Modelling the mechanical behaviour of soils using machine learning algorithms with explicit formulations: a comparative study

Pin ZHANG[1], Zhen-Yu YIN[1,*], Yin-Fu JIN[1,*], Xian-Feng LIU[2]

1 Department of Civil and Environmental Engineering, Hong Kong Polytechnic University, Hung Hom, Kowloon, Hong Kong, China

2 Key Laboratory of High-speed Railway Engineering of Ministry of Education, School of Civil Engineering, Southwest Jiaotong University, Chengdu 610031, China

* Corresponding author: Dr Zhen-Yu YIN, E-mail: zhenyu.yin@polyu.edu.hk; zhenyu.yin@gmail.com; Dr. Yin-Fu JIN, E-mail: yinfu.jin@polyu.edu.hk; Tel: +852 3400 8470; Fax: +852 2334 6389

Abstract: This study systematically presents the application of machine learning (ML) algorithms for constructing a constitutive model for soils. A genetic algorithm was integrated with ML algorithms to determine the global optimum model, and the *k*-fold cross-validation method was used to enhance the models' robustness. The modelling performance of three typical ML algorithms with formulations explicitly expressed (i.e., back-propagation neural network [BPNN], extreme learning machine [ELM] and evolutionary polynomial regression [EPR]) were comprehensively compared. The effect of using the total or incremental stress–strain strategy on the construction of ML-based soil models was investigated through synthetic and experimental data. All results indicate that a BPNN-based constitutive model using the incremental stress–strain strategy performs best in modelling the mechanical behaviour of soils in terms of interpolation and extrapolation abilities, followed by ELM and then EPR.

Keywords: Constitutive model; Soils; Neural network; Extreme learning machine; Evolutionary computation; Optimization

1. Introduction

Experimental investigations show that the mechanical behaviour of soils is very complicated, involving elements such as state-dependence [50], contraction-dilation [51], anisotropy [66], destructuration [39, 68], stress-path dependence [20], time-dependence [69], and non-coaxiality [53]. Accurate description of such soil behaviours is vitally important in engineering practice [31, 43, 59, 80]. To describe such soil behaviours, numerous constitutive models have been developed during the past few decades. These models can be classified as (1) linear-elastic, (2) elastic perfectly plastic (such as the Mohr-Coulomb model), (3) nonlinear (such as the hardening soil [56] and nonlinear Mohr-Coulomb [26] models, (4) critical state–based advanced (such as the modified cam-clay model [47], Nor-Sand model [23], CSAM model [76], Severn–Trent model [10], UH models [62-64], SANISAND model [52], SIMSAND model [24-26] and ANICREEP model [74]), hypoplasticity [34, 40, 57, 58] and (5) micromechanical models [4, 60, 70-73]. The last two categories are usually called advanced soil models [26, 74]. However, traditional soil models have three main disadvantages in modelling soil behaviours: (1) Most constitutive models are developed based on certain assumptions [65, 66, 69] (e.g., the associated or non-associated flow rule, non-coaxiality), (2) each model is suitable only for a specific type of soil or specific stress-paths and (3) although the mathematical formulas in a constitutive model are developed based on some theories (e.g., elastoplasticity theory) or derived from finite experimental data (e.g., the critical state line from triaxial tests), the formula's form gives good accuracy for selected tests, but at the same time limits the model's simulation ability for other stress paths. For example, the Modified Cam-

Clay (MCC) was derived from the triaxial tests of saturated remoulded clay, and thus the MCC model is difficult to predict other kind of tests or other soils. In addition, the mathematical formulas become increasingly complicated when involving many parameters, resulting in difficulties of parameter identification and further limiting their engineering applications.

Soil normally exhibits highly nonlinear characteristics. To simulate such characteristics, ML algorithms are very powerful and can thus be employed as an alternative way to construct data-driven constitutive models. ML algorithms have three following advantages in developing soil models: (1) ML algorithms can directly extract the stress–strain relationship from the experimental data without making any assumptions [8, 9, 11]. More stable and accurate results can be obtained by ML based models if the physical mechanism is implied in training data and/or incorporated into the training process; (2) ML algorithms have a strong ability to capture complicated non-linear relationships [1, 5, 16] and (3) the prediction accuracy of ML-based models can rise with the increase of experimental datasets. Numerous ML-based soil models have already been developed, and they can be categorized according to the model's training strategy, whether (1) training models using the total values of stress and strain or (2) training models in incremental form [36]. ~~Because ML algorithms can directly learn the stress–strain relationship from the experimental data, the increment-based training method might not be better at modelling the stress–strain under a simple path.~~ However, up to now there is no comparative study to discuss which one is more suitable to develop ML based model for describing soil behaviours. Accordingly, the performance of two stress–strain strategies in developing ML-based constitutive models deserves investigation.

To construct an ML-based soil model, a large number of ML algorithms can be adopted, such as a back-propagation neural network (BPNN) [2, 15, 18, 44, 45, 55], evolutionary neural network (ENN) [30], recurrent neural network (RNN) [46, 81], support vector machines (SVMs) [33], evolutionary polynomial regression (EPR) [7, 22, 42] and genetic programming (GP) [3]. To find an ML algorithm that efficiently models soils' stress–strain relationship, a performance comparison of different ML algorithms is needed. A comprehensive process for constructing ML-based models consists of the training, validation and testing phases. The validation phase is used to examine the robustness of trained ML models before evaluating model performance via the test set. However, existing ML-based soil constitutive models did not include the validation phase. The robustness of these ML models cannot be guaranteed. The testing phase in most ML-based soil constitutive models is the validation phase. Furthermore, the performance of an ML-based constitutive model is usually evaluated using testing data that belong to the same distribution as the training data (interpolation ability) but that are merely evaluated based on the unseen data (extrapolation ability). All these problems are worth investigating.

This study aims to comprehensively demonstrate the process of constructing an ML-based constitutive model. To this end, three representative ML algorithms that can give explicit expression – BPNN, extreme learning machine (ELM) and EPR – were selected. The *k*-fold cross-validation method was employed in the validation phase to enhance the robustness of ML-based constitutive models. A genetic algorithm (GA) was used to optimize parameters for developing the global optimum model. A synthetic database based on a simple soil model was first built. This model uses the real capabilities of BPNN, ELM and EPR to model soil behaviours for comparison, including interpolation and extrapolation abilities and the effects of the total and incremental stress–strain strategies. Finally, BPNN's, ELM's and EPR's capacities for modelling soil behaviour are further examined using real test data for Kaolinite clay.

2. Methodology of machine learning

2.1 Back-propagation neural network

A BPNN is a feedforward neural network characterized by propagation of errors from the output layer to find a set of weights and biases able to ensure that the output value of the network is identical to the actual output value [48]. A BPNN includes an input layer, any number of hidden layers and an output layer. The performance of BPNN is mainly affected by its framework, i.e. the number of hidden layers and hidden neurons. Based on a given framework, the purpose of other hyper-parameters such as activation function is to further improve the training efficiency or optimize the model. Considering that this study focuses on simulating mechanical behaviours of soils using ML algorithms, the deep investigation regarding the effect of each hyper-parameter on the model performance is not conducted. Herein, the optimum framework of BPNN based model is carefully investigated, whereas remaining hyper-parameters are set as the default value in Matlab toolbox. ~~,The hyper-parameters of the BPNN are the number of hidden layers and hidden neurons, which affect the BPNN's performance.~~ Once the hyper-parameters are determined, weighting and bias values can be calculated by gradient descent or optimization algorithms. Figure 1(a) illustrates a typical BPNN with one hidden layer. Taking the numbers of inputs and hidden and output neurons to be *r*, *p* and *q*, respectively, and assuming that there are *n* datasets in the training set, the output of the hidden and output layers can be expressed as

$$\mathbf{H} = f\left(\mathbf{WX} + \theta\right) \tag{1}$$

$$\mathbf{O} = g\left(\mathbf{VH} + \theta_{\mathbf{o}}\right) \tag{2}$$

where **X** **=** matrix of input variables ($r \times n$); **H** = matrix of the hidden layer output ($p \times n$); **O** = matrix of output variables ($q \times n$); **W**, **V** = weights matrix on the connections between input and hidden neurons ($p \times r$) and between hidden and output neurons ($q \times p$), respectively; $\boldsymbol{\theta}$, $\boldsymbol{\theta_o}$ = bias vectors on the connections between input and hidden neurons ($p \times 1$) and between hidden and output neurons ($q \times 1$), respectively; and *f*, *g* = activation functions in hidden and output layers, respectively~~, which are *tansig* and *purlin* in this study and can be formulated as follows:~~. The default activation functions in the hidden and output layers in the Matlab toolbox are *tanh* and *purlin*, respectively. Such two activation functions are used in this study, because *tanh* (see Eq. [3]) as the activation function for the hidden layer tends to show excellent performance in the shallow BPNN [6, 77, 78], and *purlin* (see Eq. [4]) is a activation function used in the output layer for regression problem.

$$\text{tansig}: f\left(x\right) = \frac{2}{1+e^{-2x}} - 1 \tag{3}$$

$$purlin: g\left(x\right) = x \tag{4}$$

~~Note that the training set used in the BPNN has been normalized into an interval (−1, 1) using Eq. (5) because doing so can eliminate the effect of different magnitudes of input~~

~~variables on the model’s performance and can clearly reduce computational costs:~~

~~$x_{norm} = \frac{x - x_{\min}}{x_{\max} - x_{\min}}\left(\bar{x}_{\max} - \bar{x}_{\min}\right) + \bar{x}_{\min}$ (5)~~

~~where x = actual value of input variables, $x_{\min}$ = minimum value of input variables and $x_{\max}$ = maximum value of input variables. $\bar{x}_{\min}$ = – 1; $\bar{x}_{\max}$ = 1.~~

2.2 Extreme learning machine

An ELM is a type of feedforward neural network characterized by a single hidden layer (see Fig. 1[b]). The hyper-parameters in the ELM equal the number of hidden neurons. The weights of the input layer and the biases of the hidden layer are assigned randomly, and the weights of the hidden layer (**β**) are determined analytically through a simple generalized inverse operation of the hidden layer output matrices [21], as shown in Eqs. (5)–(6), making the ELM’s learning speed thousands of times faster than seen in traditional feedforward networks:

$$\mathbf{H} = f\left(\mathbf{WX} + \theta\right) \tag{5}$$

$$\min_{\boldsymbol{\beta}}\left\|\mathbf{H}\boldsymbol{\beta} - \mathbf{O}\right\| \tag{6}$$

where **X** = matrix of input variables (r×n), **H** = matrix of the hidden layer output (p×n), **O** = matrix of output variables (q×n), **W** = weights matrix on the connections between input and hidden neurons (p×r), $\boldsymbol{\theta}$ = the bias vector of the connections between input and hidden neurons (p×1), **β** = the weight matrix connecting the hidden and the output layers (q×p) and f = the activation function in the hidden layer. For a fair comparison with BPNN, activation function *tanh* is also used in ELM.~~—*tansig* in this study. As with the BPNN method, the training set used in the ELM also needs to be normalized into the interval (−1, 1) using Eq. (5).~~

2.3 Evolutionary polynomial regression

EPR is a genetic programming method characterized by the modelling of a system using a mathematical expression in the form of polynomial structures. Constructing an EPR-based model consists of two phases: (1) structure identification and (2) parameter estimation [13]. During the first phase, optimization algorithms such as the genetic algorithm and particle swarm optimization are used to search for symbolic structures – that is, to determine the exponent matrix. During the second phase, the parameters' values are estimated by solving a least squares (LS) linear problem. Compared with BPNN and ELM, the training set in the EPR does not require normalization. A typical EPR expression can be formulated as

$$\mathrm{y} = \sum_{j=1}^{m} F\left(\mathbf{X}, f_j\left(\mathbf{X}\right), a_j\right) + a_0 \tag{7}$$

where $\boldsymbol{y}$ = predicted output, $\mathbf{X}$ = matrix of input variables, F = a function constructed by the process, f_{j} ($\mathbf{X}$) = jth transformed variable, a_j = an adjustable parameter for the jth term and a_0 = an optional bias. f_{j} ($\mathbf{X}$) is determined by the optimization algorithm, and a_j and a_0 are determined by the LS.

The EPR's key objective is to identify the number of transformed variables and a combination of vectors of independent input variables. Herein, the transformed variable is obtained via

$$f_j\left(\mathbf{X}\right) = \boldsymbol{x}_1^{\mathbf{ES}(j,1)} \bullet \ldots \bullet \boldsymbol{x}_i^{\mathbf{ES}(j,\ i)} \bullet \ldots \bullet \boldsymbol{x}_k^{\mathbf{ES}(j,\ k)} \tag{8}$$

where $\boldsymbol{x}_i$ = *i*th input variable, *k* = a total number of input variables and $\mathbf{ES}_{\mathrm{m\times k}}$ = exponent matrix.

2.4 Genetic algorithm

A genetic algorithm (GA) is a meta-heuristic optimization algorithm inspired by natural evolution [19]. It has been extensively employed in geotechnical engineering for tasks such as identification constitutive models' parameters [24, 26, 67, 75], model selection [25], slope [37, 54], embankment [14, 41], tunnelling [35, 38], pile foundation [27, 29] and excavation [28]. In this study, the GA was selected to optimize weights and biases in BPNN and ELM algorithms and to search for symbolic structures in the EPR algorithm. In GA, a population of individuals is first generated. A chromosome based on a coding scheme (real-coded GA) is then employed to represent each individual. After calculating the fitness value of each individual, the best individual having the lowest fitness value in the population is selected and then evolves through crossover and mutation operations to generate a new population. The process continues until it satisfies the termination criterion – that is, whether it reaches the maximum generation. Meanwhile, the fitness value converges at a constant value.

2.5 K-fold cross-validation

Three phases are involved in the integrated process of constructing an ML model: training, validation and testing. The validation phase seeks to improve the robustness of the training model and avoid overfitting. Currently, the *k*-fold cross-validation (CV) method is widely used to validate models [49]. In this method, the original training set is randomly divided into *k* sub-datasets. Herein, *k*–1 sub-datasets, which form a new sub-training set, are employed to train models, and the performance of the trained model is validated by the remaining sub-dataset. Each sample in the training set thus has an opportunity to train and validate models. Because [32] demonstrated that ~~K~~*k* tends to be set at 10, the ten-fold CV method was used in this study.

At each round, the ML model with a fixed set of hyper-parameters was trained ten times based on nine sub-training sets, ~~and~~thereafter the performance of this ML model was evaluated by the mean value of the sum of squared errors (MSSE) for the remaining sub-dataset. Such process eliminates the effect of allocation of training and testing sets on the model performance. Meanwhile the model performance evaluated by the increasing *k* validation subsets instead of only one validation set can prevent overfitting problem [79]. ~~It is defined as~~Therefore, the fitness function in the GA~~, which~~ can be expressed as

$$Fitness = \frac{\sum_{i=1}^{m}\left(\bar{y}_i - y_i\right)^2}{k} \tag{9}$$

where $\bar{y}_i$ = predicted output, y_i = actual output, m = the number of datasets in the remaining sub-dataset and k = the number of CV sets.

2.6 Evaluation indicators

Two commonly used evaluation indicators – mean absolute error (MAE) and mean absolute percentage error (MAPE) – were used to evaluate the performance of ML models in this study. The combination of MAE and MAPE helps overcome the deficiencies of both, so that both are used extensively to evaluate model performance [5, 61]. Low values of these two indicators indicate that a model has excellent performance. The expression of MAE and MAPE can be obtained by

$$MAE = \frac{1}{n}\sum_{i=1}^{n}|r_i - p_i| \tag{10}$$

$$MAPE = \frac{1}{n}\sum_{i=1}^{n}\left|\frac{r_i - p_i}{r_i}\right| \times 100\% \tag{11}$$

where r = actual output value, p = predicted output value and n = the total number of datasets.

2.7 Model framework

Figure 2(a) presents the flowchart for constructing a ML-based constitutive model. This type of data-driven model starts from the collection of datasets with which to form a database. In the ML domain, 80% (for training the model) and 20% (for testing the model) is a widely acknowledged scheme for data separation ratio in the community. Such separation ratio can ensure the ML based model being well trained and tested, which has been theoretically proved [12]. ~~Of the data,~~Therefore, 80 % are used to train the model and 20 % are used to test it in this study. The total or incremental stress–strain strategy is selected beforehand; thereafter, the corresponding features or input variables can be determined. At the next step, the ten-fold cross-validation method is used to divide the training set into ten subsets for training and validating models. At each round, GA is employed to identify the general parameters of ML algorithms. The hyper-parameters are determined by trial and error, with the corresponding hyper-parameters of the model that generates the lowest fitness value regarded as the optimal hyper-parameter. After determining three optimal constitutive models based on BPNN, ELM and EPR, their performance is compared using the test set.

Figures 2(b) and 2(c) illustrate the schematic view of the total and incremental stress–

strain strategy, respectively. In the total stress–strain strategy, the stress in the *i*th step is affected only by the strain at the *i*th step and by the physical parameters (see Eq. [12]). In the incremental stress–strain strategy, the stress in the *i*th step is affected by the strain, the stress at the (*i*–1)th step, the strain increment at the *i*th step and the physical parameters (see Eq. [13)]):

$$\sigma^{i} = f\left(\mathrm{X}, \varepsilon^{i}\right) \tag{12}$$

$$\sigma^{i} = f\left(\mathrm{X}, \sigma^{i-1}, \varepsilon^{i-1}, \Delta\varepsilon^{i}\right) \tag{13}$$

where $\boldsymbol{X} = [x_1, x_2, \ldots, x_r]$, the vector of independent variables; σ^i, $\sigma^{i\text{-}1}$ = stress at the *i*th and (*i*–1)th steps; ε^i, ε^{i-1} = strain at the *i*th and (*i*–1)th steps; $\Delta\varepsilon^i$ = axial strain increment at the *i*th step; and *f* = formulation of stress–strain relationship, as determined by the ML algorithms in this study.

It should be noted that in the incremental stress–strain strategy, the predicted stress at the *i*th step needs to update the input stress variable in real time to predict the stress at the (*i*+1)th step. In addition, the strain ε at the (*i*+1)th step is updated by the following:

$$\varepsilon^{i} = \varepsilon^{i-1} + \Delta\varepsilon^{i} \tag{14}$$

To eliminate the effect of scales of parameters on the model performance and improve convergence, all datasets need to be preprocessed. Considering the distribution of parameters used in this study is different, thereby Minmax normalization method instead of standardization method is used in this study, as presented in Eq. [15].

$$x_{norm} = \frac{x - x_{\min}}{x_{\max} - x_{\min}}\left(\bar{x}_{\max} - \bar{x}_{\min}\right) + \bar{x}_{\min} \tag{15}$$

where x = actual value of input variables, $x_{\min}$ = minimum value of input variables and $x_{\max}$ = maximum value of input variables. $\bar{x}_{\min} = -1$; $\bar{x}_{\max} = 1$.

3 ML–based constitutive models using synthetic data

3.1 Synthetic data by a simple soil model

To comprehensively compare the performance of three ML algorithms and two modelling strategies ~~when~~ on developing constitutive models, a simple sand shear constitutive model was first used to generate synthetic datasets in this study (see Eq. [16]). The ~~three~~ purpose of ML-based constitutive models developed based on~~were compared using~~ synthetic datasets ~~a theoretical function~~ rather than directly based on the experimental data is to eliminate the interference of experimental and measurement errors ~~related to~~ on the mapping capability of ML algorithms. Moreover, the experimental data tend to be limited and insufficient for comparison of ML algorithms' performance, whereas the data can be generated infinitely by a theoretical function.

$$\tau = \sigma_{n0} \frac{\mu\gamma}{1/G + \gamma} \tag{16}$$

where σ_{n0} = vertical stress; τ = shear stress; γ = shear strain; G = shear modulus, 1000 kPa in this study; and μ = friction angle, tan(π/6) in this study.

A total of fourteen curves were generated to develop ML-based constitutive models. Herein, the axial strain γ ranges from 0 % to 10 %, and a fixed set of axial strain increment $\Delta\gamma$, including 0.01 %, 0.05 %, 0.1 %, 0.15 % and 0.2 %, was chosen consistently for ten curves. Each curve consists of 91 data points. Nine curves (σ_{n0} = 25, 50, 100, 200, 250, 300, 400, 500 and 600 kPa) with a total of 819 data points were employed to train the ML-based constitutive models, and the remaining five curves (σ_{n0} = 15, 150, 350, 650 and 700 kPa) were used to test the models.

According to the stress–strain strategy, as mentioned in Eqs. (13)–(14), the vector ***X*** of independent variables in this soil model is σ_{n0}. As a result, the total and incremental stress–strain strategies have two and four input variables, respectively. Both have an output variable. The corresponding total and incremental stress–strain strategy can be written as

$$\tau^{i} = f\left(\sigma_{n0}, \gamma^{i}\right) \tag{17}$$

$$\tau^{i} = f\left(\sigma_{n0}, \tau^{i-1}, \gamma^{i-1}, \Delta\gamma^{i}\right) \tag{18}$$

where the definitions of τ, γ and $\Delta\gamma$ are similar to those of σ, ε and $\Delta\varepsilon$ in Eqs. (13)–(14).

3.2 Determination of parameters in ML algorithms

The parameters to be determined in the ML algorithms include hyper-parameters and general parameters. Table 2 presents the hyper-parameters in the BPNN, ELM and EPR models. The trial-and-error method was employed in this study to determine the optimal hyper-parameter of ML algorithms. A single-layer BPNN was used to construct constitutive models in this study, there being only several input variables. A single hidden layer is sufficient to capture the stress–strain relationship. Table 3 summarizes several methods of determining the optimal number of hidden neurons. The optimal number of hidden neurons ranged from one to five in the total stress–strain strategy and from one to ten in the incremental stress–strain strategy. Because there is no method for determining the optimal number of transformed terms in the ELM and EPR, the ranges of hidden neurons and transformed terms in these two algorithms increase continuously until the number of hidden neurons and transformed terms cannot improve the model’s performance. In this way the ranges of hyper-parameters in three ML algorithms can be determined, as Table 2 shows.

In addition to the hyper-parameters, the weights and biases in the BPNN and ELM as well as the exponent matrix in the EPR were determined using the optimization algorithm GA in this study, guaranteeing that a global optimum constitutive model can be obtained. Note that the values of exponents must be non-negative in the EPR algorithm, because the datasets include the initial stress–strain stage (0, 0); indeed, negative exponents are wrong in this condition. The values of exponents were thus limited to [0, 1, 2, 3]. Table 4 presents the parameter values in the GA. Note that BPNN and EPR are set to a maximum of 500 generations, whereas for the ELM, because of its different convergence rate, the figure is 5000.

3.3 Results of the validation set

The training model seeks to determine the optimal hyper-parameters in each ML algorithm, and the model’s performance with a fixed set of hyper-parameters is evaluated by the ten CV sets. The hyper-parameters of an ML-based constitutive model that produces the lowest convergent fitness value are optimal. Figure 3 presents the evolution of fitness value generated by three types of ML-based constitutive models using the total stress–strain strategy. It can be clearly observed that the convergence rates of BPNN and EPR are much faster than that of ELM. The fitness value roughly holds steady when the generation exceeds 350 and 200 in BPNN and EPR, respectively, whereas the fitness value remains roughly constant when the generation reaches 4000 in ELM. From the perspective of the convergent fitness value, as shown in Figure 3, the optimal numbers of hidden neurons in BPNN and ELM are four and eight, respectively, and the optimal number of transformed terms in EPR is eleven.

The evolution of fitness value generated by three ML algorithms using the incremental stress–strain strategy is shown in Figure 4. Overall, the convergence rate of the three types of ML-based constitutive models using the incremental stress–strain strategy is faster than when using the total stress–strain strategy. The fitness value roughly holds steady when the generation reaches 250, 1000 and 100 in the BPNN, ELM and EPR, respectively. From the perspective of the convergent fitness value, as shown in Figure 4, the optimal numbers of hidden neurons in BPNN and ELM are four and ten, respectively, and the optimal number of transformed terms in EPR is eleven. Note that the optimal fitness values are much less than those yielded using the total stress–strain strategy.

3.4 Results of the training set

The optimal hyper-parameters of the three ML algorithms using two stress–strain strategies are determined as heretofore mentioned. Accordingly, three optimal ML-based constitutive models of each stress–strain strategy are constructed based on the training set. Figure 5 presents the predicted stress–strain curves using three optimally trained models on the basis of the total stress–strain strategy, compared with the measured curves. It is clear that the predicted results of BPNN show perfect agreement with the measured curves, whereas the results predicted by ELM and EPR deviate from the measured curves. In particular, the prediction error for the ELM- and EPR-based constitutive models is much greater at the initial stage (0, 0), and these models also yield a large degree of error at the early stage of stress–strain curves, attributable to the principles of these two algorithms as described in section 3.3.

Figure 6 presents the predicted stress–strain curves using three optimally trained models based on the incremental stress–strain strategy, compared with the measured curves. It can be seen that all three models can accurately capture the stress–strain curves, which indicates that the incremental stress–strain strategy for simulating stress–strain relationship shows a significant improvement.

3.5 Results of the test set

During the last phase, the performance of ML-based constitutive models is evaluated against the test set, with σ_{n0} in the training set ranging from 50 to 600 kPa. Generally, test datasets are taken from within the range of training datasets, so that test sets with σ_{n0} = 150 and 350 kPa are taken into consideration. To investigate the ability of ML-based constitutive models to extrapolate beyond the range of training datasets, test sets for which σ_{n0} = 15, 650 and 700 kPa are also conducted in this study. Table 5 summarizes the values of indicators for these five test sets. For the interpolated test sets, Figure 7 presents the results of simulation using three optimal ML-based constitutive models based on the total stress–strain strategy. The predicted stress–strain curve using BPNN largely agrees with the measured curve, and the corresponding MAE and MAPE values are also lower than those produced by ELM and EPR. Notably, ELM and EPR cannot accurately predict initial stress when strain equals zero, and ELM- and EPR-based constitutive models cannot accurately predict the evolution of stress with increases in strain. Figure 8 presents the results of simulation using three optimal ML models based on the incremental stress–strain strategy for the test set. The predicted stress–strain curves using the BPNN-based constitutive model still agree perfectly with the measured curves and outperform the ELM- and EPR-based constitutive models. The performance of the ELM-based constitutive model is clearly better than that of the total stress–strain strategy, and it also accurately captures the evolution of stress. Nevertheless, the change in the performance of EPR-based constitutive model is different from others, perfectly predicting the stress–strain relationship for σ_{n0} = 350 but exhibiting worse performance at predicting the stress–strain relationship for σ_{n0} = 150. Note that prediction performance at the initial stage of ELM- and EPR-based constitutive models is clearly improved from that seen with the total stress–strain strategy. Overall, ML-based constitutive models that use the incremental stress–strain strategy offer reliable performance for interpolated test sets, and a BPNN-based constitutive model exhibits the best performance.

~~For t~~The extrapolated test sets are used to further examine the generalization ability of ML algorithms themselves. Figure 9 presents the results of simulation using three optimal ML-based constitutive models based on the total stress–strain strategy. For σ_{n0} = 650 and 700 kPa, three ML-based constitutive models can still capture the stress–strain relationship. The BPNN-based constitutive model performs perfectly, followed by the EPR and ELM-based constitutive models. However, for σ_{n0} = 15 kPa, the predicted stress–strain curve by the BPNN-based constitutive model deviates from the actual stress–strain curve. The results of simulation using three optimal ML-based constitutive models based on the incremental stress–strain strategy are shown in Figure 10. In addition to stress–strain curves for σ_{n0} = 650 and 700 kPa, it can be observed that the BPNN-based constitutive model's ability to predict the stress–strain relationship for σ_{n0} = 15 kPa improves significantly. Meanwhile, the performance of the ELM-based constitutive model improves dramatically with lower MAE and MAPE values, whereas the prediction performance of EPR-based constitutive model decreases.

Overall, ML-based constitutive models are better at predicting stress–strain relationships within the range of the training datasets than at extrapolating beyond the range of the training datasets. ML-based constitutive models developed using the incremental stress–strain strategy outperform those developed using the total stress–strain strategy. A BPNN-based constitutive model developed using the incremental stress–strain strategy is thus recommended for describing the stress–strain relationship, because this model makes highly accurate predictions capturing the stress–strain relationship for the interpolated and extrapolated test sets.

4. ML–based constitutive models using real data

4.1 Database

To investigate ML-based constitutive models' ability to predict soil behaviour in engineering practice, this study uses datasets from twelve sets of triaxial compression shear tests conducted by [17] on Kaolinite clays having various over-consolidation ratios (OCRs). The results of shear and void ratio behaviour are collected as shown in Figure 11. Herein, datasets from nine tests having OCRs of 1, 2, 2.25, 2.5, 2.7, 4, 5, 10 and 20 are used to train the model, and the remaining three, with OCRs of 3, 8 and 50 kPa, are used to test it.

4.2 Selection of a simulation strategy

According to previous comparisons, a BPNN algorithm that integrates the incremental stress–strain strategy is used to model Kaolinite clays' behaviour, including that related to deviatoric stress and void ratio. According to the incremental stress–strain strategy seen in Eq. (14), the vector ***X*** of independent variables is the OCR, and there are two output variables: deviatoric stress *q* and void ratio *e*. Accordingly, ML-based Kaolin clays' constitutive models can be obtained by

$$q^i = f\left(p^{i-1}, q^{i-1}, e^{i-1}, \varepsilon_1^{i-1}, \Delta\varepsilon_1^i\right) \tag{19}$$

$$e^i = g\left(p^{i-1}, q^{i-1}, e^{i-1}, \varepsilon_1^{i-1}, \Delta\varepsilon_1^i\right) \tag{20}$$

where p^{i-1}, q^{i-1}, e^{i-1}, ε_1^{i-1} = mean stress, deviatoric stress, void ratio and axial strain at the (*i*–1)th steps, respectively; q^i, e^i, $\Delta\varepsilon_1^i$= deviatoric stress, void ratio and axial strain increment at the *i*th step, respectively; and *f*, *g* = formulations of deviatoric stress–strain and void ratio–strain relationships.

Figure 12 presents the framework of BPNN-based constitutive models for predicting Kaolinite clay behaviour. Note that the predicted deviatoric stress and void ratio at the i^{th} step must update the deviatoric stress and void ratio in real time to predict deviator stress and volumetric strain at the $(i+1)^{\text{th}}$ step. Updates to strain ε at the $(i+1)^{\text{th}}$ step follow Eq. (15). After training, formulations of BPNN-based Kaolinite constitutive models are found and summarized in Appendix.

4.3 Results of simulation

Validation results of these simulations with which to determine the optimal parameters of BPNN-based Kaolinite constitutive models are not presented, but Appendix A presents the formulation of optimal BPNN-based Kaolinite clay constitutive models in detail. It can be observed that the optimal number of hidden neurons in BPNN is eight. Figure 13 presents the results of the training set predicted by the optimal BPNN model, compared with the measured results, showing that BPNN-based constitutive model can accurately capture non-linear deviatoric stress–strain and void ratio–strain relationships.

Figure 14 presents the results of the predicted deviatoric stress–strain and void ratio–strain relationships for the interpolated test set. Because this study uses the recursive simulation strategy, prediction error accumulates gradually with increases in strain. The accumulated error is negligible up to strain of 20 % for simulation of the deviatoric stress–strain relationship. By contrast, the predicted void ratio–strain curve gradually deviates from the accrual curve when strain exceeds 10 %. Overall, the BPNN-based constitutive model better simulates the deviatoric stress–strain relationship, likely because the void ratio–strain relationship is more complicated than the deviatoric stress–strain relationship. Figure 14 shows that deviatoric stress increases monotonically with strain for all experimental tests, whereas the void ratio–strain relationship differs for different OCR values because of the dilatant behaviour associated with high OCR and the contractive behaviour associated with low.

Figure 15 presents the predicted deviatoric stress–strain and void ratio–strain relationships for the extrapolated test set. Absent actual experimental results for OCR = 22.5, 25, 27.5 and

30, the reasonability of predicted curves is referred from the results for OCR = 20 and 50. For OCR = 50, a predicted deviatoric stress–strain curve using the BPNN-based constitutive model agrees well with the actual curve, although no experimental data are available in the training set beyond OCR = 20. Predicted curves for OCR = 22.5, 25, 27.5 and 30 also suggest reasonable trends (with deviatoric stress increasing monotonically with increases in strain and peak deviatoric stress increasing with decreases in OCR), with all results falling into the range OCR = 20 and OCR = 50. However, the BPNN-based constitutive model's predicted void ratio–strain curves obviously deviate from the actual curves when strain exceeds 5 %. The reason prediction of the void ratio–strain relationship is less accurate than prediction of the deviatoric stress–strain relationship has already been stated, but overall, BPNN-based constitutive models do well (in terms of both interpolation and extrapolation) at simulating actual soil behaviour so long as datasets are sufficient. What's more, the ability to obtain simple, explicit function can further extend the application of the BPNN-based constitutive model.

4.4 Comparison of different ML based models

To further compare the performance of BPNN for modelling soil behaviours with ELM and EPR, the latter two ML algorithms are also used to predict the behaviours of Kaolinite clays. It should be noted that the training, testing datasets and also the modelling framework for ELM and EPR are consistent with that used in BPNN. Herein, the optimum number of hidden neurons in ELM based model is identified as 7, and the optimum transformed terms in EPR based model is identified as 8. For brevity, the process for determining the hyper-parameters of ELM and EPR are not presented in detail.

The predicted stress-strain relationships on the testing datasets using the optimum ELM based model are presented in Fig. 16. It can be seen from Figs. 16(a) and (b) that the predicted results on the interpolated datasets show a good agreement with experimental results. In Figs. 16(c) and (d), the strain softening and volumetric contraction are obviously observed from the predicted results on extrapolated datasets, which severely violates the measured results. Such factors indicate the ELM based model can well describe the known soil behaviours, but may be not suitable to predict the soil behaviours on the unseen datasets. Fig. 17 presents the predicted stress-strain relationships on the testing datasets using the optimum EPR based model. Similar to the results presented in Section 3, the predicted error on both interpolated and extrapolated datasets are larger than that generated by BPNN and ELM based models. It indicates the generalization ability of EPR is inferior to the neural networks based algorithms; thereby it has difficulty in modelling complicated soil behaviours. Overall, the generalization ability of BPNN is excellent and it can be used to simulate soil behaviours on both known and unknown datasets.

It should be noted that ML based model is a kind of data-driven model, thereby its application scope can be expanded as the type and information of datasets increase. For example, if the database involves data under unloading and different stress paths, the ML based model can be well trained and used to simulate soil behaviours under such conditions. Otherwise, more effective physical mechanism needs to be added to refine the ML based model. Future work will focus on such issues.

5. Conclusions

Determination of soil constitutive models is vitally important to engineering practice. ML algorithms have been used to model soil behaviour, because ML-based constitutive models are free of assumptions and offer strong non-linear mapping capabilities. This study systematically demonstrated the application of ML algorithms for construction of a soil constitutive model, using three commonly employed ML algorithms able to present an explicit formulation – BPNN, ELM and EPR – to develop models and comprehensively comparing their modelling performance.

A database based on a simple sand shear constitutive model was first built to reflect the three ML algorithms' ability to model soil behaviour, with the intent of eliminating potential interference from noise-corrupted experimental data. Although the ML algorithm can learn directly from data, an incremental stress–strain strategy able to take loading path into consideration was more suitable than the total stress–strain strategy for constructing ML-based constitutive models. ML-based models' hyper-parameters can be determined through trial and error, and the genetic algorithm should identify general hypermeters for developing the global optimum model. The application of *k*-fold cross validation can enhance the robustness of ML-based models, facilitating the application of these ML-based models to engineering practice.

Simulation results for theoretical and experimental data indicated that BPNN-based constitutive models are more stable and accurate, including for interpolation and extrapolation when modelling soil behaviour, than the ELM- and EPR-based constitutive models. Notably, the BPNN-based constitutive model's predictions of deviatoric stress–strain and void ratio–

strain relationships for Kaolinite clay largely agreed with actual experimental data.

Overall, ML-based constitutive models can directly capture non-linear soil behaviour based on limited experimental data without making any assumptions; what’s more, an explicit formulation for the constitutive model can be determined that guarantees application of ML-based constitutive models in numerical analysis and engineering practice. Meanwhile, the model’s performance and scope of application will increase as the database expands, producing more accurate predictions than are to be had from traditional constitutive models.

Acknowledgments

This research was financially supported by the Research Grants Council (RGC) of Hong Kong Special Administrative Region Government (HKSARG) of China (Grant No.: 15209119, PolyU R5037-18F); Joint research project between SiChuan Province and National Universities funded by Science & Technology Department of Sichuan Province (No.2019YFSY0015), and Open research grant of MOE Key Laboratory of High-speed Railway Engineering.

Appendix A. Formulations of BPNN-based Kaolinite constitutive models

$$\mathbf{H} = f\left(\mathbf{WX} + \theta\right) \tag{1A}$$

$$\mathbf{O} = g\left(\mathbf{VH} + \theta_{\mathrm{o}}\right) \tag{2A}$$

where, **X = [*p*, *q*, *e*, *ε*1, Δ*ε*1]**, matrix of input variables; **H** = matrix of the hidden layer output; **O** = [***q***, ***e***], matrix of output variables; *f* = *tansig* formulation; *g* = *purlin* formulation. Herein,

$$\mathbf{W} = \begin{bmatrix} -1.35315 & 1.907756 & 0.915302 & -2.02378 & -2.16087 \\ 2.595596 & -0.32869 & -2.48042 & 2.352501 & -2.32131 \\ -1.36652 & 0.169634 & 3.622397 & -2.44086 & -3.21463 \\ -0.72626 & -0.01880 & -0.22374 & 0.908566 & 1.41438 \\ -0.02852 & 0.013743 & 0.634650 & -0.57525 & -0.11921 \\ -0.13418 & 0.036866 & 0.933846 & -1.36281 & -1.07577 \\ -0.19825 & -0.51040 & -6.04189 & -2.26296 & -1.74243 \\ -0.19073 & 0.055012 & 1.177738 & -0.69086 & -2.03872 \end{bmatrix}$$

$$\theta = \left[2.077058;\ 0.827601;\ 0.480095;\ -0.95727;\ -0.88088;\ -1.21919;\ 0.862033;\ -2.24696\right]$$

$$\mathbf{V} = \begin{bmatrix} -0.1350 & -0.04501 & -2.23939 & -1.11558 & 1.894278 & -1.86286 & -2.12292 & 1.08516 \\ 0.065208 & -0.02499 & -0.12495 & 0.603675 & 0.853076 & -0.0923 & -0.06472 & -0.63716 \end{bmatrix}$$

$$\theta_{\mathrm{o}} = \left[0.31;\ 0.113215\right]$$

References

[1] Badawy MF, Msekh MA, Hamdia KM, Steiner MK, Lahmer T, Rabczuk T (2017) Hybrid nonlinear surrogate models for fracture behavior of polymeric nanocomposites. Probabilist Eng Mech 50:64-75

[2] Basheer IA (2000) Selection of methodology for neural network modeling of constitutive hystereses behavior of soils. Comput-Aided Civ Inf 15:440–458

[3] Cabalar AF, Cevik A (2011) Triaxial behavior of sand–mica mixtures using genetic programming. Expert Syst Appl 38(8):10358-10367

[4] Chang CS, Hicher PY (2005) An elasto-plastic model for granular materials with microstructural consideration. Int J Solids Struct 42(14):4258-4277

[5] Chen RP, Zhang P, Kang X, Zhong ZQ, Liu Y, Wu HN (2019) Prediction of maximum surface settlement caused by EPB shield tunneling with ANN methods. Soils Found 59(2):284–295

[6] Chen RP, Zhang P, Wu HN, Wang ZT, Zhong ZQ (2019) Prediction of shield tunneling-induced ground settlement using machine learning techniques. Front Struct Civ Eng 13(6):1363–1378

[7] Faramarzi A, Javadi AA, Alani AM (2012) EPR-based material modelling of soils considering volume changes. Comput Geosci-UK 48:73-85

[8] Feng XT, Chen BR, Yang CX, Zhou H, Ding X (2006) Identification of visco-elastic models for rocks using genetic programming coupled with the modified particle swarm optimization algorithm. Int J Rock Mech Min 43(5):789-801

[9] Feng XT, Li SJ, Liao HJ, Yang CX (2002) Identification of non-linear stress-strain-time relationship of soils using genetic algorithm. Int J Numer Anal Met 26(8):815-830

[10] Gajo A, Wood M (1999) Severn–Trent sand: a kinematic-hardening constitutive model: the q–p formulation. Geotechnique 49(5):595-614

[11] Gao W, Ge M, Chen D, Wang X (2016) Back analysis for rock model surrounding underground roadways in coal mine based on black hole algorithm. Eng Comput-Germany 32(4):675-689

[12] Gholamy A, Kreinovich V, Kosheleva O (2018) Why 70/30 or 80/20 Relation Between Training and Testing Sets: A Pedagogical Explanation. International Journal of Intelligent Technologies and Applied Statistics 11(2):105–111

[13] Giustolisi O, Savic DA (2006) A symbolic data-driven technique based on evolutionary polynomial regression. J Hydroinform 8(4):235-237

[14] Guo X, Dias D, Carvajal C, Peyras L, Breul P (2018) Reliability analysis of embankment dam sliding stability using the sparse polynomial chaos expansion. Eng Struct 174:295-307

[15] Habibagahi G, Bamdad A (2003) A neural network framework for mechanical behavior of unsaturated soils. Can Geotech J 40(3):684-693

[16] Hasanipanah M, Noorian-Bidgoli M, Jahed Armaghani D, Khamesi H (2016) Feasibility of PSO-ANN model for predicting surface settlement caused by tunneling. ENG COMPUT-GERMANY 32(4):705-715

[17] Hattab M, Hicher PY (2004) Dilating behaviour of overconsolidated clay. Soils Found 44(2):27-40

[18] He S, Li J (2009) Modeling nonlinear elastic behavior of reinforced soil using artificial neural networks. Appl Soft Comput 9(3):954-961

[19] Holland JH. Adaptation in natural and artificial system: University of Michigan Press, 1975.

[20] Hu X, Zhang Y, Guo L, Wang J, Cai Y, Fu H, et al. (2018) Cyclic behavior of saturated soft clay under stress path with bidirectional shear stresses. Soil Dyn Earthq Eng 104:319-328

[21] Huang GB, Zhu QY, Siew CK (2006) Extreme learning machine: Theory and applications. Neurocomputing 70:489-501

[22] Javadi AA, Rezania M (2009) Applications of artificial intelligence and data mining techniques in soil modeling. Geomech Eng 1(1):53-74

[23] Jefferies M (1993) Nor-Sand: a simle critical state model for sand. Geotechnique 43(1):91-103

[24] Jin Y-F, Wu Z-X, Yin Z-Y, Shen JS (2017) Estimation of critical state-related formula in advanced constitutive modeling of granular material. Acta Geotech 12(6):1329-1351

[25] Jin Y-F, Yin Z-Y, Shen S-L, Hicher P-Y (2016) Investigation into MOGA for identifying parameters of a critical-state-based sand model and parameters correlation by factor analysis. Acta Geotech 11(5):1131-1145

[26] Jin Y-F, Yin Z-Y, Shen S-L, Hicher P-Y (2016) Selection of sand models and identification of parameters using an enhanced genetic algorithm. Int J Numer Anal Met 40(8):1219-1240

[27] Jin Y-F, Yin Z-Y, Wu Z-X, Daouadji A (2018) Numerical modeling of pile penetration in silica sands considering the effect of grain breakage. Finite Elem Anal Des 144:15–29

[28] Jin Y-F, Yin Z-Y, Zhou W-H, Huang H-W (2019) Multi-objective optimization-based updating of predictions during excavation. Eng Appl Artif Intel 78:102–123

[29] Jin YF, Yin ZY, Wu ZX, Zhou WH (2018) Identifying parameters of easily crushable sand and application to offshore pile driving. Ocean Eng 154:416-429

[30] Johari A, Javadi AA, Habibagahi G (2011) Modelling the mechanical behaviour of unsaturated soils using a genetic algorithm-based neural network. Comput Geotech 38(1):2-13

[31] Kim J, Hwang W, Kim Y (2018) Effects of hysteresis on hydro-mechanical behavior of unsaturated soil. Eng Geol 245:1-9

[32] Kohavi R. A study of Cross-Validation and bootstrap for accuracy estimation and model selection. International joint conference on artificial intelligence: Morgan Kaufmann Publishers Inc., 1995. p. 1137-1143.

[33] Kohestani VR, Hassanlourad M (2016) Modeling the mechanical behavior of carbonate sands using artificial neural networks and support vector machines. Int J Geomech 16(1):04015038

[34] Kolymbas D. A generalized hypoelastic constitutive law. Proc XI Int Conf Soil Mechanics and Foundation Engineering. San Francisco: Balkema, Rotterdam, 1985. p. 2626.

[35] Koopialipoor M, Jahed Armaghani D, Haghighi M, Ghaleini EN (2017) A neuro-genetic predictive model to approximate overbreak induced by drilling and blasting operation in tunnels. B Eng Geol Environ:1-10

[36] Lefik M, Schrefler B (2003) Artificial neural network as an incremental non-linear constitutive model for a finite element code. Computer Methods in Applied Mechanics and Engineering 192(28-30):3265-3283

[37] Liu C, Jiang Z, Han X, Zhou W (2019) Slope displacement prediction using sequential intelligent computing algorithms. Measurement 134:634-648

[38] Liu K, Liu B (2019) Intelligent information-based construction in tunnel engineering based on the GA and CCGPR coupled algorithm. Tunnell Undergr Space Technol 88:113-128

[39] Liu WZ, Shi ML, Miao LC, Xu LR, Zhang DW (2013) Constitutive modeling of the destructuration and anisotropy of natural soft clay. Comput Geotech 51:24-41

[40] Mašín D (2005) A hypoplastic constitutive model for clays. Int J Numer Anal Methods Geomech 29(4):311-336

[41] Müthing N, Zhao C, Hölter R, Schanz T (2018) Settlement prediction for an embankment on soft clay. Comput Geotech 93:87-103

[42] Nassr A, Esmaeili-Falak M, Katebi H, Javadi A (2018) A new approach to modeling the behavior of frozen soils. Eng Geol 246:82-90

[43] Ng CWW, Akinniyi DB, Zhou C, Chiu CF (2019) Comparisons of weathered lateritic, granitic and volcanic soils: Compressibility and shear strength. Eng Geol 249:235-240

[44] Penumadu D, Zhao RD (1999) Triaxial compression behavior of sand and gravel using artificial neural networks (ANN). Comput Geotech 24:207-230

[45] Rashidian V, Hassanlourad M (2014) Application of an Artificial Neural Network for Modeling the Mechanical Behavior of Carbonate Soils. Int J Geomech 14(1):142-150

[46] Romo MP, Garcίa SR, Mendoza MJ, Taboada‐Urtuzuάstegui V (2001) Recurrent and

constructive - algorithm networks for sand behavior modeling. Int J Geomech 1(4):371-387

[47] Roscoe KH, Burland J. On the generalized stress-strain behaviour of wet clay. Engineering Plasticity. Cambridge, UK: Cambridge University Press, 1968. p. 535-609.

[48] Rumelhart DE, Hinton GE, Williams RJ (1986) Learning representations by back-propagating errors. Nature 323(9):533–536

[49] Stone M (1974) Cross-validatory choice and assessment of statistical predictions. J R Stat Soc C-appl 36(2):111-147

[50] Su D, Yang ZX (2019) Drained analyses of cylindrical cavity expansion in sand incorporating a bounding-surface model with state-dependent dilatancy. Appl Math Model 68:1-20

[51] Su L-J, Yin J-H, Zhou W-H (2010) Influences of overburden pressure and soil dilation on soil nail pull-out resistance. Comput Geotech 37(4):555-564

[52] Taiebat M, Dafalias YF (2008) SANISAND: Simple anisotropic sand plasticity model. Int J Numer Anal Met 32(8):915-948

[53] Tian Y, Yao YP (2017) Modelling the non-coaxiality of soils from the view of cross-anisotropy. Comput Geotech 86:219-229

[54] Tran C, Srokosz P (2010) The idea of PGA stream computations for soil slope stability evaluation. Cr Mecanique 338(9):499-509

[55] Turk G, Logar J, Majes B (2001) Modelling soil behaviour in uniaxial strain conditions by neural networks. Adv Eng Softw 32:805-812

[56] Vermeer P (1978) A double hardening model for sand. Geotechnique 28(4):413-433

[57] Wang S, Wu W, Yin Z-Y, Peng C, He X-Z (2018) Modelling time-dependent behaviour of granular material with hypoplasticity. Int J Numer Anal Methods Geomech 42(12):1331-1345

[58] Wu W, Bauer E, Kolymbas D (1996) Hypoplastic constitutive model with critical state for granular materials. Mech Mater 23(1):45-69

[59] Xie X, Qi S, Zhao F, Wang D (2018) Creep behavior and the microstructural evolution of loess-like soil from Xi'an area, China. Eng Geol 236:43-59

[60] Xiong H, Nicot F, Yin Z (2017) A three - dimensional micromechanically based model. Int J Numer Anal Methods Geomech 41(17):1669-1686

[61] Yang B, Yin K, Lacasse S, Liu Z (2019) Time series analysis and long short-term memory neural network to predict landslide displacement. Landslides

[62] Yao Y, Hou W, Zhou A (2009) UH model: three-dimensional unified hardening model for overconsolidated clays. Geotechnique 59(5):451-469

[63] Yao Y, Sun D, Luo T (2004) A critical state model for sands dependent on stress and

density. Int J Numer Anal Methods Geomech 28(4):323-337

[64] Yao Y, Sun D, Matsuoka H (2008) A unified constitutive model for both clay and sand with hardening parameter independent on stress path. Computers and Geotechnics 35(2):210-222

[65] Yao YP, Hou W, Zhou AN (2009) UH model: three-dimensional unified hardening model for overconsolidated clays. Géotechnique 59(5):451-469

[66] Yin Z-Y, Chang CS, Karstunen M, Hicher P-Y (2010) An anisotropic elastic–viscoplastic model for soft clays. Int J Solids Struct 47(5):665-677

[67] Yin Z-Y, Jin Y-F, Shen S-L, Huang H-W (2016) An efficient optimization method for identifying parameters of soft structured clay by an enhanced genetic algorithm and elastic–viscoplastic model. Acta Geotech 12(4):849-867

[68] Yin Z-Y, Karstunen M (2011) Modelling strain-rate-dependency of natural soft clays combined with anisotropy and destructuration. Acta Mech Solida Sin 24(3):216-230

[69] Yin Z-Y, Karstunen M, Chang CS, Koskinen M, Lojander M (2011) Modeling Time-Dependent Behavior of Soft Sensitive Clay. J Geotech Geoenviron Eng 137(11):1103-1113

[70] Yin Z-Y, Zhao J, Hicher P-Y (2014) A micromechanics-based model for sand-silt mixtures. Int J Solids Struct 51(6):1350-1363

[71] Yin ZY, Chang CS (2009) Microstructural modelling of stress-dependent behaviour of clay. Int J Solids Struct 46(6):1373-1388

[72] Yin ZY, Chang CS, Hicher PY (2010) Micromechanical modelling for effect of inherent anisotropy on cyclic behaviour of sand. Int J Solids Struct 47(14-15):1933-1951

[73] Yin ZY, Chang CS, Hicher PY, Karstunen M (2009) Micromechanical analysis of kinematic hardening in natural clay. Int J Plast 25(8):1413-1435

[74] Yin ZY, Chang CS, Karstunen M, Hicher PY (2010) An anisotropic elastic-viscoplastic model for soft clays. International Journal of Solids and Structures 47(5):665-677

[75] Yin ZY, Jin YF, S SJ, Hicher PY (2017) Optimization techniques for identifying soil parameters in geotechnical engineering: comparative study and enhancement. Int J Numer Anal Met 42(1):1-25

[76] Yu H (1998) CASM: A unified state parameter model for clay and sand. Int J Numer Anal Methods Geomech 22(8):621-653

[77] Zhang P, Wu H-N, Chen R-P, Chan THT (2020) Hybrid meta-heuristic and machine learning algorithms for tunneling-induced settlement prediction: A comparative study. Tunnell Undergr Space Technol 99:103383

[78] Zhang P, Yin ZY, Jin YF, Chan T, Gao FP (2020) Intelligent modelling of clay compressibility using hybrid meta-heuristic and machine learning algorithms. Geoscience Frontiers:doi: 10.1016/j.gsf.2020.1002.1014

[79] Zhang P, Yin ZY, Jin YF, Ye GL (2020) An AI-based model for describing cyclic characteristics of granular materials. Int J Numer Anal Met:doi: 10.1002/nag.3063

[80] Zhou YF, Tham LG, Yan WM, Dai FC, Xu L (2014) Laboratory study on soil behavior in loess slope subjected to infiltration. Eng Geol 183:31-38

[81] Zhu JH, Zaman MM, Anderson SA (1998) Modelling of shearing behaviour of a residual soil with Recurrent Neural Network. Int J Numer Anal Met 22(8):671-687

Table

Table 1. Previous research works for identifying constitutive models of geomaterials

Strategy	Machine learning algorithm	Validation method	Methods to determine architecture	References
Total stress-strain	BPNN	No	Trial & error	He and Li (2009)
	BPNN	No	Trial & error	Rashidian and Hassanlourad (2014)
	BPNN	No	Trial & error	Kohestani (2016)
	SVM	No	Trial & error	Kohestani (2016)
	GP	No	Optimization	Cabalar and Cevik (2011)
Incremental stress-strain	BPNN	No	Trial & error	Ellis et al. (1995)
	BPNN	No	Trial & error	Penumadu and Zhao (1999)
	BPNN	No	Optimization	Basheer (2000)
	BPNN	No	Trial & error	Habibagahi and Bamdad (2003)
	BPNN	No	Trial & error	Turk et al. (2001)
	ENN	No	Trial & error	Johari et al. (2011)
	RNN	No	Trial & error	Zhu et al. (1998)
	RNN	No	Optimization	Romo et al. (2001)
	EPR	No	Optimization	Javadi and Rezania (2009)
	EPR	No	Optimization	Faramarzi et al. (2012)
	EPR	No	Optimization	Nassr et al. (2018)

Table 2. Hyper-parameters in three selected ML algorithms

Algorithm	Hyper-parameters	Description	Range (T/I)
BPNN	*l*_hidden layers	Number of hidden layers	1/1
	*n*_neurons	Number of neurons in hidden layers	1–5/1–9
ELM	*n*_neurons	Number of neurons in hidden layers	1–11/1–11
EPR	*z*_transformed term	Number of transformed variables	1–11/1–11

Note: T = total stress-strain strategy; I = incremental stress-strain strategy.

Table 3. Methods for determining the number of hidden neurons

Methods	References	Number of hidden neurons (T/I)
$\leq 2N_i+1$	Nielsen (1987)	5/9
$\dfrac{2+N_i \times N_o+0.5N_o \times \left(N_o^2+N_i\right)-3}{N_o+N_i}$	Paola (1994)	1/1
$2N_i/3$	Wang (1994)	2/3
$\sqrt{N_i \times N_o}$	Masters (1994)	2/2
$2N_i$	Kaastra and Boyd (1996)	4/8

Note: N_i = number of input variables; N_o = number of output variables; T = total stress-strain strategy; I = incremental stress-strain strategy.

Table 4. Values of parameters in the GA algorithm

Algorithm	p_cross	p_mutation	Population	Generation
GA	0.7	0.1	20	500/5000

Note: 500 = maximum generation for the BPNN and EPR; 5000 = maximum generation for the ELM.

Table 5. Values of indicators for the test set

Strategy	Algorithm	Interpolation (σ_{n0}, kPa)				Extrapolation (σ_{n0}, kPa)					
		MAE		MAPE		MAE			MAPE		
		150	350	150	350	15	650	700	15	650	700
T	BPNN	0.58	1.36	1.22	1.02	1.95	2.84	3.35	28.81	1.13	1.21
	ELM	9.42	22.58	18.88	20.63	8.88	34.16	41.34	142.46	19.60	19.64
	EPR	6.28	14.66	14.06	14.06	0.63	27.23	29.32	14.06	14.06	14.06
I	BPNN	2.07	3.69	4.17	2.97	0.88	5.02	8.03	12.79	2.45	3.13
	ELM	4.44	5.46	9.05	4.09	6.96	15.36	18.88	151.64	7.92	9.31
	EPR	26.95	10.25	33.25	6.33	23.90	56.08	49.59	290.27	17.98	18.06

Note: T = total stress-strain strategy; I = incremental stress-strain strategy.

Figure captions

Fig. 1 Schematic view of ML algorithms: (a) BPNN; (b) ELM

Fig. 2 Model framework: (a) flowchart of constructing ML-based constitutive models; (b) schematic view of the total stress-strain strategy; (c) schematic view of the incremental stress-strain strategy

Fig. 3 Evolution of fitness value using the total stress-strain strategy for: (a) BPNN; (b) ELM; (c) EPR

Fig. 4 Evolution of fitness value using the incremental stress-strain strategy for: (a) BPNN; (b) ELM; (c) EPR

Fig. 5 Predicted results on the training set using the total stress-strain strategy for: (a) BPNN; (b) ELM; (c) EPR

Fig. 6 Predicted results on the training set using the incremental stress-strain strategy for: (a) BPNN; (b) ELM; (c) EPR

Fig. 7 Predicted results on the test set (interpolation) using the total stress-strain strategy: (a) BPNN; (b) ELM; (c) EPR

Fig. 8 Predicted results on the test set (interpolation) using the incremental stress-strain strategy: (a) BPNN; (b) ELM; (c) EPR

Fig. 9 Predicted results on the test set (extrapolation) using the total stress-strain strategy: (a) BPNN; (b) ELM; (c) EPR

Fig. 10 Predicted results on the test set (extrapolation) using the incremental stress-strain strategy: (a) BPNN; (b) ELM; (c) EPR

Fig. 11 Experimental data of Kaolinite clay

Fig. 12 Framework for predicting Kaolinite clay behaviours

Fig. 13 Predicted results on the training set using BPNN-based constitutive model with incremental stress-strain strategy: (a) q–ε_1; (b) e–ε_1

Fig. 14 Predicted results on the test set (interpolation) using BPNN based on the incremental stress-strain strategy: (a) q–ε_1; (b) e–ε_1

Fig. 15 Predicted results on the test set (extrapolation) using BPNN based on the incremental stress-strain strategy: (a) q–ε_1; (b) e–ε_1

Fig. 16 Predicted results on the test set using ELM based on the incremental stress-strain strategy: (a) q–ε_1 (interpolation); (b) e–ε_1 (interpolation); (c) q–ε_1 (extrapolation); (d) e–ε_1 (extrapolation)

Fig. 17 Predicted results on the test set using EPR based on the incremental stress-strain strategy: (a) q–ε_1 (interpolation); (b) e–ε_1 (interpolation); (c) q–ε_1 (extrapolation); (d) e–ε_1 (extrapolation)

Fig. 1

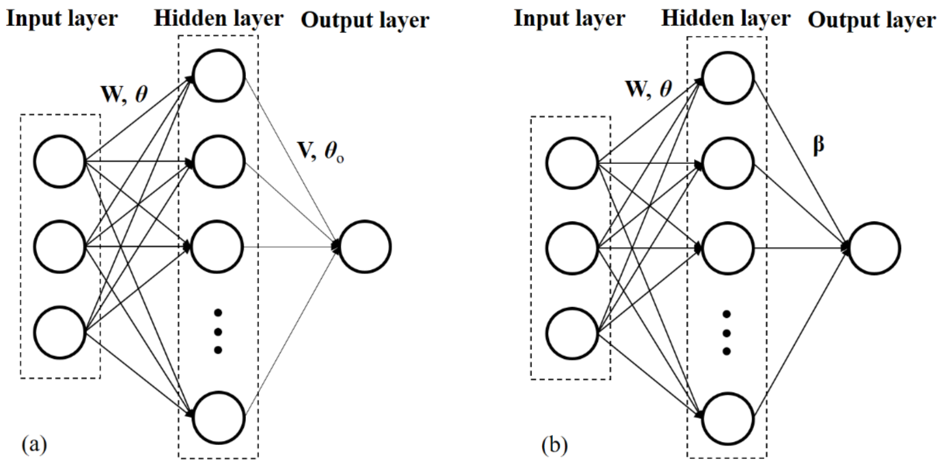

Input layer
Hidden layer
Output layer
W, θ
V, θ_o
(a)
Input layer
Hidden layer
Output layer
W, θ
β
(b)

Fig. 2

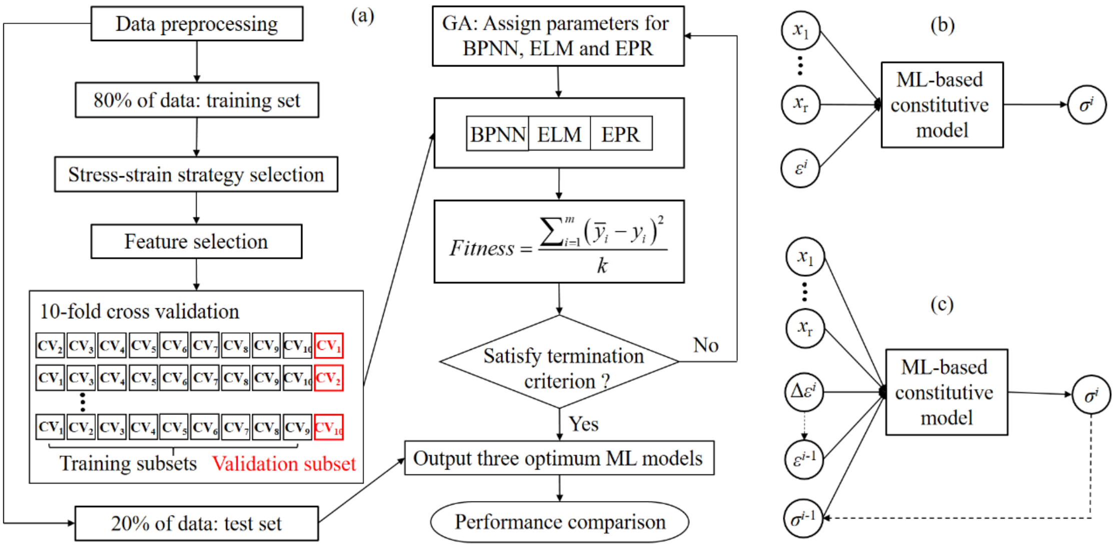

(a)
Data preprocessing
80% of data: training set
Stress-strain strategy selection
Feature selection
10-fold cross validation
CV_2 CV_3 CV_4 CV_5 CV_6 CV_7 CV_8 CV_9 CV_{10} CV_1
CV_1 CV_3 CV_4 CV_5 CV_6 CV_7 CV_8 CV_9 CV_{10} CV_2
CV_1 CV_2 CV_3 CV_4 CV_5 CV_6 CV_7 CV_8 CV_9 CV_{10}
Training subsets
Validation subset
20% of data: test set
GA: Assign parameters for BPNN, ELM and EPR
BPNN
ELM
EPR
$Fitness = \frac{\sum_{i=1}^{m}(\overline{y}_i - y_i)^2}{k}$
Satisfy termination criterion ?
No
Yes
Output three optimum ML models
Performance comparison
(b)
x_1
x_r
ε^i
ML-based constitutive model
σ^i
(c)
x_1
x_r
$\Delta\varepsilon^i$
ε^{i-1}
σ^{i-1}
ML-based constitutive model
σ^i

Fig. 3

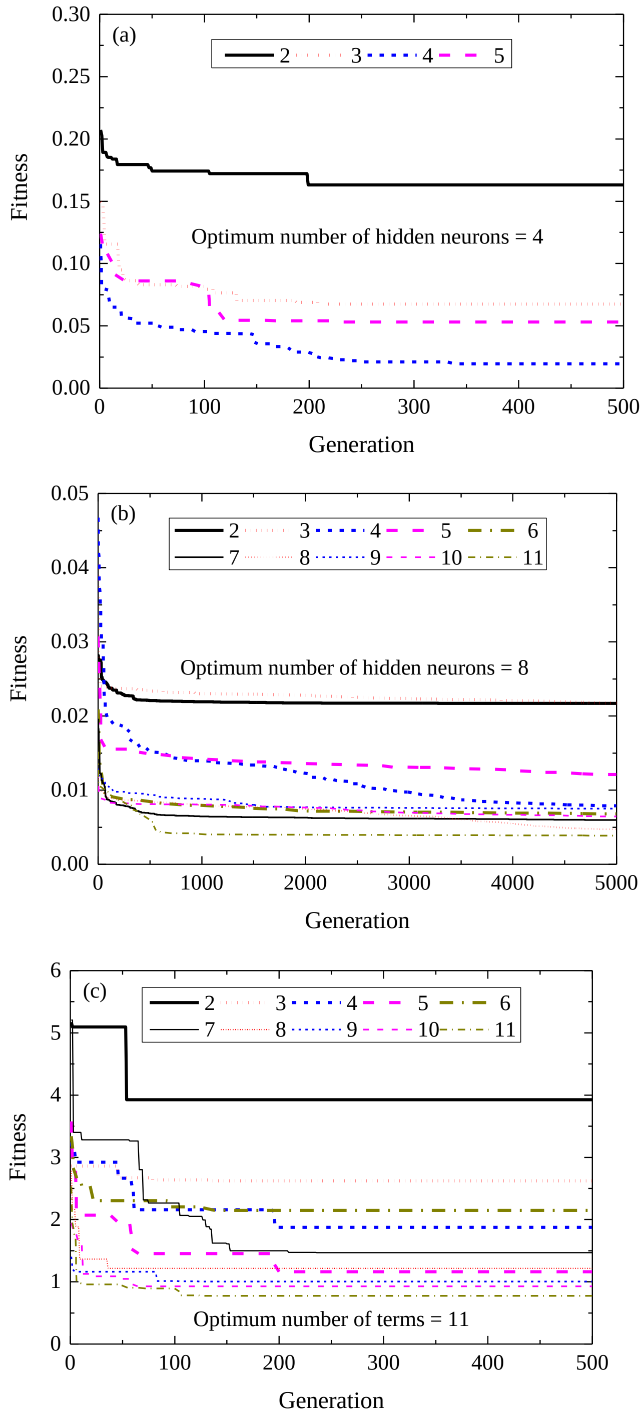

(a)
0.30
0.25
0.20
0.15
0.10
0.05
0.00
Fitness
0
100
200
300
400
500
Generation
2
3
4
5
Optimum number of hidden neurons = 4
(b)
0.05
0.04
0.03
0.02
0.01
0.00
Fitness
0
1000
2000
3000
4000
5000
Generation
2
3
4
5
6
7
8
9
10
11
Optimum number of hidden neurons = 8
(c)
6
5
4
3
2
1
0
Fitness
0
100
200
300
400
500
Generation
2
3
4
5
6
7
8
9
10
11
Optimum number of terms = 11

Fig. 4

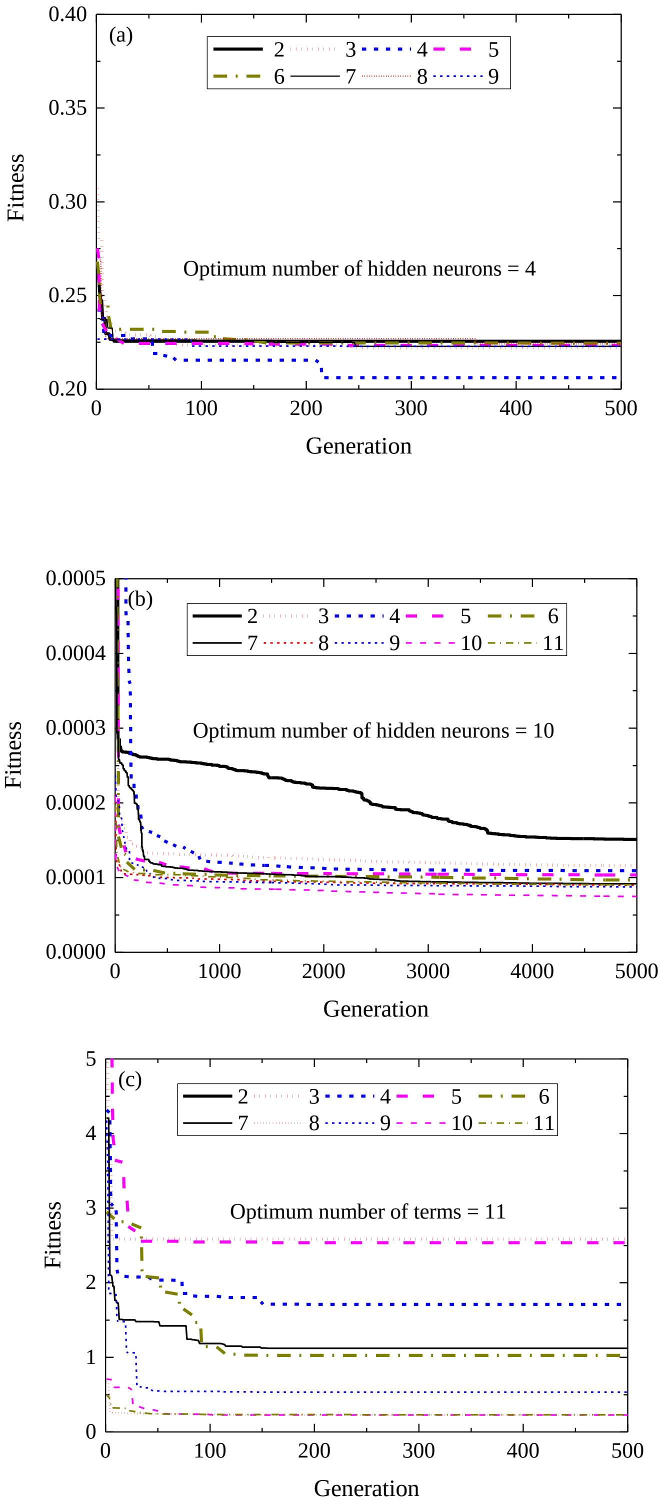

(a)
2
3
4
5
6
7
8
9
Optimum number of hidden neurons = 4
Fitness
0.40
0.35
0.30
0.25
0.20
0
100
200
300
400
500
Generation
(b)
2
3
4
5
6
7
8
9
10
11
Optimum number of hidden neurons = 10
Fitness
0.0005
0.0004
0.0003
0.0002
0.0001
0.0000
0
1000
2000
3000
4000
5000
Generation
(c)
2
3
4
5
6
7
8
9
10
11
Optimum number of terms = 11
Fitness
5
4
3
2
1
0
0
100
200
300
400
500
Generation

Fig. 5

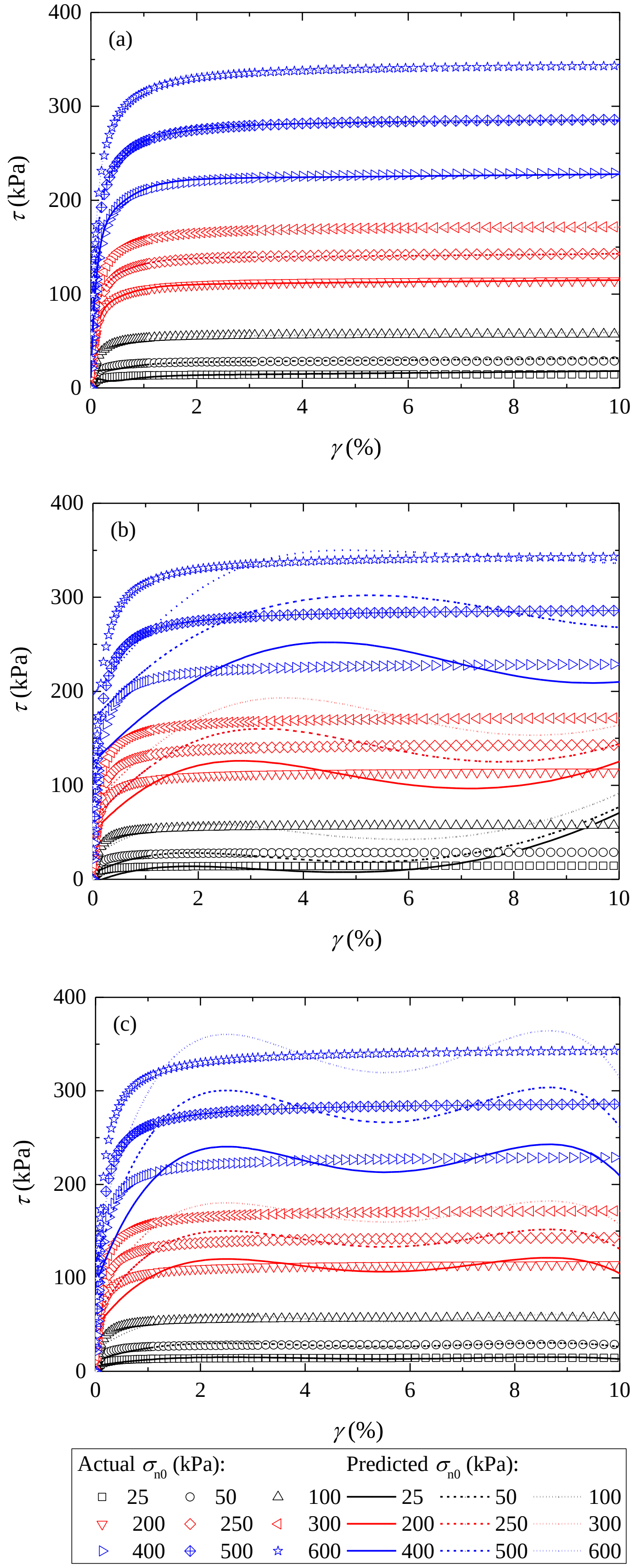

(a)
(b)
(c)
τ (kPa)
γ (%)
0
100
200
300
400
0
2
4
6
8
10
Actual σ_{n0} (kPa):
□ 25 ○ 50 △ 100
▽ 200 ◇ 250 ◁ 300
▷ 400 ◈ 500 ☆ 600
Predicted σ_{n0} (kPa):
25
50
100
200
250
300
400
500
600

Fig. 6

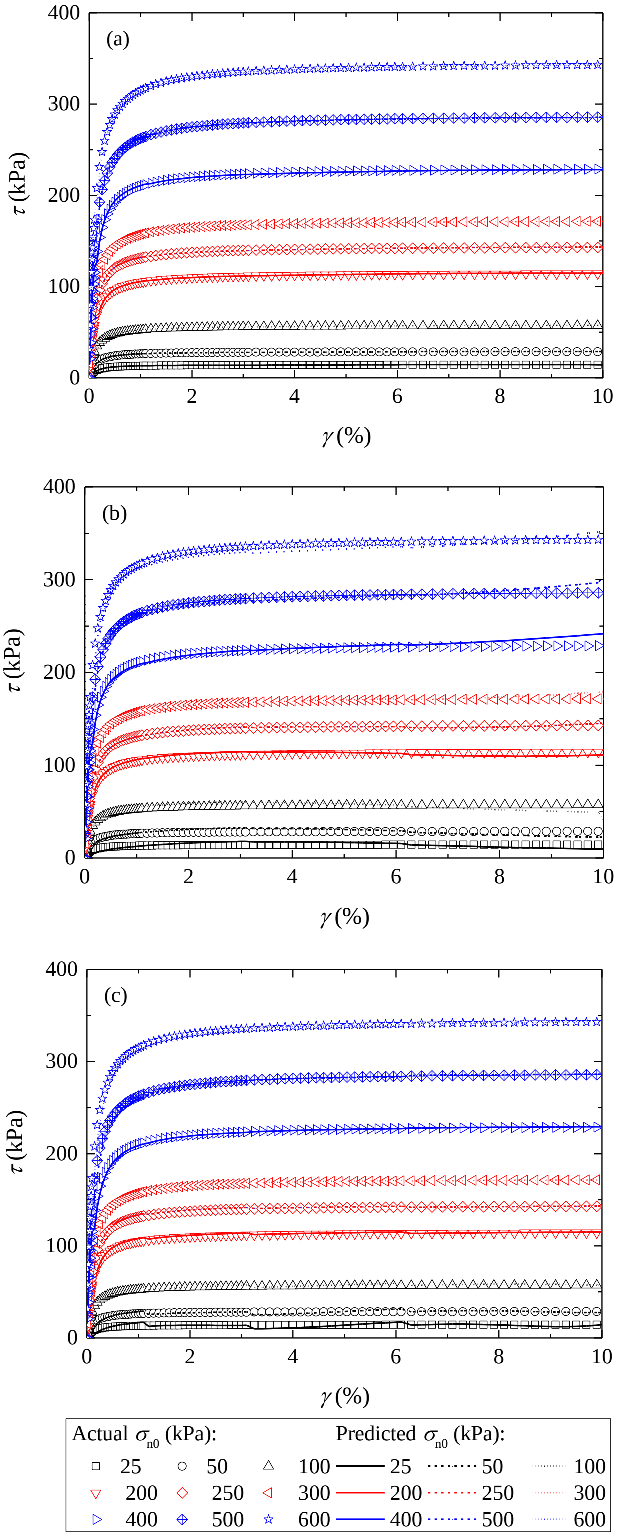

(a)
(b)
(c)
τ (kPa)
γ (%)
0
100
200
300
400
0
2
4
6
8
10
Actual σn0 (kPa):
25
50
100
200
250
300
400
500
600
Predicted σn0 (kPa):
25
50
100
200
250
300
400
500
600

Fig. 7

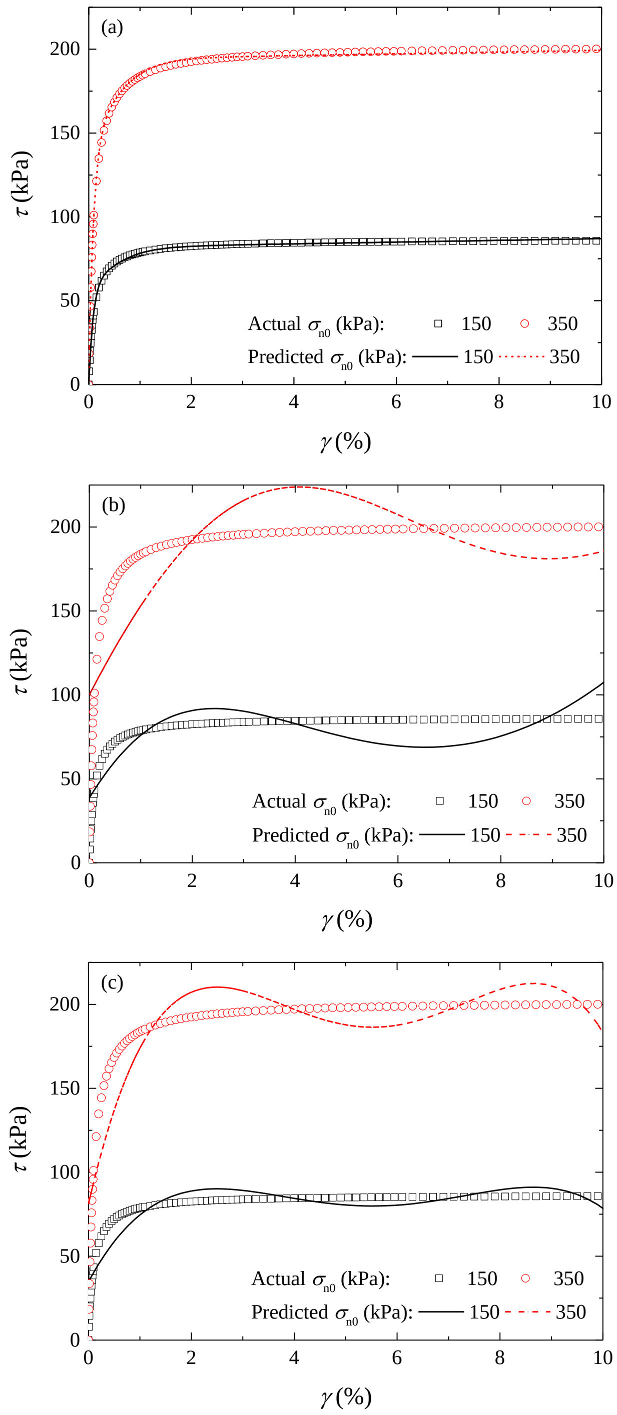

(a)
τ (kPa)
0
50
100
150
200
0
2
4
6
8
10
γ (%)
Actual σ_{n0} (kPa): □ 150 ○ 350
Predicted σ_{n0} (kPa): —— 150 ······ 350
(b)
τ (kPa)
0
50
100
150
200
0
2
4
6
8
10
γ (%)
Actual σ_{n0} (kPa): □ 150 ○ 350
Predicted σ_{n0} (kPa): —— 150 - - - 350
(c)
τ (kPa)
0
50
100
150
200
0
2
4
6
8
10
γ (%)
Actual σ_{n0} (kPa): □ 150 ○ 350
Predicted σ_{n0} (kPa): —— 150 - - - 350

Fig. 8

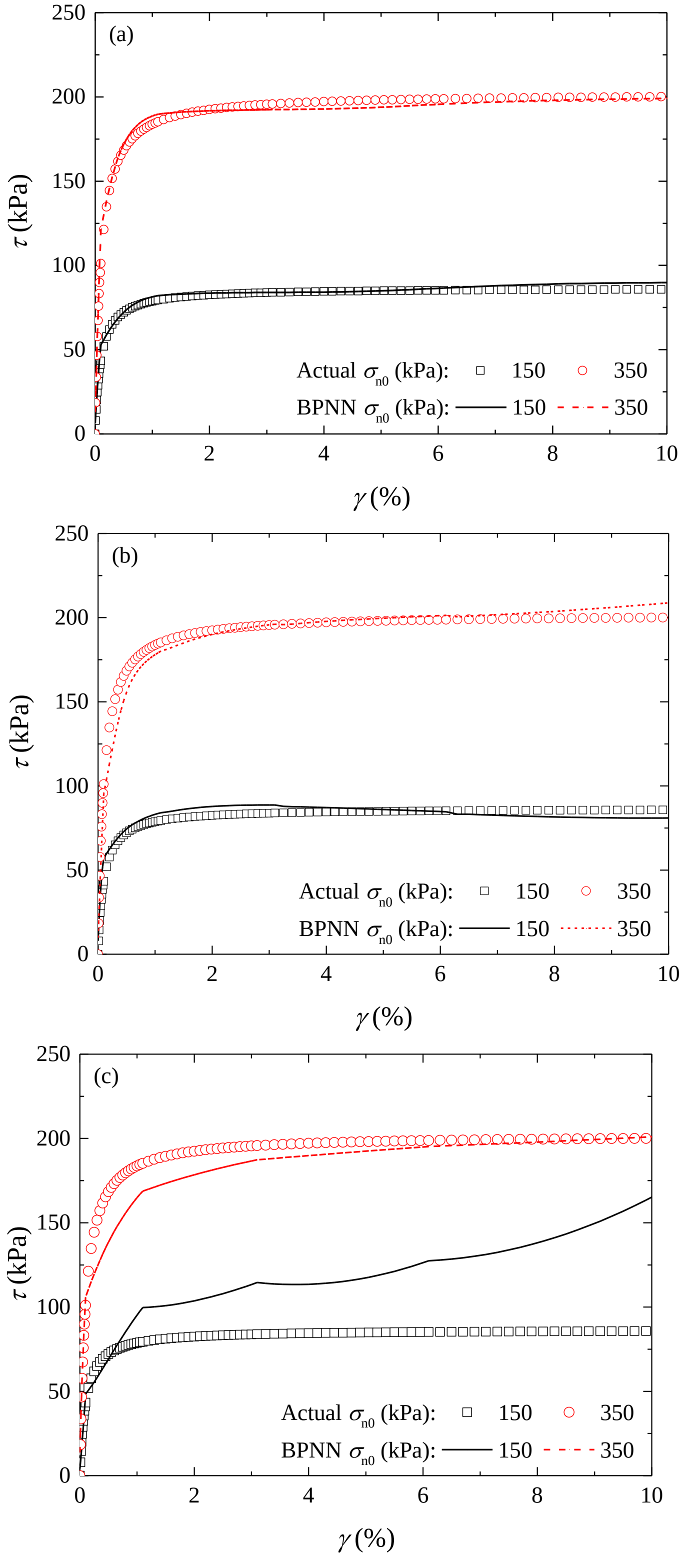

(a)
(b)
(c)
250
200
150
100
50
0
0
2
4
6
8
10
τ (kPa)
γ (%)
Actual σn0 (kPa): □ 150 ○ 350
BPNN σn0 (kPa): —— 150 - - - 350
BPNN σn0 (kPa): —— 150 ······ 350

Fig. 9

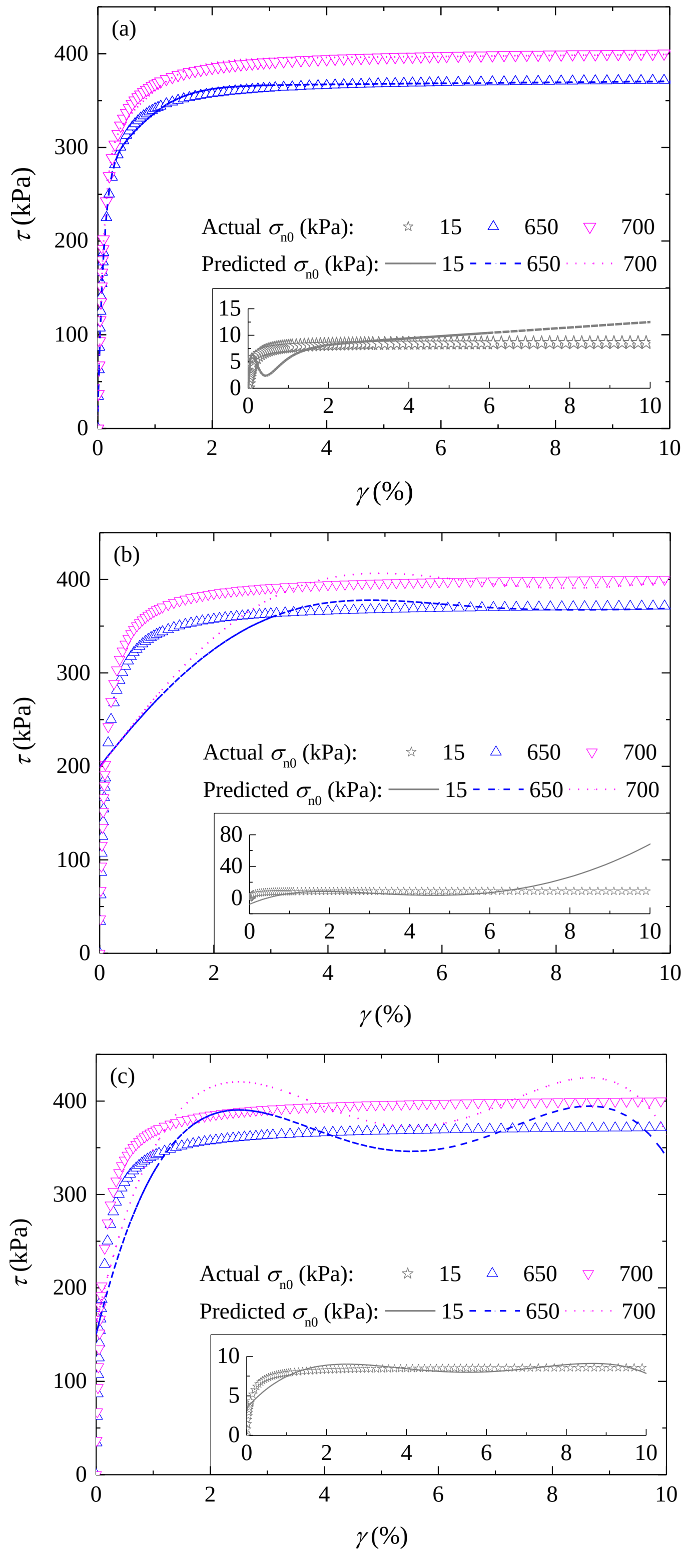

(a)
(b)
(c)
τ (kPa)
γ (%)
0
100
200
300
400
0
2
4
6
8
10
Actual σ_{n0} (kPa): ☆ 15 △ 650 ▽ 700
Predicted σ_{n0} (kPa): 15 650 700
15
10
5
80
40

Fig. 10

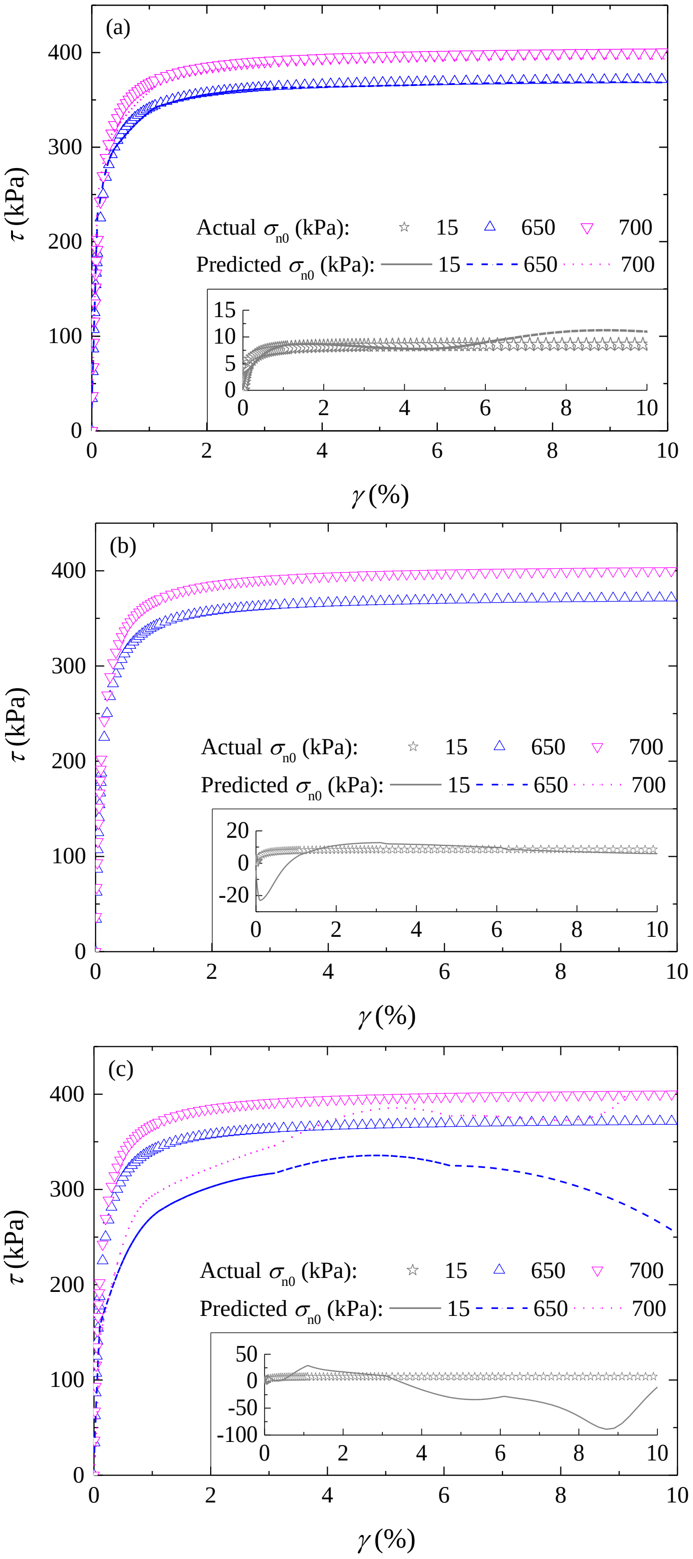

(a)
(b)
(c)
τ (kPa)
γ (%)
Actual σ_{n0} (kPa): ☆ 15 △ 650 ▽ 700
Predicted σ_{n0} (kPa): 15 650 700
0
100
200
300
400
0
2
4
6
8
10
15
10
5
20
-20
50
-50
-100

Fig. 11

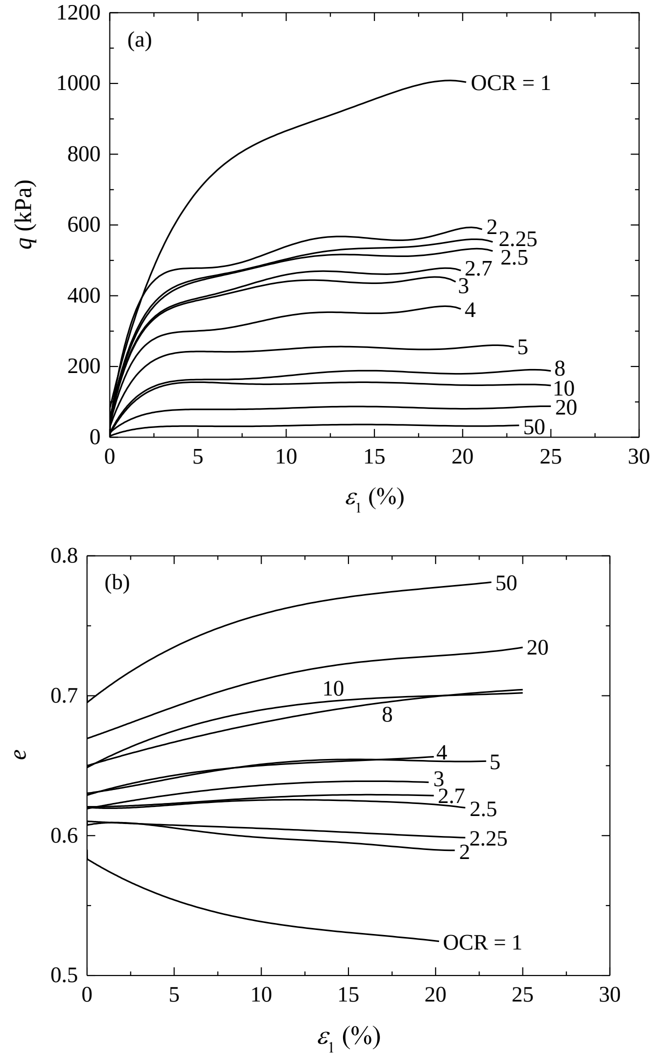

(a)
q (kPa)
1200
1000
800
600
400
200
0
0
5
10
15
20
25
30
ε_1 (%)
OCR = 1
2
2.25
2.5
2.7
3
4
5
8
10
20
50
(b)
e
0.8
0.7
0.6
0.5
0
5
10
15
20
25
30
ε_1 (%)
50
20
10
8
4
5
3
2.7
2.5
2.25
2
OCR = 1

Fig. 12

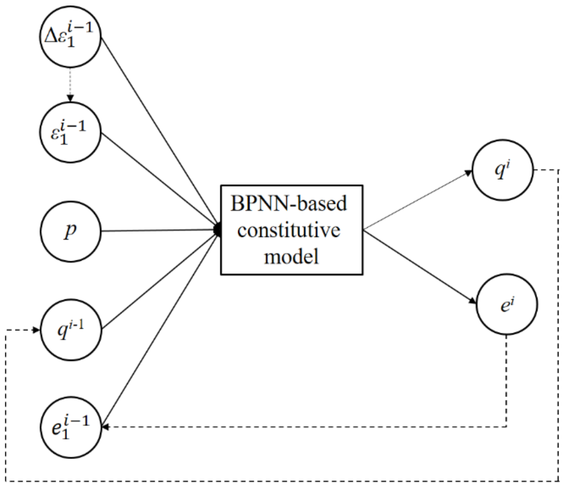
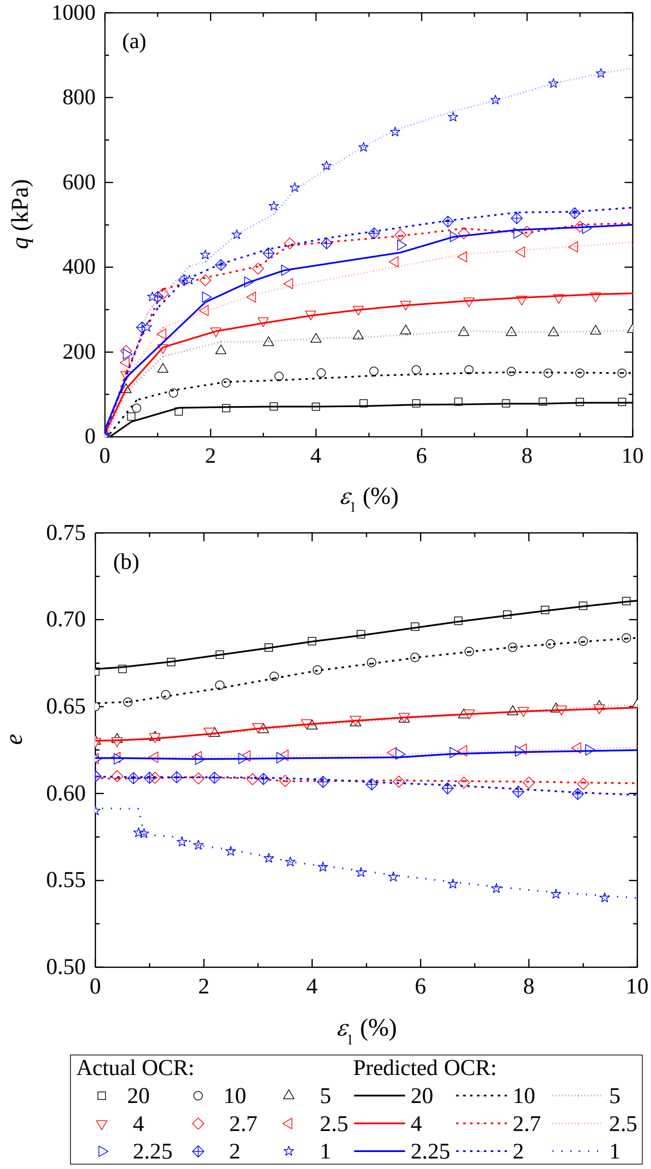

$\Delta\varepsilon_1^{i-1}$
ε_1^{i-1}
p
$q^{i\text{-}1}$
e_1^{i-1}
BPNN-based constitutive model
q^i
e^i

Fig. 13



(a)
1000
800
600
400
200
0
q (kPa)
0
2
4
6
8
10
ε_1 (%)
(b)
0.75
0.70
0.65
0.60
0.55
0.50
e
0
2
4
6
8
10
ε_1 (%)
Actual OCR:
□ 20
○ 10
△ 5
▽ 4
◇ 2.7
◁ 2.5
▷ 2.25
⊕ 2
☆ 1
Predicted OCR:
20
10
5
4
2.7
2.5
2.25
2
1

Fig. 14

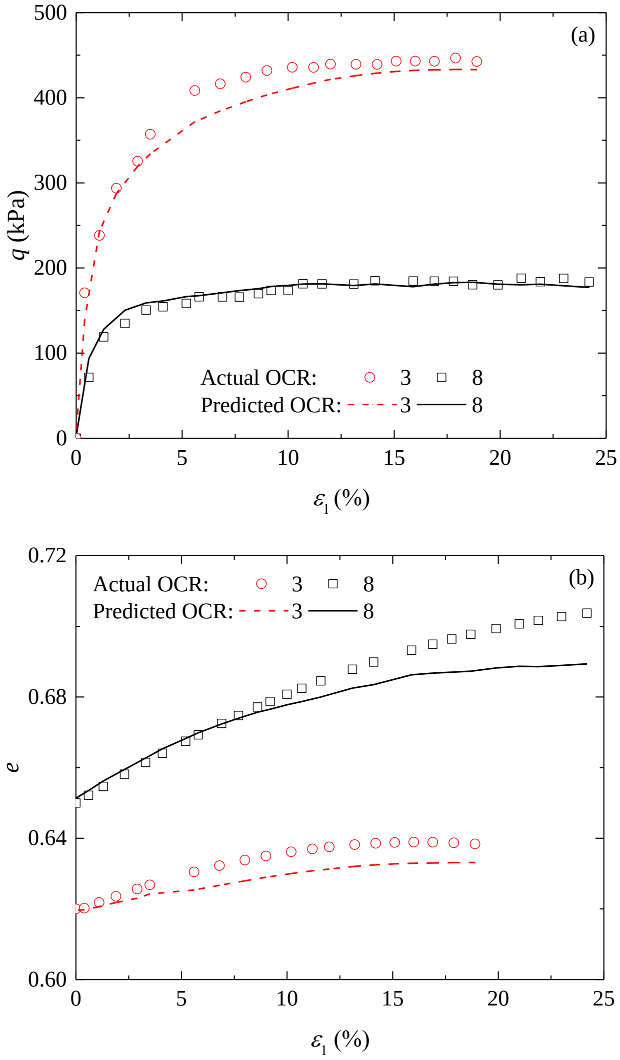

(a)
q (kPa)
500
400
300
200
100
0
0
5
10
15
20
25
ε1 (%)
Actual OCR: ○ 3 □ 8
Predicted OCR: - - - 3 —— 8
(b)
e
0.72
0.68
0.64
0.60
0
5
10
15
20
25
ε1 (%)
Actual OCR: ○ 3 □ 8
Predicted OCR: - - - 3 —— 8

Fig. 15

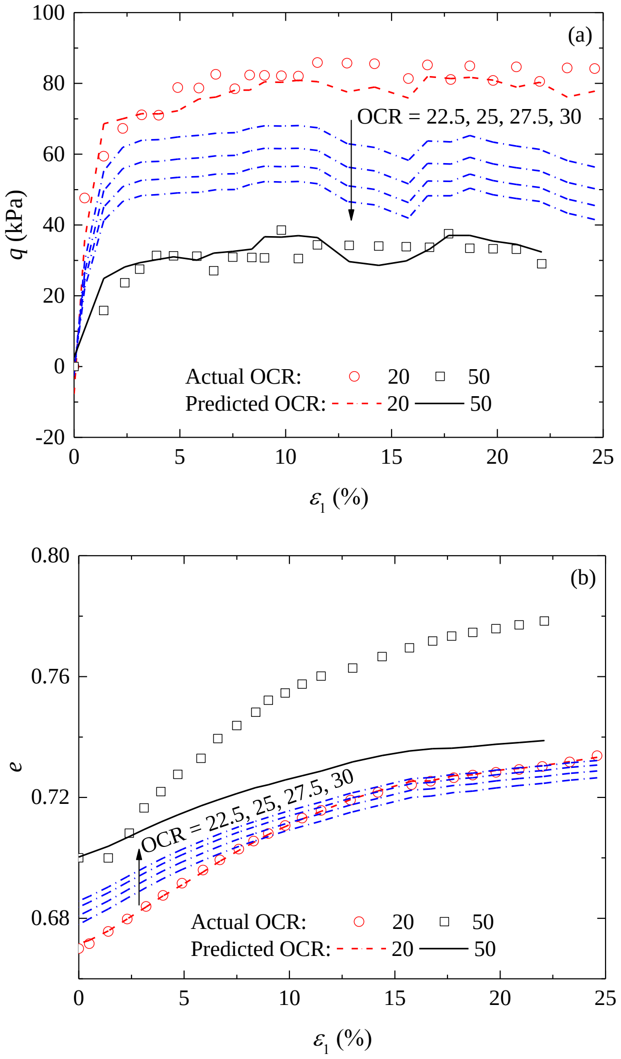
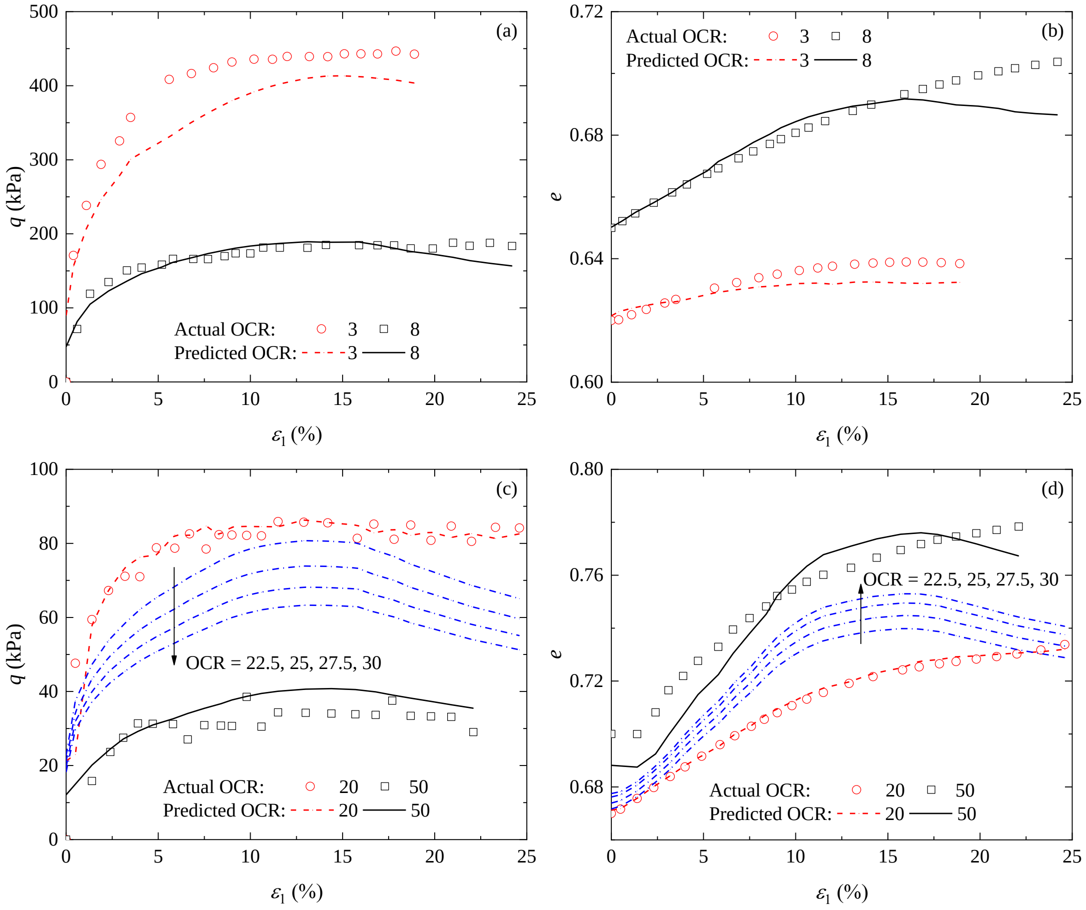
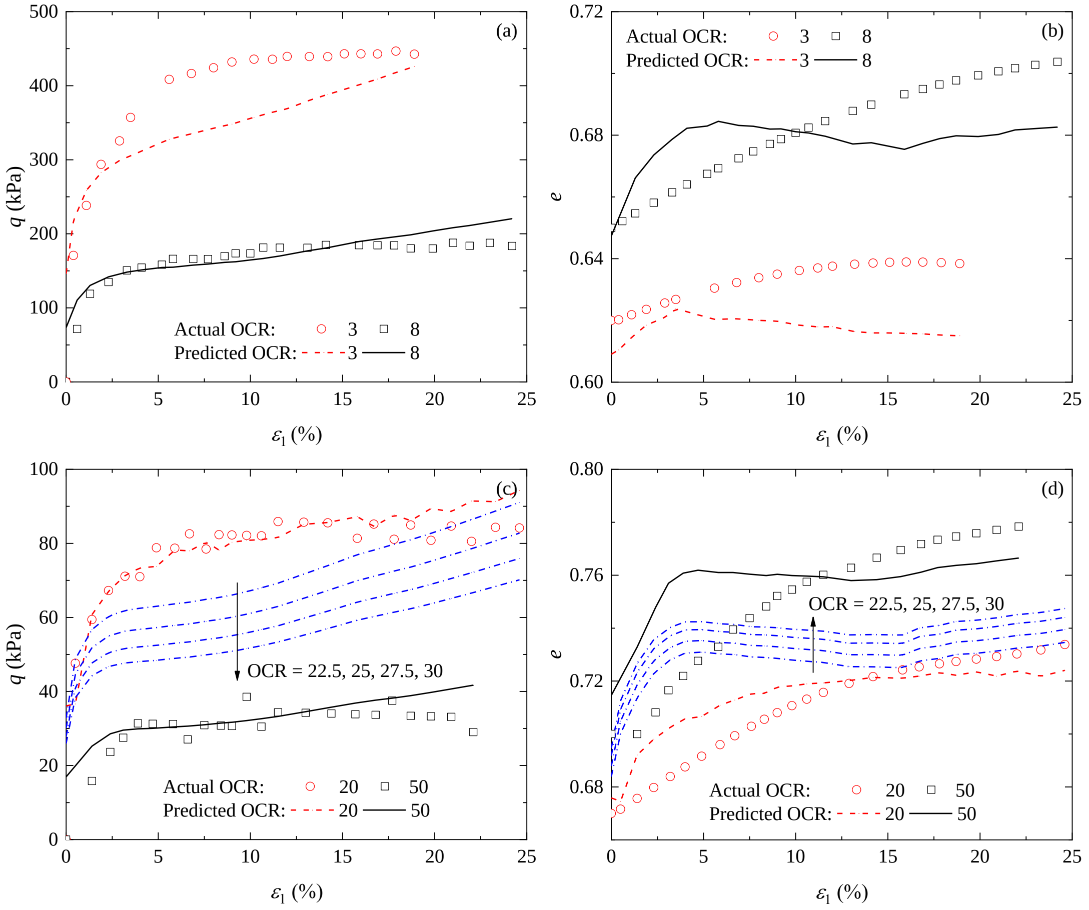

(a)
q (kPa)
100
80
60
40
20
0
-20
0
5
10
15
20
25
ε_1 (%)
OCR = 22.5, 25, 27.5, 30
Actual OCR: ○ 20 □ 50
Predicted OCR: - - - 20 —— 50
(b)
e
0.80
0.76
0.72
0.68
0
5
10
15
20
25
ε_1 (%)
OCR = 22.5, 25, 27.5, 30
Actual OCR: ○ 20 □ 50
Predicted OCR: - - - 20 —— 50

Fig. 16



(a)
q (kPa)
ε1 (%)
Actual OCR: ○ 3 □ 8
Predicted OCR: - - - 3 —— 8
(b)
e
ε1 (%)
Actual OCR: ○ 3 □ 8
Predicted OCR: - - - 3 —— 8
(c)
q (kPa)
ε1 (%)
OCR = 22.5, 25, 27.5, 30
Actual OCR: ○ 20 □ 50
Predicted OCR: - - - 20 —— 50
(d)
e
ε1 (%)
OCR = 22.5, 25, 27.5, 30
Actual OCR: ○ 20 □ 50
Predicted OCR: - - - 20 —— 50

Fig. 17



(a)
q (kPa)
500
400
300
200
100
0
0
5
10
15
20
25
ε_1 (%)
Actual OCR: ○ 3 □ 8
Predicted OCR: - - - 3 —— 8
(b)
e
0.72
0.68
0.64
0.60
0
5
10
15
20
25
ε_1 (%)
Actual OCR: ○ 3 □ 8
Predicted OCR: - - - 3 —— 8
(c)
q (kPa)
100
80
60
40
20
0
0
5
10
15
20
25
ε_1 (%)
OCR = 22.5, 25, 27.5, 30
Actual OCR: ○ 20 □ 50
Predicted OCR: - - - 20 —— 50
(d)
e
0.80
0.76
0.72
0.68
0
5
10
15
20
25
ε_1 (%)
OCR = 22.5, 25, 27.5, 30
Actual OCR: ○ 20 □ 50
Predicted OCR: - - - 20 —— 50