Earth Sciences
2022; 11(4): 194-203
http://www.sciencepublishinggroup.com/j/earth
doi: 10.11648/j.earth.20221104.15
ISSN: 2328-5974 (Print); ISSN: 2328-5982 (Online)

Elastic Impedance Inversion with GRU-CNN Hybrid Deep Learning: Visualizing the Black Box

Xiujuan Liu[1, 2, 3], Lifeng Liang[1, 2, 3, *], Zhiming Kang[1], Qiang Guo[4]

[1]Department of Geography, Lingnan Normal University, Zhanjiang, China

[2]Guangdong Coastal Economic Belt Development Research Center, Lingnan Normal University, Zhanjiang, China

[3]Mangrove Institute, Lingnan Normal University, Zhanjiang, China

[4]College of Information Engineering, China Jiliang University, Hangzhou, China

Email address:

121436068@qq.com (Lifeng Liang)

[*]Corresponding author

To cite this article:

Xiujuan Liu, Lifeng Liang, Zhiming Kang, Qiang Guo. Elastic Impedance Inversion with GRU-CNN Hybrid Deep Learning: Visualizing the Black Box. *Earth Sciences*. Vol. 11, No. 4, 2022, pp. 194-203. doi: 10.11648/j.earth.20221104.15

Received: June 22, 2022; **Accepted**: July 13, 2022; **Published**: July 22, 2022

Abstract: The process of hybrid deep learning is highly integrated with the seismic inversion, leading to the *black box* phenomena, which hampers the understanding of deep-learning inversion process. In this paper, a numerical example is presented to visualize the process of deep learning from the perspective of elastic inversion. Synthetic seismic data is generated by forward modeling on a wedge model, after which the GRU-CNN hybrid deep learning algorithm is applied to obtain the inverted impedance method. In specific, the extraction of local seismic features by CNN, the extraction of low-frequency seismic features by GRU, activation layer, Adam and learning rate schedules, initialization model, loss function, and training process are detailed illustrated and visualized, all of which reveal the internal operating mechanism of the *black box*. The results show that: 1) after required epoch iterations, the features extracted by CNN becomes close to the real impedance, while the features extracted by GRU is close to the low-frequency information involved in conventional seismic inversion (which is consistent with the cognition from commercial software, e.g., Jason, Geoview, etc.), 2) The learning rate is a very critical parameter in the optimization process. Comparing with the constant learning rate, the cosine learning rate converges faster with better performance, and 3) the initial impedance model in hybrid deep learning is to initialize the weights of all neurons, which is essentially different from those of conventional seismic inversion scheme, e.g., constrained sparse pulse inversion.

Keywords: Elastic Impedance Inversion, GRU, CNN, Wedge Model, The Black Box

1. Introduction

Deep learning has been widely explored since the early twentieth century [1]. The remarkable performance of deep-learning-based technique has also gained popularity in the petroleum exploration industry [2].

The rapid development of deep learning is primarily due to three reasons: algorithm, big data, and computing power. The back-propagation algorithm is the basic algorithm of deep learning, which has been well developed [3]. Big data is available since the prosperity of the Internet provides tremendous data for use and analysis. Computing power is greatly enhanced by GPU, TPU, etc., of which performance has far exceeded that of CPU. In the past four years, deep learning has made a spurt of development in the field of seismic exploration. In summary, 2017 can be regarded as the year of awakening for the intelligent application, and 2018 can be regarded as the explosive year of intelligent application research, and 2019 is the year of initial results of intelligent application research in petroleum exploration [4]. The application of deep learning includes seismic structural interpretation (e.g., fault, horizon, salt dome, channel, cave, etc.) [5], noise suppression and signal enhancement [6], litho-facies identification[7], reservoir parameter prediction [8], seismic forward modeling [9], seismic velocity modeling, first arrival picking, seismic data interpolation [10], seismic

attribute analysis, and microseismic data analysis, etc. The adopted machine learning techniques include deep neural networks, dictionary learning, generative confrontation network, random forest algorithm, and cluster analysis algorithms [5]. In particular, Alfarraj et al. proposed using integrating Gated Recurrent Units (GRU) to extract low-frequency information of long-term series, and using convolutional neural networks (CNN) to extract high-frequency features of seismic data, to improve the accuracy of seismic inversion [11].

However, unlike the linear system which can be clearly explained, the mechanism behind deep learning has a characteristic of inexplicability. Researchers often use *black box* to interpret such mechanisms of deep learning. In the field of geophysics, the mechanism of deep learning is poorly understood. For example, the meaning of manually extracting features is clear to us, while the neural network of deep learning is trained to get as close to the training set as possible. Although the neural network can indeed solve any specific questions by minimizing the objective function using the reverse gradient algorithm, the change of weights and the physical meaning of extracted features remain unclear to us. Therefore, we are unable to disassemble it and internally analyze the function of each part [12-14].

In this paper, we illustrate the *black box* in detail via a numerical example of seismic inversion. The GRU-CNN hybrid deep learning algorithm proposed by Alfarraj is applied to synthetic seismic data [11]. In specific, the extraction of local seismic features by CNN and of low-frequency seismic features by GRU, activation layer, Adam and learning rate schedules, initialization model, loss function, and training process are detailed illustrated and visualized.

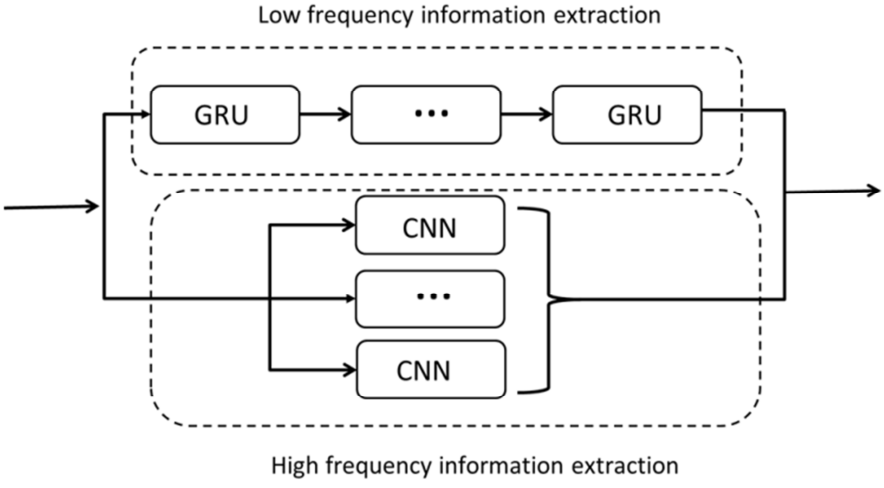
2. Methodologies

2.1. GRU-CNN Hybrid Deep Learning

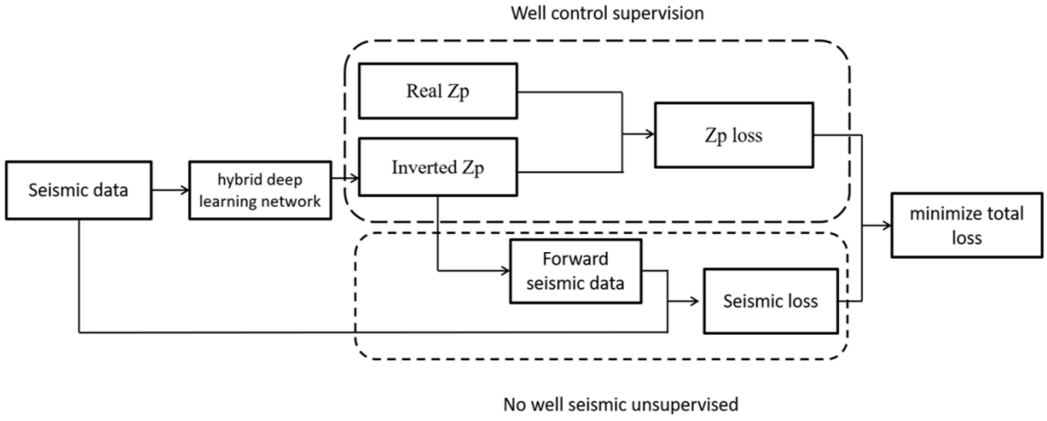
We employ the deep learning algorithm proposed by Alfarraj et al. to illustrate the mechanism of black boxes [11]. The network structure is a hybrid deep learning combining Gated Recurrent Units (GRU) and convolutional neural network (CNN). The GRU network can extract low-frequency information from long-term series, which is consistent with the characteristic of seismic data. Notably, the GRU network is developed based on Recurrent Neural Network (RNN) and (Long Short Term Memory) LSTM, which can be selected accordingly in practical applications.

Figure 1a shows the network structure of GRU-CNN. A total of three layers of GRU networks are selected in this article (GRU networks with more layers are not discussed here). The extraction of high-frequency information is achieved by connecting three CNNs in parallel. Different local features of seismic data are mainly achieved by setting different convolution kernel sizes and step lengths. Similarly, a total of three-layer CNN network is analyzed here while the CNN network with more layers is suppressed for simplicity. In summary, the seismic inversion results containing low-frequency and high-frequency information are jointly extracted through the combination of GRU and CNN networks.

The inversion process using the hybrid deep learning network is outlined in Figure 1b. The synthetic data of a wedge model is the input training data, and P-wave impedance (*Z*p) is inverted as output data through the hybrid deep learning network, In particular, at the well location, the mean square error between the inverted and true impedance (i.e., logging curves) can be calculated, which is the *well-control supervision part*. The modeled seismic responses (based on the convolution between reflectivities and wavelet) and the (input) observed seismic data make up the misfit (loss) of seismic data, which is the *non-well seismic unsupervised part*. Through the forward and backward propagation of multiple epochs of the deep learning network, the total loss of the two parts is minimized and the multi-dimensional nonlinear inversion problem can be solved.



(a)



(b)

Figure 1. *(a) The network structure of the GRU-CNN hybrid deep learning and (b) the work-flow of the GRU-CNN-based seismic impedance inversion. The ellipsis represents the number of layers of GRU network in (a) and the number of parallel CNN network layers in (b).*

2.2. Loss Function

For the back-propagation process of the hybrid deep learning network based on gradient descent method, the loss function (objective function) should be first defined, then partial derivative of the loss function to the gradient is calculated, and finally the weights and bias parameters along the gradient descent direction can be calculated. The setting of the loss function is of significance in calculating the gradient. Here, we employ the loss function based on mean square error, which is the most commonly used loss function.

The loss function of the hybrid deep learning inversion consists of two parts [11]: the seismic loss and the elastic impedance loss, which takes the form as

$$L(\theta) = \alpha L_1(\theta) + \beta L_2(\theta) \tag{1}$$

where θ represents the angle of seismic incident, $L_1(\theta)$ represents the elastic impedance loss, $L_2(\theta)$ represents the seismic loss, α and β represent the regularization parameters which have different values and represent the degree of confidence in the seismic data or logging elastic impedance.

The seismic loss $L_2(\theta)$ is the mean square error between the synthetic seismic and the input seismic, i.e.,

$$L_2(\theta) = \frac{1}{M}\sum_i \left\| d_i - F(F_\theta^+(d_i)) \right\|_2^2 \tag{2}$$

where m_i represents the elastic impedance of ith trace, $F_\theta(d_i)$ represents the EI inversion of the seismic data d_i, F represents the forward inversion of the EI from the inversion.

The impedance loss $L_1(\theta)$ is the mean square error between the predicted EI and the real EI on the training seismic trace, i.e.,

$$L_1(\theta) = \frac{1}{N}\sum_i \left\| m_i - F_\theta^+(d_i) \right\|_2^2 \tag{3}$$

In the next section, we try to explain each part of Figure 1b in detail through a numerical example. With the intention of revealing the black box, we illustrate and visualize the inversion process and discuss some key parameters involved in the hybrid deep learning.

3. Black Box Visualization

3.1. Synthetic Model

We designed a simple geological model with three lithologic layers [12], in which the first and third layers are clay, the second layer is sandstone. The first and third layers (clay) can be considered as two semi-spaces, of which thickness is beyond the seismic resolution. The thickness of the second layer ranges between 0 and 30 m, as shown in Figure 2. In addition, the impedance of clay and sandstone is set as 4500 m/s*g/cc and 5500 m/s*g/cc, respectively. The model reflectivity is calculated by

$$R_i = \frac{Z_{i+1} - Z_i}{Z_{i+1} + Z_i} \tag{4}$$

where Z represents the impedance and the subscript i represents the ith layer. According to Equation 4, the reflectivities between the clay and sandstone are 0.1 (top layer, R_1) and -0.1 (bottom layer, R_2), respectively, as expressed by

$$R_i = (0.1, -0.1)^T \tag{5}$$

For simplicity, we employ a three-point source wavelet given by

$$w = (-1, 3, -1)^T \tag{6}$$

Based on the convolutional model, synthetic seismic data d can be generated by convolving the reflectivities with source wavelet as

$$d = R * w \tag{7}$$

The synthetic data of the wedge model is shown in Figure 3a (where the last trace exhibits no seismic reflection since

the thickness approaches 0). For simplicity, we only select the last three traces (Figure 3b) which involve tuning effect for analysis. Since the hybrid deep learning algorithm is a semi-supervised learning method, a certain amount of labeled data is required as a supervised sample. For this reason, the first and third traces are set as labeled data, of which impedance is known, and the impedance as well as the seismic data are treated as training dataset.

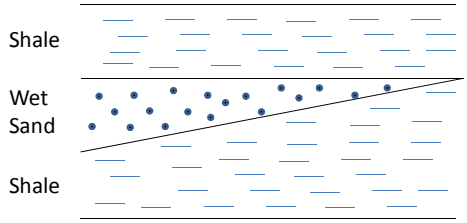


Figure 2. *A simple geological wedge model.*

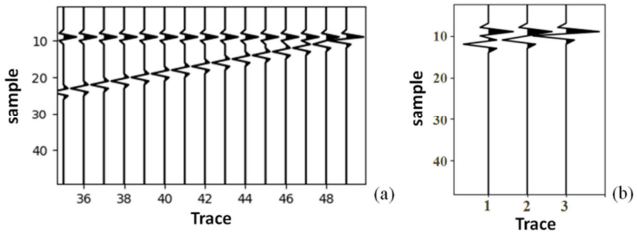


Figure 3. *(a) Synthetic data of the wedge model with its (b) 47th, 48th, and 49th traces for analysis.*

3.2. Local Seismic Feature Extraction by CNN

The basic (calculating) unit for feature extraction by CNN is convolution. In order to illustrate the calculating process, we take the first trace of seismic data in Figure 3b as an example (which is listed in the first row of Table 1). Performing the convolution between the seismic data and a kernel by

$$Y(n) = x(n) * h(n) \tag{8}$$

where $x(n)$ represents the seismic data, $h(n)$ represents the convolution kernel, and * represents the convolution. The convolution kernel is also known as the weight, is the learnable part in deep learning. By means of optimization algorithm and reverse gradient transfer, the weight (kernel) can be solved and updated.

Specifically, in *Pytorch*, one can check the weight of the convolution kernel through the following command:

If isinstance(m, nn.Conv1d) or isinstance(m, nn.ConvTranspose1d):

print (nn.init.xavier_uniform_(m.weight.data))

Next, we will illustrate the convolution process by demonstrating the seismic data, weights, and extracted features. The second and third rows of Table list the initial value of the convolution kernel and the convolved results, respectively.

Table 1. *Parameters for the convolution process.*

Seismic data	-0.5257	1.5770	-0.5257	0.5257	-1.5770	0.5257	-1.5632e-17
Weights	0.2149	-0.1999	0.1051	-0.1135	0.0651		
Results	-0.646	0.7127	-0.4436	0.4836	-0.4440	0.1130	-0.646

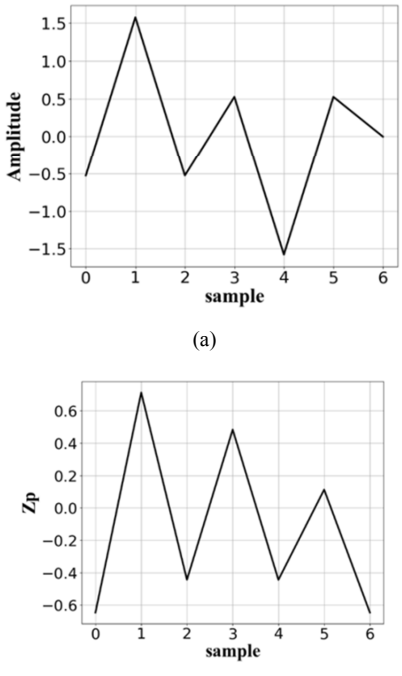


Figure 4. *(a) Initial seismic data and (b) the extracted feature by 1-D convolution kernel.*

In order to illustrate the hybrid deep learning network clearly, we graphically display the results during the convolution process. Figure 4a shows the seismic data in Table 1, which is also an enlarged display of the first seismic trace in Figure 3b. Figure 4b shows the convolution result of Table 1, which is one of the extracted features. In order to extract different features from the synthetic data (image), the neural network requires a number of different convolution kernels. For this reason, we set up 8 convolution kernels and extracted 8 seismic features with one training round (epoch=1), and the results are shown in Figure 5, from which we can see some of these features are similar to seismic data and are easy to understand, while some are quite different from the data, which are related to the initial value of the convolution kernel. In addition, Figure 5 considers the boundary expansion (padding=2) and the hole convolution (dilation=1). Figure 5b shows the extracted features after 100 training rounds (epoch=100). It can be seen that some of them have shown some real impedance features. Generally, a two-layer neural network is sufficient to fit any functions, however, in reality, a complex feature is often composed of multiple simple features. The DeepLab network structure could help understand the image from the part to the whole. It shows that this hierarchical structure is very suitable for revealing images from part to whole.

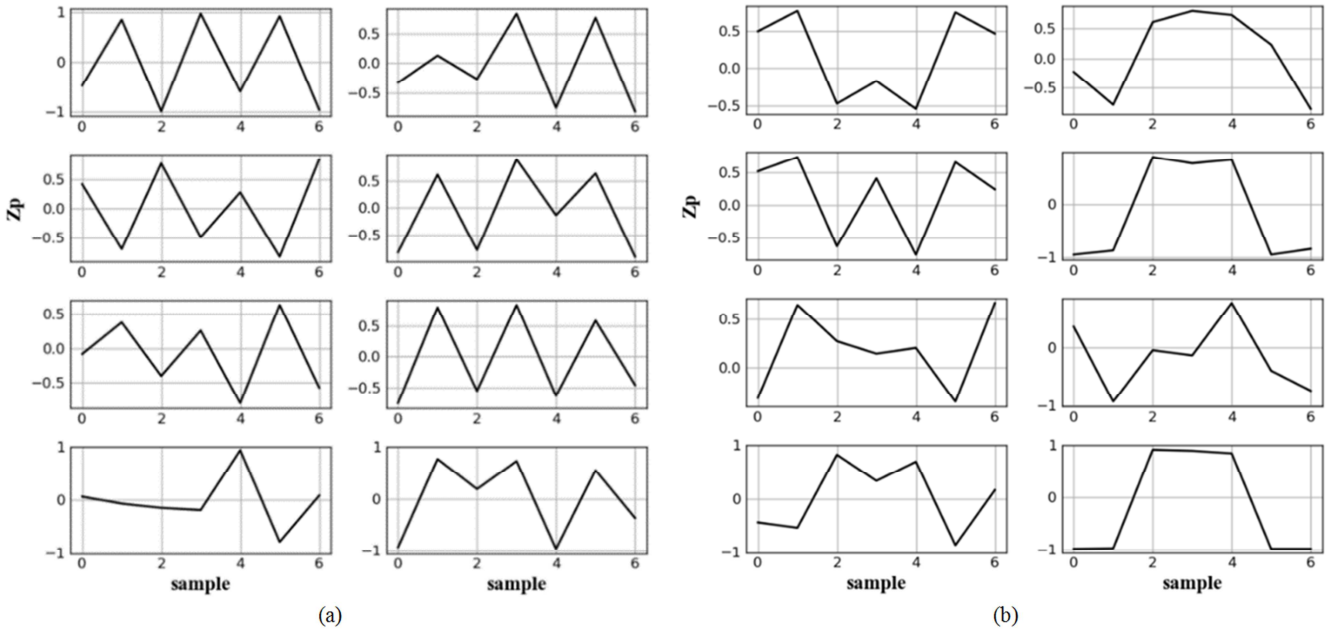


Figure 5. *Extracted local features after (a) 1 and (b) 100 epochs by CNN.*

3.3. Low-Frequency Feature Extraction by GRU

In conventional impedance inversion, the low-frequency components of the impedance data are also required in addition to the mid- and high-frequency ones. For constrained sparse spike inversion, low-frequency components can generally be obtained from seismic horizons, or can be estimated by interpolating or laterally extrapolating logging data (reference). However, this type of method is not driven by seismic data, which leads to low accuracy. In deep learning, seismic data can be considered as sequence samples that change over time, which are similar to voice, video, etc. Therefore, adjacent seismic data is related, which requires the neural network to have the capability to store information in order to effectively process sequence samples. However, CNN network does not have the capability to change the internal weight structure and deal with sequence data. To this end, the Gated Recurrent Unit (GRU) network was proposed (Figure 6). GRU is a variant of LSTM and is featured by processing sequences of any length by using neurons with self-feedback. GRU can effectively save the historical information of sequences and can remember long-term dependencies (low-frequency trends on seismic data), therefore, it is more suitable for processing seismic data. Compared with the LSTM model, GRU has only update gates and reset gates, with fewer parameters and faster training speed.

Similarly, we apply the GRU network to the synthetic data in Figure 3b. Here, three GRU networks are connected in series, and 8 seismic features are extracted as shown in Figure 7. Comparing Figure 7 with Figure 5, even if there is only one training round, the information extracted by GRU has certain low-frequency features. After 100 training rounds, the extracted features (Figure 7b) exhibit apparent low-frequency trend, which is completely different from those extracted by CNN. It shows that the GRU network has the capability to store information and extract low-frequency features of seismic data.

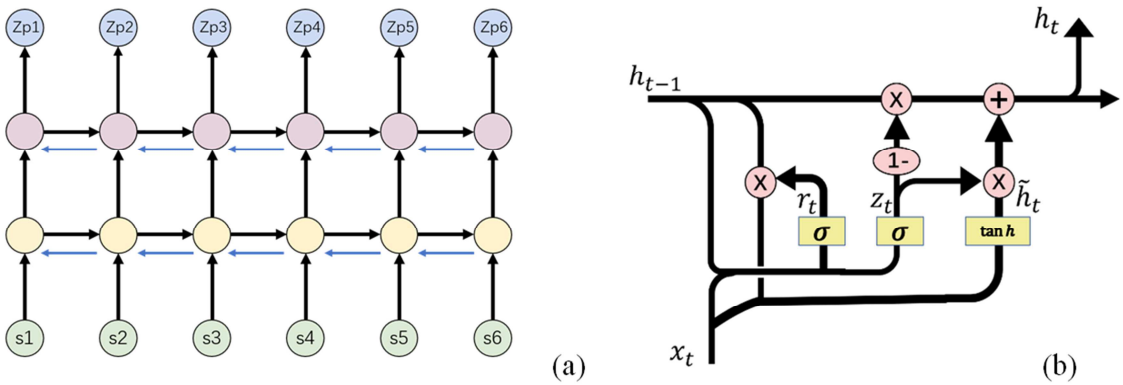


Figure 6. *The network structure of (a) GRU and (b) its neuron.*

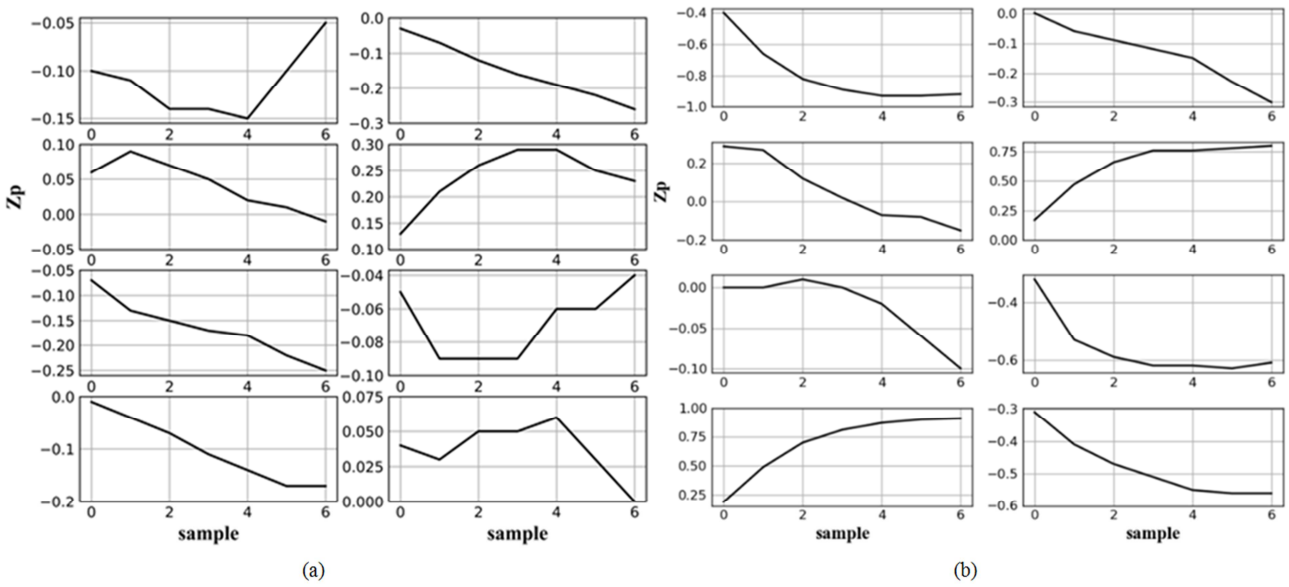


Figure 7. *Extracted low-frequency features after (a) 1 and (b) 100 epochs by GRU.*

3.4. Activation Layer

Inspired by neuroscience, deep learning networks are composed of layered structures, so the artificial neuron activation function can also be arranged on a unified level with the layered structure of neurons. In order to achieve functional separation and increase flexibility, the deep learning framework generally only performs one operation on a single-layer structure, and then combines multiple layers to construct a new neural network. Therefore, the two parts of each neuron, the weighted sum function and the activation function are divided into two calculation units during processing. According to the adopted function, the activation layer can be divided into *sigmoid* activation layer, *tanh* activation layer, *ReLU* activation layer, etc. The goal of activation layer is to add nonlinear capabilities to the neural network so that a large number of complex nonlinear problems can be handled. In this article, we make use of the tanh activation function, which takes the form as

$$f(x)=\frac{1-e^{-2x}}{1+e^{-2x}} \tag{9}$$

Figure 8a shows the *tanh* function distribution with the impedance curve before (Figure 8b) and after (Figure 8c) expressed by the function. Comparing Figure 8b and 8c,, it can be seen that, the original impedance which has a range of (-3,3) falls into a range of (-1,1) after the activation, so the activation function can provide nonlinear characteristics for the neural network. Besides, compared with the *sigmoid* activation function, the *tanh* function solves the non-zero mean value problem existing in the sigmoid function, and it converges faster with the output centered around zero.

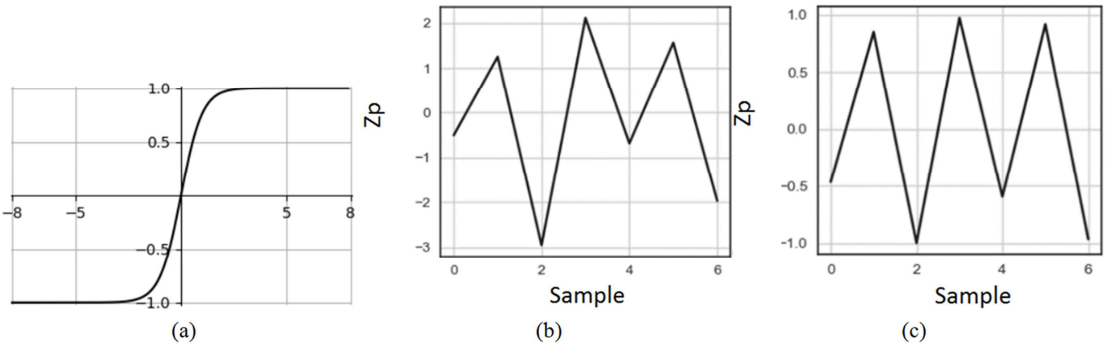


Figure 8. *(a) Standard tanh activation function and the impedance curve (b) before and (c) after the activation.*

3.5. Adam and Learning-Rate Schedules

The neuron nodes (weights) in deep learning networks are numerous and complex. Analyzing how the weights change during the learning process could help us better understand the internal mechanism of hybrid deep learning for seismic inversion. The update of the weights is mainly determined by the optimization algorithm. The basic gradient-based update algorithm takes the form as [6].

$$\omega_{t+1} \leftarrow \omega_t - \alpha \frac{\partial E}{\partial \omega_t} \tag{10}$$

which ω_t represents the current weight value, E represents the gradient of the objective function (also known as loss function), and α is the learning rate. Here, we employ the optimization algorithm Adam. According to Equation 10, the learning rate is a key parameter that controls the change of the weight value and determines the convergence speed of deep learning. In general, if the learning rate is set too large, the objective function is likely to oscillate and it is difficult to converge, while if the learning rate is too small, although the loss function can converge, it suffers from low convergence speed, which will greatly influence computation efficiency for large deep learning networks, and it is easy to fall into a local minimum.

To address the problem mentioned above, Smith proposed the cosine-curve learning rate [15], by which α can be expressed as

$$\alpha_t = \alpha_{\min} + \frac{1}{2}(\alpha_{\max} - \alpha_{\min})\left[1 + \cos\left(\frac{T_{cur}}{T_{\max}}\pi\right)\right] \quad (11)$$

where $\alpha_{\min}$ and $\alpha_{\max}$ represent the minimum and maximum learning rates, respectively, T_{cur} represents the range of epoch, T_{max} represents the maximum number of iteration. In this study, $\alpha_{\min}$ and $\alpha_{\max}$ are set as 0.001 and 0.05, respectively, T_{cur} and T_{max} are set as (0,100) and 33, respectively.

Given epoch (t) ranges from 0 to 100, the distribution of the learing rate in Equation 11 is shown in Figure 9, from which it can been seen that, when epoch increases from 0 to 33, the initial learning rate is 0.05, and then gradually decreases to 0.001, which exhibits a cosine-like trend of change; when the epoch increases from 34 to 66, the learning rate increases from 0.001 to 0.05; finally, the learning rate decreases according to a cosine function. The advantage of the cosine-curve learning rate is that the learning rate can be changed periodically between a large and small step size. A large step size can make the objective function jump out of locally optimal solutions, based on which a small step size helps a fine-tune to find the global optimal solution. For the cosine learning rate, the rate at the last epoch should return to its minimum value.

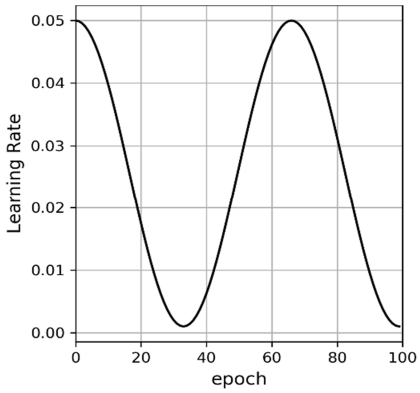


Figure 9. *Distribution of cosine-curve learning rate.*

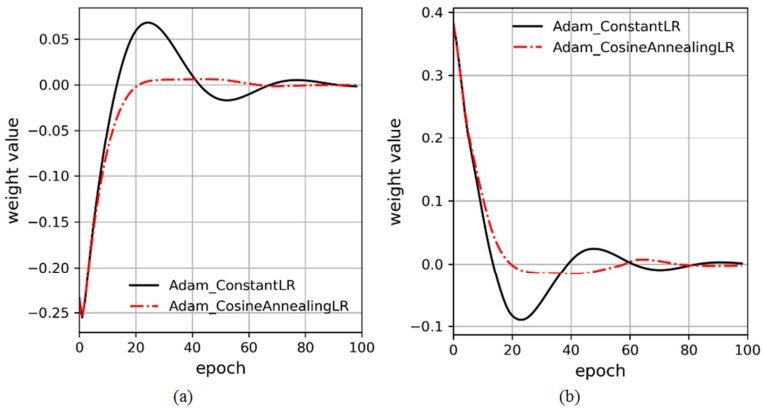


Figure 10. *The change of weight using different learning rates for (a) input-hidden of the 2nd layer and (b) hidden-hidden of the 1st layer for GRU.*

We compare the change of weights by using two different learning rates, i.e., the constant and cosine-curve learning rate. The optimization algorithm is Adam and the results are shown in Figure 9. In specific, the weight values can be checked by calling the following command:

for name,parameters in inverse_net.named_parameters():
parm[name]=parameters.detach().cpu().numpy()
aa=parm['gru.weight_hh_l0'][0, 2]
bb=parm['gru.weight_ih_l2_reverse'][0,2] % reverse means backward direction

Figure 10a displays the input-hidden weights (ω_{ii}, ω_{if}, ω_{ig}, ω_{io}) of the second layer for GRU network (labeled as *gru.weight_ih_l2_reverse*). Since GRU transmits information in both forward and backward directions. Figure 10b displays the hidden-hidden weights (ω_{hi}, ω_{hf}, ω_{hg}, ω_{ho}) of the kth layer for GRU network (labeled as *gru.weight_ih_l2_reverse*). For the cosine-curve learning rate, it can be seem that when the epoch is greater than 20, the weight changes more smoothly. It is equivalent to finding the relatively optimal solution directly after a few iterations. For the constant learning rate, when the epoch ranges from 20 to 60, the weight changes more drastically, indicating that the deep learning network is searching for the optimal solution in the continuous iteration process. In other words, the constant learning rate requires a large number of iterations to be converged, which increases the computing time of deep learning.

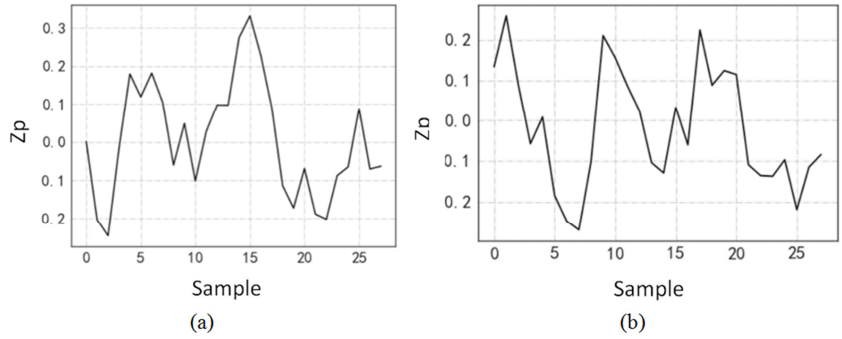


Figure 11. *The generated initial impedance given (a) random_seed=30 and (b) random_seed=2. Note that, given different random seeds (weight parameter), although the input data is the same, the output is different.*

3.6. Model Initialization and Training Process

The training dataset is the labeled data. Each training dataset consists of seismic data and the corresponding impedance data. It can be regarded as a seismic trace along a well and the impedance data can be calculated by the logging curves. The training dataset itself establishes the mapping relation between seismic and impedance data (of course, we don't know the mapping function in advance). The ultimate goal of deep learning is to build an objective (loss) function, train the data set iteratively, and find this mapping relation (or called the mapping function $y=f(x)$). Different from conventional gradient-based seismic inversion. Deep learning inversion does not require an initial (impedance) model, but only an initialized weight parameters of the deep learning network.

In summary, the initialization process is as: first, initialize the weight parameters of the hybrid deep learning network through *inverse_net.train()* and *random_seed*. (note that, given different random_seed, the generated weight parameters will be differed); second, input seismic data to the deep learning network, and calculate *y_pred* through *y_pred = inverse_net(x)*, where **x** is the input data for training (i.e., synthetic seismic data), and *y_pred* represents the P-wave impedance generated by the inversion (i.e., mapping function). It should be noted that, when the weight parameters are different, even if the input data is the same, different initialized impedance models will be generated, as shown in Figure 11.

For conventional seismic inversion, the initial impedance model is directly given (e.g., constrained sparse pulse inversion); in contrast, for deep learning seismic inversion, the initial model is actually the initialized weight parameters (for example, the weights in the second row, Table 1), after which the initial impedance model can be generated through *y_pred= inverse_net(x)*. Therefore, the two ways of initializing the model are essentially different.

Figure 12 displays the comparisons between the inverted and true impedance as well as the corresponding seismic data during the training process. It can be seen the inverted impedance almost recovers the true impedance after 60 training rounds. The loss function converges and approaches zero after 50 training rounds (Figure 13). The value of the loss function at the specific training round is given as L=1*1.002+0.2*1.124=1.227 (Figure 12a), L = 1*0.04+0.2*0.297 = 0.099 (Figure 12b), L = 1*0.009+0.2*0.025 = 0.014 (Figure 12c), L = 1*0.004+0.2*0.002 = 0.004 (Figure 12d), L = 1*0.002+0.2*0 = 0.002 (Figure 12e).

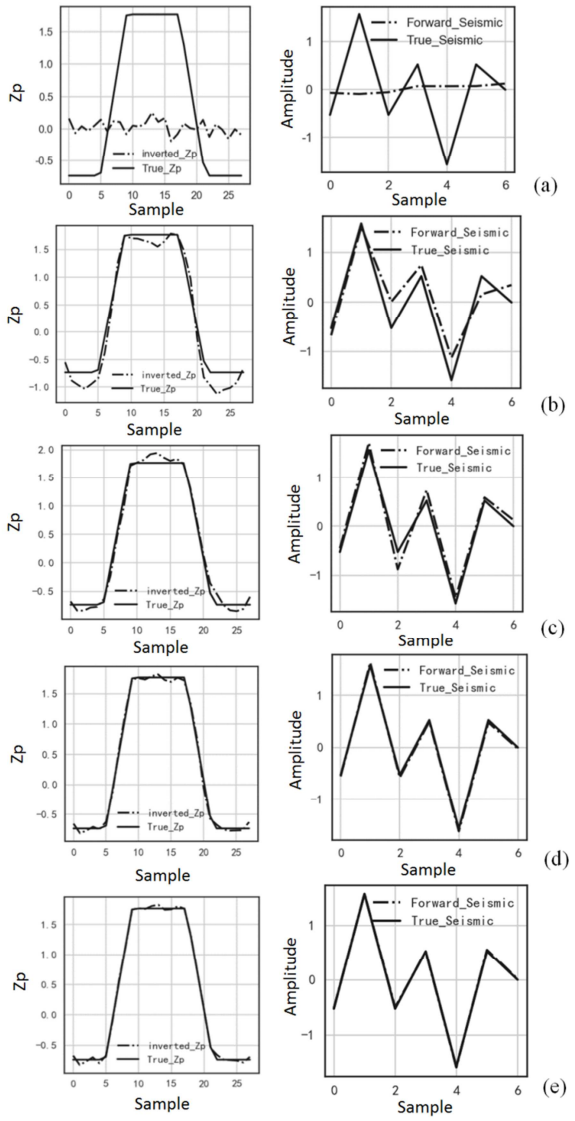


Figure 12. *The inverted and true impedance (left-hand side) and the synthetic and true seismic data (right-hand side) after (a) 1 (b) 20 (c) 40 (d) 60 (e) 80 train rounds.*

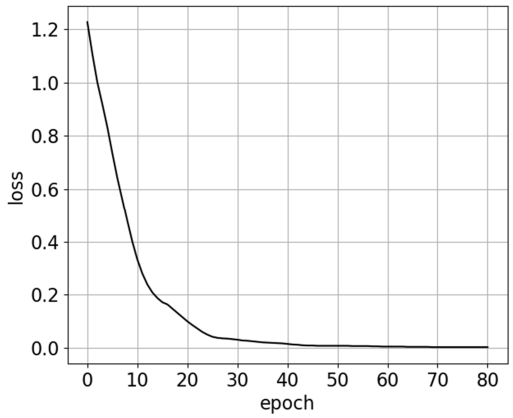


Figure 13. *The evolution of the loss function for the hybrid deep learning inversion.*

4. Conclusion

The complexity of subsurface structure, anisotropy of speed, and seismic noise contamination, all lead to a highly nonlinear seismic-geological forward model, to which the deep learning scheme is very applicable. The deep learning process is highly integrated with the seismic inversion, and the process is a highly complex systematic project with many concepts, parameters, and modules, which leads to the black box phenomena of deep learning inversion process which is difficult to understand.

(1) In this article, we explain and visualize the extraction of local seismic features by CNN, extraction of low-frequency seismic features by GRU, activation layer, Adam and learning rate from the GRU network, the initialization model, loss function calculation, and training process, all of which reveal the internal operating mechanism as a decomposition of the building block. We think the detailed illustration could play a certain role in avgood understanding of the deep learning inversion process.

(2) We extracted 8 different local features of the seismic data by CNN. The numerical process shows that the initial value is the key reason for the difference of the 8 features, likewise the difference between the 8 low-frequency features extracted by GRU. After multiple epoch iterations, the 8 features extracted by CNN is close to the low-frequency information involved in conventional seismic inversion (which is consistent with the cognition in commercial software, e.g., Jason, Geoview, etc.).

(3) The learning rate is a very critical parameter in the optimization algorithm. Comparing with the constant learning rate, the cosine annealing learning rate converges faster with better performance. By simulating the distribution curve of neuron nodes (weights), it facilitates understanding the reasons for the different convergence speeds of different learning rate schemes.

(4) The initial impedance model in hybrid deep learning is to initialize the weights of all neurons, after which the initial impedance can be generated. It is essentially different from those of conventional seismic inversion scheme, e.g., constrained sparse pulse inversion.

Acknowledgements

The authors appreciate the supports provided by the Fund Project of Guangdong Provincial Philosophy and Social Science Planning Foundation, No. GD20XYJ04; Talent Special Foundation of Lingnan Normal University, No. ZL1936; Guangdong Coastal Economic Belt development Research Center, Lingnan Normal University (20222L03); the Open Project of Mangrove Research Institute, Lingnan Normal University (YBXM11).

References

[1] HINTONS G E, SALAKHUTDINOV R R., 2006, Reducing the dimensionality of data with neural networks: Science, 313, 504-507. doi: 10.1126/science.1127647.

[2] KRIZHEVSKY A, SUTSKEVER I, HINTON G E., 2012, Imagenet classification with deep convolutional neural networks: Proceedings of the International Conference on Neural Information Processing Systems (NIPS), 2012: 1097-1105.

[3] RUMELHART D, HINTON G, WILLIAMS R., 1986, Learning representations by back propagating errors: Nature, 323, 533-536. doi: 10.1038/323533a0.

[4] ZHAO Gaishan., 2019, Road to intelligent petroleum geophysical exploration: From automatic to intelligent: Geophysical Prospecting for Petroleum, 58, 791-810. doi: 10.3969/j.issn.1000-1441.2019.06.002.

[5] Wu X M, Shi Y Z, Fomel S, et al., 2019, FaultNet3D: Predicting Fault Probabilities, Strikes, and Dips With a Single Convolutional Neural Network: IEEE Transactions on Geoscience and Remote Sensing, 57, 9138-9155. doi: 10.1109/TGRS.2019.2925003.

[6] HAN Wei-xue, ZHOU Ya-tong, CHI Yue., 2018, Deep learning convolutional neural networks for random noise attenuation in seismic data: Geophysical Prospecting for Petroleum., 57, 862-869. doi: 10.3969/j.issn.1000-1441.2018.06.008. (in Chinese).

[7] LIU Li-hui, LU Rong, YANG Wen-kui., 2019, Seismic lithofacies inversion based on deep learning: Geophysical Prospecting for Petroleum, 58, 123-129. doi: 10.3969/j.issn.1000-1441.2019.01.014. (in Chinese).

[8] AN Pen, CAO Dan-ping, ZHAO Bao-yin, et al., 2019, Reservoir physical parameters prediction based on LSTM recurrent neural network: Progress in Geophysics, 34, 1849-1858, doi: 10.6038/pg2019CC0366. (in Chinese).

[9] ZHANG Dong-xiao, CHEN Yun-tian, MENG Jin., 2018, Synthetic well logs generation via Recurrent Neural Networks: Petroleum Exploration and Development, 45, 598-607. doi: 10.11698/PED.2018.04.06. (in Chinese).

[10] WANG Wen-qiang, MENG Fan-shun, SUN Wenliang., 2019, Trace editing based on optimized convolutional neural network: Progress in Geophysics., 34, 0214-0220, doi: 10.6038/pg2019BB0387. (in Chinese).

[11] Alfarraj M, AlRegib G., 2019, Semisupervised sequence modeling for elastic impedance inversion: Interpretation, 7, 237-249. doi: 10.1190/int-2018-0250.1.

[12] Russell, B., 2019, Machine learning and geophysical inversion –A numerical study: The Leading Edge, 38, 498-576. doi: 10.1190/tle38070512.1.

[13] Kim, Y., and N. Nakata, 2018, Geophysical inversion versus machine learning in inverse problems: The Leading Edge, 37, 894–901, doi: 10.1190/tle37120894.1.

[14] Naeini, E. Z., and Prindle, K., 2018, Machine learning and learning from machines: The Leading Edge, 37, 886–893. doi: 10.1190/tle37120886.1.

[15] Smith L N., 2017, Cyclical Learning Rates for Training Neural Networks: 2017 IEEE Winter Conference on Applications of Computer Vision (WACV), IEEE, 2017: 464-472. doi: 10.1109/WACV.2017.58.