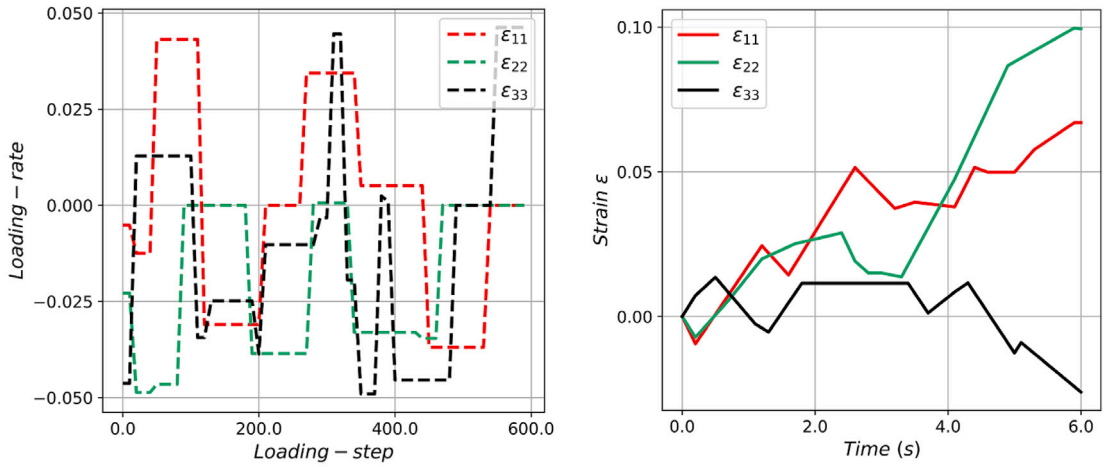
Before constructing the particle samples, the fractal particle size distribution, which is defined as $F(d) = (d^{3-\beta} - d_{min}^{3-\beta})/(d_{max}^{3-\beta} - d_{min}^{3-\beta})$, is used to guarantee that the samples with different PSDs comprise comparable particle numbers by adjusting the minimum diameter d_{min} and maximum diameter d_{max} of one sample, where β is the fractal distribution. In this section, as shown in Fig. 5(a), four RVEs, e.g., the Original, Constant, Binary, and Uniform, with different PSDs are adopted. Corresponding to each RVE, the detailed parameters,



(a) Four different granular samples.

(b) The stress loading paths when the p value or σ_{33}=1MPa.

Fig. 5. Different granular samples and loading paths.



(a) The loading rate of each loading step.

(b) The strain of each loading step

Fig. 6. The random strain loading path.

Table 1
The granular parameters of the different samples.

PSD	Particle size range (m)	Mean particle diameter (m)	Particle number
Original	0.006–0.0175	0.0125	10 000
Constant	0.0107–0.0118	0.0112	
Binary	0.009–0.018	0.0180	
Uniform	0.0022–0.018	0.0151	

including particle size range, mean particle diameter, etc., are listed in Table 1.

In addition to considering different PSDs, the constitutive behaviour of the granular material under different loading paths is also simulated using discrete element modelling. Different from the TTC testing of Section 3.1, where the p values of the CP cases and b values of the CB cases are kept constant, both the p and b values of the TTC cases are set to different values in this section. The specified loading paths can be found in Table 2 and Fig. 5(b). Especially, when $b = 0$ or 1, the particle sample is subjected to triaxial compression and triaxial extension states, respectively.

Eventually, a total of 450 loading cases are performed in the DEM modelling, where 90 TTC loading cases with different PSDs, p values, and b values are selected as the test set.

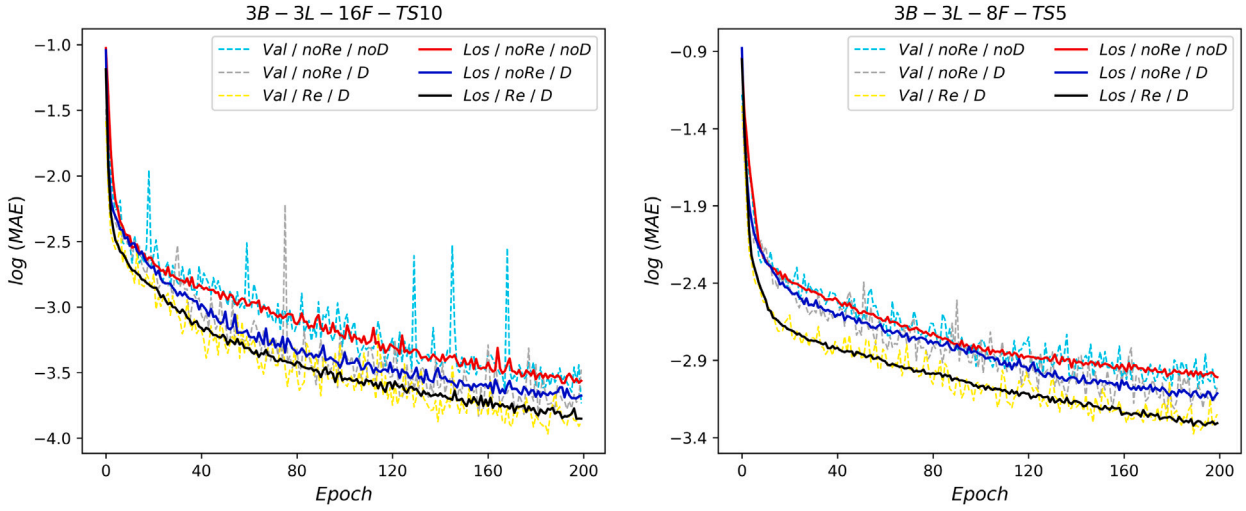
3.3. Random strain loading experiments

To further explore the potential of the data-driven model, training cases with random strain loading paths are generated to validate the prediction capability of the TCN network in this work Each random loading path is realised by randomising the loading step and loading rate. In one loading case, as shown in Fig. 6(a), the maximum loading rate and step are limited to 0.05 and 600 (100 loading steps in one second), respectively. Before calculating the strain path, the loading steps are randomly separated into different segments, and the loading rate used in each segment is randomly selected and kept constant. After determining the loading rate of each loading step, the strain path at each loading step can be obtained by integrating the production of the loading rate and step, as demonstrated in Fig. 6(b). Then, the random strain paths are used to generate different loading cases (see Fig. 6).

In this work, 300 random strain loading cases are performed in the DEM modelling, where 284 cases are used to train the TCN network, and the rest 16 cases are for the test set.

Table 2
The loading paths for TTC testing.

Loading path	Controlled variable 1	Controlled variable 2
CP with different p, b	p = 1.0, 2.0, 4.0 (MPa)	b = 0.00, 0.25, 0.50, 0.75, 1.0
CB with different p, b	σ_{33} = 1.0, 2.0, 4.0 (MPa)	b = 0.00, 0.25, 0.50, 0.75, 1.0



(a) The training results of experiment 1.

(b) The training results of experiment2.

Fig. 7. The training results of two experiments.

4. The training process for the data-driven model

In this section, both the TCN and GRU networks are trained with the generated experimental data in Section 3.1. Before training the TCN network, the hyperparameters are first determined by the Bayesian method, and then the transfer learning scheme is employed in the training stage of the TCN network. For the GRU network, the hyperparameters used in Qu et al. (2021a) are adopted in this work.

4.1. The determination of the hyperparameters in the TCN network

4.1.1. The residual structure and dilation factor

Different from 1D-CNN (Vlassis and Sun, 2021), the dilation factor and residual structure are introduced to the TCN network as the architecture hyperparameters. Consequently, it is worth investigating the function of the dilation factor and the residual connection in the TCN network, before determinating other hyperparameters with the Bayesian optimisation method.

For this purpose, two experiments are executed. In each experiment, three TCN networks consisting of the same number of TCN blocks, DCC layers, filters, and kernel size are trained with or without the dilation factor and residual architecture. The training results of two experiments are plotted in Fig. 7, where B, L, F, and TS refer to the TCN block, DCC layer, filter, and time step, respectively; the Val, Los, Re, and D denote the validation loss, loss, residual architecture, and dilation factor, respectively. More detail about the dilation factor can be found in Fig. 34(d) of Appendix A.

Fig. 7 shows that the TCN network with the residual architecture and dilation factor obtains a lower MAE in a certain epoch. Comparing Fig. 7(a) and (b) indicates that the added residual architecture and dilation factor can also improve the training accuracy of the TCN network and accelerate the convergence of the training MAE even though the TCN network has a different hyperparameter combination.

Table 3
The ranges of hyperparameters of the TCN network.

Hyperparameter	Range
TCN blocks	(2, 3)
DCC layers in one TCN block	(3, 4)
Filters	(16, 64)
Kernel size	(2, 3, 4)

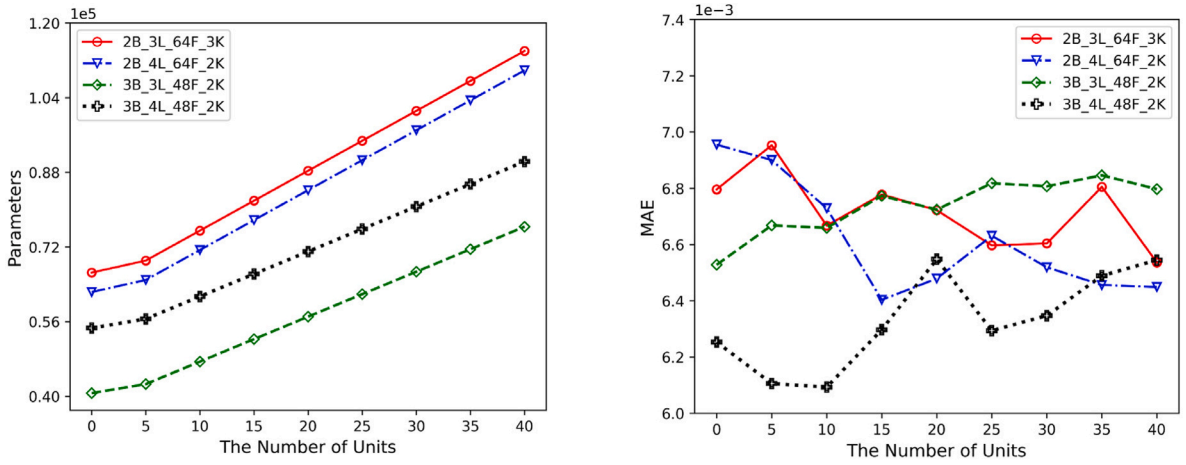
4.1.2. Bayesian optimisation method

The prediction of the TCN network is a high dimensional function of hyperparameters (including the numbers of TCN blocks, DCC layers, filters, the kernel size of each filter, etc.) that decide the architecture of the TCN network. To maximise the potential of the TCN network, Bayesian optimisation, a sequential design strategy for global optimisation of black-box functions, is introduced to search for the optimum solution (i.e., the best prediction precision of the TCN network) of a high dimensional function.

After having validated the effect of the residual architecture and dilation factor on the performance of the TCN network, the other hyperparameters are determined by the Bayesian optimisation experiment to be described below. However, the parameter space (i.e., the range of each hyperparameter) should be first determined to improve the computational efficiency. To this end, a total of 11 comparison experiments are carried out to determine the range of each key hyperparameter. The specific experiment parameters and the training results of each experiment are attached in Appendix B. Based on the experiment results and considering the balance between the computational efficiency and the training accuracy of TCN networks, the range of each hyperparameter is determined and listed in Table 3.

After having determined the ranges of the hyperparameters, it is still tedious to find what is the most suitable combination of these hyperparameters to form the TCN network. To address this problem, the Bayesian optimisation method is introduced.

Based on the ranges of TCN blocks and DCC layers in Table 3, four Bayesian search experiments are conducted. In each Bayesian experiment, the number of filters increases from 16 to 64 with a gap



(a) The number of parameters in the dense layer.

(b) The MAEs with different units in the dense layer

Fig. 8. The influence of the number of units in the dense layer on the training result of the TCN network.

Table 4
Best 4 results obtained via Bayesian optimisation method.

Architecture	Parameters	Filters	Kernel size	Learning rate	Loss value	Batch size	Time (s/epoch)
2B-3L	66 499	64	3	0.0001	0.006796	64	10
2B-4L	62 339	64	2	0.0001	0.006955	64	10
3B-3L	40 659	48	2	0.001	0.006528	256	8
3B-4L	52 627	48	2	0.001	0.006253	256	15

of 16; the kernel size changes in the range of two to four. In addition to the hyperparameters related to the architecture of the TCN network, other hyperparameters including learning rate and batch size are also investigated. The parameter spaces of learning rate and batch size are set to be (0.1, 0.01, 0.001. 0.0001) and (64, 128, 256) in four Bayesian search experiments, respectively.

In each Bayesian search, 100 different combinations of hyperparameters are explored and the combinations with the top five minimum mean absolute errors (MAEs) are selected, which are listed in Tables 10 to 13 in Appendix C. Subsequently, one best combination is chosen from each Bayesian search experiment. The principle is that if the difference of their MAEs of two combinations is less than 1e-3, then the combination with fewer parameters is adopted. Thus, four TCN networks are chosen. Finally, these four TCN networks are trained again with longer epochs. The training results are listed in Table 4.

Based on Table 4, by considering the balance between the precision of the TCN network and the computational cost, the third architecture (3B-3L) which has the following hyperparameters: three TCN blocks, three DCC layers in each TCN block, 48 filters in each DCC layer with the kernel size of two, the learning rate of 0.001 and batch size of 256, is chosen.

4.1.3. The hyperparameters of the FC part in the TCN network

All the experiments proceeded above with the Bayesian optimisation method have only one fully-connected layer (i.e., the output layer). Now a dense layer is inserted between the flatten layer and the output layer in the four networks listed in Table 4. The optimal number of units in the dense layer is determined through the trial-and-error method at an interval of five.

Fig. 8 shows that with the increase of the units, no apparent improvement is observed in the final performance of the TCN network. Consequently, the TCN network only uses the output layer as the fully-connected layer.

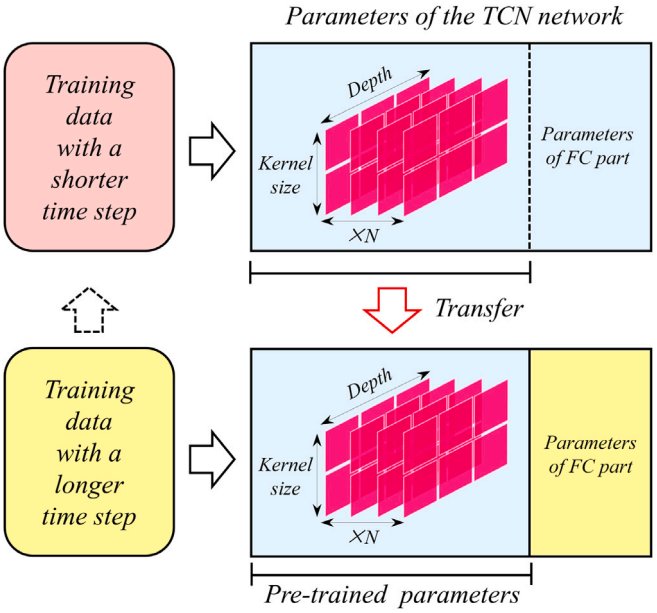
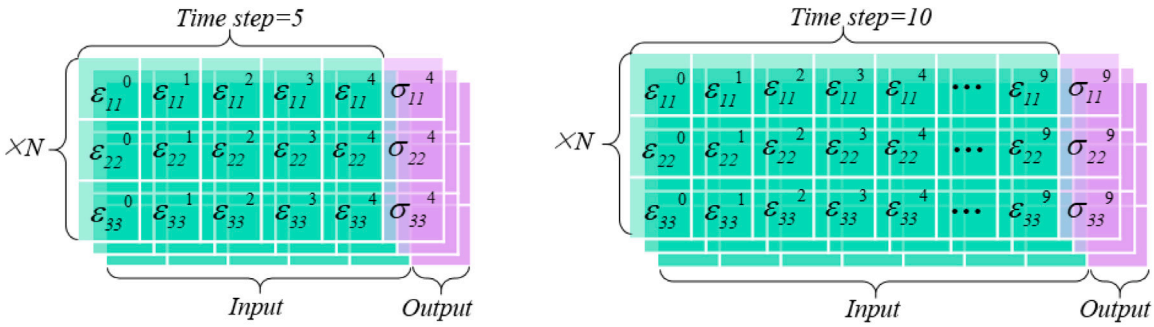


Fig. 9. The transfer learning process in the TCN network.

4.2. The transfer learning

To further improve the training efficiency and accuracy of the TCN network, transfer learning, a strategy that applies the experience gained while addressing one problem to different but related problems, is leveraged in this work. Based on this conception, the following transfer learning scheme, also illustrated in Fig. 9, is adopted.

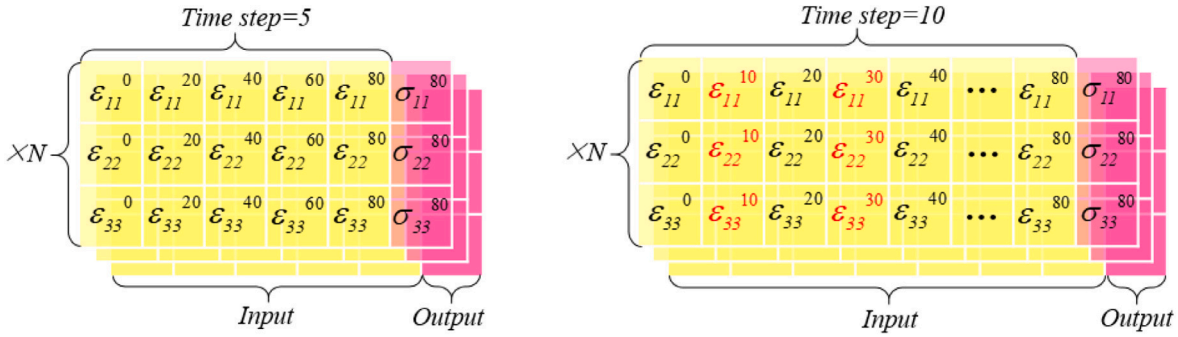
A TCN network, which includes N 2D filter blocks, is first trained with a (small) time step, and the trained weights and biases of the TC part of the TCN network are stored in each filter block. Subsequently,



(a) The training data with the time step of 5.

(b) The training data with the time step of 10.

Fig. 10. The temporal appending method.



(a) The training data with the time step of 5.

(b) The training data with the time step of 10.

Fig. 11. The resolution refinement method.

Table 5
Training result of four methods with the increasing of time-step.

No.	Time step = 5	Time step = 10	Time step = 20	Time step = 40	Time step = 80
1	0.0103	0.0086	0.0067	0.0053	0.0051
2	0.0103	0.0075	0.0057	0.0046	0.0044
3	0.0130	0.0086	0.0061	0.0047	0.0048

the trained N filter blocks are transferred to a new TCN network, and the newly-built TCN network is trained again using the training data with a larger time step. Then, the trained network with the current time step is regarded as the one with a smaller time step and thus can be used to initialise the parameters for the next network with a larger time step. The whole process is repeated until a TCN network trained with a designated time step is obtained. This method is feasible as only the time step of the training data is altered, while the shape of each filter (e.g. the kernel size) is kept constant.

4.2.1. The data generation method

In the transfer learning scheme, two different approaches are used to group the input data series from the training data required with different time steps. One is called the *temporal appending* (TA), and the other is the *resolution refinement* (RR). In the TA method, which is shown in Fig. 10(a), a total of N stress–strain pairs are first grouped as the input at a time step of five. Then, to obtain the input training data for a longer time step, the time series is simply extended to 10 (see Fig. 10(b)), and N new input data series are yielded. By repeating this process, the training data with the final time step of 80 can be obtained.

While for the RR approach, as illustrated in Fig. 11(a), the principal strains at time instances 0, 20, 40, 60, and 80 with a gap or resolution of 20 are first selected to form the input training data series for the time step of five. Then, the principal strains at extra time instances at a halved resolution of 10 are also added to form the input data for the time step of 10, as shown in Fig. 11(b). By further reducing the resolution, the input training series for the final time step of 80 can be obtained.

4.2.2. The training results with the transfer learning scheme

Using the training data generated in Section 3.1 and the two approaches to (re-)group the input data series, three experiments are designed to explore the effect of the transfer learning scheme proposed, and five TCN networks are trained in each experiment.

In the first experiment, the TCN networks are trained with different time steps (e.g. 5, 10, 20, 40, and 80) directly without invoking the transfer learning scheme. The transfer learning is applied in the second and third experiments where the input data series are formed by the temporal adding and resolution refinement approaches respectively. The training results of the three experiments are plotted in Fig. 12 and also listed in Table 5.

It is clear from Table 5 that transfer learning can enhance the training precision using identical training data. Besides, although the performance of the third experiment is not well when the time step is less than 20, it still acquires a similar training precision to the second experiment when the time step increases to 40.

Compared with training TCN networks with a regular method, as demonstrated in Fig. 12(a), it is found that the transfer learning scheme can remarkably accelerate the convergence rate in the initial stage of the training process. Furthermore, from the training results of the second and third experiments, as shown in Fig. 12(b) and (c), the training efficiency has been significantly improved when training the TCN

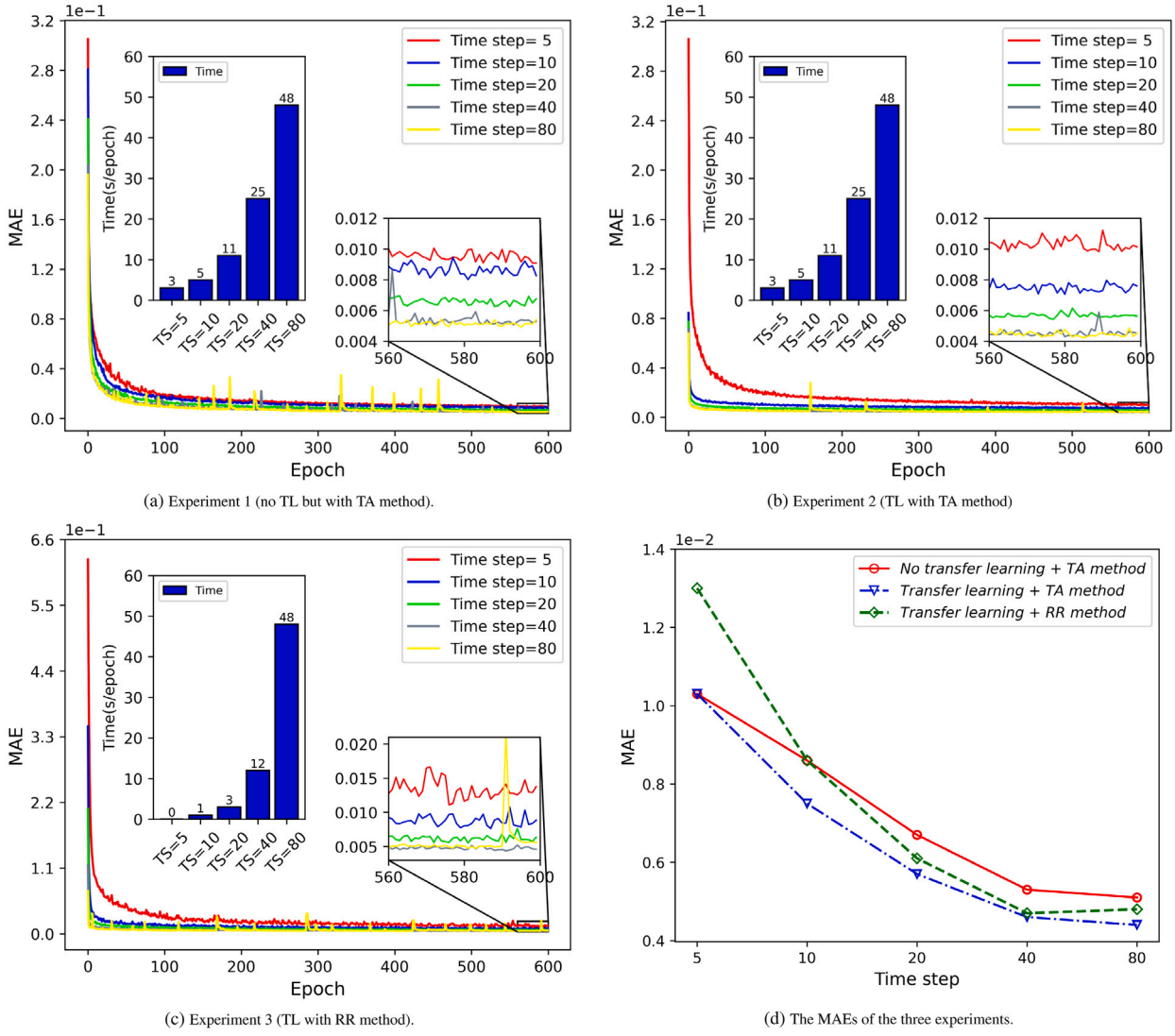


Fig. 12. The training results of three different experiments.

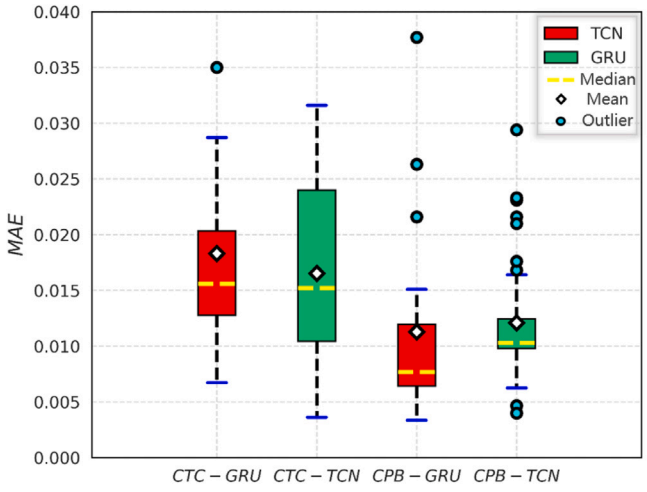


Fig. 13. The MAE distribution of the 80 test samples with the same PSD.

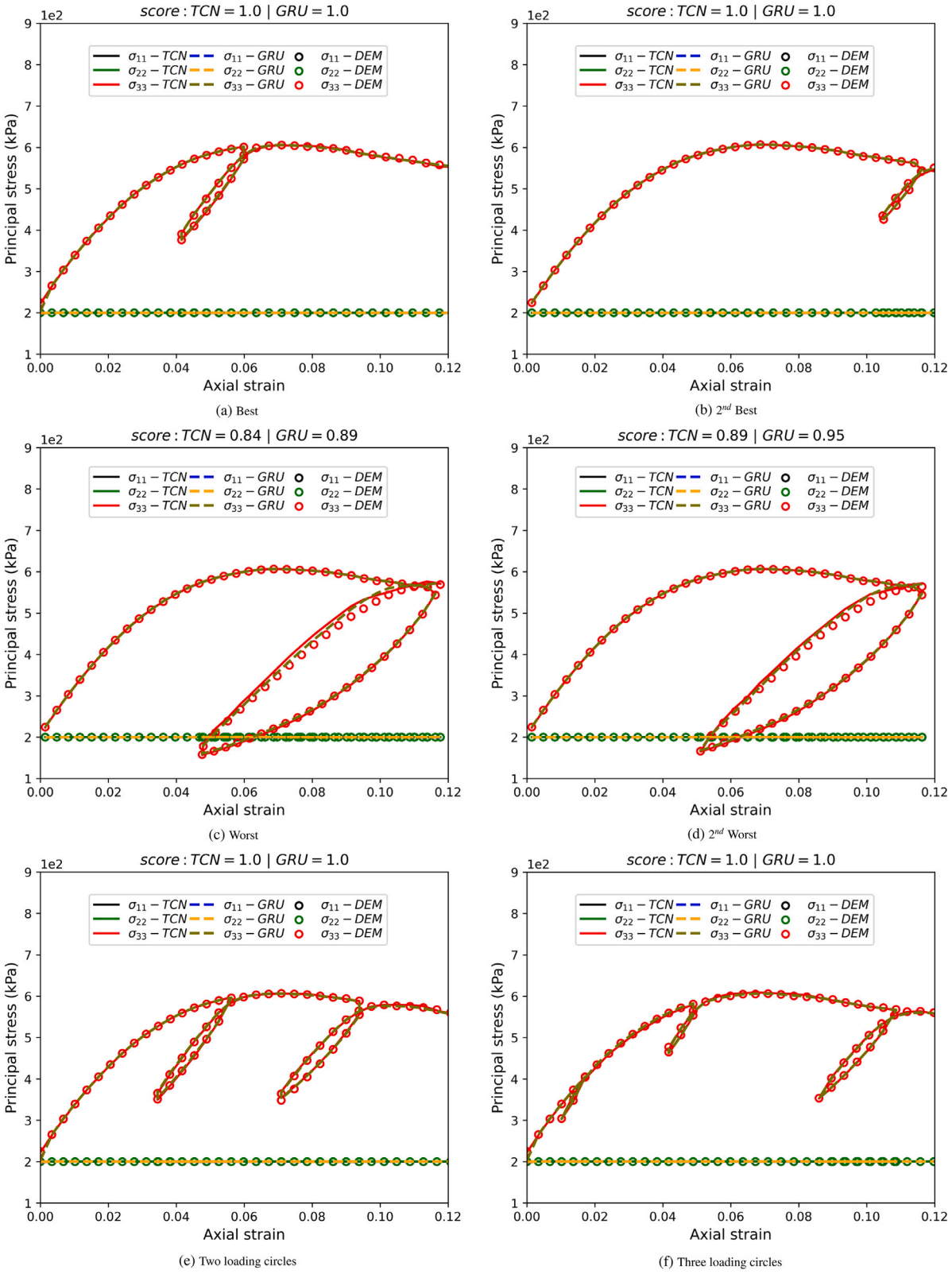
networks with the data generated by the resolution method. Fig. 12(d) plots all MAE values of the three experiments and demonstrates that increasing the time step can improve the training accuracy until the time step reaches 40.

5. Results analysis

In this section, three datasets generated in Section 3 are used to validate the prediction ability of the TCN network. In each dataset, in addition to the MAE of the test cases, the evaluation proposed in Section 2.4 is also employed to provide a direct score for each prediction result. To offer a more intuitive demonstration of the capability of the TCN network, some representative prediction results are plotted from each dataset. To further demonstrate the prediction capability of the TCN network, the stress–strain mapping of each case is also predicted by the trained GRU network with the same dataset.

5.1. The prediction result for two types of tri-axial loading

In this section, a total of 80 test cases, including 36 CP/B and 44 CTC cases, are used to validate the prediction ability of the TCN network. An overall MAE distribution of the 80 test cases with the TCN and GRU networks is demonstrated in Fig. 13. The result shows that the GRU and TCN have similar performance under CTC and CP/B loading



(a) Best (b) 2nd Best (c) Worst (d) 2nd Worst (e) Two loading circles (f) Three loading circles

Fig. 14. Prediction results of the TCN network under the CTC loading condition.

conditions. Both the TCN and GRU networks have better performance in the CP/B loading cases than that in CTC cases. Furthermore, except for some outliers, the MAEs of the test cases are located in a similar range under each loading condition in the TCN and GRU network. In addition, the median and mean MAE of test samples in one certain loading condition are almost the same using the TCN and GRU network.

Using the TCN and GRU networks, the 44 CTC samples obtain an average score of 0.973 and 0.0989, respectively, and the 36 CP/B cases acquire an average score of 0.977 and 0.986, respectively. Some representative prediction results of CTC and CP/B cases of the TCN network are demonstrated in Figs. 14 to 16, and the prediction result of the GRU is also given for direct comparison in each case.

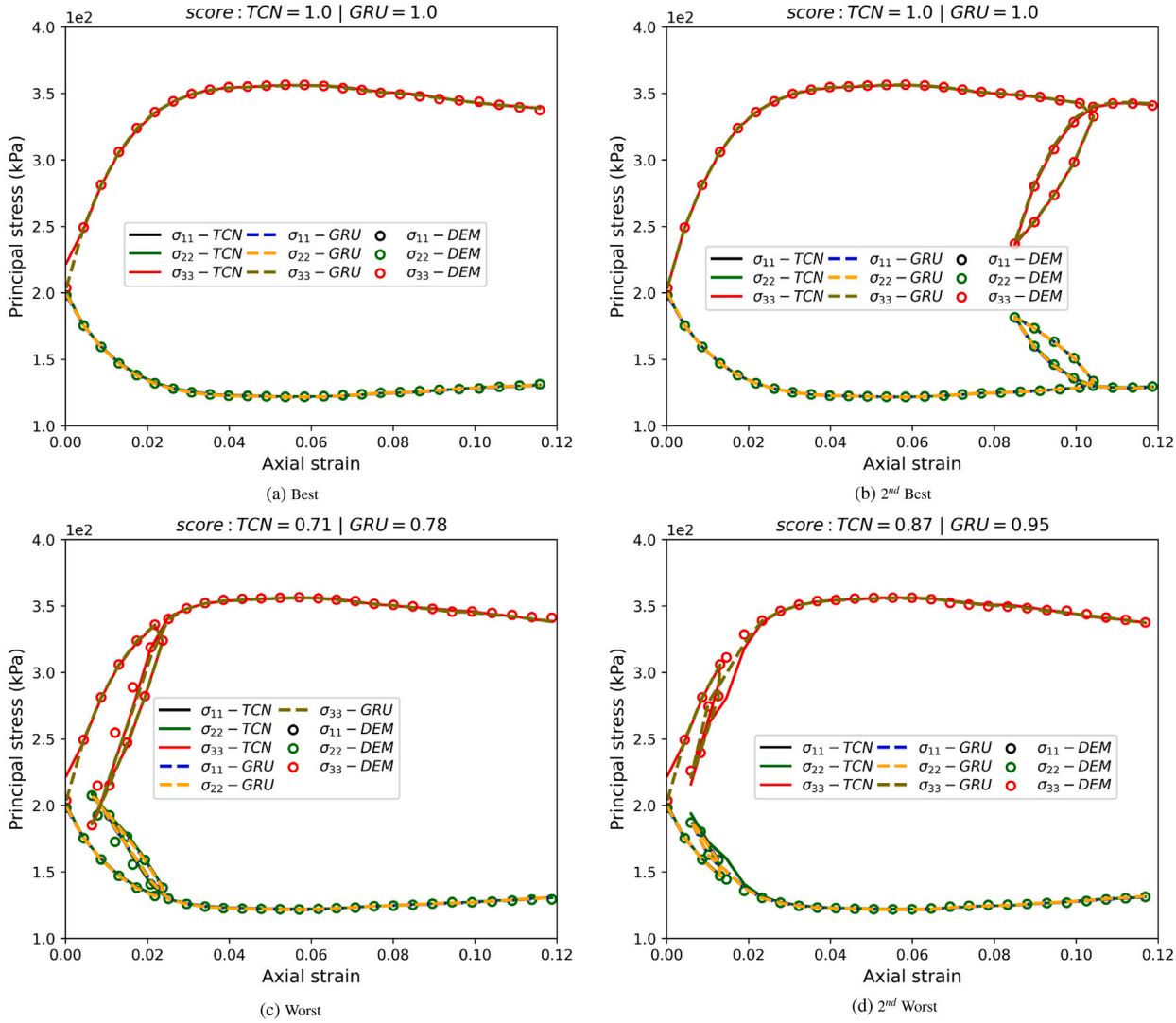


Fig. 15. Prediction results of the TCN network under the CP loading condition.

Fig. 14 shows that both the TCN and GRU networks demonstrate a strong capability of capturing the mechanical features of particle assemblies in CTC experiments, and both the best and worst predictions stem from the one unloading–reloading loop condition. Especially, Fig. 14(c) and (d) demonstrate that the worst prediction of the TCN network can still capture the stress–strain response satisfactorily even though they have lower scores. Besides, Fig. 14(e) and (f) show that the trained TCN and GRU networks can also forecast multi-cycle loading cases well, including two and three unloading–reloading loops, which are not included in the training data.

Four prediction cases, the best, 2nd best, worst, and 2nd worst of the TCN network under the CP condition are plotted in Fig. 15. Especially, Fig. 15(c) and (d) show that both the TCN and GRU networks with the worst evaluation results can still capture most of the stress–strain features of granular materials in the CP experiments.

The best and worst predictions under the CB condition are plotted in Fig. 16(a) and (b), and both cases obtain a score of 1 with the TCN and GRU networks.

In conclusion, the TCN and GRU networks have a similar ability to model the stress–strain response for the same granular material in the tri-axial compression experiment.

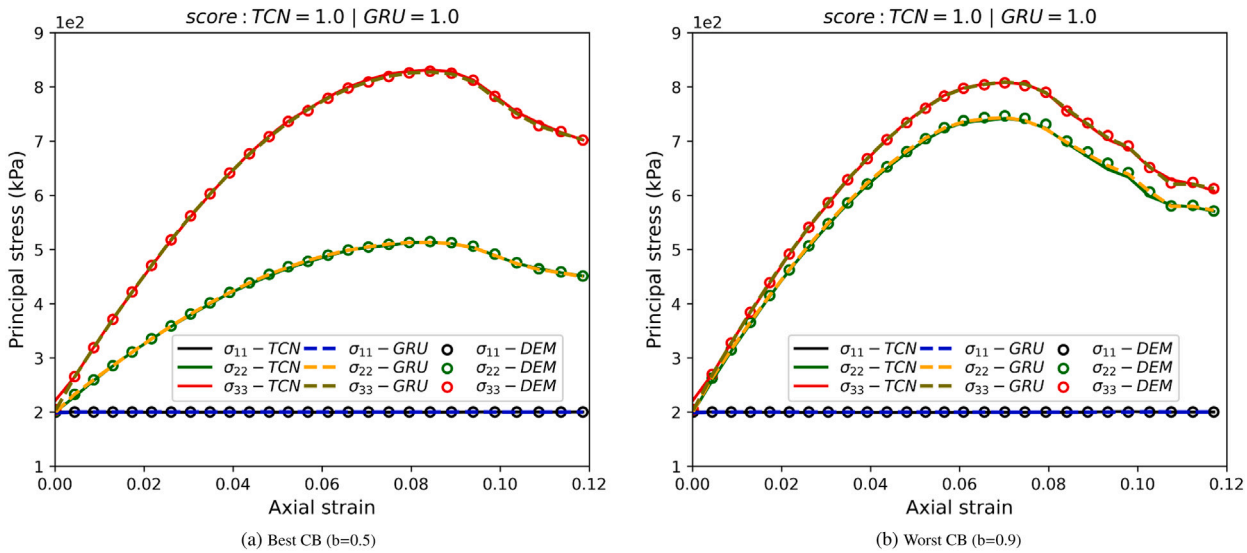
5.2. The prediction result for different granular samples and loading paths

This section investigates the performance of both the TCN and GRU networks for granular samples with different PSDs and initial states in the TTC experiment. The hyperparameters and training procedure used in Section 4 are also adopted here. The two networks are trained again with the 360 TTC cases in Section 3.2, and the remaining 90 samples, including 45 CB and 45 CP cases with different RVEs, b values, and initial stress states, are selected as the test set.

An overall MAE distribution of the 90 test cases is demonstrated in Fig. 17. It is found that the test cases obtain similar best, median, and mean MAE value using the TCN and GRU networks under the same condition. However, the MAE distribution of the CB and CP cases has a lower upper bound in the GRU network than that in the TCN network.

All 90 selected test cases obtain a score of 1 with both the TCN and GRU networks. To provide a more intuitive demonstration of the prediction ability of the TCN and GRU networks, the prediction results of CB and CP test cases with different PSDs, b values, and p values are plotted in Figs. 18 and 19, where c represents the confining stress; C, U, and O represent the particle samples of Constant, Uniform, and Original (see Table 1), respectively.

In the CB condition, Fig. 18(a) and (b) respectively plot the predicted principal stresses of the TCN and GRU networks in different particle samples with the same confining stress. Fig. 18(c) demonstrates



(a) Best CB (b=0.5) (b) Worst CB (b=0.9)

Fig. 16. Prediction results of the TCN network under the CB loading condition.

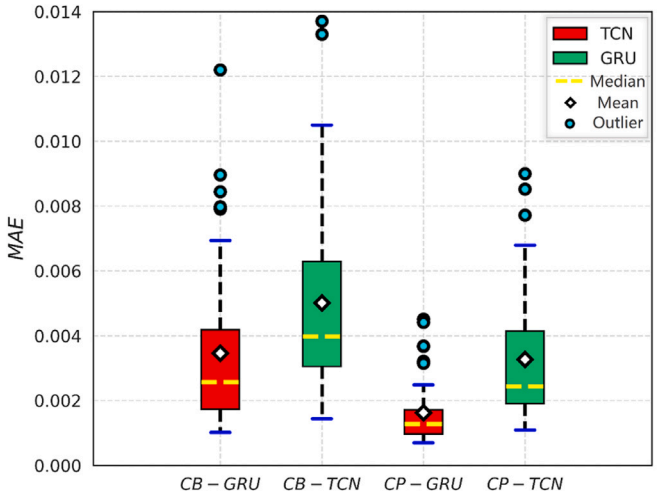


Fig. 17. The MAE distribution of the 90 test samples with different PSDs.

that both the TCN and GRU networks can accurately describe the constitutive relations for different particle samples in different loading paths.

A similar conclusion also drawn under the CP loading condition from the prediction results of Fig. 19.

All the above results demonstrate that both the TCN and GRU can not only predict the mechanical behaviour of the same particle assembly but also forecast the stress–strain response for granular materials with different PSDs and loading paths.

5.3. The prediction results for random stain loading

To further explore the potential of the TCN network, the dataset with random strain loading is also considered. Two data-driven models based on the TCN and GRU networks are constructed, respectively, utilising the random strain loading cases. The training method and hyperparameters used here are kept consistent with Section 4.

In this section, 16 test cases with random strain loading paths are selected to further test the capability of the TCN network. The prediction results of the TCN and GRU networks are plotted in Fig. 20, where the MAEs and scores of the 16 random strain loading cases are presented in Fig. 20(a) and (b), respectively.

The results in Fig. 20 show that both the TCN and GRU have a similar prediction ability in the random strain loading cases. Based on the GRU and TCN networks, the 16 test cases obtain an almost identical average MAE (0.0102 and 0.0113) and average score (0.990, and 0.986), respectively.

To offer a more intuitive exhibition of the capability of the TCN network, four prediction cases, the best, 2nd best, worst, and 2nd worst, of the TCN network are plotted, together with the prediction result of the GRU network, in Figs. 21 to 24. In each case, the predicted principal stress is plotted against the DEM simulated stress as the ground truth, respectively.

As demonstrated in Figs. 21 and 22, the predictions of the TCN and GRU networks are almost identical for the best two results, and both the TCN and GRU networks are capable of accurately capturing the unloading–reloading features of each principal stress.

In the two worst cases results, as shown in Figs. 23 and 24, although there are some differences between the prediction results and the ground truth, both the TCN and GRU networks can still reasonably capture the stress–strain responses.

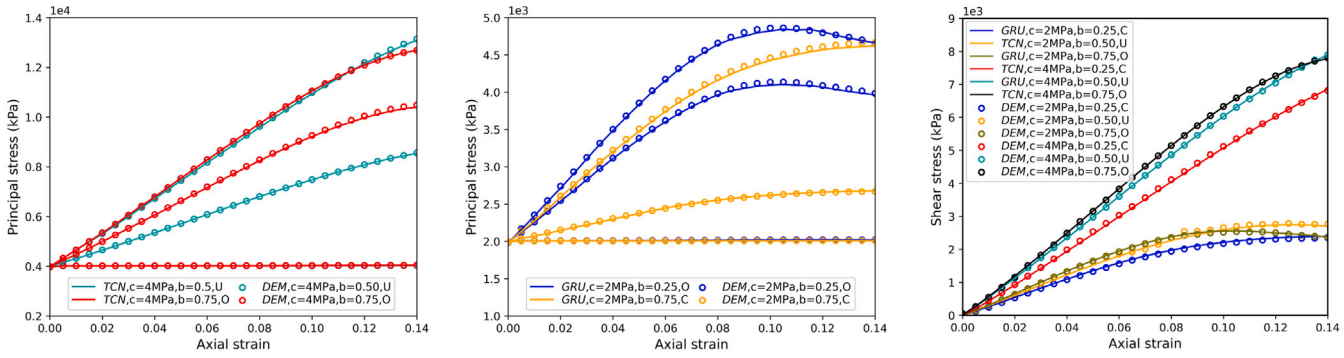
All the prediction results in this subsection show that the TCN and GRU networks have a similar prediction ability in random strain loading cases, and both of them perform well in describing the stress–strain response under a complex loading path.

5.4. The robustness of the TCN network

Generally speaking, some low levels of data noise, due to random and/or system errors, are inevitably present in practical applications. In this section, the performance of the TCN network with different noise levels of training datasets is first investigated. Then, the capability of the network to filter out noise data is also studied. The contaminated training datasets used for the investigation are generated by adding artificial noise data to the pure DEM modelling datasets used in Section 5.1. The time step of the dataset and hyperparameters of the TCN network used here are the same as in Section 4.1.

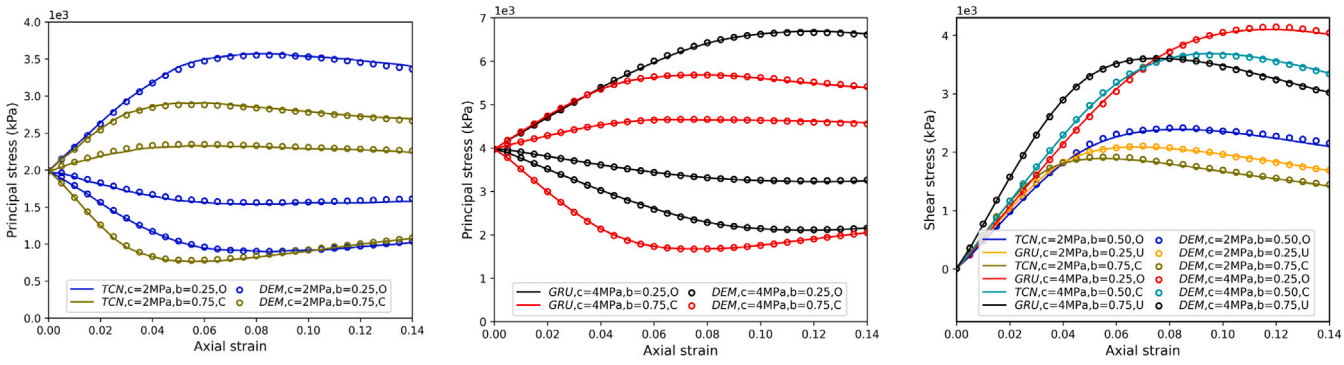
5.4.1. Effects of data noise on network performance

The noise data are yielded via normal distribution functions with different standard deviations (sd). For each sd value, three different columns of noise data are generated with different random seeds and added to the three principal stress sequences of each stress–strain



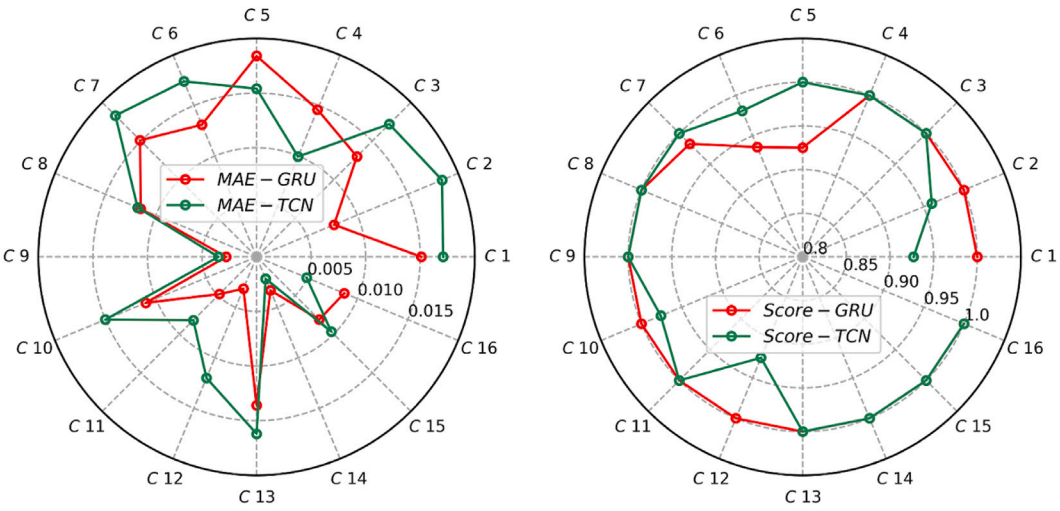
(a) The prediction of the TCN. (b) The prediction of the GRU. (c) The prediction of the shear stress.

Fig. 18. The prediction results of the TCN and GRU networks under different CB loading paths and PSDs.



(a) The prediction of the TCN. (b) The prediction of the GRU. (c) The prediction of the shear stress.

Fig. 19. The prediction results of the TCN and GRU networks under different CB loading paths and PSDs.



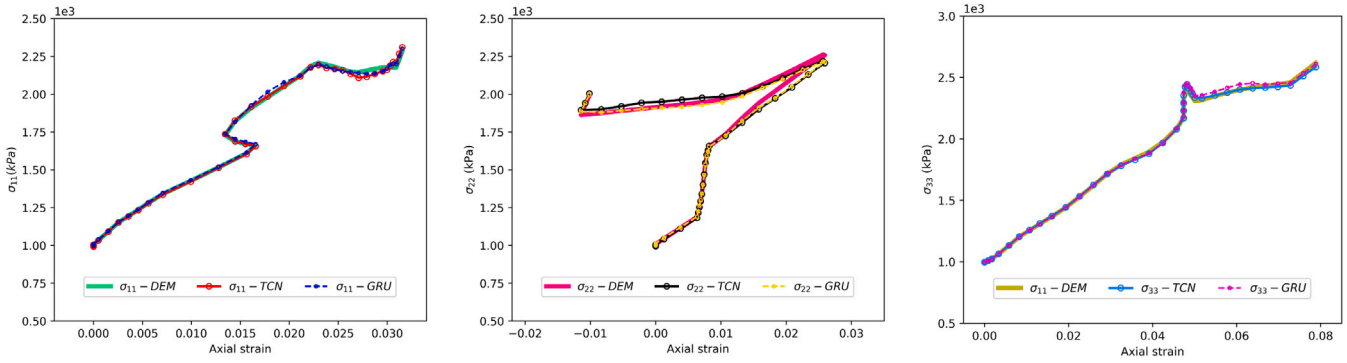
(a) The MAE of each case. (b) The score of each case.

Fig. 20. The MAE and score of the 16 random strain loading cases.

curve. The *sd* value ascends from 0.5% to 5% at an interval of 0.5%, and consequently, 10 groups of contaminated datasets with different *sd* values are created to train and validate new TCN networks. The prediction capability of newly-trained TCN networks is tested via 80 uncontaminated test samples.

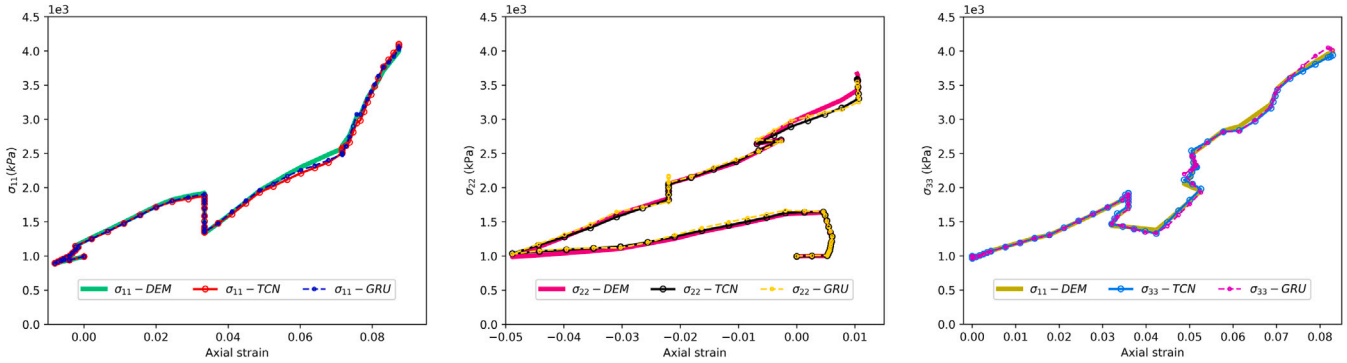
The noise level has an impact not only on the training stage but also on the prediction ability of the TCN network. For the former, the MAE value rises with the increase of the *sd* value, as shown in Fig. 25.

To explore the influence of noise level on the prediction capability of the TCN network, datasets with three different *sd* values (0.5%, 1.5%, 2.5%, 3.5%, and 4.5%) are used to train TCN networks. The



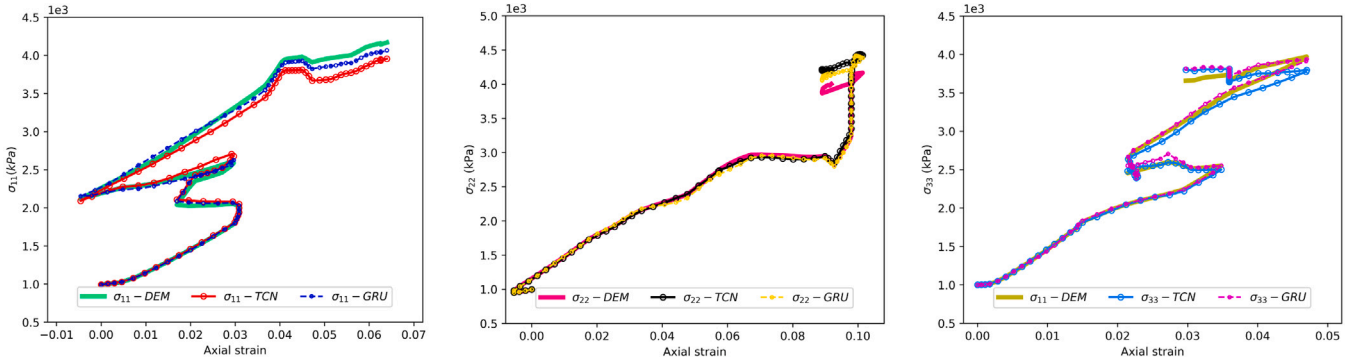
(a) σ_{11} of the case. (b) σ_{22} of the case. (c) σ_{33} of the case.

Fig. 21. The best prediction result of random strain loading cases.



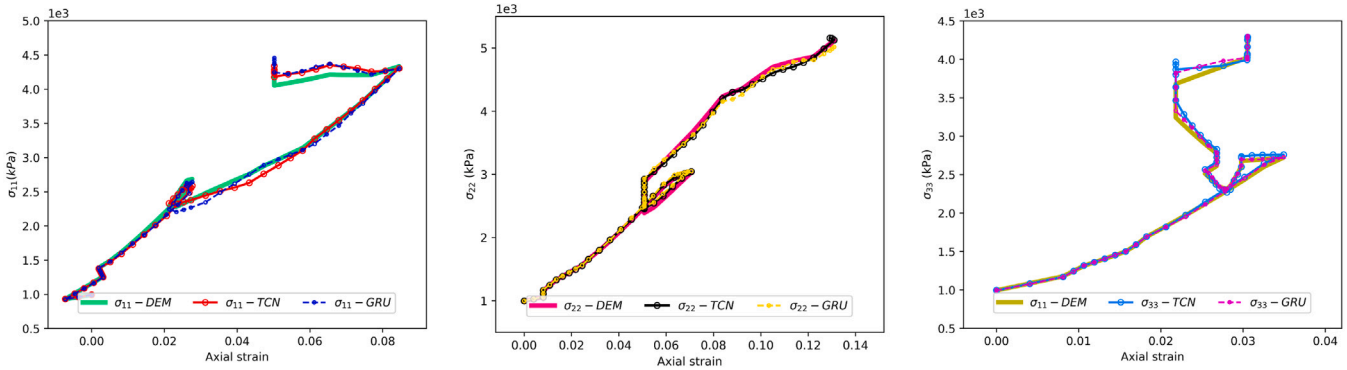
(a) σ_{11} of the case. (b) σ_{22} of the case. (c) σ_{33} of the case.

Fig. 22. The 2^{nd} best prediction result of random strain loading cases.



(a) σ_{11} of the case. (b) σ_{22} of the case. (c) σ_{33} of the case.

Fig. 23. The worst prediction result of random strain loading cases.



(a) σ_{11} of the case. (b) σ_{22} of the case. (c) σ_{33} of the case.

Fig. 24. The 2^{nd} worst prediction result of random strain loading cases.

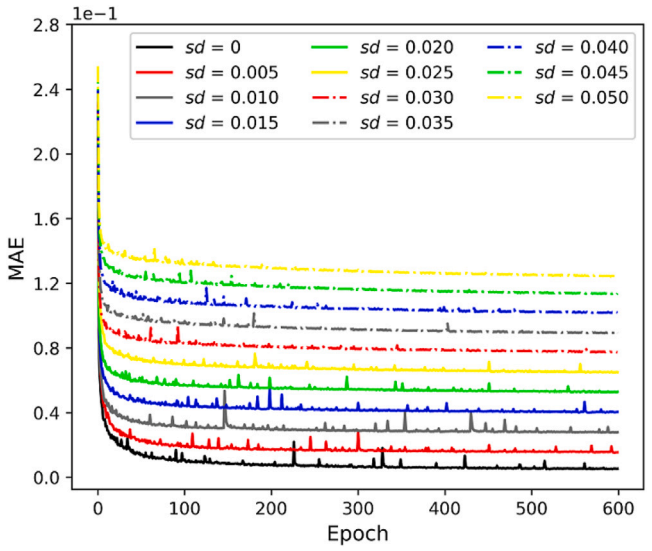


Fig. 25. The learning curves of models with different noise levels.

trained networks referred to as contaminated or polluted networks later are tested for the original uncontaminated 80 triaxial compression cases in Section 3.1. Corresponding to different sd values, the detailed empirical cumulative distribution (ECD) of MAEs and scores for all 80 test cases are provided in Fig. 26. The result shows that the maximum MAE and the lowest score of the 80 test cases increase with the growth of the sd values.

To further demonstrate the performance of the TCN networks with polluted datasets, the loading cases in Section 5.1 are predicted again using the contaminated TCN networks when the sd is equal to 0.5%, 2.5%, and 4.5%, respectively.

For the CTC loading condition, the best and worst prediction results are given in Fig. 27. It is found that the polluted TCN networks can still describe the main constitutive features of granular materials, although they obtain lower scores and higher MAEs. However, for multi-cycle loading cases, the polluted TCN networks perform worse when the sd value reaches 4.5% and seem no longer make accurate predictions.

Fig. 28 provides a direct demonstration of the robustness of the TCN networks under the CP loading condition. Fig. 28(a) demonstrates that TCN networks are capable of making good predictions for the CP loading condition with different noise levels. For the worst loading cases (Fig. 28(b)), the polluted TCN networks can still predict the most stress–strain response of the granular system.

A similar conclusion can also be drawn for the CB loading cases, as shown in Fig. 29.

5.4.2. Data noise filtration of the TCN network.

The cases shown in Figs. 27 to 29 also suggest that the TCN network might be able to filter out data noise to some extent. Thus eight representative cases in Section 5.4.1 are selected to further investigate this seemingly desirable feature. As both the bias and variance significantly affect the level of data noise, two normal distributions with the same sd value of 4.5% but different bias (μ) values (0.15 and −0.15) are used as data noise to generate the contaminated dataset again to train the network. The prediction result of each test case, corresponding to the polluted stress, are plotted in Figs. 30 to 32. Then comparing the contaminated strain–stress curve with the prediction in each case should illustrate to what level randomly added noise may be filtered out by the TCN network.

For the CTC loading condition, compared to the results in Fig. 27, Fig. 30 demonstrates that the added bias value in the noise data has a slight influence on the prediction of the TCN. Furthermore, the TCN network can filter out most of the noise and make good predictions for cases with one loading–unloading cycle (Fig. 30(a) and (b)). However, the network gradually loses this favourable feature for cases with multiple loading–unloading cycles (Fig. 30(c) and (d)), as there are no such cases that are included in the training, and thus the network has a weaker resistance to the disturbance of noise data in the corresponding loading condition, although the TCN network performs well in uncontaminated multiple-cycles loading test cases (see Fig. 14(e) and (f)).

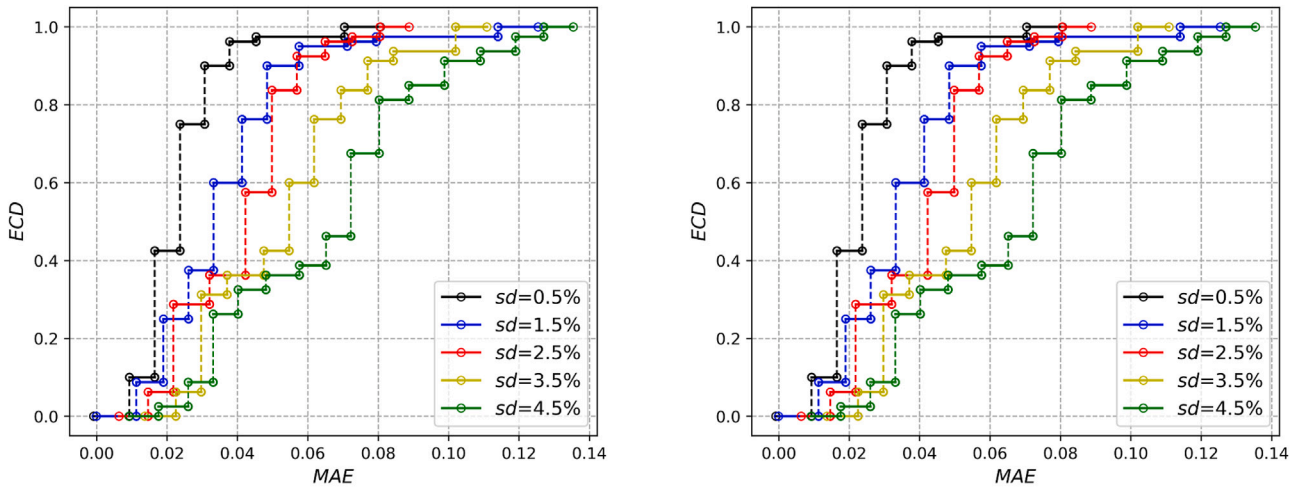
For the cases in both CP and CB conditions shown in Figs. 31 and 32, respectively, the filtering effect of the TCN network to the noise data with different sd and bias values is confirmed. Thus it can be concluded that TCN networks have the capability of filtering out randomly generated noise data to a certain extent, and thus generally enhance the prediction performance.

Meanwhile, the MAEs of the polluted and predicted stresses in each case are also calculated and plotted in Fig. 33. This favourable feature is in fact due to the nature of neural networks that can learn a representation to optimally fit the given data if over-fitting is avoided in the training, and hence should also be possessed by other neural networks.

6. Conclusion

This paper has attempted to construct a TCN network-based constitutive model to capture the history/path-dependent nature of granular stress–strain relation. To train and validate the TCN network, three types of numerical experiments, including CTC, TTC, and random strain loading experiments, are implemented to generate datasets via discrete element modelling of representative volume elements (RVE) of particle systems. Furthermore, to maximise the potential of the proposed model, the Bayesian optimisation method is employed to find the optimum architecture of the TCN network. Meanwhile, the training data generation strategy is optimised and a transfer learning (TL) scheme is innovatively embedded into the TCN algorithm. It is found that the training efficiency and accuracy obtained can be significantly improved when applying the optimised data generation method and the transfer learning scheme together to the TCN algorithm. The prediction results of the trained model demonstrate that the TCN is reliable in predicting mechanical features of the granular materials. In addition, the noise data with different variances and biases are artificially added to training datasets to test and analyse the robustness of TCN networks. The result shows that TCN networks have the capability of filtering out randomly generated noise data to a certain extent, which is crucial in engineering. This favourable feature should also be possessed by other neural networks, although the numerical investigation has been conducted here.

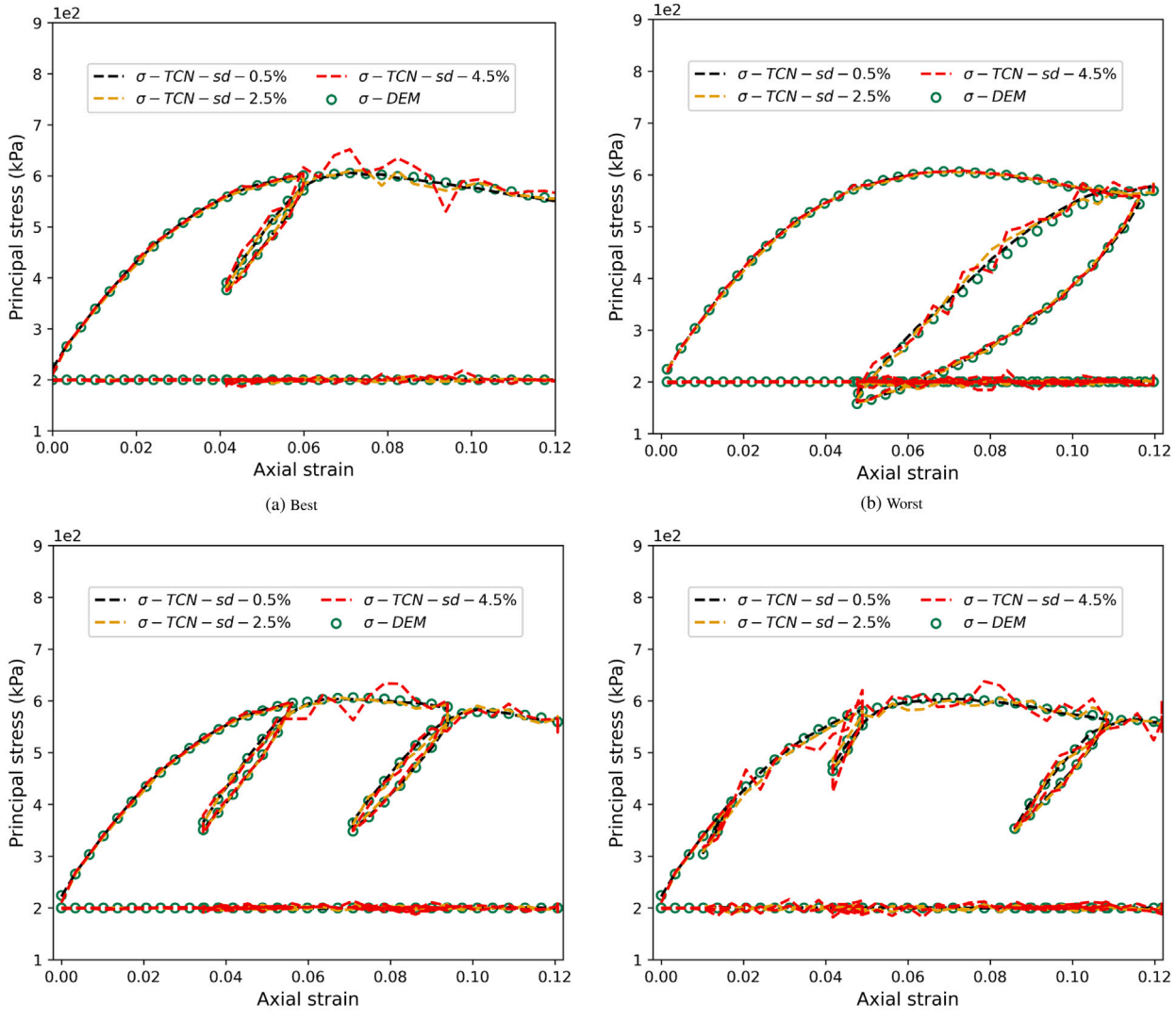
However, some issues have not been covered or considered in the current work. For instance, the homogenisation assumption is not adopted when generating training data, and thus the TCN model is trained in the principal stress–strain space, instead of in an invariance space (Lefik et al., 2009). In addition, the strain–stress pairs on each stress–strain curve in the training database are sampled with more or less similar strain increments. Thus the TCN network trained on the current dataset may perform badly when making predictions for stress–strain datasets with different or variable strain increments, as pointed out in Qu et al. (2021a). Besides, all time-sequence neural networks are of a multiple-time-step nature which is different from conventional material constitutive models and therefore are generally incompatible with the standard finite element solution procedure where a single step strain or increment is provided for evaluating the corresponding stress state. All the above issues will be addressed and reported in our further work.



(a) The MAEs of the test cases with different sd values.

(b) The scores of the test cases with different sd values.

Fig. 26. The prediction results of the polluted TCN networks.



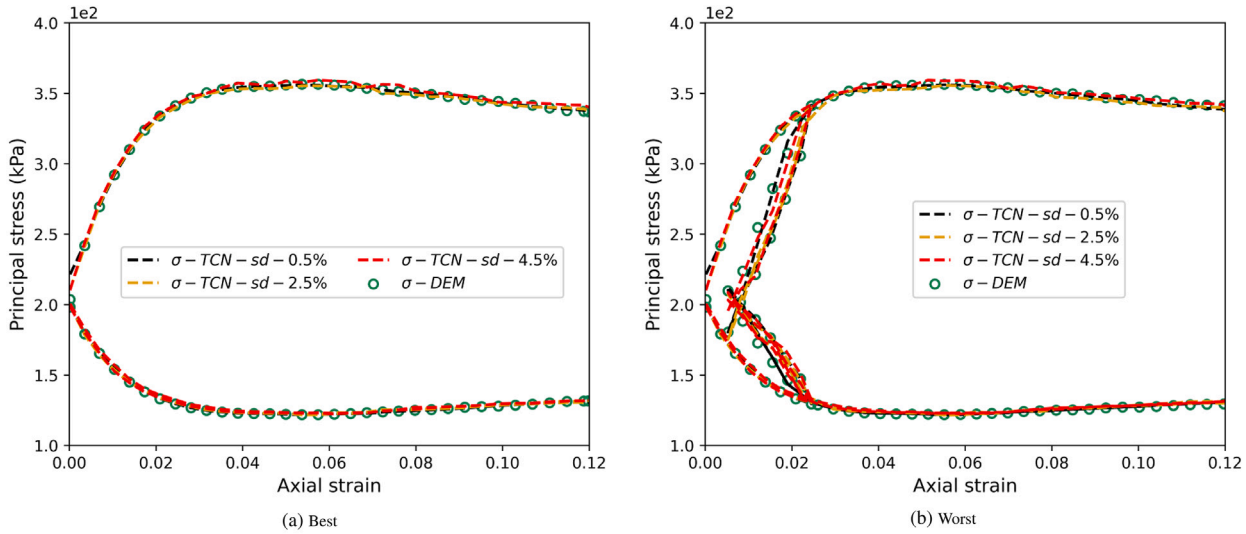
(a) Best

(b) Worst

(c) Two loading cycles

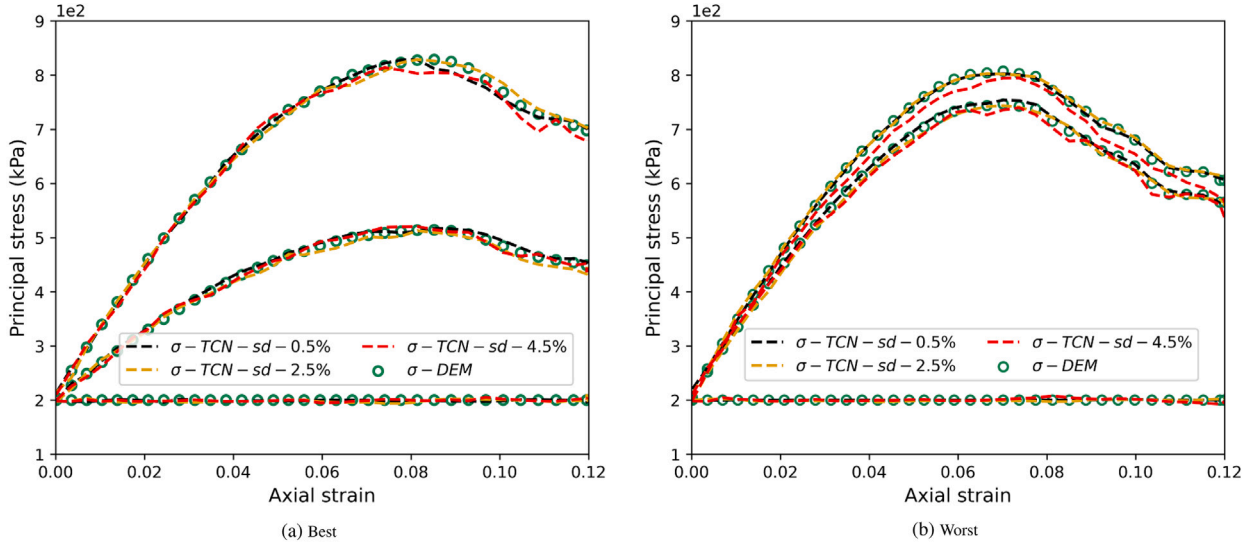
(d) Three loading cycles

Fig. 27. Prediction results of the polluted TCN network under the CTC loading condition.



(a) Best (b) Worst

Fig. 28. Prediction results of the polluted TCN network under the CP loading condition.



(a) Best (b) Worst

Fig. 29. Prediction results of the polluted TCN network under the CB loading condition.

CRediT authorship contribution statement

Mengqi Wang: Methodology, Software, Writing – original draft. **Tongming Qu:** Conceptualization, Data curation. **Shaoheng Guan:** Data curation. **Tingting Zhao:** Software. **Biao Liu:** Validation. **Y.T. Feng:** Supervision, Reviewing and editing, Project administration.

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.

Data availability

Data will be made available on request.

Acknowledgements

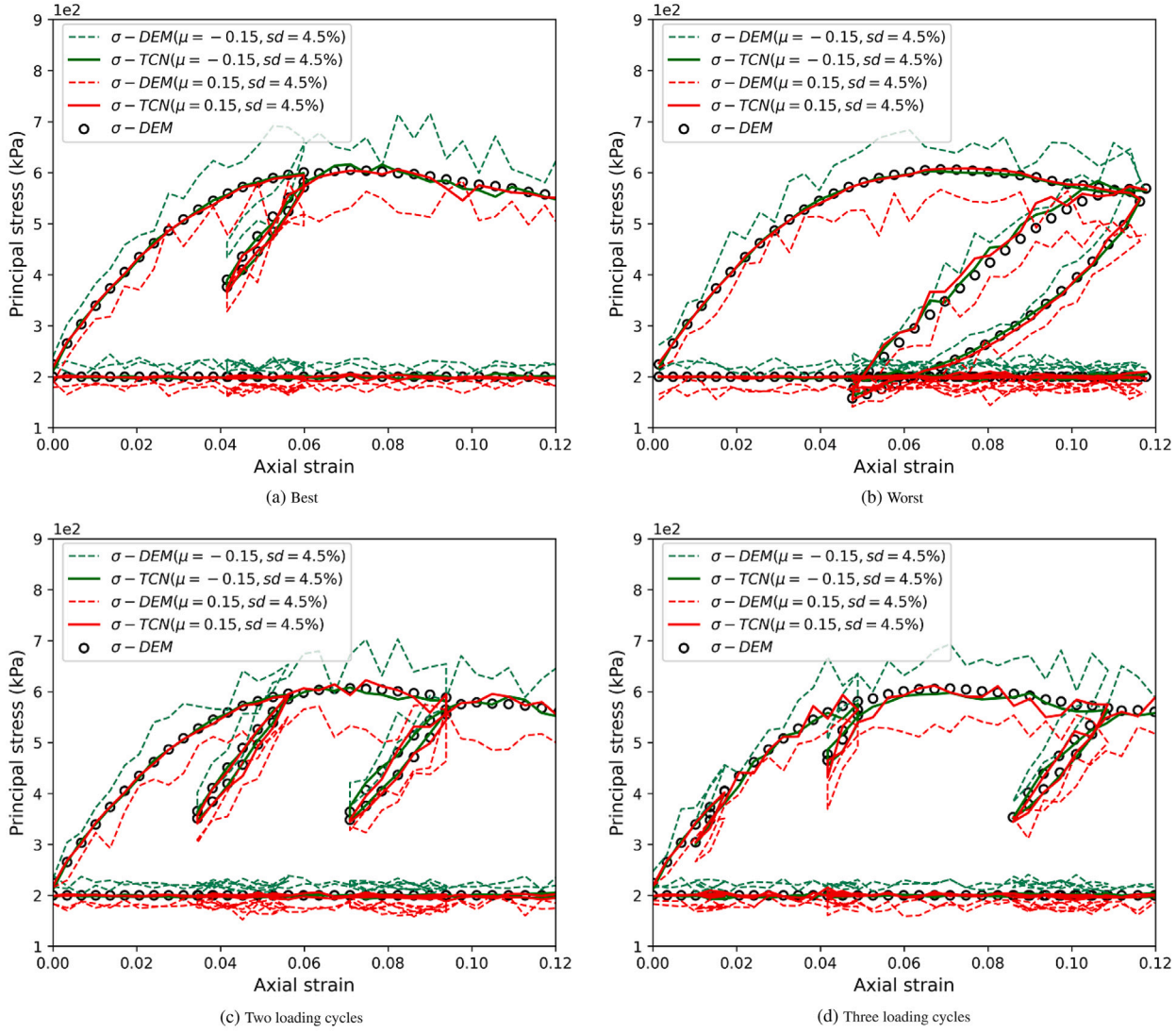
This work is supported by the National Natural Science Foundation of China (NSFC) (Grant Nos. 12102294 and 12072217). The support is greatly acknowledged.

Appendix A. The DCC layer and TCN block in the TCN network

(1) The 1D-DCC layer

A dilated casual convolution layer is a modified casual convolution neural network that is evolved from the standard 1D-convolution layer. A standard 1D-convolution layer is shown in Fig. 34(a), where $(a_0, a_1, \ldots, a_t)$ is the input sequential data with time-step t, and $(z_0, z_1, \ldots, z_t)$ is the output sequence obtained by applying a convolution calculation with a (three-element) filter (an array that stores weights and bias). To maintain the identical length of the input and output data, both ends of the input data are padded with zeros first. In the convolution operation, the filter slides along the direction of the time sequence from the beginning of the zero-padded input data at a stride of one. However, this type of architecture involves future information in the current convolution calculation, e.g., the output z_i depends on the input a_{i+1}. This is incorrect for time sequence data.

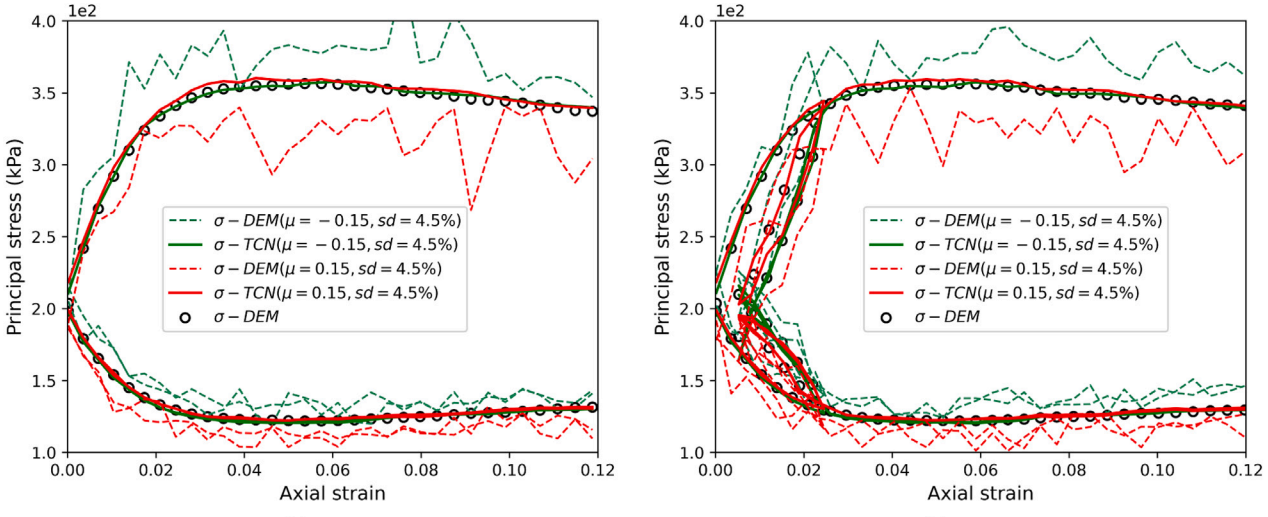
To overcome the above defect, the casual convolution neural network (Bai et al., 2018) is proposed, as presented in Fig. 34(b), which adopts the manner of one-end zero padding to the input data to guarantee that only past or historical data is included for each convolution



(a) Best (b) Worst

(c) Two loading cycles (d) Three loading cycles

Fig. 30. The filtering effect of the TCN network on noise data under the CTC loading condition.



(a) Best (b) Worst

Fig. 31. The filtering effect of the TCN network on noise data under the CP loading condition.

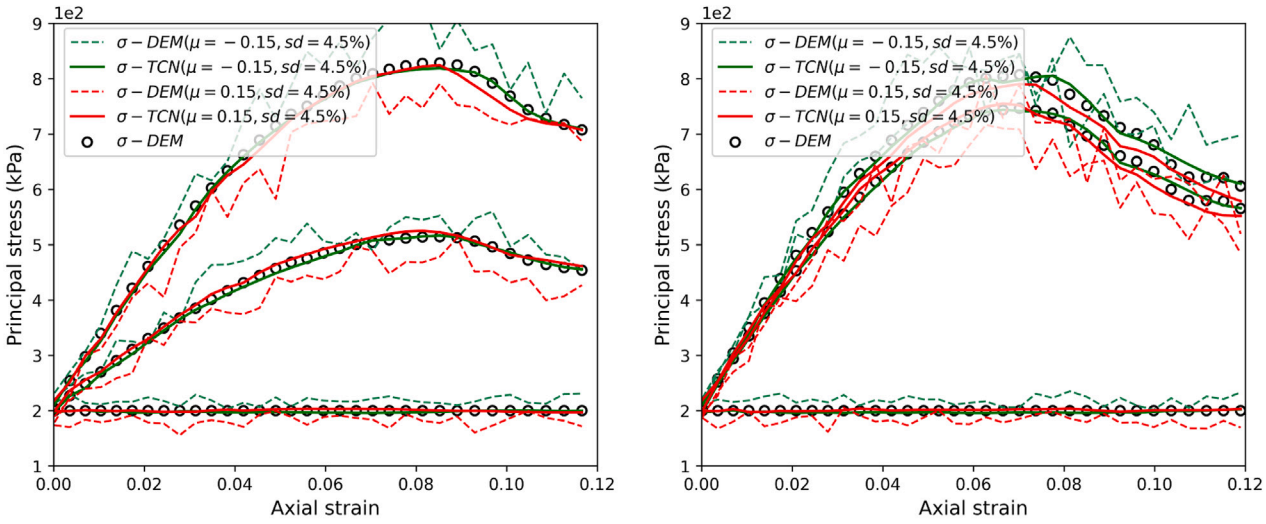


Fig. 32. The filtering effect of the TCN network on noise data under the CB loading condition.

Table 6
Training results with the different number of TCN blocks.

No.	DCC layers	Filters	Kernel size	TCN blocks	Loss value	Time (s/epoch)
1	3	32	2	1/2/3/4	0.0139/0.0084/0.0077/0.0074	2/4/6/9
2	3	32	4	1/2/3/4	0.0105/0.0076/0.0073/0.0072	3/6/9/13
3	3	64	2	1/2/3/4	0.0089/0.0067/0.0065/0.0065	4/8/12/18

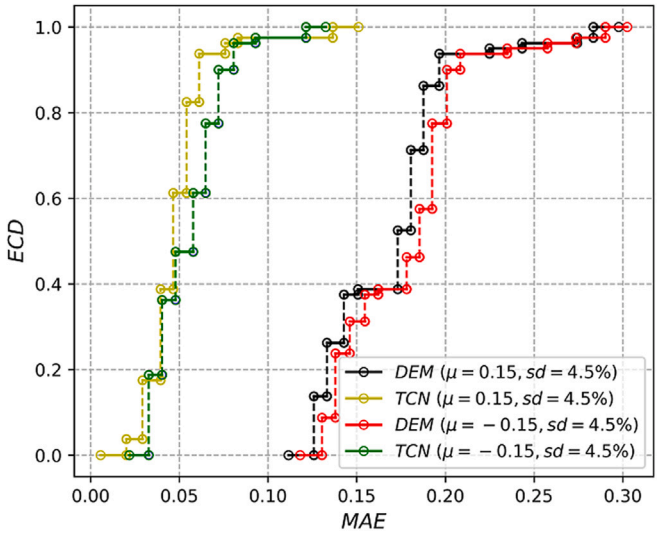


Fig. 33. The MAEs of the polluted TCN networks and polluted samples.

calculation. This makes the network suitable for addressing time series prediction problems.

Meanwhile, to improve the prediction accuracy of the casual convolution layer, a dilated factor D is introduced to invoke more past information in each convolution calculation, resulting in a dilated causal convolution (DCC) layer (Bai et al., 2018), as shown in Fig. 34(c). To further take the advantage of the dilation, multiple DCC layers with increasing dilation factors D are stacked together to form one casual convolution unit in the TCN network. Such a casual convolution unit with two (hidden) DCC layers is shown in Fig. 34(d).

When assuming that there are a total of L layers in one TCN block, the output and input layer of the TCN block is regarded as the first and the L^{th} layer, respectively. The relationship between the involved historical information length (IHIL) of each layer and the hyperparameters kernel size K, dilation factor D in one convolution calculation can be expressed as:

$$\begin{cases} IHIL^{(1)} = M^{(1)} \\ IHIL^{(i+1)} = [(K-1)D^{(i+1)}+1]M^{(i)} - D^{(i+1)}N^{i+1} \end{cases} \quad (K \geqslant 2) \qquad (11)$$

where i denotes the ith layer of one TCN block, and $i \in (1, L-1)$. In the current work, the dilation factor at the ith layer is taken to be:

$$D^i = 2^{(L-i)} \qquad (12)$$

To generate one element in the output layer, the number of elements involved in one convolution calculation of each layer can be expressed by M, where $M^{(1)}$ represents one element in the output layer of the TCN block. N is the number of elements that are repeatedly involved when the filter slides along the historical strain sequence. The relation between the M and N can be formulated as:

$$\begin{cases} M^{(i+1)} = KM^{(i+1)} - N^{(i)}(K-2) \\ N^{(i)} = M^{(i)} - 1 \end{cases} \qquad (13)$$

As the input of the TCN network is numerous 2D arrays (three principal strain sequences), each DCC layer is extended from 1D to 2D by adding multiple columns. The added dimension is called $depth$. Consequently, each 1D filter is also extended to a 2D filter block. The shape of this 2D filter block can be described in a form of $C \times kernelsize$, where C represents the number of columns in the depth in one filter block and $kernelsize$ refers to the number of elements along the direction of the time step of the filter. Note that the number of filters blocks used for processing one DCC layer is the same as the depth of the next layer and different C's can be used between different layers (see Fig. 35).

Appendix B. The sensitivity analysis of hyperparameters

Tables 6 to 9 provide the parameters used in experiments that are designed to determine the hyperparameter space. The learning

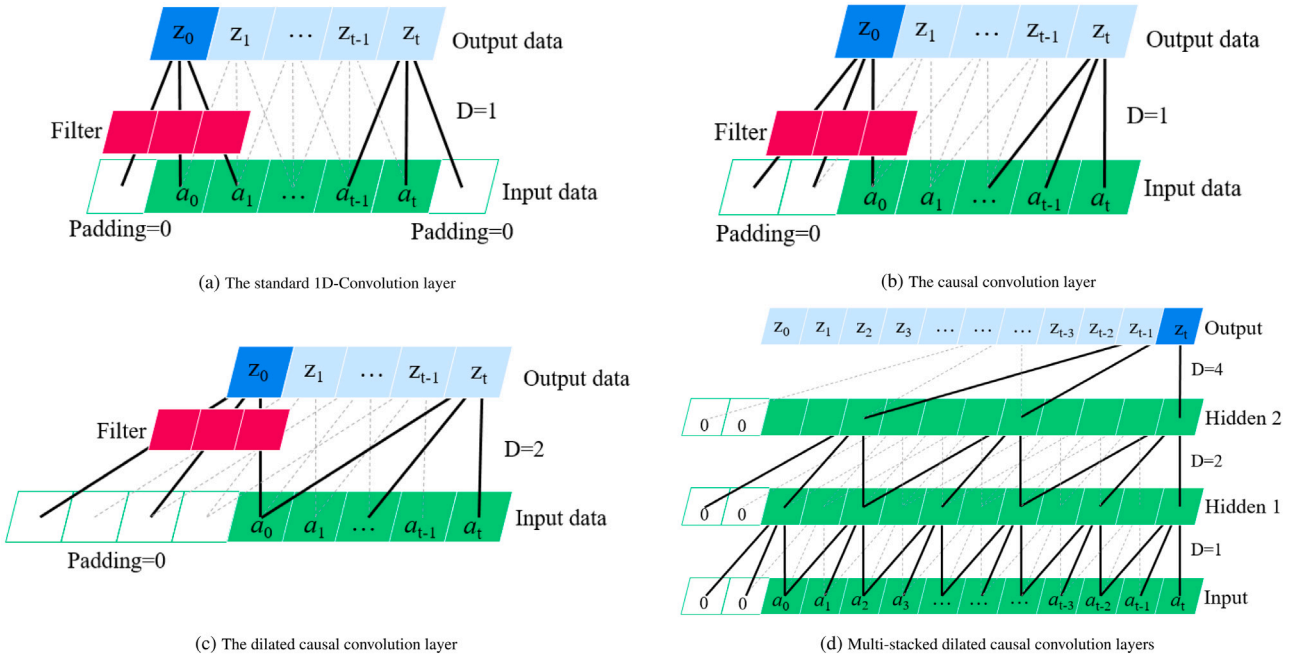


Fig. 34. The 1D-convolution neural networks.

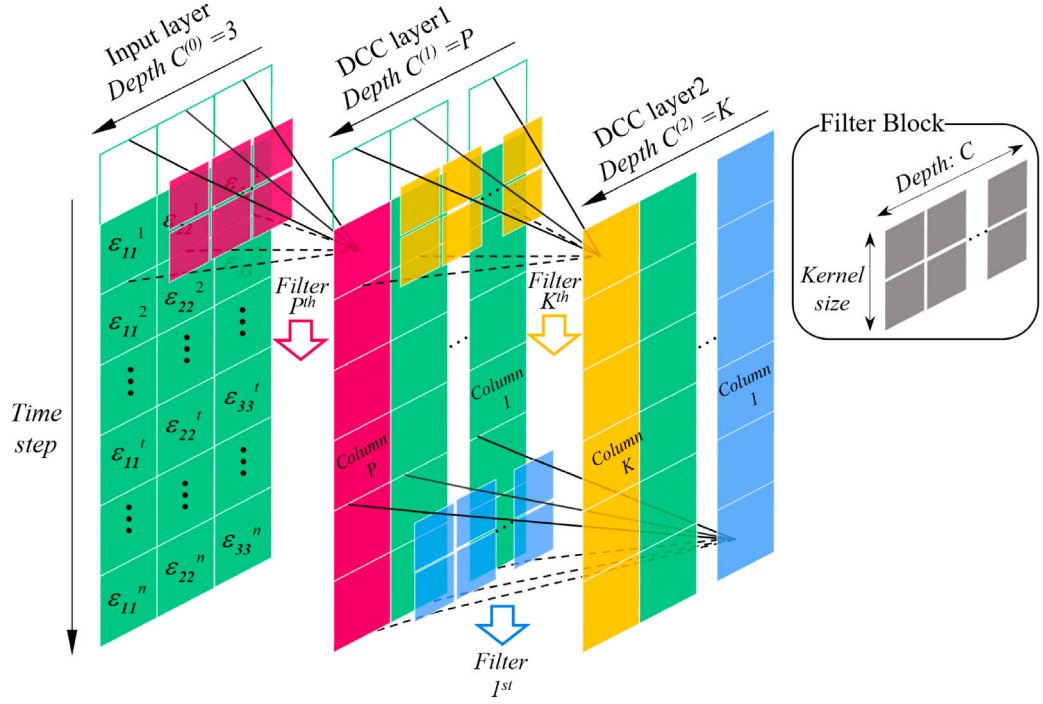


Fig. 35. The TCN block with multiple 2D filter blocks.

curves and the training time of one epoch in each experiment are demonstrated in Figs. 36 to 39, where V, L, b, f, and k represent the validation loss curve, loss curve, the number of TCN blocks, filters, and kernel size, respectively.

(1) TCN blocks As listed in Table 6, to determine the range of TCN blocks in one TCN network, three experiments are implemented. In each experiment, four TCN networks are established with one, two, three, and four TCN blocks, respectively. Other hyperparameters, including the numbers of the DCC layers, the filters, and the kernel size, are kept the same but are set to different values for different experiments.

Fig. 36 indicates that the accuracy of the network has the most notable improvement when the number of TCN blocks increases from one to two but has almost no change when the number of TCN blocks increases from three to four. Consequently, the maximum number of the TCN blocks is set to be three.

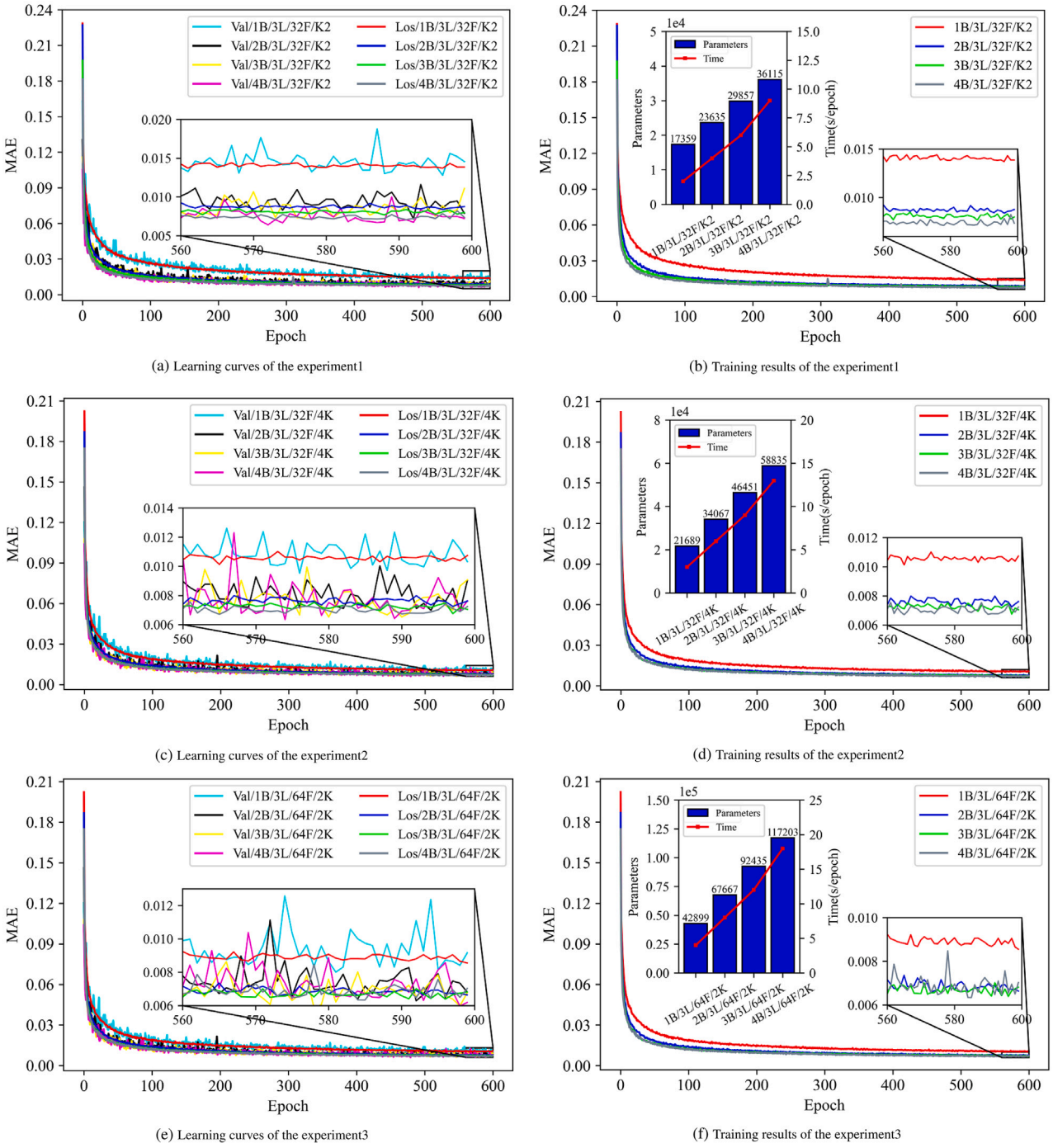


Fig. 36. Learning curves with different number of TCN blocks.

(2) DCC layers

Then, another three comparison experiments are implemented to identify the range of DCC layers in one TCN block. As listed in Table 7, in each experiment, three TCN networks with the same number of TCN blocks, but with two, three, and four DCC layers in one block respectively, are constructed. The other hyperparameters (the filters, and kernel size) are kept constant.

Fig. 37 shows that the MAE of the networks decreases with the increase of DCC layers in one TCN block, especially when it increases from two to three, and there is still a slight improvement in the accuracy when continuing to increase the number of DCC layers to four. Therefore, the maximum number of the DCC layers is limited to four.

(3) Filters

Furthermore, two extra comparison experiments are used to explore the number of filters in one DCC layer, based on one certain architecture of the TCN network (i.e., a certain number of TCN blocks and the DCC layers in one block). In each experiment, as listed in Table 8, four TCN networks are built with 32, 64, 96, and 128 filters in each DCC layer, respectively.

Fig. 38 indicates that the value of MAE in different TCN networks decreases sharply when increasing the filters from 32 to 64 in each DCC

Table 7
Training results with the different number of DCC layers in one TCN block.

No.	DCC layers	Filters	Kernel size	TCN blocks	Loss value	Time (s/epoch)
1	2/3/4	32	2	3	0.0099/0.0077/0.0071	3/6/8
2	2/3/4	32	4	3	0.0085/0.0071/0.0068	6/9/12
3	2/3/4	64	2	2	0.0085/0.0067/0.0062	5/8/11

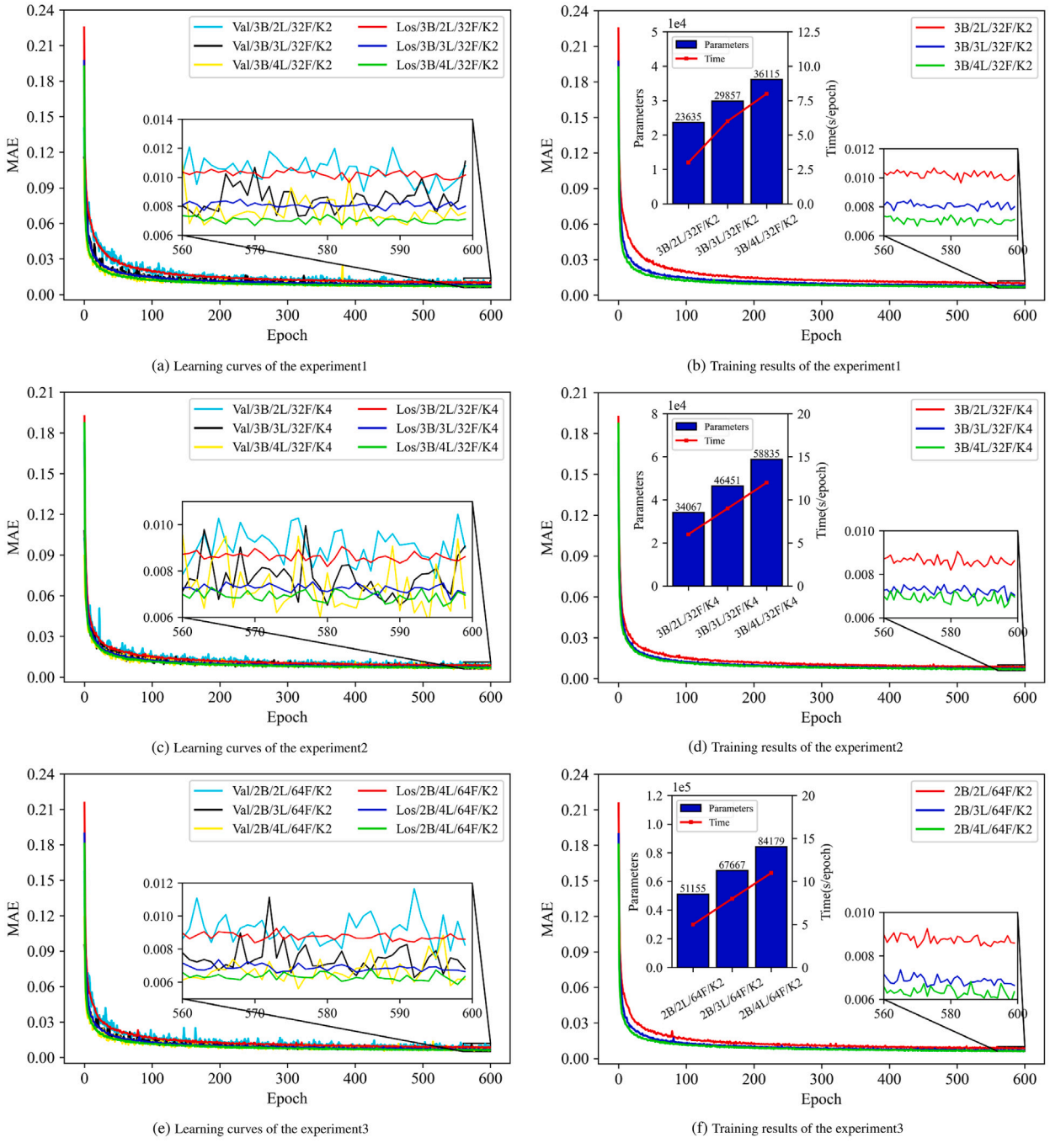


Fig. 37. Learning curves with different number of DCC layer in one TCN block.

Table 8
Training results with the different number of filters in one TCN block.

No.	DCC layers	Filters	Kernel size	TCN blocks	Loss value	Time (s/epoch)
1	3	32/64/96/128	2	1	0.0139/0.0089/0.0079/0.0077	2/4/6/10
2	3	32/64/96/128	2	2	0.0084/0.0067/0.0062/0.0061	4/8/13/22

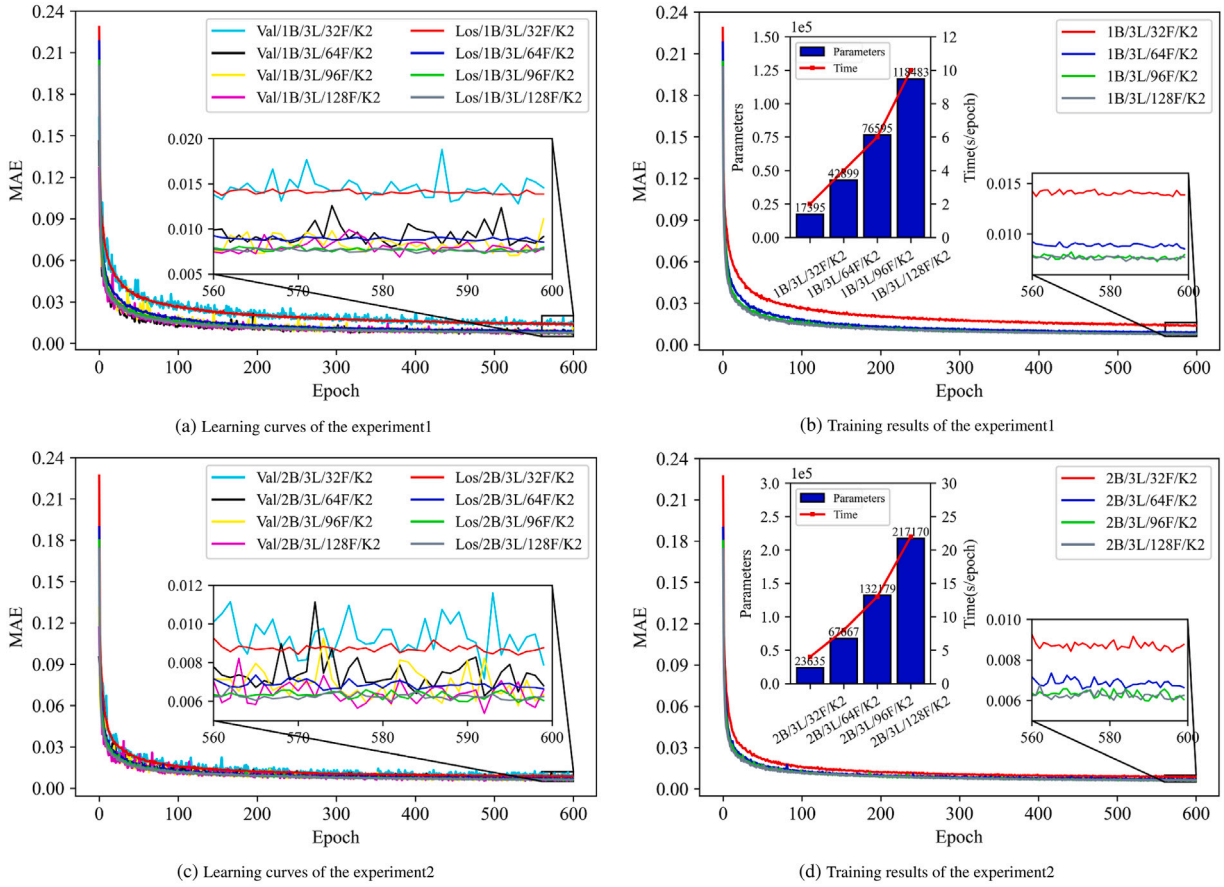


Fig. 38. Learning curves with the different number of filters in one DCC layer.

Table 9
Training results with different Kernel sizes of one filter.

No.	DCC layers	Filters	Kernel size	TCN blocks	Loss value	Time (s/epoch)
1	3	32	2/4/6/8	2	0.0084/0.0076/0.0075/0.0072	4/6/7/9
2	3	32	2/4/6/8	3	0.0077/0.0071/0.0071/0.0070	6/9/10/14
3	4	32	2/4/6/8	2	0.0075/0.0068/0.0068/0.0067	5/8/11/14

layer. It has less effect on enhancing the precision of the network when increasing the number of filters to 128 but results in a rapid rise in the computational cost. As a result, the maximum number of filters in one TCN network should be limited to 64.

(4) Kernel size

Finally, the range of kernel size is determined by three comparison experiments after determining the number of TCN blocks, DCC layers, and the corresponding number of filters in one DCC layer. As listed in Table 9, the kernel sizes of two, four, six, and eight are adopted in one TCN network in each experiment to investigate the influence of kernel size on the training accuracy of the TCN network.

As shown in Fig. 39, the MAE values of TCN networks have a rapid decline (i.e., an improvement in accuracy) when increasing the kernel size from two to four. However, this phenomenon almost disappears when the kernel size increases from four to six and eight. Accordingly, the upper limit of the kernel size is set to four.

Appendix C. The results of the Bayesian search experiment

See Tables 10–13.

Table 10
The results of Bayesian search trial 1 with 2 TCN blocks and 3 DCC layers in one block.

Trial	Parameters	Filters	Kernel size	Learning rate	Batch size	Loss value
1	66 499	64	3	0.0001	64	0.015125
2	50 019	48	4	0.0001	64	0.017683
3	87 171	64	4	0.0001	128	0.020305
4	66 499	64	3	0.0001	128	0.020535
5	26 691	64	2	0.0001	64	0.020557

Table 11
The results of Bayesian search trial 2 with 2 TCN blocks and 4 DCC layers in one block.

Trial	Parameters	Filters	Kernel size	Learning rate	Batch size	Loss value
1	91 203	64	3	0.0001	64	0.014678
2	62 339	64	2	0.0001	64	0.015043
3	52 275	48	3	0.0001	64	0.015780
4	68 547	48	4	0.0001	64	0.016139
5	120 067	64	4	0.0001	64	0.017517

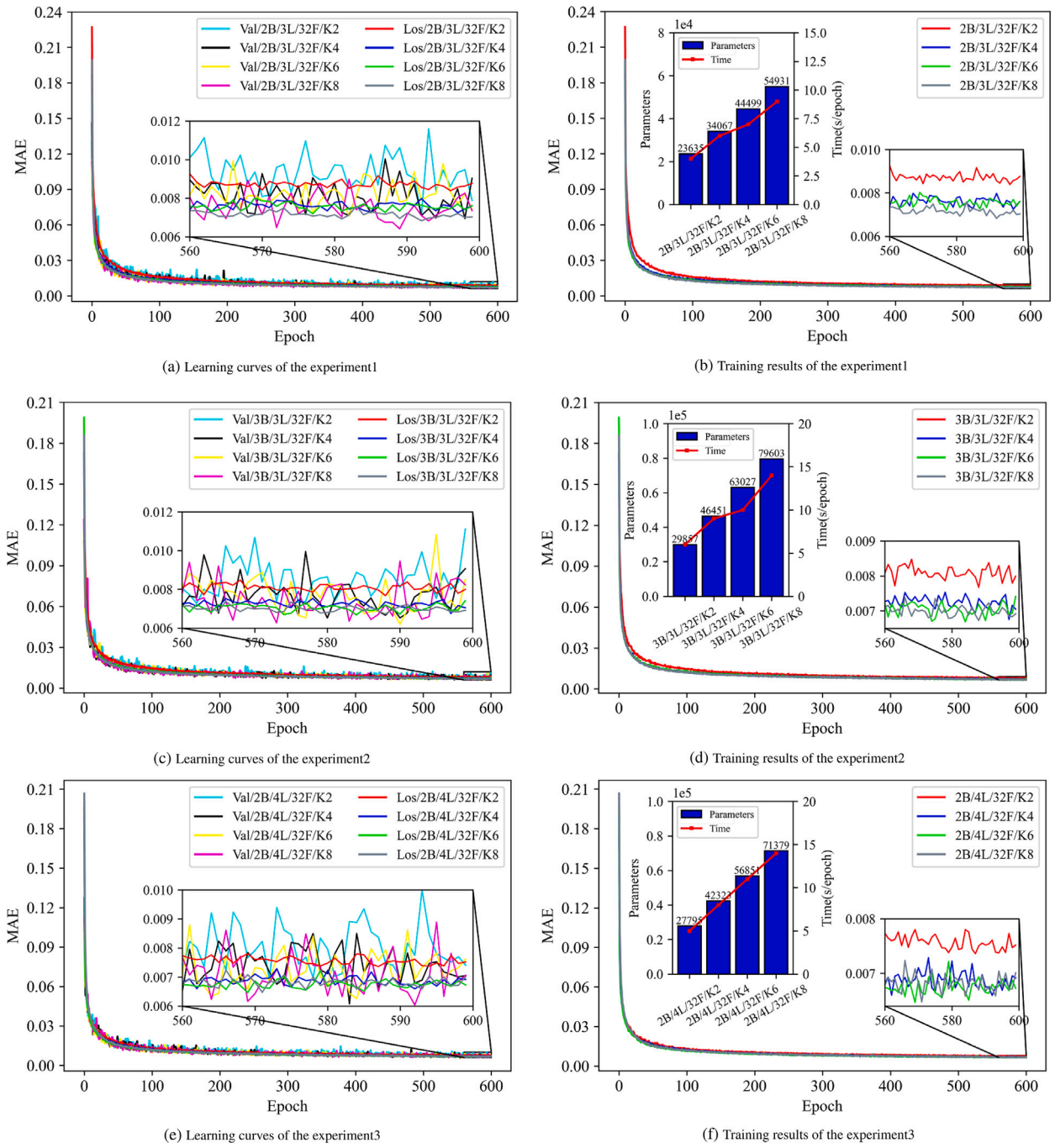


Fig. 39. Learning curve with different kernel size in one filter.

Table 12
The results of Bayesian search trial 3 with 3 TCN blocks and 3 DCC layers in one block.

Trial	Parameters	Filters	Kernel size	Learning rate	Batch size	Loss value
1	40 659	48	2	0.001	256	0.012501
2	136 515	64	4	0.0001	64	0.012586
3	103 555	64	3	0.0001	64	0.012767
4	59 235	64	2	0.0001	64	0.014421
5	18 915	32	2	0.001	256	0.014540

Table 13
The results of Bayesian search trial 4 with 3 TCN blocks and 4 DCC layers in one block.

Trial	Parameters	Filters	Kernel size	Learning rate	Batch size	Loss value
1	140 611	64	3	0.001	256	0.010109
2	52 627	48	2	0.001	256	0.011159
3	95 363	64	2	0.001	256	0.011261
4	185 859	64	4	0.0001	64	0.011490
5	140 611	64	3	0.0001	64	0.011858

References

Abueidda, D.W., Almasri, M., Ammourah, R., Ravaioli, U., Jasiuk, I.M., Sobh, N.A., 2019. Prediction and optimization of mechanical properties of composites using convolutional neural networks. Compos. Struct. 227, 111264.

Abueidda, D.W., Koric, S., Sobh, N.A., Sehitoglu, H., 2021. Deep learning for plasticity and thermo-viscoplasticity. Int. J. Plast. 136, 102852.

Ali, U., Muhammad, W., Brahme, A., Skiba, O., Inal, K., 2019. Application of artificial neural networks in micromechanics for polycrystalline metals. Int. J. Plast. 120, 205–219.

Alipour, M., Lashkari, A., 2018. Sand instability under constant shear drained stress path. Int. J. Solids Struct. 150, 66–82.

Anandarajah, A., 2008. Multi-mechanism anisotropic model for granular materials. Int. J. Plast. 24 (5), 804–846.

Bai, S., Kolter, J.Z., Koltun, V., 2018. An empirical evaluation of generic convolutional and recurrent networks for sequence modeling. arXiv preprint arXiv:1803.01271.

Bréchet, Y., Louchet, F., 1988. Localization of Plastic Deformation, Vol. 3. Trans Tech Publ.

Cybenko, G., 1989. Approximation by superpositions of a sigmoidal function. Math. Control Signals Systems 2 (4), 303–314.

Das, S.K., Das, A., 2019. Influence of quasi-static loading rates on crushable granular materials: A DEM analysis. Powder Technol. 344, 393–403.

Desrues, J., Andò, E., 2015. Strain localisation in granular media. C. R. Phys. 16 (1), 26–36.

Ezzein, F.M., Bathurst, R.J., 2011. A transparent sand for geotechnical laboratory modeling. Geotech. Test. J. 34 (6), 590–601.

Fernández, M., Rezaei, S., Mianroodi, J.R., Fritzen, F., Reese, S., 2020. Application of artificial neural networks for the prediction of interface mechanics: a study on grain boundary constitutive behavior. Adv. Model. Simul. Eng. Sci. 7 (1), 1–27.

Fukuoka, R., Suzuki, H., Kitajima, T., Kuwahara, A., Yasuno, T., 2018. Wind speed prediction model using LSTM and 1D-CNN. J. Signal Process. 22 (4), 207–210.

Ghaboussi, J., Sidarta, D., 1998. New nested adaptive neural networks (NANN) for constitutive modeling. Comput. Geotech. 22 (1), 29–52.

Gorji, M.B., Mozaffar, M., Heidenreich, J.N., Cao, J., Mohr, D., 2020. On the potential of recurrent neural networks for modeling path dependent plasticity. J. Mech. Phys. Solids 143, 103972.

Gu, G.X., Chen, C.-T., Buehler, M.J., 2018. De novo composite design based on machine learning algorithm. Extreme Mech. Lett. 18, 19–28.

Guo, N., Zhao, J., 2014. A coupled FEM/DEM approach for hierarchical multiscale modelling of granular media. Internat. J. Numer. Methods Engrg. 99 (11), 789–818.

He, X., Wu, W., Wang, S., 2019. A constitutive model for granular materials with evolving contact structure and contact forces—Part I: framework. Granul. Matter 21 (2), 16.

Heider, Y., Wang, K., Sun, W., 2020. SO (3)-invariance of informed-graph-based deep neural network for anisotropic elastoplastic materials. Comput. Methods Appl. Mech. Engrg. 363, 112875.

Hewage, P., Behera, A., Trovati, M., Pereira, E., Ghahremani, M., Palmieri, F., Liu, Y., 2020. Temporal convolutional neural (TCN) network for an effective weather forecasting using time-series data from the local weather station. Soft Comput. 24 (21), 16453–16482.

Hornik, K., Stinchcombe, M., White, H., 1989. Multilayer feedforward networks are universal approximators. Neural Netw. 2 (5), 359–366.

Jeong, H., Signetti, S., Han, T.-S., Ryu, S., 2018. Phase field modeling of crack propagation under combined shear and tensile loading with hybrid formulation. Comput. Mater. Sci. 155, 483–492.

Lara-Benítez, P., Carranza-García, M., Luna-Romera, J.M., Riquelme, J.C., 2020. Temporal convolutional networks applied to energy-related time series forecasting. Appl. Sci. 10 (7), 2322.

Lea, C., Flynn, M.D., Vidal, R., Reiter, A., Hager, G.D., 2017. Temporal convolutional networks for action segmentation and detection. In: Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition. pp. 156–165.

Lefik, M., Boso, D., Schrefler, B., 2009. Artificial neural networks in numerical modelling of composites. Comput. Methods Appl. Mech. Engrg. 198 (21–26), 1785–1804.

Liu, G., Li, Q., Msekh, M.A., Zuo, Z., 2016. Abaqus implementation of monolithic and staggered schemes for quasi-static and dynamic fracture phase-field model. Comput. Mater. Sci. 121, 35–47.

Ma, G., Guan, S., Wang, Q., Feng, Y., Zhou, W., 2022. A predictive deep learning framework for path-dependent mechanical behavior of granular materials. Acta Geotech. 1–16.

Mozaffar, M., Bostanabad, R., Chen, W., Ehmann, K., Cao, J., Bessa, M., 2019. Deep learning predicts path-dependent plasticity. Proc. Natl. Acad. Sci. 116 (52), 26414–26420.

Qu, T., Di, S., Feng, Y., Wang, M., Zhao, T., 2021a. Towards data-driven constitutive modelling for granular materials via micromechanics-informed deep learning. Int. J. Plast. 103046.

Qu, T., Di, S., T Feng, Y., Wang, M., Zhao, T., Wang, M., 2021b. Deep learning predicts stress–strain relations of granular materials based on triaxial testing data. CMES Comput. Model. Eng. Sci. 128 (LA-UR-21-21583).

Qu, T., Feng, Y., Wang, M., 2021c. An adaptive granular representative volume element model with an evolutionary periodic boundary for hierarchical multiscale analysis. Internat. J. Numer. Methods Engrg. 122 (9), 2239–2253.

Qu, T., Feng, Y., Wang, Y., Wang, M., 2019a. Discrete element modelling of flexible membrane boundaries for triaxial tests. Comput. Geotech. 115, 103154.

Qu, T., Wang, S., Fu, J., Hu, Q., Zhang, X., 2019b. Numerical examination of EPB shield tunneling–induced responses at various discharge ratios. J. Perform. Constr. Facil. 33 (3), 04019035.

Shaoheng Guan, Tongming Qu, Feng, Y., Gang Ma, Wei Zhou, 2022. A machine learning based multi-scale computation framework for granular materials. Acta Geotech. (in press).

Ueda, K., Iai, S., 2019. Constitutive modeling of inherent anisotropy in a strain space multiple mechanism model for granular materials. Int. J. Numer. Anal. Methods Geomech. 43 (3), 708–737.

Vlassis, N.N., Sun, W., 2021. Sobolev training of thermodynamic-informed neural networks for interpretable elasto-plasticity models with level set hardening. Comput. Methods Appl. Mech. Engrg. 377, 113695.

Wang, H.-L., Cui, Y.-J., Lamas-Lopez, F., Calon, N., Saussine, G., Dupla, J.-C., Canou, J., Aimedieu, P., Chen, R.-P., 2018. Investigation on the mechanical behavior of track-bed materials at various contents of coarse grains. Constr. Build. Mater. 164, 228–237.

Wang, K., Sun, W., 2019. Meta-modeling game for deriving theory-consistent, microstructure-based traction–separation laws via deep reinforcement learning. Comput. Methods Appl. Mech. Engrg. 346, 216–241.

Wang, X., Zhang, H., Yin, Z.-Y., Su, D., Liu, Z., 2022. Deep-learning-enhanced model reconstruction of realistic 3D rock particles by intelligent video tracking of 2D random particle projections. Acta Geotech. 1–24.

Wei, L., Yang, J., 2014. On the role of grain shape in static liquefaction of sand–fines mixtures. Géotechnique 64 (9), 740–745.

Yang, C., Kim, Y., Ryu, S., Gu, G.X., 2020a. Prediction of composite microstructure stress-strain curves using convolutional neural networks. Mater. Des. 189, 108509.

Yang, Z., Liao, D., Xu, T., 2020b. A hypoplastic model for granular soils incorporating anisotropic critical state theory. Int. J. Numer. Anal. Methods Geomech. 44 (6), 723–748.

Zhang, P., Yin, Z.-Y., 2021. A novel deep learning-based modelling strategy from image of particles to mechanical properties for granular materials with CNN and BiLSTM. Comput. Methods Appl. Mech. Engrg. 382, 113858.

Zhu, Q., Shao, J.-F., Mainguy, M., 2010. A micromechanics-based elastoplastic damage model for granular materials at low confining pressure. Int. J. Plast. 26 (4), 586–602.