https://dx.doi.org/10.1016/j.ijrmms.2023.105525


# A deep learning-based approach with anti-noise ability for identification of rock microcracks using distributed fibre optic sensing data

Shuai Zhao, Daoyuan Tan*, Shaoqun Lin, Zhenyu Yin, Jianhua Yin

*Department of Civil and Environmental Engineering, The Hong Kong Polytechnic University, Hong Kong, China*

**Abstract:** Most of the existing deep learning-based crack identification models can achieve high accuracy when being trained and tested using data split from the same dataset with minimal noise, while perform poorly on field monitoring data with certain level of noise. This research developed a hybrid attention convolutional neural network (HACNN) for rock microcrack identification with enhanced anti-noise ability for distributed fibre optic sensing data. A hybrid attention module was designed and placed next to some certain convolutional layers to enhance the nonlinear representation ability of the proposed model. Two training interference strategies, namely small mini-batch training and adding dropout in the first convolutional layer, were employed to interfere with the training of the HACNN to enhance its robustness against noise. A series of experiments are designed based on the properties of the two training interference strategies to optimize the model parameters. Results showed that the optimized HACNN achieved higher accuracy on datasets with different signal-to-noise ratios compared to other machine learning algorithms, including the support vector machine, the multilayer perceptron, and an existing one-dimensional convolutional neural network. This research demonstrates the potential of establishing a robust DL-based model for identification of rock microcracks from noisy distributed fibre sensing optic data, even when training the model only with a smoothed dataset.

**Keywords:** Rock microcrack identification; Convolutional neural network; Hybrid attention module; Anti-noise; Fibre optic sensing data

## 1. Introduction

The presence of cracks in structures can cause a reduction in local stiffness, leading to nonaffine deformation of structures. In the field of rock engineering, cracks serve as a visual indicator of rock failure patterns. Accurate identification of cracks and measurement of strain are crucial for analysing the mechanical properties and failure mechanisms of rocks.[1-3] Therefore, it is vital to conduct accurate and reliable crack identification and measurement of rock strain.

Currently, deep learning (DL)-based approaches are widely used to identify cracks owing to their strong nonlinear representation abilities. For example, Miao et al.[4] combined the digital image correlation and convolutional neural network (CNN) to identify crack mechanism of red sandstone, namely mode I, mode II, and mixed-mode I/II. With regard to the application of DL-based approaches on-site engineering, Chen et al.[5] designed a FraSegNet to identify the fracture trace of rock tunnel face using the images acquired from the tunnel face. It is well-known the computational intensity and the training time will be high when using the two-dimensional (2D) CNN-based architectures to identify cracks among large numbers of images. Comparing with the 1D CNN, the 2D CNN-based architectures are not lightweight and efficient enough, hindering their real-time application in crack detection. Therefore, some researchers transformed images from a 2D pixel space to a 1D frequency space via Fourier transformation, and then established 1D-CNNs to conduct crack classification among the frequency space.[6, 7] However, this crack detection approach based on Fourier transformation is an indirect method, and some errors may be introduced into the 1D frequency space during the transformation process. Distributed fibre optic sensing (DFOS) techniques has become a promising approach to overcome the aforementioned limitations, since they can directly provide 1D signal data (e.g. strain data) for 1D-CNNs.

DFOS techniques have been used in some fields for deformation monitoring of frozen soil[26], displacement monitoring of cast-in situ piles[27], crack monitoring[28], in-situ pullout behavior of grouted anchor cables[29], and magnetostrictive strain monitoring[30], because of their unique advantages, such as high-spatial-resolution distributed sensing and immunity to electromagnetic and electric interference. In addition, DFOS techniques have also demonstrated the superiority in measuring rock strain in the laboratory experiments[8, 9] and on-site rock engineering.[10, 11, 31] The strain data monitored by the DFOS system can reveal the crack development of structures, which means cracks can be identified through the monitored strain data. Specifically, strains of structures will have a sharply increase to form a localized peak at locations where cracks appear, such that detection of cracks can be achieved through identifying localized peaks. A number of studies have conducted the microcrack identification using the strain data

obtained from DFOS systems. For example, Song et al.[12] applied the support vector machine (SVM) and robust principal component analysis (RPCA) techniques to detect microcracks of a steel beam. In their study, RPCA techniques were adopted to denoise the original strain data, and then the denoised strain data was used to train a SVM model to perform microcrack detection. To avoid tedious denoising steps for the original stain data, Song et al.[13] subsequently tried a self-designed 1D-CNN for microcrack detection among the original data without denoising, and a satisfy accuracy (i.e. 98.65%) was obtained. However, the generalization or anti-noise ability for their proposed 1D-CNN was not verified among other noisy datasets. It is essential to note that while a deep learning-based network can achieve high accuracy on testing datasets that are split from the same data as the training dataset, its performance may be poor on noisy data. Therefore, further validation and testing are necessary on a wider range of datasets with varying noise levels to assess the efficacy of a model in identifying microcracks of structures.

In practical engineering applications, the data acquired by sensors or optical fibres are often mixed with Gaussian noises that vary extensively under different environments. Therefore, it is highly necessary to design models that possess the generalization or anti-noise ability, that is to say, to make a model trained with data that has minimal noise achieve high accuracy among other data that has large noise. Another important consideration when designing models is that the length of the 1D strain matrix corresponding to microcrack region is typically short. As a result, the number of convolutional layers that can be used in the model may be limited if the model follows the typical structure with a pooling layer (stride=2) following a convolutional layer. For instance, the number of convolutional layers will not exceed 5 if using the aforementioned typical structure for an input matrix with a size of $32 \times 1$. However, a small number of convolutional layers may limit the nonlinear representation ability of a model, which can affect the generalization ability of the model. Gratifyingly, shortcut connections can solve this limitation by allowing an increase in the number of convolutional layers without degrading the accuracy of the model.[14] Additionally, attention mechanism is also beneficial to enhance the nonlinear representation ability of a model.[15-17]

In this study, a hybrid attention convolutional neural network (HACNN) is developed for rock microcrack identification. To enhance the nonlinear representation ability of the developed model, the shortcut connections are applied to increase the number of convolutional layers, and moreover, a hybrid attention (HA) module is designed and placed following to some certain convolutional layers. But more than that, two training interference strategies, i.e. small mini-batch training and adding dropout in the first convolutional layer, are used to enhance the anti-noise ability of the proposed HACNN. The rest of this

paper is organized as follows. In Section 2, the methodology for microcrack is systematically introduced, and the unique characteristics that make the HACNN possess anti-noise ability are detailed. In Section 3, the anti-noise ability of the HACNN is validated by a serious of comparative experiments among datasets with different signal-to-noise ratios (SNRs), and the features learned by the HACNN are visualized via t-distributed stochastic neighbor embedding (t-SNE) to explain the experimental results. In Section 4, adding dropout to the first different convolutional layers with kernels of different sizes is implemented to analyse the effect of kernel size on the performance of the HACNN. In Section 5, the concluding remarks are drawn from the aforementioned analysis.

## 2. Methodology

A systematic framework, as presented in Fig. 1, is proposed for rock microcrack identification. The proposed methodology consists of three steps: data collection from experiment, dataset establishment, and deep learning model development for microcrack identification using 1D-CNN. Each step is elaborated in the following sections.

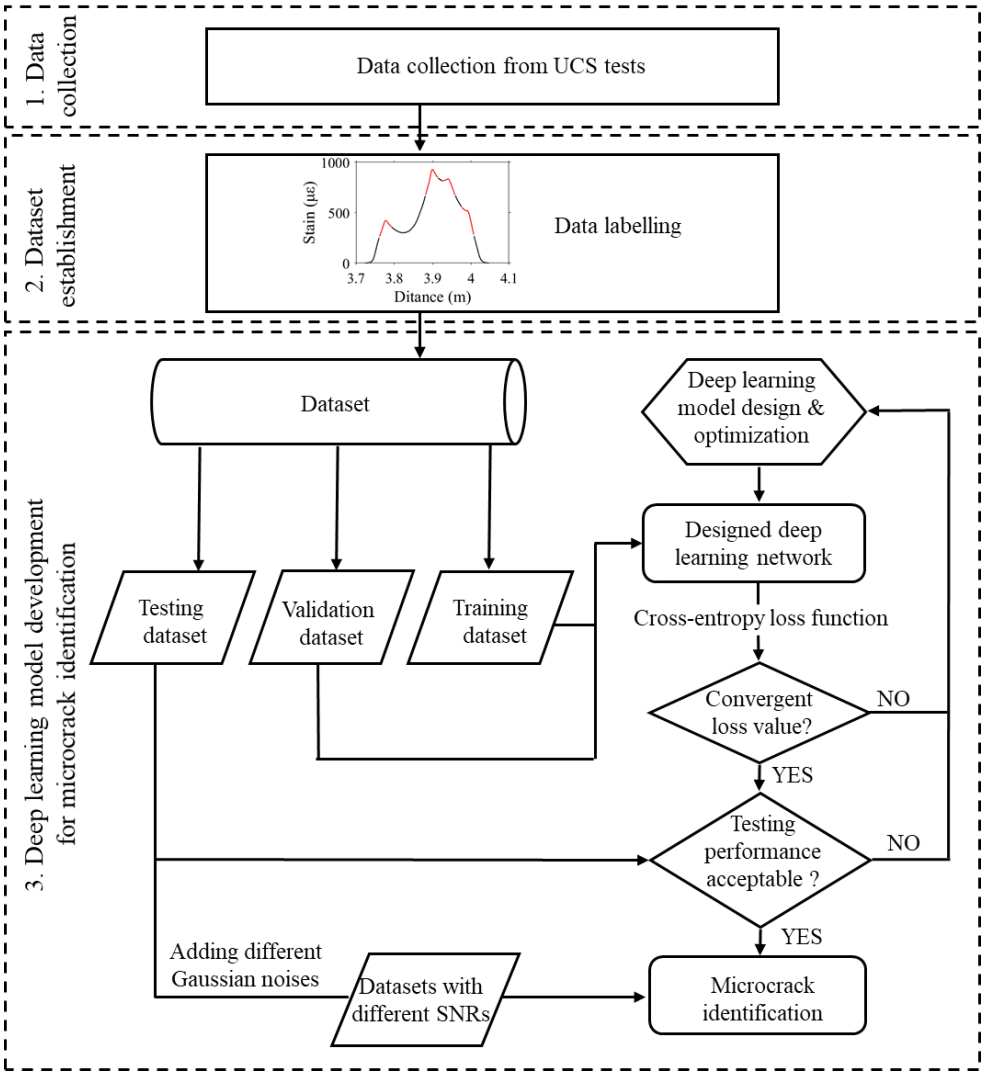



**Fig. 1.** Overall framework of the proposed approach.

### *2.1. Data collection from an experiment*

The rock strain data are acquired from the previous uniaxial compressive strength (UCS) test conducted by the authors on a cylindrical specimen of granite.[8] A granite cylindrical sample with a diameter of 69.2 mm and height of 140.3 mm was tested on the uniaxial compression tester (UCT), as displayed in Fig. 2. A whole fibre was attached to the surface of the specimen forming five hoop fibres (marked as 'H') and a spiral fibre (marked as 'S'), as displayed in Fig. 2. These five hoop fibres marked as '$H_1$~$H_5$' were at an interval of 20 mm, and '$H_1$' and '$H_5$' were located at 30.3 mm and 30 mm away from the top and bottom of the specimen, respectively. The 'S' fibre intersected all hoop fibres at an angle of 20.2°. The whole fibre consisting of 'S' fibre and 'H' fibres was connected to an optical frequency domain reflectometry (OFDR)-based interrogator (Fig. 2) with a spatial resolution of 1 mm, which acquired strain data every 5 s. More details about the experimental process of UCS test can be referred to the authors' previous work.[8]

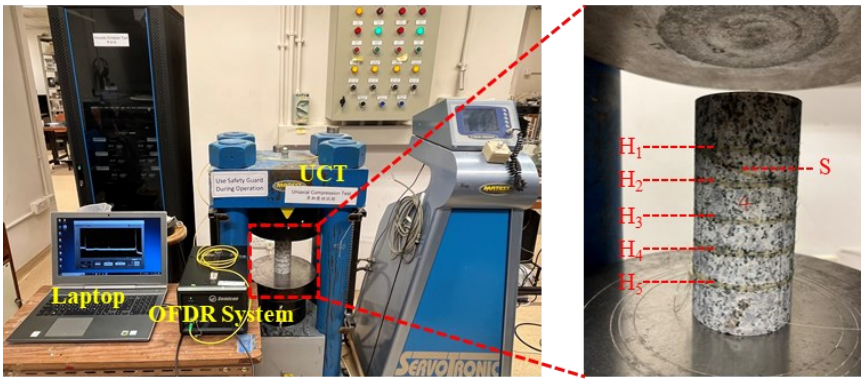



**Fig. 2.** Uniaxial compressive strength test.

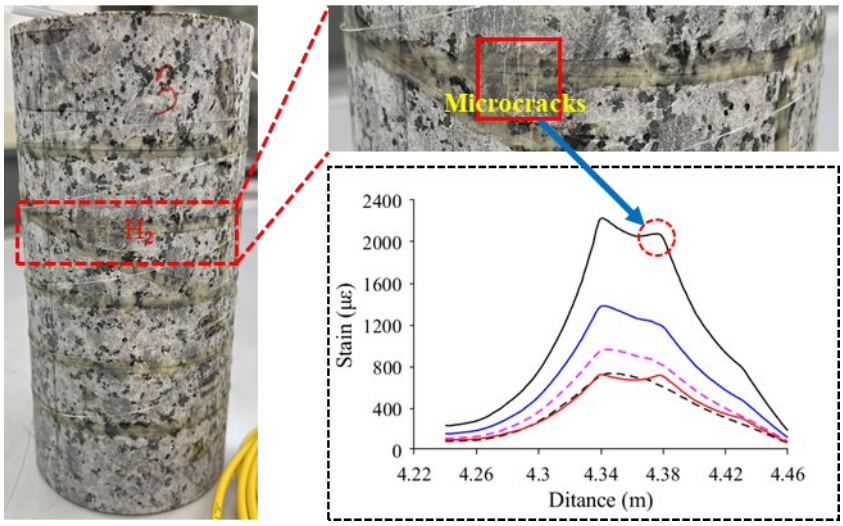



**Fig. 3.** Photograph of the granite specimen after loading.

The maximum axial stress was set to 80.0 MPa during the loading process. Tensile wing cracks were discovered to grow on the surface of the granite specimen, as shown in Fig. 3. Three microcracks were

found at the location of $H_2$ hoop fibre (Fig. 3). The strain signals monitored by $H_2$ hoop fibre had a sharp peak at the location of each microcrack. Moreover, the peak value increased with the crack opening, as displayed in Fig. 3. The strain peak or strain singularity is one key feature for some machine learning algorithms (e.g. SVM[12] and CNN[13]) to identify microcracks.

There were 4, 3, 1, 1, and 1 microcrack appeared at location of $H_1$, $H_2$, $H_3$, $H_4$, $H_5$ hoop fibre when the axial stress came up to maximum of 80.0 MPa. These 10 microcracks were represented by the monitored strain singularities, as shown in Fig. 4. The sample interval of the OFDR-based system was set to 1mm, and part of the monitored strain data by the 6 fibres are presented in Fig. 4.

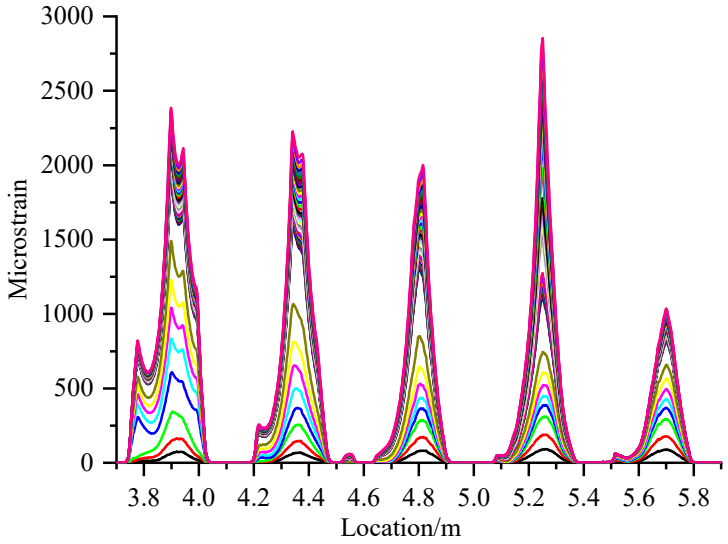



**Fig. 4.** Measured strain profiles from five measuring fibres.

*2.2. Dataset establishment*

The collected data are strain sequence, and Fig. 5 presents the distributed strain fragment along the $H_1$ hoop fibre at a certain moment. In the UCS test, microcracks were discovered at 3.740 m, 3.898 m, 3.943 m, and 3.992 m position, and each position corresponded to a strain peak, as presented in Fig. 5. Thirty-two strain points with the strain peak point as centre (red curve part in Fig. 5) were sampled and labelled as a crack strain sample that was given a positive label of 1. Therefore, each positive crack strain sample was a 32 × 1 matrix. Similarly, thirty-two continuous strain points were sampled from the non-crack region (i.e. black curve part in Fig. 5) as non-crack strain sample that was given a negative label of 0. With this labelling method, a total of 3120 crack strain samples and 3120 non-crack strain samples were obtained using the strain data monitored in the UCS test. Among the 6240 strain matrix samples, 80% of the samples were randomly selected out as training and validation dataset, and the ratio of the training samples to the validation samples is 4:1. The remaining 20% of the samples were set aside for testing. The statistics of the dataset information are presented in Table 1.

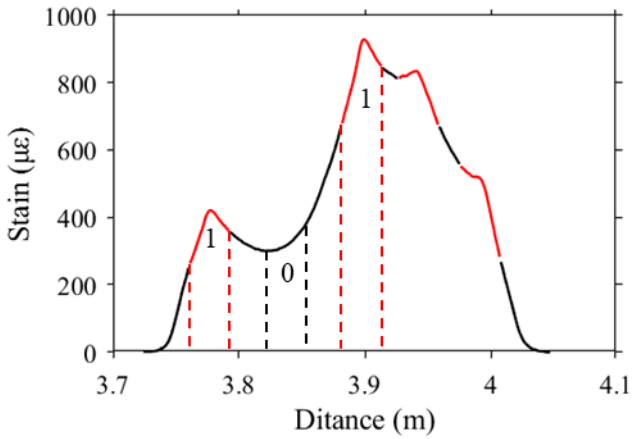



**Fig. 5.** Schematic of the P-zones and N-zones.

**Table 1**
Number of images in the dataset.

| Category | Training and validation (80%) | | Testing (20%) |
|---|---|---|---|
| | Training samples (80%) | Validation samples (20%) | |
| Crack strain sample | 1997 | 499 | 624 |
| Non-crack strain sample | 1997 | 499 | 624 |

### *2.3. The proposed rock microcrack identification approach*

A HACNN, as shown in Fig. 6, is developed to automatically identify microcracks among the established rock strain dataset. The HACNN consist of 11 convolutional (Conv) layers, a dropout layer, 2 pooling layers, 5 HA modules, a SoftMax layer, a fully connected (FC) layer, 11 batch normalization (BN) layers, and 11 Tanh layers. Details of the proposed HACNN and its basic parameters are shown in Table 2. These functional layers can be trained in a supervised way to map each input (32 × 1 matrix) to the marked classification label ($[1\ 0]^T$ or $[0\ 1]^T$).

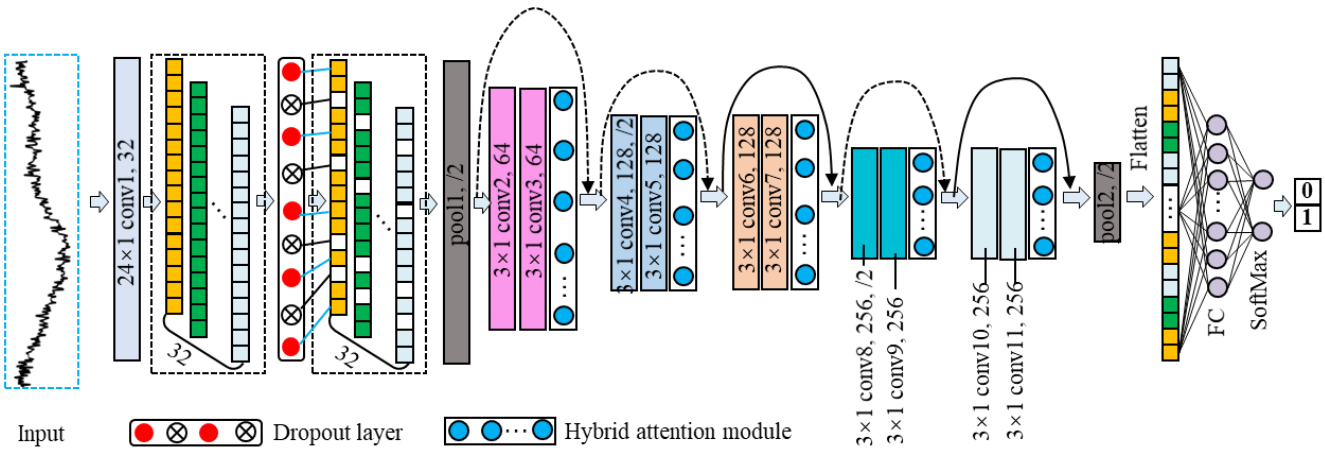



**Fig. 6.** Schematic of the proposed HACNN.

The convolutional layer uses its filters (or kernels) to sweep the input data slice by slice to extract features that characterize the pattern in the input data. The BN layer acts as a normalizer for input data

from each training mini-batch, which reduces internal covariance shift thus accelerating the training process.[18] The Tanh layer, one type of activation function, helps improve the nonlinear representation ability of the HACNN through performing nonlinear transformation for the inputs to learn more complicated nonlinear features. The max pooling layer is used to reduce the spatial size of the extracted data features through performing local max operation (a type of down-sampling operation) over the features, and location-invariant features can be obtained through max operation. The features from the second max pooling layer (marked as pool2 in Fig. 6) need to be flattened to one dimension before they are fed into a FC layer, which acts as a 'classifier' to build the mapping between the learned features and the output. The SoftMax layer weights and exponentially normalizes outputs from the FC layer to generate a vector $[p_1\ p_2]$, which represents the probability that the given input data fragment (32 × 1 matrix) is sampled from crack or non-crack region.

Two techniques are applied when designing the proposed model in order to improve the nonlinear representation ability of the HACNN. The two techniques are as following:

First, the shortcut connections (the curve with arrow in Fig. 6) are used to increase the number of convolutional layers. The typical structure with a pooling layer (stride=2) following a convolutional layer is not used herein, because the number of convolutional layers will not exceed 5 if using this typical structure for the input with a size of 32 × 1. However, small number of convolutional layers cannot guarantee a network to extract abundant nonlinear features, and simply stacking convolutional layers introduces higher training error. Therefore, the depth of the proposed model is enlarged through shortcut connections, which can bring accuracy gains for a network.[14] The number of convolutional layers is increased to 11 using shortcut connections in order to extract more diverse nonlinear features from the input data, and each convolutional layer adopt the 'same' padding to do convolutional operations for its inputs. Same padding prevents the loss of information at the edges of the strain matrix fragment through adding elements with a value of zero around the edges of the strain matrix fragment. Moreover, large kernels with a size of 24 × 1 are used in the first convolutional layer, and small kernels (3 × 1) are used in the subsequent convolutional layers. One purpose of using a large kernel in the first convolutional layer is to suppress high-frequency noise of the input data.

Second, a HA module is designed and placed next to some convolutional layers. It is known that a convolutional layer will blend cross-channel and spatial information together when it extracts informative features. The representation of interests (i.e. targets) can be improved if the meaningful features can be emphasized in channel and spatial dimensions.[16] Therefore, a channel attention module and a spatial module are placed next to a convolutional layer in a sequential manner to boost representation power of the HACNN. More information about the attention module are introduced in Section 2.3.1.

**Table 2**

Details of the proposed HACNN.

| No. | Layer type | Input size | Kernel size/stride | Kernel number | Output size | Padding |
|---|---|---|---|---|---|---|
| 1 | Conv1 | 32×1 | 24×1 / 1 | 32 | 32×32×1 | Yes (same) |
| 2 | Dropout | 32×32×1 | -- | -- | 32×32×1 | -- |
| 3 | BN | 32×32×1 | -- | -- | 32×32×1 | -- |
| 4 | Tanh | 32×32×1 | -- | -- | 32×32×1 | -- |
| 5 | Max Pooling1 | 32×32×1 | 3×1 / 2 | 32 | 32×16×1 | Yes (same) |
| 6 | Conv2 | 32×16×1 | 3×1 / 1 | 64 | 64×16×1 | Yes (same) |
| 7 | BN | 64×16×1 | -- | -- | 64×16×1 | -- |
| 8 | Tanh | 64×16×1 | -- | -- | 64×16×1 | -- |
| 9 | Conv3 | 64×16×1 | 3×1 / 1 | 64 | 64×16×1 | Yes (same) |
| 10 | BN | 64×16×1 | -- | -- | 64×16×1 | -- |
| 11 | HA module | 64×16×1 | -- | -- | 64×16×1 | -- |
| 12 | Tanh | 64×16×1 | -- | -- | 64×16×1 | -- |
| 13 | Conv4 | 64×16×1 | 3×1 / 2 | 128 | 128×8×1 | Yes (same) |
| 14 | BN | 128×8×1 | -- | -- | 128×8×1 | -- |
| 15 | Tanh | 128×8×1 | -- | -- | 128×8×1 | -- |
| 16 | Conv5 | 128×8×1 | 3×1 / 1 | 128 | 128×8×1 | Yes (same) |
| 17 | BN | 128×8×1 | -- | -- | 128×8×1 | -- |
| 18 | HA module | 128×8×1 | -- | -- | 128×8×1 | -- |
| 19 | Tanh | 128×8×1 | -- | -- | 128×8×1 | -- |
| 20 | Conv6 | 128×8×1 | 3×1 / 1 | 128 | 128×8×1 | Yes (same) |
| 21 | BN | 128×8×1 | -- | -- | 128×8×1 | -- |
| 22 | Tanh | 128×8×1 | -- | -- | 128×8×1 | -- |
| 23 | Conv7 | 128×8×1 | 3×1 / 1 | 128 | 128×8×1 | Yes (same) |
| 24 | BN | 128×8×1 | -- | -- | 128×8×1 | -- |
| 25 | HA module | 128×8×1 | -- | -- | 128×8×1 | -- |
| 26 | Tanh | 128×8×1 | -- | -- | 128×8×1 | -- |
| 27 | Conv8 | 128×8×1 | 3×1 / 2 | 256 | 256×4×1 | Yes (same) |
| 28 | BN | 256×4×1 | -- | -- | 256×4×1 | -- |
| 29 | Tanh | 256×4×1 | -- | -- | 256×4×1 | -- |
| 30 | Conv9 | 256×4×1 | 3×1 / 1 | 256 | 256×4×1 | Yes (same) |
| 31 | BN | 256×4×1 | -- | -- | 256×4×1 | -- |
| 32 | HA module | 256×4×1 | -- | -- | 256×4×1 | -- |
| 33 | Tanh | 256×4×1 | -- | -- | 256×4×1 | -- |
| 34 | Conv10 | 256×4×1 | 3×1 / 1 | 256 | 256×4×1 | Yes (same) |
| 35 | BN | 256×4×1 | -- | -- | 256×4×1 | -- |
| 36 | Tanh | 256×4×1 | -- | -- | 256×4×1 | -- |
| 37 | Conv11 | 256×4×1 | 3×1 / 1 | 256 | 256×4×1 | Yes (same) |
| 38 | BN | 256×4×1 | -- | -- | 256×4×1 | -- |
| 39 | HA module | 256×4×1 | -- | -- | 256×4×1 | -- |
| 40 | Tanh | 256×4×1 | -- | -- | 256×4×1 | -- |
| 41 | Max Pooling2 | 256×4×1 | 3×1 / 2 | 256 | 256×2×1 | Yes (same) |
| 42 | FC | 256×2×1 | -- | 1024 | 1024×1 | -- |
| 43 | SoftMax | 1024×1 | -- | 2 | 2×1 | -- |

To make the proposed HACNN have anti-noise or generalization ability, two training interference strategies, i.e. small mini-batch training and adding dropout in the first convolutional layer, are adopted. More information about the two training interference strategies is introduced in Section 2.3.2.

### *2.3.1. Hybrid attention module*

Attention mechanism has brought substantial performance gain for various CNN-based models. The trade-off between performance and complexity of a model can reflect the pros and cons of a used attention module. A good designed attention module should help a network achieve better performance at the cost of lower network complexity. The convolutional block attention module (CBAM)[16] is a popular attention module since it brings satisfactory performance gain for some models. An important characteristic of the CBAM is that it combines channel attention and spatial attention and arranges them in a scientific way, i.e. sequential manner with spatial attention following channel attention. The channel attention, concentrating on learning 'what' is meaningful feature for a given input, is complementary to spatial attention, which concentrates on learning 'where' to attend. However, the use of both max-pooling and average-pooling in the channel attention module of CBAM makes the CBAM not lightweight enough. Additionally, the channel attention module of the CBAM have to reduce the dimensions of its intermediate inputs and then restore it to improve the non-linearity of the intermediate inputs. However, this dimensionality reduction process is inefficient and destroys the direct correspondence between channel and its weight.[17] To address aforementioned limitations of channel attention module, Wang et al.[17] proposed an efficient channel attention (ECA) module. This ECA module achieves cross-channel interaction through efficiently implementing a 1D convolution with self-adaptive kernel size, and it avoids the side effect brought by dimensionality reduction.

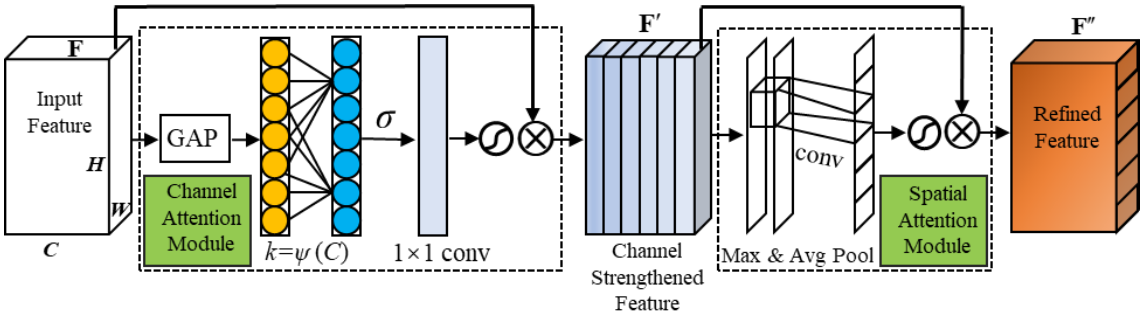



**Fig. 7.** The proposed hybrid attention (HA) module.

Taking a cue from the advantage of CBAM and ECA module, a HA module suitable for processing 1D signal data, as presented in Fig. 7, is designed through replacing the channel attention module of CBAM with ECA module. In comparison with CBAM, the proposed HA module adopts ECA module to

strengthen features in channel dimensions, and the 2D convolutional layer of the spatial channel attention module of CBAM is modified to a 1D convolutional layer (kernel size=1, filters=1) in the HA module. As can be seen from Fig. 7, the intermediate feature map $\mathbf{F} \in \mathbb{R}^{C\times W\times H}$ first goes through a global average pooling (GAP) to obtain the aggregated feature $\boldsymbol{y} \in \mathbb{R}^{C}$, and the c-th channel of $\boldsymbol{y}$ is calculated as:

$$y_C = \frac{1}{H\times W}\sum_{i=1}^{H}\sum_{j=1}^{W}\mathbf{F}_C(i,j) \tag{1}$$

where $C$, $H$, and $W$ are the total channel number, the height, and the width of $\mathbf{F}$, respectively.

Then the feature $\boldsymbol{y}$ is put into a 1D convolutional layer with an adjustable kernel size of $k$ to capture local cross-channel interaction. The $k$ is the coverage of local cross-channel interaction and is adaptively determined according to channel number $C$. The $k$ is computed as following:

$$k = \psi(C) = \left|\frac{\log_2(C)}{\gamma} + \frac{b}{\gamma}\right|_{odd} \tag{2}$$

where $|x|_{odd}$ obtains the nearest odd number of $x$. The γ and b are constants, and they are set to 2 and 1, respectively, following a study by Wang et al.[17]

The features extracted by the 1D convolutional layer subsequently goes through a sigmoid ($\sigma$) operation. The 1D convolution operation combining the σ operation enables the fast computation of aggregated channel weights. The channel-strengthened features, $\mathbf{F}'$, are computed as:

$$\mathbf{F}' = \mathbf{F} \otimes \sigma(\mathrm{Conv1D}_k(\boldsymbol{y})) \tag{3}$$

where $\otimes$ is element-wise multiplication.

And then, the spatial attention module (Fig. 7) starts processing the $\mathbf{F}'$. Specifically, an average-pooling and a max-pooling operation act on the $\mathbf{F}'$ along the channel axis to obtain average-pooled features and max-pooled features. Then, a 1D convolutional layer concatenates them to generate spatial-strengthened features $\boldsymbol{\chi}$, which is computed as:

$$\boldsymbol{\chi} = \sigma(\mathrm{Conv1D}(\mathit{AvgPool}(\mathbf{F}'); \mathit{MaxPool}(\mathbf{F}'))) \tag{4}$$

The final refined features, $\mathbf{F}''$, are gained by multiplying $\boldsymbol{\chi}$ with $\mathbf{F}'$ as:

$$\mathbf{F}'' = \mathbf{F}' \otimes \boldsymbol{\chi} \tag{5}$$

In summary, the channel attention module of the HA module captures channel relationship of the input features, while the spatial attention module of the HA module captures the spatial dependencies between any two elements through assigning different weights to elements of the feature matrix. Therefore, the useful features are emphasized, while the less useful ones are suppressed when the intermediate feature

map **F** flows past the proposed HA module, which contributes to the distinction between crack and non-crack strain matrix.

*2.3.2. Training techniques for the proposed model*

The idea is to make the HACNN have a certain adaptation ability for different noisy data through training it with training interference. In this study, two training interference strategies are adopted.

First, the strategy that adds dropout to the first convolutional layer is used (Fig. 6). Srivastava et al. [19] mentioned that adding dropout to convolutional layers was equivalent to adding noise to the inputs, which brought additional performance gain for a CNN. Therefore, this strategy is applied herein to improve the generalization ability of the proposed HACNN. It is known that the dropout rate of 0.5 produces greatest regularization effect.[19, 20] The dropout rate higher than 0.5 removes too many parameters without improving regularization effect for a network. Thus, it is better to choose a value less than 0.5 since this can not only reasonably interfere with the training (i.e. provide noisy inputs) but also maintain a certain regularization effect for a network. The dropout rate is set to 0.1 by trial and error in this study, and details about the effect of dropout rate value on the performance of the HACNN are presented in Section 3.3. Compared with a small kernel (3 × 1) that generally make 1 or 2 neurons deactivated at a time, the large kernels (24 × 1) of the first convolutional layer of the HACNN can ensure more neurons to be deactivated at the same time, which achieves a better effect of modelling the addition of noises. More details about the effect of kernel size on the performance of the HACNN are presented in Section 4.

Second, small mini-batch training is used. A study conducted by Keskar et al.[21] demonstrated a network can possess a better generalization ability when using small batch size to calculate an approximation of the gradient and then update weights. Moreover, as is known, a BN layer computes the expected value $\mathrm{E}[y^{l(i,j)}]=\frac{1}{p\,|b|}\sum_{k=1}^{|b|}\sum_{j=1}^{p}y_k^{l(i,j)}$ for each $y^{l(i,j)}$ from the given input $\mathbf{y}^l=(y^{l(i,1)},\ldots, y^{l(i,j)},\ldots, y^{l(i,p)})$. For two batch size of $b_1$ and $b_2$, if $|b_1|\gg|b_2|$, following the central limit theorem, $\mathrm{Var}(\mathrm{E}[y^{l(i,j)}]_{b_1}<\mathrm{Var}(\mathrm{E}[y^{l(i,j)}]_{b_2}$, which indicates the average value of the given input has larger fluctuations when using small batch size. It means if the number of samples contained in a mini-batch is smaller during the training process, the fluctuation range of the average value will be larger, and this satisfies the requirement of adding interference to the average value and variance of the model being trained. If a model can converge to a satisfactory loss value even in the case of large fluctuations of the average value in a BN layer, the anti-noise ability of the model should be stronger.

It is suggested the batch size should be at least no smaller than the number of classes to be classified, in order to prevent the average value and variance in a BN layer from deviating too much from those values of the overall training samples. Otherwise, the average value and variance in the BN layer for the mini-batch samples will always deviate from those values for the overall training sample, which is not conducive to the training of a model. Therefore, the batch size is set to 10 in this study through trial and error approach. More details about the effect of batch size on the performance of the HACNN are presented in Section 3.3.

### *2.3.3. Model training*

The binary cross-entropy loss function was selected to calculate the gradient so as to update the weights in the model, and it is defined as follows:

$$L = -\frac{1}{N}\sum_{i=1}^{N}\left[ p_i \log p_i^c + (1-p_i)\log(1-p_i^c) \right] \tag{6}$$

where $p_i$ = 1/0 indicates the *i*-th input strain matrix is sampled from crack or non-crack region, $p_i^c$ is the probability output by SoftMax and represents the possibility of the *i*-th input strain matrix is the crack strain matrix, and *N* is the number of training samplings.

The initial learning rate was set to 0.001, and the Adam optimizer was adopted to update the weights. The dropout rate value and batch size were set to 0.1 and 10, respectively, through trial and error approach. All the inputs are crack or non-crack strain matrix with a size of 32 × 1.

## 3. Experimental results

### *3.1. Microcrack identification performance on the testing dataset*

The proposed HACNN was executed using the training parameters introduced in Section 2.3.3 on a desktop equipped with one Intel Core i9-12900K CPU, one NVIDIA RTX 3090 GPU, and 64 GB of RAM. The computational software environment for the desktop was configured with TensorFlow framework.

The training and validation process continued for a maximum of 50 epochs. As can be seen from Fig. 8, although the loss value of the HACNN has large fluctuations in the early epochs, the loss value of the HACNN drops to a lower value in the last 4 epochs among the training dataset, and converges to a lower value among the validation dataset. Therefore, the HACNN trained for 50 epochs is used to conduct experiments.

The accuracy which describes the average identification correctness is used to evaluate the identification performance of the proposed HACNN. The accuracy is computed as:

accuracy: (TP+TN)/(TP+TN+FP+FN)

where TP (i.e. true positive) is the number of correctly classified crack strain matrix; TN (i.e. true negative) is the number of correctly classified non-crack strain matrix; FP (i.e. false positive) is the number of falsely classified non-crack strain matrix; FN (i.e. false negative) is the number of falsely classified crack strain matrix.

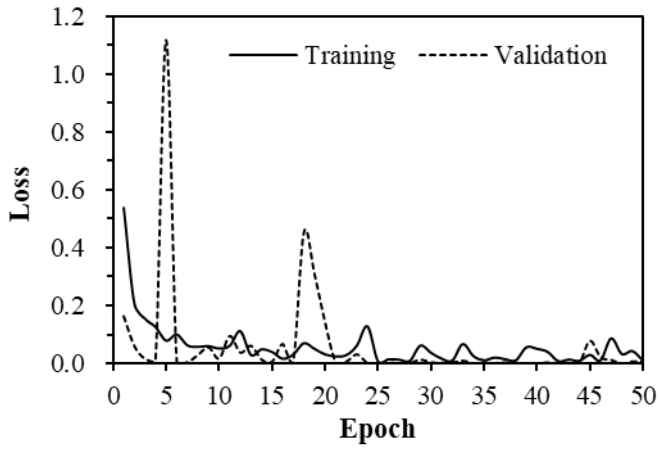



**Fig. 8.** Loss curve during training and validation process.

The proposed HACNN achieves a classification accuracy of 99.92% with 1 error (i.e. 1 FN) among the testing dataset with 1248 strain matrix samples. Fig. 9 shows a data fragment, and all the crack stain matrixes (red parts of the curve) are correctly classified. As can be seen from Fig. 4 and 9, the original data, including the training, validation, and testing dataset, are clean with less noise. Therefore, it is understandable that the HACNN has high classification accuracy on the testing dataset. However, the goal of this study is to make the HACNN have better performance on real noisy data even when using clean data to train the HACNN. The adaption ability of the trained HACNN for noisy data are discussed in Section 3.2.

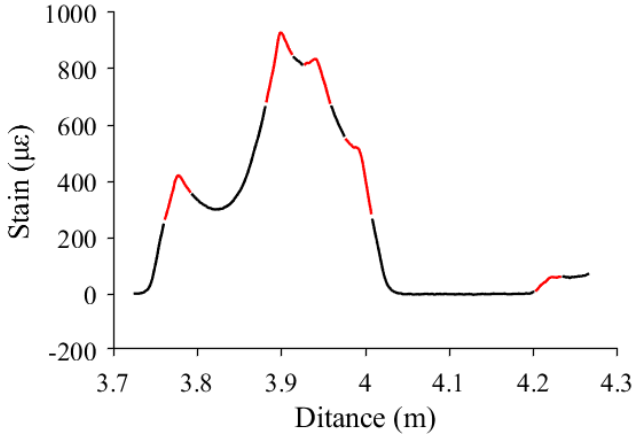



**Fig. 9.** A monitored strain sample that is correctly identified by HACNN.

### *3.2. Performance comparisons under noisy environment*

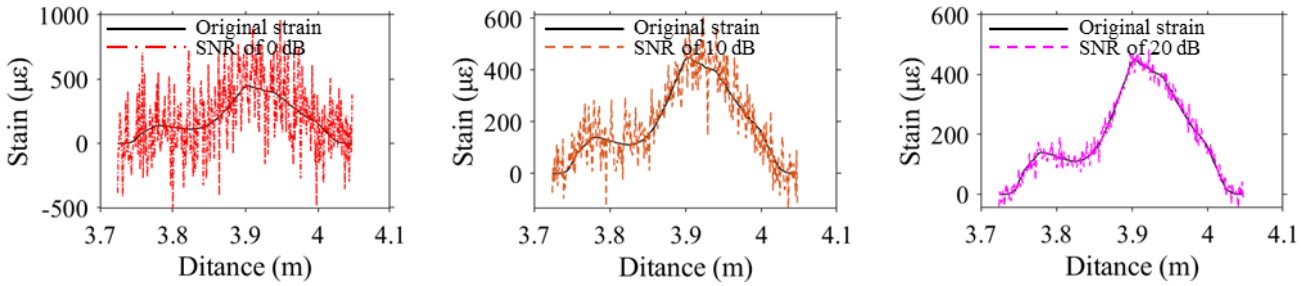
Training the HACNN on clean data and evaluating their performance on noisy data is a common

practice in engineering and data science, since real-world data collected from sensors or optical fibres are typically subject to various types of noise, such as Gaussian noise, which vary widely depending on the environment. The SNR is universally used to compare the desired signal intensity to the background noise intensity, and is computed as:

$$\mathrm{SNR} = 10\log_{10}\left(\frac{P_{\mathrm{signal}}}{P_{\mathrm{noise}}}\right) \tag{7}$$

where $P_{\mathrm{noise}}$ and $P_{\mathrm{signal}}$ are the power of noises and signals, respectively. The unit of SNR is dB.

In this study, the additive white Gaussian noises are added to the established testing dataset to compose signals with different SNRs, and the composite noisy signals with a SNR ranging from 0 to 20 dB are used to investigate the anti-noise ability of the proposed HACNN. Fig. 10 presents a monitored signal fragment and its corresponding composite noisy signals with SNR of 0, 10, and 20 dB. It can be seen that the composite signals with SNR of 0 fluctuate violently around the original monitored signal.




(a) (b) (c)

**Fig. 10.** A monitored strain sample that is adding Gaussian noises with different SNR values.

To demonstrate the superiority of the proposed HACNN in terms of anti-noise ability, the accuracy of the HACNN is compared with that for the support vector machine (SVM), multilayer perceptron (MLP), and the CNN proposed by Song et al.[13] (denoted as CNN-Song). As introduced in Section 2.3.3, the initial learning rate, batch size, and dropout rate for the HACNN are set to 0.001, 10, and 0.1, respectively. The radial basis function is selected as kernel function of the SVM. The penalty parameter ($C$) is set to 100, and kernel parameter (gamma) is set to 'scale' mode during training. Under the 'scale' mode, the gamma is calculated by 1/(n_features * X_train.var()), where the n_feature is the dimension of input data (32 for the established data) and the X_train.var() is the variance of the training data. The hidden layer number of the MLP is set to 3, and these 3 layers have 24, 16, and 8 neurons, respectively. The initial learning rate and batch size for the MLP are the same as those for the HACNN. Fig. 11 and Table 3 present the accuracy of the four mentioned models among the composite signals with different SNRs. It should be noted that each of the four mentioned models are trained 10 times using the testing data that is without adding noises, and the displayed accuracy in Fig. 11 and Table 3 is obtained from the best model among

the 10 times.

As can be seen from Fig. 11 and Table 3, all the four models achieve accuracies above 99% among the testing data, and all the four models except the CNN-Song achieve accuracies above 90% among the composite noisy dataset with SNR of 20. However, the accuracy of the SVM suffers from sharp decrease with the decrease of SNR, and the accuracy is below 60% when the SNR of the composite noisy dataset is 0 dB. The accuracy of the CNN-Song is always below 72% when SNRs of the composite noisy datasets range from 0 to 20 dB, although it achieves an accuracy of 99.92% on the testing dataset. The poor robustness against noise of the CNN-Song may be because the strides of all the convolutional layers are set to 1 and no pooling layers follow the convolutional layers to reduce the size of input data. This makes a large number of parameters need to be learned and thus results in overfitting of the CNN-Song during training. In comparison with the SVM and the CNN-Song, the MLP achieves relatively satisfactory accuracies on the composite noisy datasets with different SNRs. Nevertheless, the accuracies of the MLP are still lower than those for the proposed HACNN. The HACNN achieves an accuracy above 80% when the SNR is larger than 10, and it achieves an accuracy at closer to 70% when the SNR equals to 0. In summary, the proposed HACNN has a better anti-noise ability.

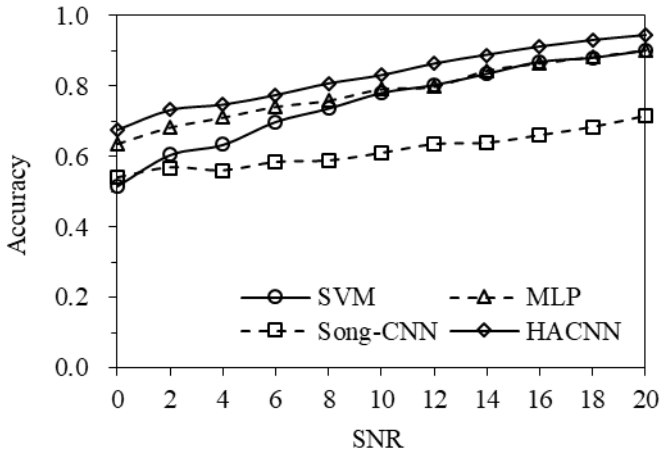



**Fig. 11.** Accuracy of different identification models on datasets with different SNR values.

**Table 3**

The accuracy of different models under different SNRs.

| Model | Testing dataset | SNR (dB) | | | | | |
|---|---|---|---|---|---|---|---|
| | | 20 | 16 | 12 | 8 | 4 | 0 |
| SVM | 99.28% | 90.06% | 86.86% | 80.29% | 73.64% | 63.38% | 51.68% |
| MLP | 99.92% | 90.14% | 86.46% | 79.81% | 75.88% | 71.07% | 63.62% |
| CNN-Song | 99.92% | 71.39% | 65.95% | 63.38% | 58.73% | 55.93% | 53.93% |
| HACNN | 99.92% | 94.39% | 91.19% | 86.38% | 80.77% | 74.76% | 67.63% |

For performance comparisons, the receiver operating characteristics (ROC) curves for the four models

among the composite dataset with SNR of 20 dB are illustrated in Fig. 12. The horizontal and vertical coordinates of the Fig. 12 are false positive rate (FPR) (FPR=FP/(FP+TN)) and true positive rate (TPR) (TPR=TP/(TP+FN). The closer the ROC curve is to the upper left corner of the ROC graph, the better the performance of the classifier is. Moreover, a classifier possesses good classification performance if the area under curve (AUC) is closer to 1. The results show that the proposed HACNN performs better than others among the composite dataset with SNR of 20 dB.

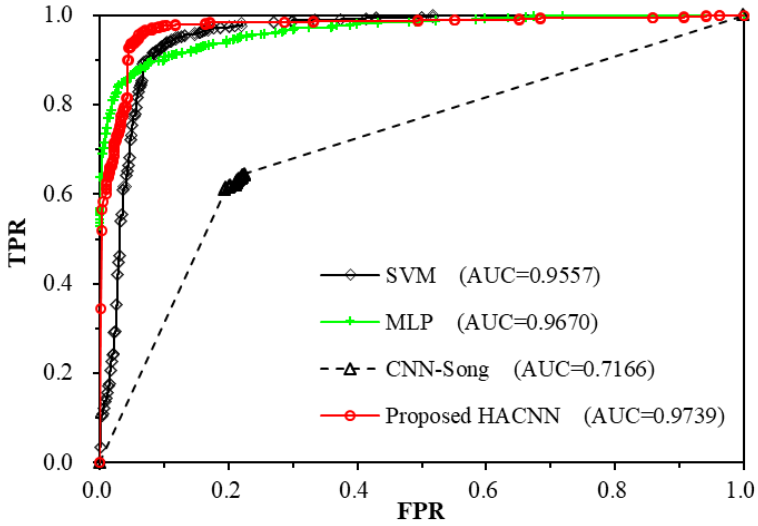
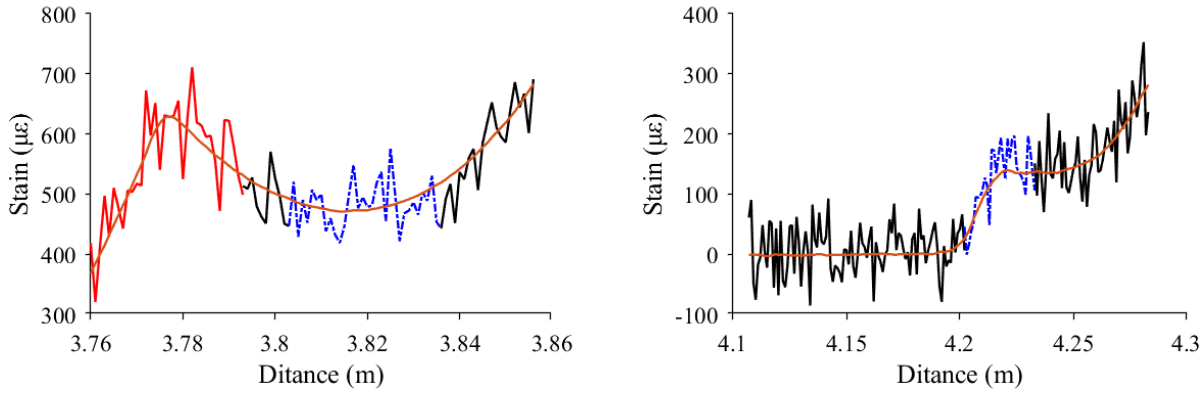



**Fig. 12.** ROC curve for different identification models.

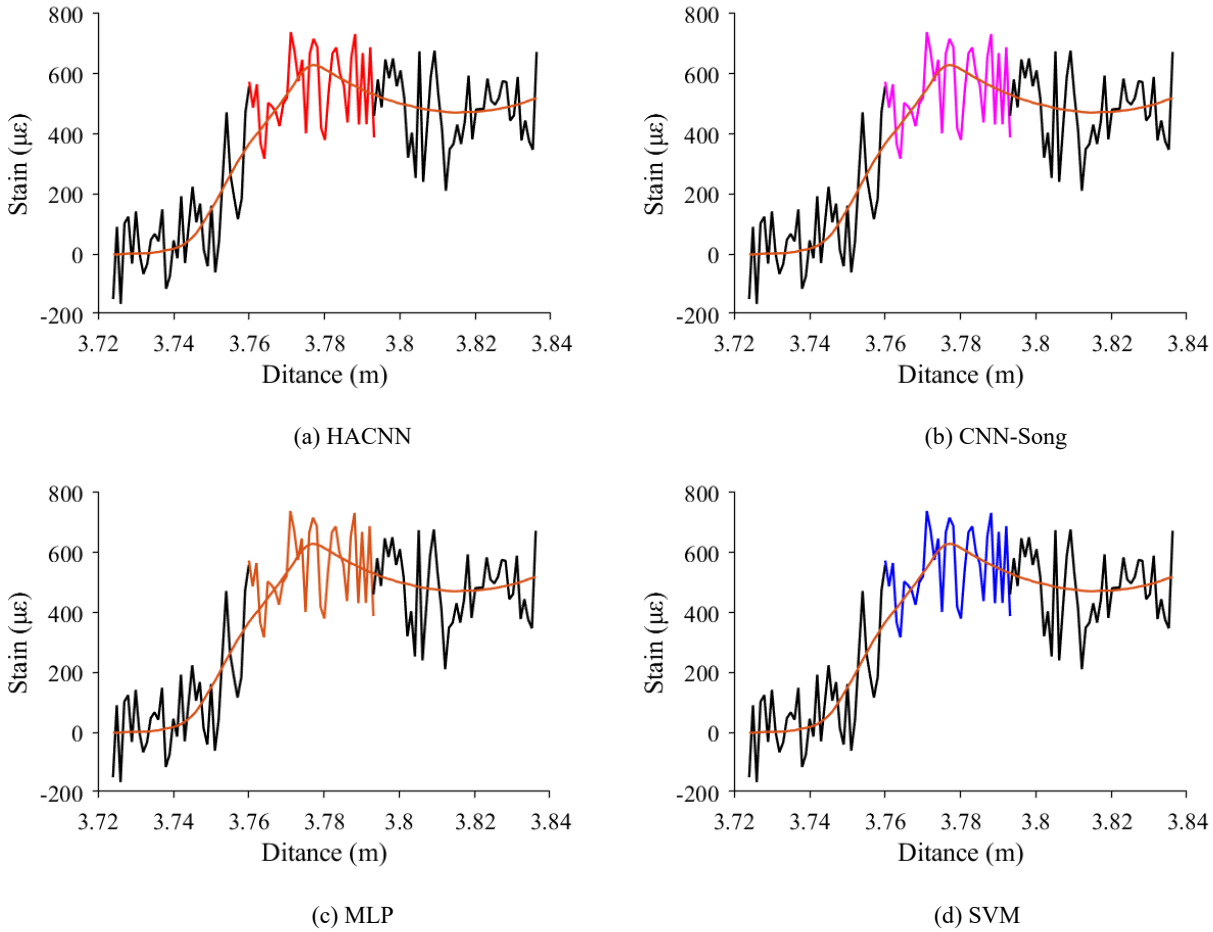



(a) (b)

**Fig. 13.** Misidentified examples: (a) strain data fragment misidentified as microcrack (FP) and (b) strain data fragment misidentified as non-crack (FN).

Fig. 13 presents two samples misidentified by the HACNN. Both samples are collected from the composite dataset with SNR of 20 dB. Fig. 13a shows the strain matrix is sampled from non-crack region, while it is misidentified as crack strain matrix by the HACNN. The strain data fragment of the original signals (dark orange curve) at the trough of wave is in non-crack region (Fig. 13a), while this data fragment has a relatively regular fluctuation with obvious local peak after adding Gaussian noises, as shown by the data fragment with blue colour in Fig. 13a. The shape of the blue data fragment is similar

to that of the red data fragment in crack region, which results in the misidentification of the data fragment with blue colour. Fig. 13b shows that the strain matrix is sampled from the crack region, while it is misidentified as non-crack strain matrix by the HACNN. The strain data fragment of the original signals (dark orange curve) at the local peak of curve is in crack region (Fig. 13b), while this data fragment fluctuates irregularly after adding additive white Gaussian noise, as shown by the data fragment with blue colour in Fig. 13b. The shape of the blue data fragment is similar to that of its neighboring data fragment in non-crack region (black curve), which causes that the data fragment with blue colour is misidentified as non-crack. To solve this misidentification problem, the adversarial training[22] and/or soft thresholding[23] techniques can be embedded into the HACNN to weaken the effect of noise to better distinguish the classes of data matrix fragment.



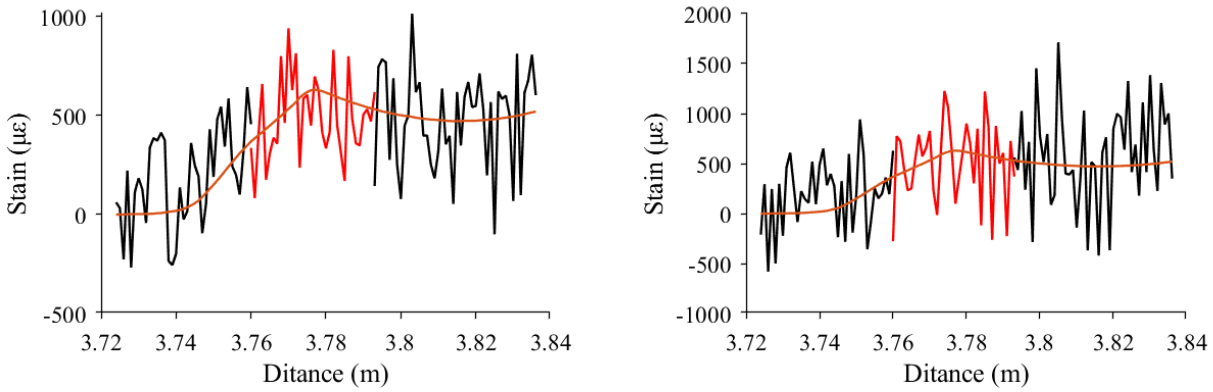

(a) HACNN (b) CNN-Song

(c) MLP (d) SVM

**Fig. 14.** Identified results of different models among a same strain data fragment with SNR of 10.

The identification results of the four crack identification models for the same strain data fragment with SNR of 10 are displayed in Fig. 14. The HACNN can accurately identify the strain sequence corresponding to microcrack (red curve in Fig. 14a), whilst the CNN-Song, MLP model, and SVM model classify this same strain sequence as non-crack strain sequence. Fig. 15 demonstrates that HACNN can accurately identify the microcrack strain sequence (red curve) for an initial strain data fragment (dark

orange curve) with SNRs of 5 and 0. These identification results indicate the HACNN has better robustness against noise than the CNN-Song, MLP model, and SVM model.




(a) SNR of 5 (b) SNR of 0

**Fig. 15.** Identified results of HACNN among strain data fragments with different SNRs.

### *3.3. The necessity of dropout, min-batch training, and hybrid attention module*

As introduced in Section 2.3.2, it is not the fact that the smaller the batch size, the more obvious the anti-noise ability of the model is improved. It had better set the batch size larger than the number of classes to be classified. Table 4 shows the effect of the batch size on the performance of the HACNN. Compared to the large batch size of 50 and 100, the batch size of 10 for the HACNN increases the accuracy by 3.82% and 3.99%, respectively, among the composite dataset with SNR of 0, and increases the accuracy by 3.11% and 3.85%, respectively, among the composite dataset with SNR of 4. Moreover, the accuracy for the HACNN with a batch size of 10 is higher than that for the HACNN with a batch size of 5 among the composite datasets with SNR ranging from 8 to 20. Taken together, the HACNN with a batch size of 10 can achieve better results among the composite datasets with different SNRs compared to other models.

**Table 4**

The accuracy (%) of HACNN trained with different batch size under dropout rate of 0.2.

| Batch size | Testing dataset | SNR (dB) | | | | | |
|---|---|---|---|---|---|---|---|
| | | 20 | 16 | 12 | 8 | 4 | 0 |
| 5 | 99.92% | 92.47% | 90.06% | 84.38% | 79.49% | 74.84% | 67.95% |
| 10 | 99.92% | 93.27% | 90.46% | 85.26% | 79.81% | 74.76% | 67.71% |
| 15 | 99.92% | 92.87% | 89.74% | 84.62% | 79.65% | 73.56% | 66.67% |
| 20 | 99.92% | 92.39% | 87.50% | 82.69% | 79.01% | 74.04% | 67.79% |
| 50 | 99.92% | 91.75% | 88.31 | 84.62% | 78.41% | 71.65% | 63.89% |
| 100 | 99.92% | 93.27% | 89.36 | 85.02% | 78.59% | 70.91% | 63.72% |

Table 5 shows the effect of the dropout rate value on the performance of the HACNN. It can be seen from Table 5 that the accuracy of the HACNN decrease to relatively low values (95.83% and 92.71%),

even on the original testing dataset when using the dropout rate of 0.8 and 0.9. It is consistent with the analysis in Section 2.3.2 that higher dropout rate removes too many parameters and thus reduces the performance of the HACNN. Compared with the HACNN whose dropout layer is removed, the HACNN with a dropout rate of 0.1 increase the accuracy by 3.07%, 2.93%, 3.50%, 3.66%, 3.57%, 2.91% among the composite datasets with SNR ranging from 0 dB to 20 dB, respectively. Taken together, the HACNN with a dropout rate of 0.1 can attain better results among the composite noisy datasets compared to other models.

**Table 5**
The accuracy of HACNN trained with different dropout rate values under batch size of 10.

| Dropout rate | Testing dataset | SNR (dB) | | | | | |
|---|---|---|---|---|---|---|---|
| | | 20 | 16 | 12 | 8 | 4 | 0 |
| Non-dropout layer | 99.92% | 91.48% | 87.62% | 82.72% | 77.27% | 71.83% | 64.56% |
| 0.1 | 99.92% | 94.39% | 91.19% | 86.38% | 80.77% | 74.76% | 67.63% |
| 0.2 | 99.92% | 93.27% | 90.46% | 85.26% | 79.81% | 74.76% | 67.71% |
| 0.3 | 99.84% | 91.19% | 86.94% | 82.53% | 78.13% | 71.88% | 66.83% |
| 0.4 | 99.84% | 91.43% | 88.62% | 85.58% | 80.85% | 73.56% | 68.43% |
| 0.5 | 99.84% | 93.03% | 89.74% | 84.21% | 78.77% | 73.08% | 66.83% |
| 0.6 | 98.88% | 92.15% | 89.58% | 85.34% | 80.37% | 74.12% | 67.31% |
| 0.7 | 99.36% | 91.03% | 86.22% | 81.49% | 78.21% | 70.59% | 65.54% |
| 0.8 | 95.83% | 90.30% | 85.82% | 82.45% | 77.16% | 71.47% | 64.66% |
| 0.9 | 92.71% | 88.06% | 85.50% | 82.21% | 79.57% | 73.64% | 63.22% |

Table 6 shows the effect of the HA module on the performance of the HACNN. The HACNN that removes HA modules is denoted as HACNN-Non-HA. Compared to the accuracy of the HACNN-Non-HA, the accuracies of the HACNN increase by 1.44%, 4.49%, 2.73%, 2.89%, 3.69%, 3.93% among the composite datasets with SNR ranging from 0 dB to 20 dB, respectively. Therefore, the HA module improves the nonlinear representation and anti-noise abilities of the HACNN.

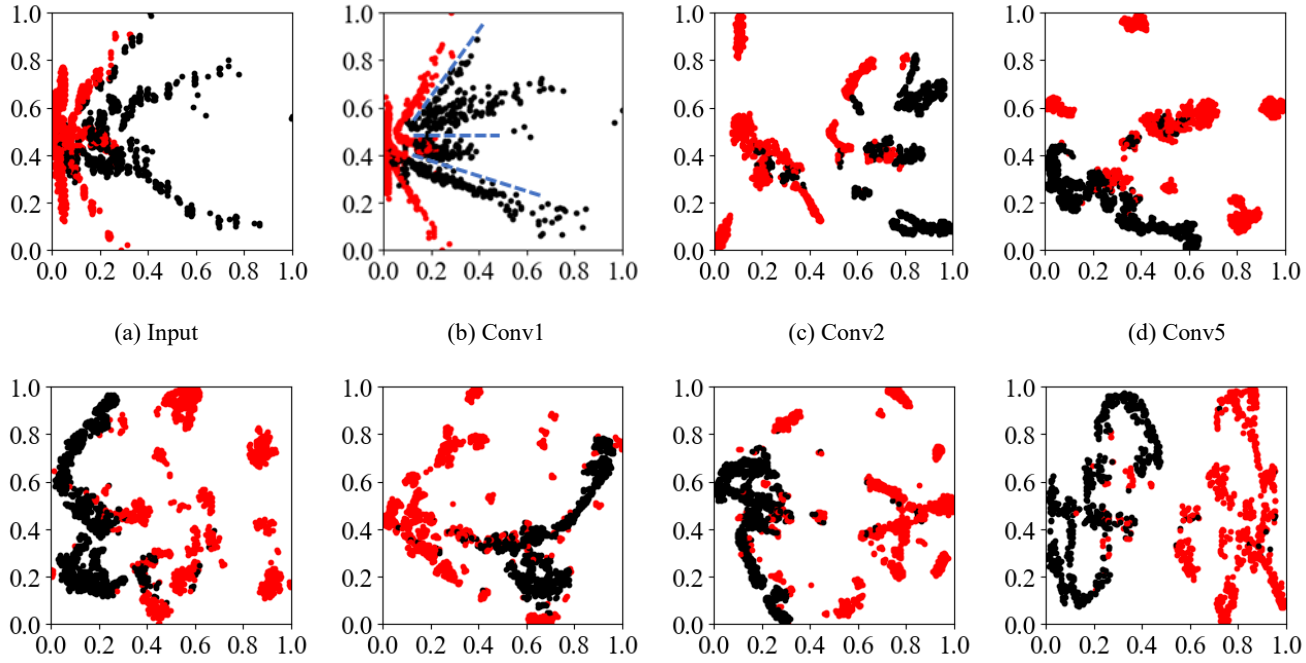
**Table 6**
The accuracy of the HACNN and the HACNN without HA modules.

| Model | Testing dataset | SNR (dB) | | | | | |
|---|---|---|---|---|---|---|---|
| | | 20 | 16 | 12 | 8 | 4 | 0 |
| HACNN-Non-HA | 99.60% | 90.46 | 87.50% | 83.49% | 78.04% | 70.27% | 66.19% |
| HACNN | 99.98% | 94.39% | 91.19% | 86.38% | 80.77% | 74.76% | 67.63% |

### *3.4. Network visualization*

To understand the inner operating mechanism of the proposed HACNN, t-SNE[24] method is used to visualize the feature distribution of the composite noisy samples with SNR of 20 dB extracted from the HACNN. Fig. 16 presents the mentioned feature distribution obtained from some convolutional layers

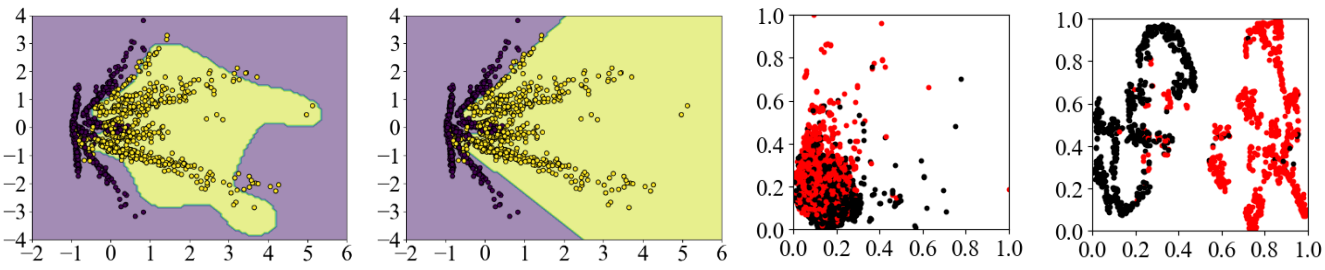
and the FC layer of the HACNN. Comparing the feature distribution of the Input layer to that of the Conv1 layer, the features start to become divisible, as can be seen from the blue dashed line in Fig. 16b. With the layer going deeper, the features become more and more divisible. The feature points of each category start to cluster together in Conv2 layer (Fig. 16c). Moreover, feature points with the same attributes in the same category will cluster as well, as can be seen from the small cluster of red or black feature points in Fig. 16c-h. In the FC layer, most of the black feature points cluster separately from the red feature points that also cluster together, which make the category of feature points easy to be divided. Nevertheless, misclassification phenomena occur, that is to say, a few red feature points are mixed into the cluster of black feature points, and vice versa (Fig. 16h). The misclassification phenomena can be mitigated or even eliminated through improving the anti-noise ability of the HACNN.



(a) Input (b) Conv1 (c) Conv2 (d) Conv5

(e) Conv7 (f) Conv9 (g) Conv11 (h) FC

**Fig. 16.** Feature representations of the HACNN among the composite noisy dataset with SNR of 20 dB. Solid black dots and red dots are crack and non-crack feature points, respectively.



(a) SVM (b) MLP (c) CNN-Song (d) Proposed HACNN

**Fig. 17.** Feature representations of four models among the composite noisy dataset with SNR of 20 dB.

The feature distributions extracted from the SVM, the MLP, and the CNN-Song are also visualized,

as displayed in Fig. 17. Some crack feature points (yellow solid points) near the blue boundary are wrongly divided into the non-crack region (purple region) by the SVM (Fig. 17a). The feature points at the region with $x$ ranging from -1 to 1 and $y$ ranging from -1 to 1 are easy to wrongly classified by the SVM and the MLP, as shown in Fig. 17a and b. The feature distribution of the composite noisy samples with SNR of 20 dB extracted from the FC layer of the CNN-Song are illustrated in Fig. 17c. It can be seen the CNN-Song trained with clean data is difficult to separate the crack feature points from the composite noisy samples. While, there are two clear separated clusters of feature points extracted from the FC layer of the HACNN. In summary, it can be also known from the visualization results the proposed HACNN has better anti-noise ability than the SVM, the MLP, and the CNN-Song.

## 4. Discussion

The dropout randomly deactivates neurons of a certain layer of a network with a certain probability during training process, and thus a part of connections between two adjacent layers are cut. Therefore, training a network using the dropout technique prevents neurons from co-adapting too much and consequently the overfitting of a network. Since the convolutional layers do not have numerous parameters, it seems that adding dropout to convolutional layers would not have much effect on overfitting. However, adding dropout to the convolutional layers is equivalent to providing noisy inputs for the following layers, which also helps to prevent overfitting.

In this study, the dropout is added to the first convolutional layer, which is equivalent to adding noise to the original input. Table 7 shows the effect of kernel size on the classification performance of the HACNN. The performance for the HACNN decreases with the decrease of kernel size. Comparing with the HACNN with kernel size of 3 × 1 in first convolutional layer, the accuracies for HACNN with kernel size of 24 × 1 increase by 8.74%, 9.46%, 12.42%, 16.67%, 14.83%, 11.06% among the composite datasets with SNR ranging from 0 dB to 20 dB, respectively. The accuracies for the HACNN with kernel size of 3 × 1 are below 70%, which indicates the anti-noise ability is relatively poor for the HACNN with kernel size of 3 × 1.

**Table 7**

The accuracy of HACNNs with different kernel size.

| Kernel size | Testing dataset | SNR (dB) | | | | | |
|---|---|---|---|---|---|---|---|
| | | 20 | 16 | 12 | 8 | 4 | 0 |
| 3 | 99.36% | 83.33% | 76.36% | 69.71% | 68.35% | 65.30% | 58.89% |
| 8 | 99.92% | 90.95% | 85.50% | 80.77% | 74.04% | 68.75% | 63.14% |
| 16 | 99.92% | 92.47% | 88.54% | 82.85% | 77.72% | 70.91% | 67.39% |
| 24 | 99.92% | 94.39% | 91.19% | 86.38% | 80.77% | 74.76% | 67.63% |

It can be concluded from the results in Table 7 that the kernel size has an important effect on the classification performance of the HACNN. As we known, there is only 1 or 2 neurons being deactivated for a small kernel with size of 3 × 1 under fixed dropout rate, which means only a relatively large number of kernels can ensure that enough neurons are deactivated. However, a large number of kernels means higher computing cost. A large or wide kernel (e.g. with a size of 24 × 1) can ensure more neurons to be deactivated at one time, and thus only a relatively small number of kernels can ensure that enough neurons are deactivated. Moreover, a wide kernel (e.g. 24 × 1) of the first convolutional layer enlarges the receptive field to gain the global features. This is useful for microcrack identification since the high-frequency part of the input data can be suppressed because of global computing. Therefore, the large convolutional kernel combining with dropout can achieve a better effect in modelling the addition of noises. This way of simulating noise addition is equivalent to adding interference during training process. If a model can converge to a satisfactory loss value with training interference, the anti-noise or generalization ability of the model should be stronger for different noisy datasets.

## 5. Conclusions

This study proposes a HACNN to achieve accurate and reliable identification of rock microcracks based on the fibre optic sensing data and aims to make the proposed HACNN perform well among data of different SNRs. For this purpose, the shortcut connections are used to increase the number of convolutional layers to 11 for enhancing the nonlinear representation ability of the HACNN. Moreover, the designed HA modules of the HACNN emphasize the meaningful features in channel and spatial dimensions, which is also beneficial to the representation of nonlinear features. In addition, two training interference strategies, i.e. small mini-batch training and adding dropout in the first convolutional layer, are used during the training process for enhancing the anti-noise ability of the HACNN.

The performance of the proposed HACNN was compared with that for the SVM, the MLP, and the CNN-song. Among the composite noisy datasets with SNRs ranging from 0 dB to 20 dB, the HACNN respectively achieves accuracies of 94.39%, 91.19%, 86.38%, 80.77%, 74.76%, 67.63%, both of which are higher than those for the SVM (90.06%, 86.86%, 80.29%, 73.64%, 63.38%, 51.68%), MLP (90.14%, 86.46%, 79.81%, 75.88%, 71.07%, 63.62%), and CNN-Song (71.39%, 65.95%, 63.38%, 58.73%, 55.93%, 53.93%). It can be seen the proposed HACNN presents a relatively better anti-noise ability. When it comes to the training interference strategy, adding dropout in the first convolutional layer, a large kernel size of 24 × 1 combining a dropout rate of 0.1 achieves better performance. It is because this parameter combination can reasonably interfere with the training and also maintain a certain regularization effect

for the HACNN. When mentioning the small mini-batch training, a batch size of 10 achieves better performance because this batch size enables the average value and variance in BN layers to fluctuate in a reasonable range.

This research shows the possibility of establishing a robust DL-based model against noises. For ongoing and future research, soft thresholding[23] can be embedded into the HACNN to eliminate unimportant features for the further improvement of anti-noise ability. The thresholds can be automatically determined through a delicately designed subnetwork which is a particular combination of GAP, FC layers, and sigmoid function.[25] Therefore, the anti-noise ability should be improved if different thresholds corresponding to noisy data of different SNRs are automatically found to suppress useless features.

## Acknowledgements

The Open Research Project Programme of the State Key Laboratory of Internet of Things for Smart City (University of Macau) (Ref. No. SKL-IoTSC(UM)-2021-2023/ORPF/A19/2022), the General Research Fund project from Research Grants Council of Hong Kong Special Administrative Region Government of China (No. 15214722), and the Start-up Fund from The Hong Kong Polytechnic University (1-BD88) are gratefully acknowledged.

## References

1. Basu A, Mishra D, Roychowdhury K. Rock failure modes under uniaxial compression, Brazilian, and point load tests. *Bull Eng Geol Environ.* 2013;72: 457-475.
2. Wang F, Ding C, Pan H, Zhang S. A mesostructure-informed cohesion-based numerical method for fracture behavior of slate with foliation structure. *Int J Rock Mech Min Sci.* 2022;160: 105252.
3. Guo TY, Wong LNY. Cracking mechanisms of a medium-grained granite under mixed-mode I-II loading illuminated by acoustic emission. *Int J Rock Mech Min Sci.* 2021;145: 104852.
4. Miao S, Pan P-Z, Zhao S, Han J, Konicek P. A new DIC-based method to identify the crack mechanism and applications in fracture analysis of red sandstone containing a single flaw. *Rock Mech Rock Eng.* 2021;54: 3847-3871.
5. Chen J, Zhou M, Huang H, Zhang D, Peng Z. Automated extraction and evaluation of fracture trace maps from rock tunnel face images via deep learning. *Int J Rock Mech Min Sci.* 2021;142: 104745.
6. Geetha GK, Sim S-H. Fast identification of concrete cracks using 1D deep learning and explainable artificial intelligence-based analysis. *Autom Constr.* 2022;143: 104572.
7. Zhang Q, Barri K, Babanajad SK, Alavi AH. Real-time detection of cracks on concrete bridge decks using deep learning in the frequency domain. *Engineering.* 2021;7(12): 1786-1796.
8. Lin S-Q, Tan D-Y, Yin J-H, Li H. A Novel Approach to surface strain measurement for cylindrical rock specimens under uniaxial compression using distributed fibre optic sensor technology. *Rock Mech Rock Eng.* 2021;54: 6605-6619.
9. Madjdabadi B, Valley B, Dusseault MB, Kaiser PK. Experimental evaluation of a distributed Brillouin sensing system for detection of relative movement of rock blocks in underground mining. *Int J Rock Mech Min Sci.* 2017;100(93): 138-151.
10. Forbes B, Vlachopoulos N, Diederichs MS, Aubertin J. Augmenting the in-situ rock bolt pull test with distributed optical fiber strain sensing. *Int J Rock Mech Min Sci.* 2020;126: 104202.

11. Liu Y, Li W, He J, Liu S, Cai L, Cheng G. Application of Brillouin optical time domain reflectometry to dynamic monitoring of overburden deformation and failure caused by underground mining. *Int J Rock Mech Min Sci.* 2018;106: 133-143.
12. Song Q, Yan G, Tang G, Ansari F. Robust principal component analysis and support vector machine for detection of microcracks with distributed optical fiber sensors. *Mech Syst Signal Proc.* 2021;146: 107019.
13. Song Q, Zhang C, Tang G, Ansari F. Deep learning method for detection of structural microcracks by brillouin scattering based distributed optical fiber sensors. *Smart Mater Struct.* 2020;29(7): 075008.
14. He K, Zhang X, Ren S, Sun J. Deep residual learning for image recognition. *Proceedings of the IEEE conference on computer vision and pattern recognition.* IEEE; 2016:770-778.
15. Zhao S, Zhang G, Zhang D, Tan D, Huang H. A hybrid attention deep learning network for refined segmentation of cracks from shield tunnel lining images. *J Rock Mech Geotech Eng.* 2023.
16. Woo S, Park J, Lee J-Y, Kweon IS. Cbam: Convolutional block attention module. *Proceedings of the European conference on computer vision (ECCV).* Springer; 2018:3-19.
17. Wang Q, Wu B, Zhu P, Li P, Zuo W, Hu Q. ECA-Net: Efficient channel attention for deep convolutional neural networks. *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition.* IEEE; 2020:11534-11542.
18. Ioffe S, Szegedy C. Batch normalization: Accelerating deep network training by reducing internal covariate shift. *International conference on machine learning.* PMLR; 2015:448-456.
19. Srivastava N, Hinton G, Krizhevsky A, Sutskever I, Salakhutdinov R. Dropout: a simple way to prevent neural networks from overfitting. *J Mach Learn Res.* 2014;15(1): 1929-1958.
20. Baldi P, Sadowski P. Understanding dropout. *Proceedings of the 26th International Conference on Neural Information Processing Systems.* Curran Associates Inc.; 2013:2814-2822.
21. Keskar NS, Mudigere D, Nocedal J, Smelyanskiy M, Tang PTP. On large-batch training for deep learning: Generalization gap and sharp minima. *arXiv preprint arXiv:160904836.* 2016.
22. Goodfellow I, Pouget-Abadie J, Mirza M, et al. Generative adversarial networks. *Commun ACM*. 2020;63(11): 139-144.
23. Donoho DL. De-noising by soft-thresholding. *IEEE Trans Inf Theory.* 1995;41(3): 613-627.
24. Van der Maaten L, Hinton G. Visualizing data using t-SNE. *J Mach Learn Res.* 2008;9(11).
25. Zhao M, Zhong S, Fu X, Tang B, Pecht M. Deep residual shrinkage networks for fault diagnosis. *IEEE Trans Ind Inform.* 2019;16(7): 4681-4690.
26. Wu B, Zhu H-H, Liu T-X, Wang D-Y, Hu L-L, Li B. Experimental investigation of interfacial behavior of fiber optic cables embedded in frozen soil for in-situ deformation monitoring. *Measurement* 2023;215: 112843.
27. Chen W, Hong C, Wang J, Wang X. Investigation on compressive and pullout behavior of cast-in situ piles using BOFDA technique. *Acta Geotech*. 2023: 1-12.
28. Liu Y, Bao Y. Automatic interpretation of strain distributions measured from distributed fiber optic sensors for crack monitoring. *Measurement* 2023;211: 112629.
29. Chen W, Hong C, Chen X, Luo G, Su D. Comparative analysis of anchor cables in pullout tests using distributed fiber optic sensors. *Can Geotech J*. 2023.
30. Pei H, Yin J, Wang Z. Monitoring and analysis of cast-in-place concrete bored piles adjacent to deep excavation by using BOTDA sensing technology. *J Mod Opt*. 2019;66(7): 703-709.
31. Wang D, Zhu H, Huang J, Yan Z, Zheng X, Shi B. Fiber optic sensing and performance evaluation of a water conveyance tunnel with composite linings under super-high internal pressures. *J Rock Mech Geotech Eng*. 2023.