APPLIED GEOPHYSICS, Vol.18, No.4 (December 2021), P. 499–514, 13 Figures.
DOI:10.1007/s11770-021-0913-3

Seismic velocity inversion based on CNN-LSTM fusion deep neural network*

Cao Wei[1], Guo Xue-Bao♦[2], Tian Feng[1], Shi Ying[2], Wang Wei-Hong[2], Sun Hong-Ri[2], and Ke Xuan[2]

Abstract: Based on the CNN-LSTM fusion deep neural network, this paper proposes a seismic velocity model building method that can simultaneously estimate the root mean square (RMS) velocity and interval velocity from the common-midpoint (CMP) gather. In the proposed method, a convolutional neural network (CNN) Encoder and two long short-term memory networks (LSTMs) are used to extract spatial and temporal features from seismic signals, respectively, and a CNN Decoder is used to recover RMS velocity and interval velocity of underground media from various feature vectors. To address the problems of unstable gradients and easily fall into a local minimum in the deep neural network training process, we propose to use Kaiming normal initialization with zero negative slopes of rectified units and to adjust the network learning process by optimizing the mean square error (MSE) loss function with the introduction of a freezing factor. The experiments on testing dataset show that CNN-LSTM fusion deep neural network can predict RMS velocity as well as interval velocity more accurately, and its inversion accuracy is superior to that of single neural network models. The predictions on the complex structures and Marmousi model are consistent with the true velocity variation trends, and the predictions on field data can effectively correct the phase axis, improve the lateral continuity of phase axis and quality of stack section, indicating the effectiveness and decent generalization capability of the proposed method.
Keywords: Velocity inversion, CNN-LSTM, fusion deep neural network, weight initialization, training strategy

Introduction

The velocity of the subsurface media is an important parameter in the field of seismic exploration. Seismic velocity inversion methods can be broadly divided into two categories. One is the conventional method based on physical mechanisms, such as ray travel time tomography (Hole, 1992), wave equation tomography (Woodward, 1992), migration velocity analysis (Liu and Bleistein, 1995), and full-waveform inversion (FWI, Pratt et al., 1998), etc. For physics-driven methods, it is generally necessary to construct forward and adjoint operators that satisfy the physical propagation law of partial or holistic

Manuscript received by the Editor February 16, 2021; revised manuscript received November 30, 2021.
*This research is financially supported by the Key Project of National Natural Science Foundation of China (No. 41930431), the Project of National Natural Science Foundation of China (Nos. 41904121, 41804133, and 41974116) and Joint Guidance Project of Natural Science Foundation of Heilongjiang Province (No. LH2020D006).
1. School of Computer and Information Technology, Northeast Petroleum University, Daqing 163318, China.
2. School of Earth Science, Northeast Petroleum University, Daqing 163318, China.
♦Corresponding author: Guo Xue-Bao (E-mail: guoxuebao1108@163.com).

information of seismic waves, and iteratively update the constructed inverse problem to obtain the final result. These methods, however, are typically time-consuming and computationally demanding (Meng and Scales, 1996; Li et al., 2012; Chai et al., 2018). Furthermore, they are easily influenced by factors like the initial model, which results in a local minimum (Lian et al., 2018; Guo et al., 2019; Biswas et al., 2020). Some researchers used regularization algorithms (Guitton, 2012; Lin and Huang, 2014) to alleviate the ill-posedness of inverse problem, improving the accuracy of velocity inversion.

The other is the data-driven deep learning (DL) method. The dominant models of DL (LeCun et al., 2015) are neural networks, which include convolutional neural networks (CNNs), deep belief networks, recurrent neural networks (RNNs), long short-term memory networks (LSTMs), generative adversarial networks, and others. CNNs are a type of deep feedforward neural network that contains convolution operators. They have the properties of local connectivity and weight sharing and have strong fine-grained feature extraction as well as classification-recognition capabilities in image processing problems. LSTMs are gated RNN models developed by Hochreiter and Schmidhuber (1997) to address the problem of gradient dispersion and explosion in sequence modeling. LSTMs rely on the gate mechanism, forget mechanism and recurrence mechanism to function properly and fully utilize the benefits of long-term memory and recurrent memory (Huang, 2020). Due to the advantages in time series data modeling, LSTMs are widely used in natural language processing (Graves et al., 2004; Gulordava et al., 2018).

Fabien-Ouellet and Sarkar (2020) adopted a deep neural network consisting of CNN and LSTM to estimate the root mean square (RMS) velocities as well as the interval velocities of the layered models. In addition to the purely data-driven methods (Li et al., 2020; Liu et al., 2021), combining neural networks with FWI can effectively improve the results of velocity inversion (Mao et al., 2019; Sun et al., 2020; Kazei et al., 2021). Alzahrani and Shragge (2021) developed a frequency-stepping approach to allow the neural network to progressively reconstruct complex velocity models. Sun et al. (2021) incorporated physical laws into the training process of the neural network, and at the same time, minimized data-driven model misfits and physics-based data residuals. In data-driven mode, this approach reduces the demand for data diversity and volume of the training dataset, significantly improving the accuracy of the predicted velocity models and stabilizing wave propagation. Methods for building velocity models based on DL can be considered low-cost solutions. Although deep neural networks require a large number of computational resources to continuously update the network's weights using the optimization algorithm during training, once the network parameter model with good generalization capability is successfully trained, the time and computational cost of testing can be avoided.

Based on the research work of Fabien-Ouellet and Sarkar (2020), this paper proposes a CNN-LSTM fusion deep neural network to automatically build the RMS velocity model and interval velocity model of subsurface media from the CMP gather. The CNN-LSTM fusion network leverages CNN and LSTM to construct feature extractors to extract features in spatial and temporal dimensions, respectively, and then constructs a new and universal CNN Decoder to recover RMS velocity and interval velocity information from different feature vectors. Additionally, we compare the CNN-LSTM fusion network and single neural network models. To address the issue of deep neural network training difficulty, this paper investigates the effects of weight initialization schemes and training strategy on network performance. The tests are carried out using synthetic model data as well as field data. The synthetic model data includes complex structures, and single-trace layered velocity models extracted from the complex structures and Marmousi model, in addition, testing data that is similar to samples in the training dataset but does not appear in the training dataset. The numerical experiments demonstrate the effectiveness and well generalization capability of the proposed CNN-LSTM fusion network.

Network architecture and theory

The conventional seismic velocity inversion methods obtain the velocity model parameters from the observed data, and the objective function can be expressed by equation (1):

$$V^* = \min\left\{\left\|F\left(V\right) - D\right\|_2^2 + \alpha J\left(V\right)\right\}. \qquad (1)$$

where $F(\bullet)$ denotes the forward operator (e.g., wave equation), which simulates synthetic seismic data from a given velocity model V, D is the observed

data, V^* represents the ultimate velocity model obtained by inversion, $||\bullet||_2$ stands for L_2 norm, $J(V)$ denotes regularization term, and α represents the scaling factor. The ultimate inversion result is obtained by iteration through an optimization algorithm.

Seismic velocity inversion methods based on a single neural network model of CNN or LSTM typically exploit the advantages of CNN in extracting spatial features or LSTM in mining temporal features to establish the functions between seismic data (e.g., common shot gather, CMP gather, velocity spectra, etc.) and velocity (e.g., RMS velocity, interval velocity, velocity error, etc.) to recover velocity information of subsurface structures. A single network model, on the other hand, cannot extract all of the velocity features. As a result, as shown in Figure 1, we propose a CNN-LSTM fusion deep neural network composed of CNN Encoder, LSTM-RMS Encoder, LSTM-Interval Encoder, RMS Decoder, and Interval Decoder. In the case of unknown seismic wave propagation theory and underlying physical laws, this data-driven velocity model building method encodes the inversion operator into network parameters, and its objective function is expressed by equation (2):

$$Y_\theta^*=L\left(\left(V_i^R;V_i^I\right),Y_\Theta\left(C_i\right)\right). \tag{2}$$

where Θ stands for network parameters, $Y_\Theta(\bullet)$ represents inversion operator, Y_θ^* denotes the optimal inversion operator, $L(\bullet)$ represents a loss function, and $\left\{C_i,\left(V_i^R;V_i^I\right)\right\}$ denotes training data pairs. Each training data pair is composed of CMP gather C_i, which services as the input of the network, time-domain 1D RMS velocity label V_i^R, and time-domain 1D interval velocity label V_i^I.

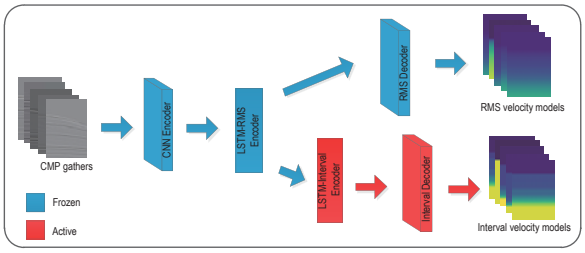

Figure 1. Architecture of CNN-LSTM fusion deep neural network.

CNN Encoder extracts the multi-scale spatial dimension information from the seismic waveform signals through fifteen "Conv2d+ReLU" modules with different convolutional kernel sizes. Following that, the temporal dimension information is extracted using the LSTM-RMS Encoder. The RMS Decoder recovers the output of the LSTM-RMS Encoder as the final RMS velocity model. In addition, an LSTM-Interval Encoder is used to capture richer velocity information on top of the feature vectors extracted by the LSTM-RMS Encoder, the output of which is eventually recovered as the interval velocity model by the Interval Decoder. The design details of each component are shown in Table 1. CNN Encoder can be represented by equation (3):

$$x^{(l+1)}=ReLU(Conv2d(x^{(l)})), \tag{3}$$

$$\begin{cases} Conv2d\left(x\right)_{(i,j)}=\sum_m\sum_n K_{m,n}\times x_{(i-1)\times s+m-2p,(j-1)\times s+n-2p} \\ ReLU\left(x\right)=\begin{cases} x & if\ x\geq 0 \\ 0 & if\ x<0 \end{cases} \end{cases}. \tag{4}$$

where l represents the index of the network layer, $Conv2d$ is 2D convolution (LeCun et al., 1989; Bouvrie, 2006), s represents the stride of convolution, p denotes padding, which is used to control the number of zeros added to the edges of feature maps in our proposed network, and $ReLU$ is rectified linear unit (Nair and Hinton, 2010). Moreover, RMS Decoder and Interval Decoder adopt the same CNN architecture, but the inputs to both are distinct, which satisfy equation (5):

$$x^{(l+1)}=LeakyReLU(BN(Conv2d(x^{(l)}))), \tag{5}$$

$$\begin{cases} BN_{\gamma,\beta}(x)=\gamma\left(\frac{x-\mu_\mathcal{B}}{\sqrt{\sigma_\mathcal{B}^2+\epsilon}}\right)+\beta \\ LeakyReLU(x)=\begin{cases} x & if\ x\geq 0 \\ ax & if\ x<0 \end{cases} \end{cases}. \tag{6}$$

where BN denotes Batch Normalization (Ioffe and Szegedy, 2015), $LeakyReLU$ is a leaky rectified linear unit (Maas et al., 2013), μ_β denotes the mini-batch mean, $\sigma_\mathcal{B}^2$ represents mini-batch variance, γ and β are learnable parameters, which represent the scaling and shift separately, ϵ is set to a smaller value for the stability of calculation. In this paper, $\epsilon = 1e\text{-}5$, $a = 0.2$.

A LSTM unit (Figure 2) is used by the LSTM-RMS Encoder and the LSTM-Interval Encoder. The cell state and gate mechanism form the heart of LSTM. Cell state moves information from earlier time steps to later time steps, avoiding short-term memory. The gate mechanism mainly relies on the sigmoid activation function (Han and Morag, 1995) to map the output vector in the

interval [0,1] to indicate the importance of information, and then multiplies the output vector with the vector to be controlled by element-wise to make the network learn which information should be retained or forgotten. Through the gate mechanism, information is added to or removed from the cell state. Graves et al. (2013) contain a detailed calculation procedure for the LSTM unit. Equation (7) can be used to express the LSTM-RMS Encoder and LSTM-Interval Encoder:

$$x^{(l+1)} = LSTM(x^{(l)}). \tag{7}$$

Table 1 Design details of CNN-LSTM fusion deep neural network

Component	Input size	Layers	Kernel size/ Input_size	Stride/ Hidden_size	Output size
CNN Encoder	(B,1,2048,22)	{Conv2d+ReLU}*15	$\begin{bmatrix}15\times1, & 16\\ 1\times9, & 16\\ 15\times1, & 32\\ 1\times9, & 32\end{bmatrix}$*1 $[15\times3,\ 32]$*7 $[1\times2,\ 32]$*4	(1,1)*11 (1,2)*4	(B,32,2048,1)
LSTM-RMS Encoder	(2048,B,32)	LSTM	32	200	(2048,B,200)
LSTM-Interval Encoder	(2048,B,200)	LSTM	200	200	(2048,B,200)
RMS Decoder	(B,200,2048,1)	{Conv2d+BN+LeakyReLU}*5	[1×1,1] *5	(1,1)*5	(B,1,2048,1)
Interval Decoder	(B,200,2048,1)	{Conv2d+BN+LeakyReLU}*5	[1×1,1] *5	(1,1)*5	(B,1,2048,1)

* "Kernel size" and "Stride" apply to components of CNN architecture, "Input_size" and "Hidden_size" apply to components of LSTM architecture.

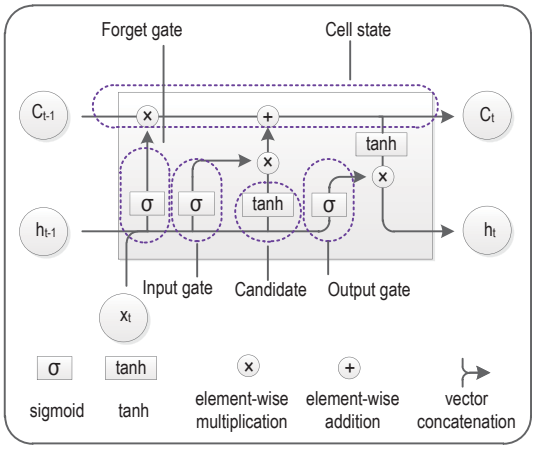

Figure 2. Architecture of the LSTM unit. The detailed calculation procedure of the LSTM unit can be found in the article of Graves et al. (2013).

Method

Dataset construction

Our dataset contains 1232 data pairs $\left\{C_i, \left(V_i^R; V_i^I\right)\right\}$, and the ratio of training sample quantities to testing sample quantities is 1200:32. Firstly, the 1D layered piecewise constant velocity models in depth-domain are randomly generated. The velocity model parameters are as follows: velocity range is from 1.50 km/s to 4.85 km/s, the velocity of the first layer is 1.50 km/s–2.00 km/s, the number of layers is 2–20, the position of the first interface is 0.5 s–0.7 s, each layer has at least 5 grid points in thickness, and the number of vertical grid points is 200. Following that, the 1D depth-domain interval velocity models are converted to labels V_i^R and V_i^I by using Seismic Unix. Finally, the 1D depth-domain interval velocity models are stretched to 2D depth-domain interval velocity models, and each velocity model is conducted forward process by utilizing the random wavelet with phase rotation to obtain CMP gather at the middle position. We solve the 2D acoustic wave equation with the time-domain finite-difference method. Detailed parameter settings are as follows: first-derivative Gaussian wavelet and Ricker wavelet are randomly adopted, the dominant frequency range is from10 Hz to 40 Hz, the phase rotation is chosen randomly from 0 to π, the middle position is the self-excited self-receiving point, the sources are placed on the left side of this point, the receivers are placed on the right side of this point, there are 22 sources with an interval of 30 m, the number and interval of receivers

are the same as the sources, the time sampling interval is 0.001 s, and the receiving time is 2.048 s. Thus, the dimension of C_i is 2048*22, and those of V_i^R and V_i^I are both 2048*1. C_i, V_i^R and V_i^I are performed normalization. Figure 3 shows two training data pairs, where the CMP gather in the first row was obtained by first-derivative Gaussian wavelet with the dominant frequency of 19 Hz and the phase rotation of 1/8 π, and the CMP gather in the second row was obtained by the Ricker wavelet with the dominant frequency of 38 Hz and the phase rotation of 1/2 π. The interval velocity mentioned below refers to the time-domain interval velocity unless otherwise stated. All experiments in this paper are implemented on the Pytorch framework.

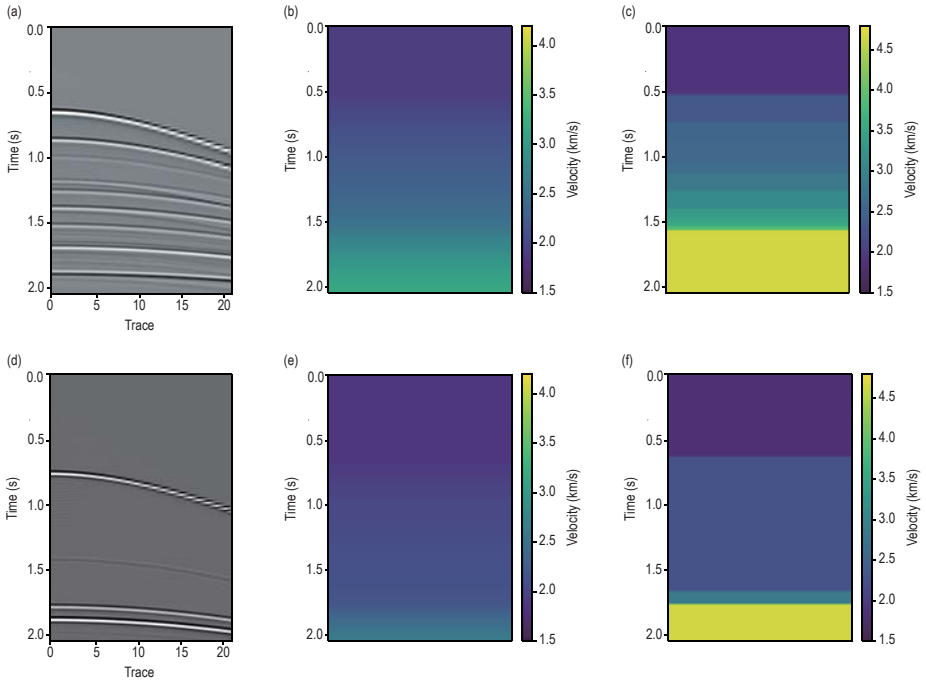

Figure 3. Two data pairs in the training dataset.
(a), (d) CMP gathers. (b), (e) Labels of RMS velocity. (c), (f) Labels of interval velocity.

Weight initialization

The weight initialization affects the neural network's convergence speed and performance, and an incorrect initialization scheme affects the nonlinear model's learning result. Normal distribution initialization, orthogonal initialization (Saxe et al., 2013), Xavier initialization (Glorot and Bengio, 2010), Kaiming initialization (He et al., 2015), and Pre-train initialization are examples of conventional initialization schemes. Xavier initialization can reduce gradient dispersion, but it has the limitation of making the output of tanh activation function in the deep layer obey the standard Gaussian distribution; it is ineffective for the ReLU activation function. He et al. (2015) considered the influence of ReLU on the output data distribution and proposed Kaiming initialization to solve the above problem. Orthogonal initialization is commonly used in RNNs. As the time step increases, repeated weight matrix multiplication makes RNNs prone to gradient explosion and vanishing, while the absolute value of orthogonal matrix eigenvalue is equal to 1, so that repeated matrix multiplication will not cause gradient explosion and vanishing.

This paper's neural network architecture uses ReLU in the CNN Encoder and LeakyReLU in the RMS Decoder and Interval Decoder. As a result, we compare the effects of normal distribution initialization, orthogonal initialization, and Kaiming initialization on the proposed network's convergence speed and performance, where normal distribution initialization, Kaiming normal initialization, and Kaiming uniform initialization allow the corresponding network parameters sampled from the distributions in equations (8)–(10):

$$W \sim N(0, std), \tag{8}$$

$$W \sim N\left(0, \sqrt{2 / \mathrm{n}_l}\right), \tag{9}$$

$$W \sim U\left(-\sqrt{6/\mathrm{n}_l}, \sqrt{6/n_l}\right). \tag{10}$$

where *std* denotes the standard deviation of the normal distribution, $n=k^2c$, k represents the size of convolution kernel, and c stands for the number of the input channel.

Training strategy

The performance of a neural network not only depends on the architecture and weight initialization scheme but also is closely related to the training strategy. The CNN-LSTM fusion deep neural network's training process aims to establish a mapping relationship between CMP gathers and RMS velocities, as well as interval velocities. We compare the effects of the MSE loss function and the SmoothL1 loss function (Girshick, 2015) on network performance when combined with a freezing factor.

When the network is optimized to minimize the MSE loss between the target and predicted values, all parameters in the network are iteratively updated according to the MSE loss between the target RMS velocities and the predicted RMS velocities until satisfactory inversion results of RMS velocities are obtained. Subsequently, the neurons are responsible for retrieving the RMS velocities (shown in blue in Figure 1) are frozen. Finally, the remaining active neurons (shown in red in Figure 1) are used to establish a mapping relationship between CMP gathers and interval velocities by minimizing the MSE loss between them. The equation (11) can be used to represent the objective function:

$$L = (1-F)\times L_{MSE}^{RMS} + F \times L_{MSE}^{Interval}, \tag{11}$$

$$\begin{cases} L_{MSE}^{RMS} = \dfrac{1}{nt \times B}\left\| \mathrm{V}^R - \widehat{\mathrm{V}}^R \right\|_2^2 \\ L_{MSE}^{\mathrm{Interval}} = \dfrac{1}{nt \times B}\left\| \mathrm{V}^I - \widehat{\mathrm{V}}^I \right\|_2^2 \end{cases}. \tag{12}$$

where F denotes the freezing factor, which is equal to 0 or 1, B represents the batch size, nt denotes the number of samples in the time-domain, and $\widehat{\mathrm{V}}$ indicates the predicted velocities. Similarly, when the network minimizes SmoothL1 loss between target and predicted velocities, the objective function can be expressed by equation (13):

$$L = (1-F)\times L_{SmoothL1}^{RMS} + F \times L_{SmoothL1}^{Interval}, \tag{13}$$

$$\begin{cases} L_{SmoothL1}^{RMS} = \dfrac{1}{nt \times B}\left\| Z^R \right\|_1 \\ L_{SmoothL1}^{Interval} = \dfrac{1}{nt \times B}\left\| Z^I \right\|_1 \\ Z = \begin{cases} 0.5\left(\mathrm{V} - \widehat{\mathrm{V}}\right)^2 & if \left|\mathrm{V} - \widehat{\mathrm{V}}\right| < 1 \\ \left|\mathrm{V} - \widehat{\mathrm{V}}\right| - 0.5 & otherwise \end{cases} \end{cases}. \tag{14}$$

Inversion examples

Synthetic model data analysis

We tested the proposed method on five types of synthetic model data in this section. The first type is the 32 samples in the testing dataset. Figures 4(a) and 4(b) are different complex synthetic models, where Figure 4(a) is processed by a smoothing operator. The second is single-trace layered models extracted from the position as indicated by the red lines in the models shown in Figures 4(a) and 4(b). The third is the complex undulating media shown in Figure 4(b). Figure 4(c) is the Marmousi model. The fourth is a single-trace layered model with relatively strong vertical velocity variation extracted from the position as indicated by the red line in the Marmousi model. The fifth is 2D complex undulating media shown in Figure 4(d). The five types of synthetic model data are tested with the same network parameter model.

The CNN-LSTM fusion deep neural network adopts Scheme 1 in Table 2 to initialize the weights, which sets the negative slopes of rectified units to zero in Kaiming normal initialization, that is, nonlinearity = relu. When establishing the mapping relationship between CMP gathers and RMS velocities, the network does not freeze the parameters, whereas when establishing the mapping relationship between CMP gathers and interval velocities, the neurons that retrieve RMS velocities are frozen. Throughout the training process, the adaptive moment estimation (Adam) algorithm (Kingma and Ba, 2014) is used to optimize the objective function represented by equation (11). In the process of establishing the two mapping relationships, the batch size is 30, and both initial learning rates are 1e-3. When the loss values no longer decrease, the training process is rolled back, and then the decay coefficient is determined from 0.1 or 0.5 through trial and error experiment. The lower limit of both learning rates is 1e-6. The CNN-LSTM fusion network infers the RMS velocities and interval velocities of 32 samples in the testing dataset. Table 3 displays the

mean MSE losses of two types of velocity in the holistic receiving time and the time from 0 to receivers acquiring information from the final reflection interface. Although the velocity information in the last layer of underground media is not carried by the seismic waveform records, it is carried by both velocity labels. Furthermore, the neural network only learns the laws hidden in the data by establishing complex functions; thus, the inference capability of the network on the velocity of the last layer affects the performance of velocity inversion in the holistic receiving time. The numerical results show that the network's inference capability on the RMS velocity of the last layer is relatively weak, which reduces the inversion performance for RMS velocity in the holistic receiving time, but the interval velocity of the last layer inferred by the CNN-LSTM fusion network is more accurate, reducing the mean MSE loss of the interval velocity in the holistic receiving time. Furthermore, the mean MSE loss of RMS velocity (Mean MSE-RMS in Table 3) is found to be less than that of interval velocity (Mean MSE-Interval in Table 3). The reason is that the neurons used to learn RMS velocities are frozen when learning the mapping relationship between CMP gathers and interval velocities, allowing the network to continue to learn the interval velocities based on learning relatively accurate RMS velocities, which is consistent with the nature of the physics-based methods. Figure 5 (green line) depicts the inversion outcomes of Scheme 1. The experimental results show that the proposed CNN-LSTM fusion network can moderately predict RMS and interval velocities.

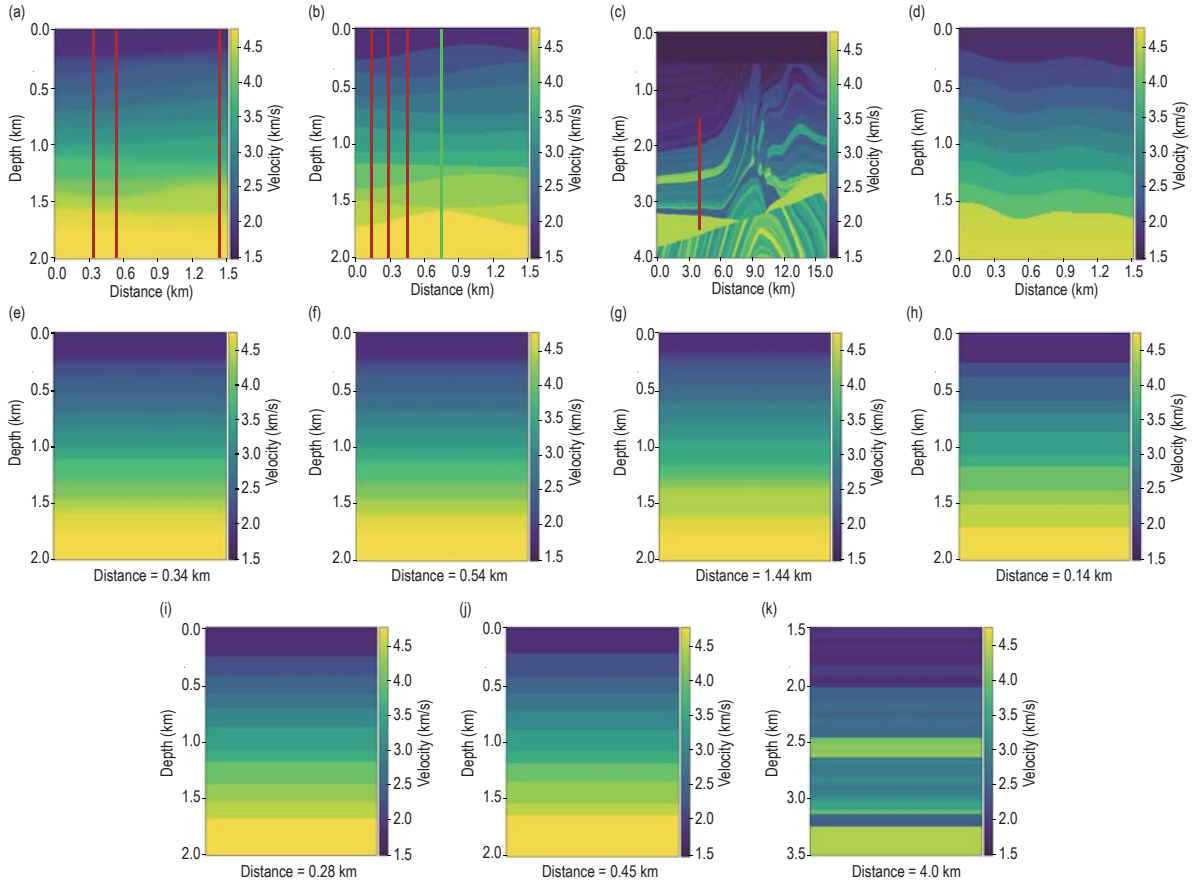

Figure 4. (a), (b), (d) Complex synthetic models. (c) Marmousi model. (e)–(g) Enlargements of the velocity models at the position of the red lines in Fig. 4(a), the distances are 0.34 km, 0.54 km, and 1.44 km, respectively. (h)–(j) Enlargements of the velocity models at the position of the red lines in Fig. 4(b), the distances are 0.14 km, 0.28 km, and 0.45 km, respectively. (k) Enlargement of the velocity model at the position of the red line in Fig. 4(c), the distance is 4.0 km, and the depth is from 1.5 km to 3.5 km.

We generate CMP gathers, RMS velocity and interval velocity labels using 2D models extended from single-trace models (Figures 4(e)–4(g), 4(h)–4(j), and 4(k)) and a 2D undulating model (Figure 4(b)). We use a 15 Hz Ricker wavelet to perform the forward process to obtain CMP gathers, with no phase rotation, while

all other parameters remain constant. Figure 6 (green line) depicts the CNN-LSTM fusion neural network's predictions on the RMS velocities and interval velocities shown in Figures 4(e)–4(g). The velocities in Figures 4(e)–4(g) increase with depth, and the thickness of each layer in the three depth-domain velocity models are different. However, after the interval velocities in the depth-domain are converted into RMS velocities and interval velocities in the time-domain, the difference between each type of velocity label of the three velocity models is relatively small, which leads to similar predictions from the network. Despite this, the CNN-LSTM fusion neural network can still provide moderately accurate predictions. Figure 7's green lines are the inversion results of RMS velocities and interval velocities corresponding to Figures 4(h)–4(j). The neural network can predict the approximate trends of the RMS velocity and interval velocity, which are close to the true velocity values. We perform the forward process on the 2D complex undulating media (Figure 4(b)) to obtain the CMP gather (Figure 8(a)) in the middle position, where the observation system matches that of "Dataset construction." Thus, the velocity model at the middle position of Figure 4(b) (green line) is the 1D depth-domain interval velocity label. Figure 8(b) shows the enlargement of the velocity model at the position of the green line in Figure 4(b), and the distance is 0.75 km, which is converted into the corresponding RMS velocity

Table 2 Weight initialization schemes for CNN-LSTM fusion deep neural network

Number	Scheme	Parameter settings			Iteration	Mean MSE-RMS	Mean MSE-Interval
		CNN Encoder	RMS Decoder	Interval Decoder			
1	Kaiming Normal Initialization	mode = fan_in, nonlinearity = relu			49480	**0.00231**	**0.01282**
2	Kaiming Normal Initialization	mode = fan_in, nonlinearity = relu	mode = fan_in, nonlinearity = leaky_relu		78440	0.05065	0.03484
3	Kaiming Uniform Initialization	mode = fan_in, nonlinearity = relu	mode = fan_in, nonlinearity = leaky_relu		19720	0.05068	0.03655
4	Orthogonal Initialization	gain = 1			73480	**0.00114**	**0.00427**
5	Normal Distribution Initialization	5-11 Conv2d:std = 1e-2, others Conv2d:std = 1e-1	std = 1e-4		42400	0.05070	0.03717
6	Normal Distribution Initialization	std = 1e-4			57880	0.02248	0.03644

* "mode = fan_in" represents the magnitude of the variance of the weights in the forward pass is preserved, "nonlinearity" denotes the nonlinear function, and "gain" stands for the optional scaling factor. Please see the "Pytorch Official Document" in the references for detailed instructions.

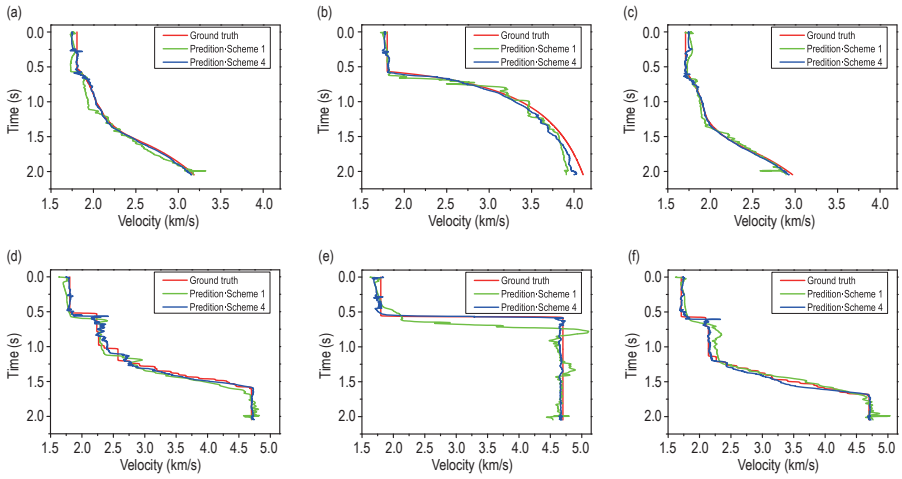

Figure 5. Inversion results of CNN-LSTM fusion deep neural network on 3 of the 32 testing samples. (a)–(c) Inversion results of RMS velocities. (d)–(f) Inversion results of corresponding interval velocities.

label and 1D time-domain interval velocity label. Figure 8(c) (green line) depicts the prediction of RMS velocity, while Figure 8(d) (green line) depicts the prediction of interval velocity. The results of the experiments show that the CNN-LSTM fusion neural network can still provide a relatively accurate velocity trends for complex

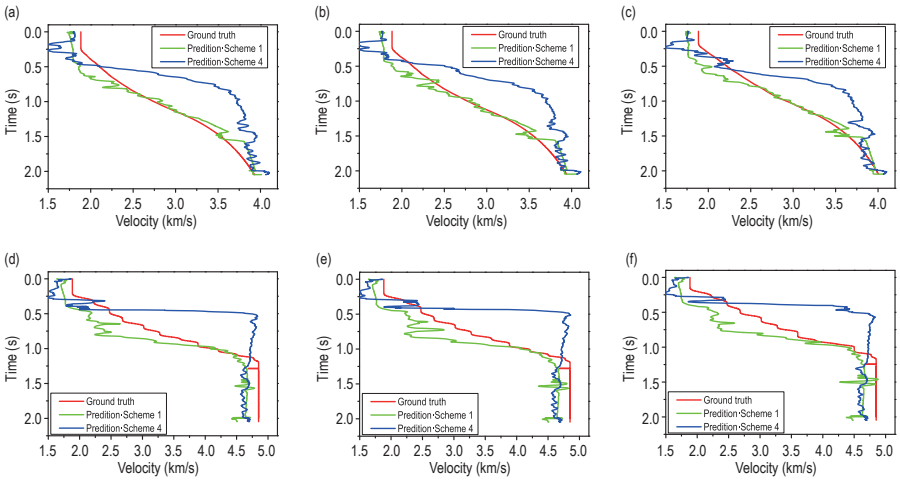

Figure 6. Inversion results of CNN-LSTM fusion deep neural network on smoothed synthetic models. (a)–(c) Inversion results of RMS velocities corresponding to Figs. 4(e)–4(g). (d)–(f) Inversion results of time-domain interval velocities corresponding to Figs. 4(e)–4(g).

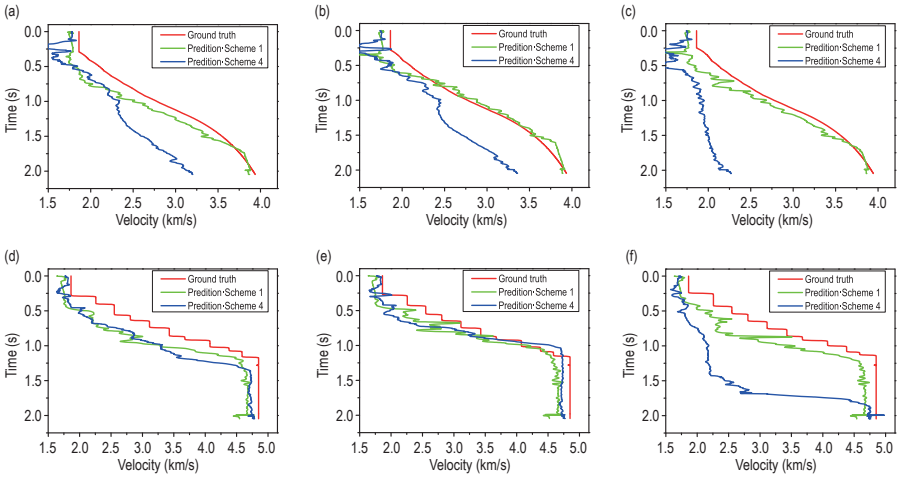

Figure 7. Inversion results of CNN-LSTM fusion deep neural network on non-smooth synthetic models. (a)–(c) Inversion results of RMS velocities corresponding to Figs. 4(h)–4(j). (d)–(f) Inversion results of time-domain interval velocities corresponding to Figs. 4(h)–4(j).

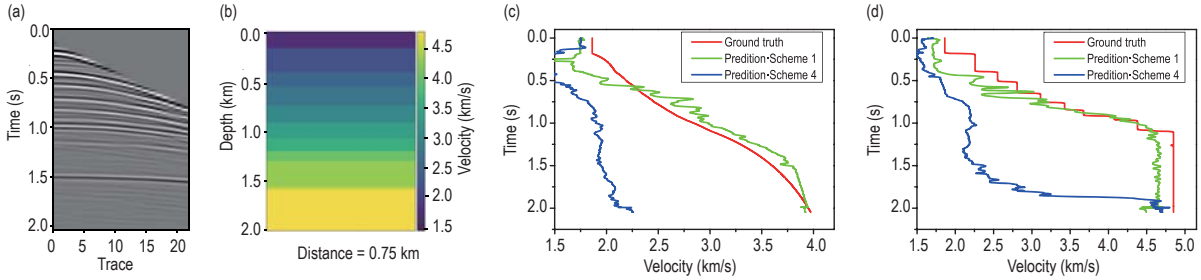

Figure 8. Inversion results of CNN-LSTM fusion deep neural network on complex undulating media. (a) CMP gather. (b) Enlargement of the velocity model at the position of the green line in Fig. 4(b), the distance is 0.75 km. (c) Inversion results of RMS velocity. (d) Inversion results of interval velocity.

structures. Figure 9's green lines depict the inversion results of RMS velocity and interval velocity, which correspond to Figure 4(k). Because both the RMS and interval velocities in the training dataset increase over time, the network does not learn how to deal with velocity reversals. The RMS velocity corresponding to Figure 4(k) roughly conforms to the law of increasing velocity with time; thus, the RMS velocity predicted by the neural network is relatively accurate, which is shown in Figure 9(a). Moreover, for interval velocity, the neural network can still provide a reasonably accurate velocity trend, except the position of velocity reversals, as shown in Figure 9(b). Although the proposed method uses CMP gathers to perform velocity analysis, it can still achieve good results even when there is horizontal undulation or relatively strong vertical velocity variation in velocity models, indicating that the CNN-LSTM fusion deep neural network has better generalization capability.

Figures 10(a) and 10(b) are the RMS velocity label and interval velocity label of the velocity model shown in Figure 4(d), respectively. All CMP gathers for the velocity model shown in Figure 4(d) are obtained using the method described in the section "Dataset construction." We use a 15 Hz Ricker wavelet to perform the forward process to obtain CMP gathers, with no phase rotation; other parameters remain unchanged. Figures 10(c) and 10(d) show the RMS velocity field and interval velocity field predicted by the CNN-LSTM

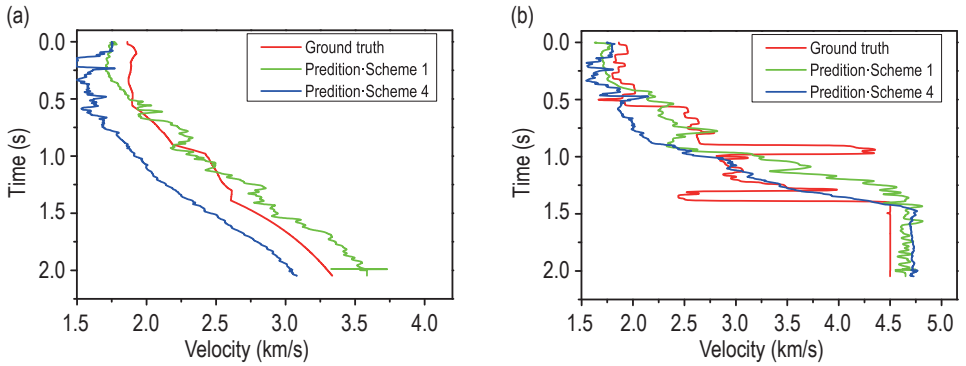

Figure 9. Inversion results of CNN-LSTM fusion deep neural network on the velocity model shown in Fig. 4(k). (a) Inversion results of RMS velocity. (b) Inversion results of time-domain interval velocity.

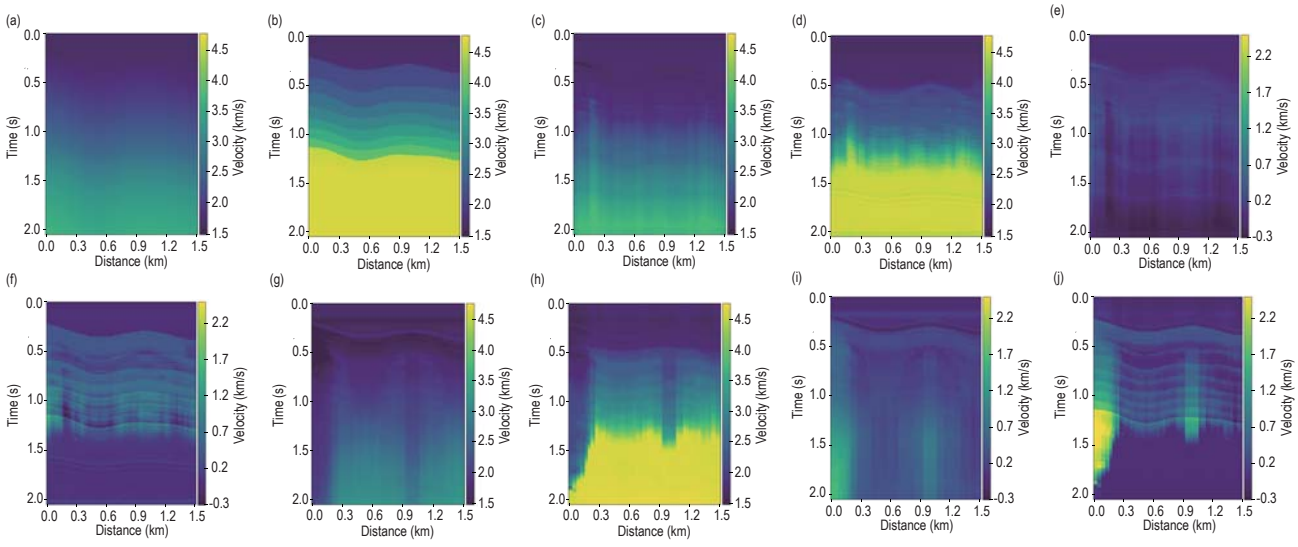

Figure 10. Inversion results of CNN-LSTM fusion deep neural network on the velocity field. (a) RMS velocity label of Fig. 4(d). (b) Interval velocity label of Fig. 4(d). (c) Inversion results of RMS velocity field when the weights are initialized with Scheme 1. (d) Inversion results of interval velocity field when the weights are initialized with Scheme 1. (e) Residual between Fig. 10(a) and Fig. 10(c), which is equal to Fig. 10(a) minus Fig. 10(c). (f) Residual between Fig. 10(b) and Fig. 10(d), which is equal to Fig. 10(b) minus Fig. 10(d). (g) Inversion results of RMS velocity field when the weights are initialized with Scheme 4. (h) Inversion results of interval velocity field when the weights are initialized with Scheme 4. (i) Residual between Fig. 10(a) and Fig. 10(g), which is equal to Fig. 10(a) minus Fig. 10(g). (j) Residual between Fig. 10(b) and Fig. 10(h), which is equal to Fig. 10(b) minus Fig. 10(h).

fusion neural network, respectively. Figure 10(e) is the residual between true RMS velocity field (Figure 10(a)) and predicted RMS velocity field (Figure 10(c)), and the residual value at each point does not exceed 0.3 km/s. Figure 10(f) is the residual between true interval velocity field (Figure 10(b)) and predicted interval velocity field (Figure 10(d)), and the average residual value of the interval velocity field is greater than that of the RMS velocity field. The CNN-LSTM fusion neural network predicts the horizontal and vertical variation trends of two types of velocity fairly accurately, but the RMS velocity field outperforms the interval velocity field. The prediction of interval velocity field can be used as the initial model of other high-resolution velocity model building methods (e.g., FWI), demonstrating the proposed method's effectiveness and strong generalization capability.

Field data test

In this section, the network parameter model used to test field data is consistent with that for testing synthetic model data. Figure 11(a) shows a CMP gather from the land seismic data in a certain exploration area, and the CMP number is 290. We perform velocity analysis on the CMP gather in Figure 11(a) using Seismic Unix, and the result is shown in Figure 11(b). Following that, we use the manually picked stacking velocity as the RMS velocity label (red line in Figure 11(c)) and then use Seismic Unix to convert the RMS velocity label to the time-domain interval velocity label (red line in Figure 11(d)). The green line in Figure 11(c) represents the

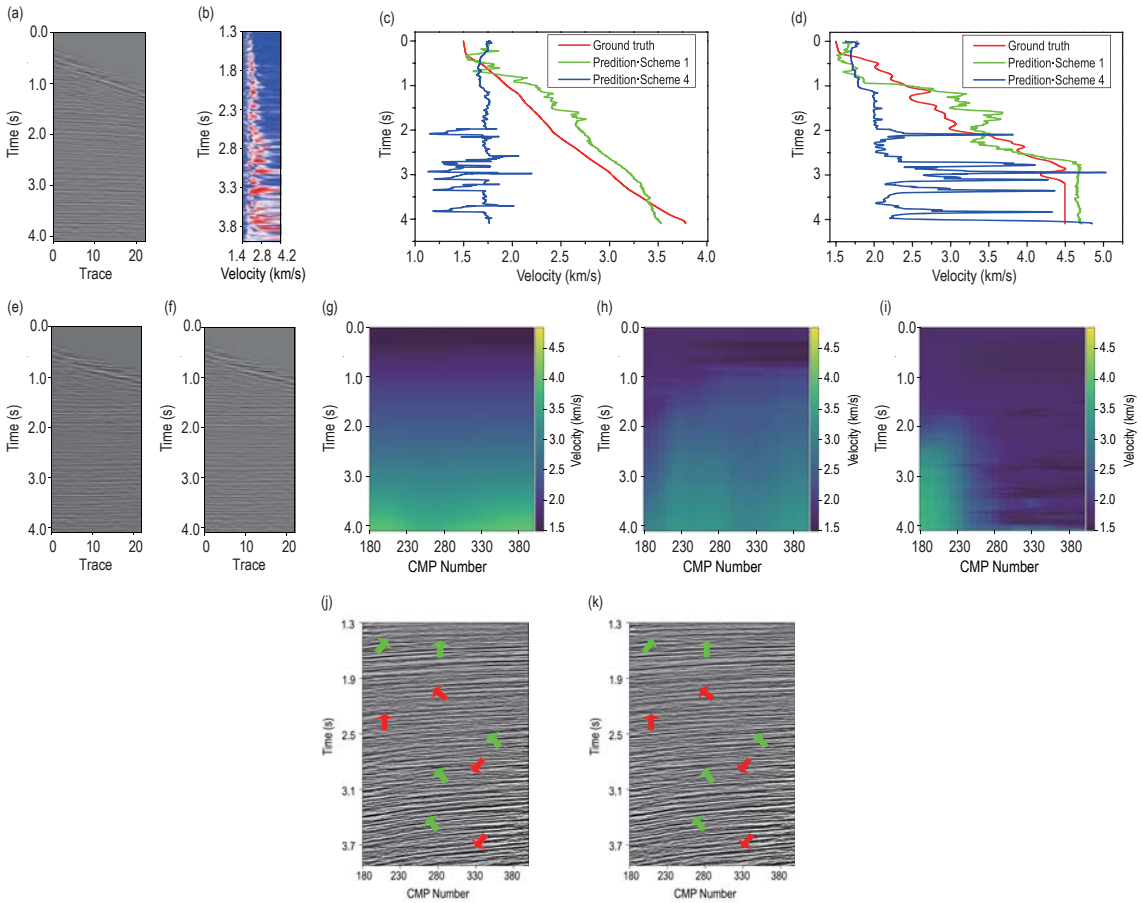

Figure 11. Inversion results of CNN-LSTM fusion deep neural network on land seismic data. (a) Land seismic data's CMP gather as the input of the CNN-LSTM fusion deep neural network. (b) Result obtained by performing velocity analysis on the CMP gather in Fig. 11(a). (c) Inversion results of RMS velocity. (d) Inversion results of interval velocity. (e) Result of NMO correction with manually picked stacking velocity. (f) Result of NMO correction with RMS velocity predicted by the network when the weights are initialized with Scheme 1. (g) Manually picked stacking velocity field. (h) RMS velocity field predicted by the network when the weights are initialized with Scheme 1. (i) RMS velocity field predicted by the network when the weights are initialized with Scheme 4. (j) Stack section obtained by the velocity field in Fig. 11(g). (k) Stack section obtained by the velocity field in Fig. 11(h).

CNN-LSTM fusion network's inversion result of RMS velocity, while the green line in Figure 11(d) represents the inversion result of interval velocity. The predictions for the two types of velocity are close to the targets. We employ the manually picked stacking velocity (red line in Figure 11(c)) and predicted RMS velocity (green line in Figure 11(c)) to perform normal moveout (NMO) correction, respectively. Figures 11(e) and 11(f) separately show the corresponding results. It can be seen that flattening the phase axis has nearly identical effects. Figures 11(g) and 11(h) depict the manually picked stacking velocity field and the RMS velocity field predicted by the CNN-LSTM fusion neural network, respectively, with CMP values ranging from 180 to 400. Figures 11(j) and 11(k) show stack sections produced by manually picked stacking velocity field and predicted RMS velocity field, respectively. Because the velocity features obtained by the two methods are relatively close, the stack section in Figure 11(k) is consistent with that in Figure 11(j). In Figures 11(j) and 11(k), the red arrows reveal that the quality of the stack section obtained by manually picked stacking velocity field is more prominent than that obtained by the predicted RMS velocity field, while the green arrows indicate the opposite, indicating that the velocity model building method based on CNN-LSTM fusion neural network can effectively improve the lateral continuity of phase axis and quality of stack section. Thus, the test results on field data illustrate that the proposed CNN-LSTM fusion neural network has good ability to retrieve the RMS velocities and interval velocities and possesses promising generalization capability.

Ablation experiment

Performance comparison of CNN-LSTM fusion deep neural network with single neural network models

We compare the performance of the CNN-LSTM fusion network to that of single neural network models to show that the proposed CNN-LSTM fusion deep neural network can simultaneously exploit the respective advantages of CNN and LSTM in feature extraction. Figure 12 shows single neural network models with only CNN architecture (a) and only LSTM architecture (b). In Table 2, Scheme 1 is used to initialize the weights of the CNN-LSTM fusion network and two single neural network models, while equation (11) serves as the objective function. Additionally, they adopt the same training strategy and training parameter settings. The predictions of 32 samples in the testing dataset provided by the three architectures are shown in Table 3. The numerical results indicate that the CNN-LSTM fusion network outperforms the single neural network models. The reason for this is that the CNN-LSTM

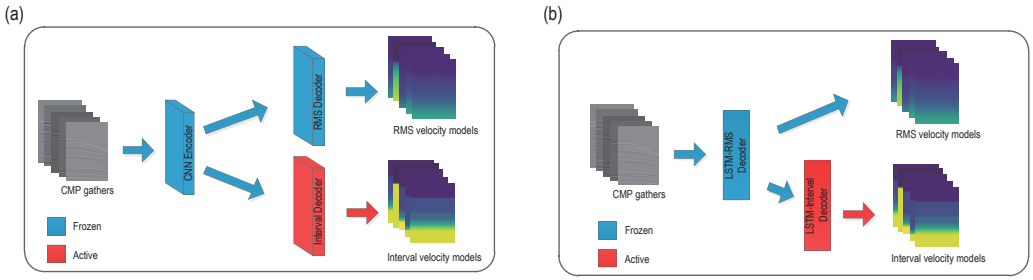

Figure 12. Two single neural network models. (a) Single neural network model with only CNN architecture. (b) Single neural network model with only LSTM architecture.

Table 3 Predictions of CNN-LSTM fusion deep neural network and single neural network models

Architecture	Mean MSE-RMS		Mean MSE-Interval	
	nt = 2048	nt = 0–final reflection interface	nt = 2048	nt = 0–final reflection interface
CNN-LSTM	**0.00231**	**0.00216**	**0.01282**	**0.01987**
CNN	0.03284	0.01053	0.07888	0.05636
LSTM	0.05158	0.02063	0.13104	0.09255

* "nt = 2048" represents the holistic receiving time and "nt = 0–final reflection interface" denotes the time from 0 to receivers acquire the information of final reflection interface.

fusion network can fully extract features in both spatial and temporal dimensions from CMP data and fully learn the velocity information of subsurface structures, whereas a single architecture can only extract a onefold feature. As a result, the inversion performance of single neural network models is limited. Furthermore, numerical experiments show that for the CNN-LSTM fusion network, the RMS velocities inferred by the RMS Decoder are converted to interval velocities by the Dix formula (Dix, 1955), which injects errors in predicted RMS velocities into the interval velocities, resulting in the interval velocities obtained through conversion being far from the targets, whereas the interval velocities predicted by the Interval Decoder being closer. This illustrates the necessity of setting RMS Decoder and Interval Decoder simultaneously in the CNN-LSTM fusion network.

Importance of weight initialization

We investigate the effects of six different initial weight settings on the performance of the CNN-LSTM fusion deep neural network using Kaiming initialization, orthogonal initialization, and normal distribution initialization. Because the interfaces are provided by the Pytorch framework, the relevant parameter settings are shown in Table 2, and detailed parameter instructions can be found in the references "Pytorch Official Document." All experiments in this paper decay the learning rate when the loss values no longer decrease, so the number of iterations is different for each experiment.

The RMS velocities and interval velocities of 32 testing samples are predicted by the final parameter model obtained from the network training. Table 2 displays the mean MSE losses in the holistic receiving time. The numerical results show that Schemes 1 and 4 can effectively converge loss curves, but orthogonal initialization requires more iterations, i.e., more computer resources. The predictions of synthetic model data and field data inferred by the network parameter model obtained from Scheme 4 show that, while Scheme 4 can obtain better inversion results than Scheme 1 on 32 testing samples, as shown in Figure 5 (blue line), the predictions obtained from Scheme 4 deviate from the RMS velocity and interval velocity labels for the single-trace velocity models extracted from the complex structures as well as the Marmousi model, and the complex undulating media corresponding to Figure 4(b). This phenomenon illustrates that the network initialized by Scheme 4 is prone to over-fit the training dataset, which degrades the generalization capability, and the corresponding experimental results are shown in Figures 6–9 (blue line). The predictions of the RMS velocity field and interval velocity field obtained by Scheme 4 are shown in Figures 10(g) and 10(h), respectively. At the position around 0–0.3 km and 0.9–1.1 km, the predicted RMS velocities and interval velocities are seriously far from the true values. Scheme 1 achieves greater lateral continuity of the RMS velocity field and interval velocity field than Scheme 4. Figure 10(i) depicts the residual between the true RMS velocity field (Figure 10(a)) and the RMS velocity field obtained by Scheme 4 (Figure 10(g)). The residual value at each point is less than 1.5 km/s, which is significantly higher than the result of Scheme 1. Figure 10(j) depicts the residual between the true interval velocity field (Figure 10(b)) and the interval velocity field obtained by Scheme 4 (Figure 10(h)). At the position around 0–0.3 km, there are residual values even above 2.4 km/s, which further indicates the superiority of Scheme 1 over Scheme 4.

The RMS velocity of field data (Figure 11(a)) predicted by the network parameter model obtained from Scheme 4 is revealed by the blue line in Figure 11(c), the prediction of interval velocity is indicated by the blue line in Figure 11(d), and the prediction of RMS velocity field with the CMP number from 180 to 400 is shown in Figure 11(i). The predictions of two types of velocity have significant vibrations and are inconsistent with the true velocity trends at the position where the CMP number is 290; additionally, the prediction of the 2D RMS velocity field deviates significantly from the manually picked stacking velocity field, indicating that Scheme 1 has superior generalization capability on field data than Scheme 4. Scheme 2, 3 and 5 cause the network to fall into local minima at various iterations during the training process, resulting in the objective function being unable to continue to converge and the quality of final inversion results being inaccurate; mode collapse also appears in these schemes when inferring RMS velocities. In addition, both RMS velocities and interval velocities predicted by the network parameter model obtained from Scheme 6 suffer from mode collapse.

Scheme 1 is currently the most appropriate weight initialization strategy for the proposed CNN-LSTM fusion deep neural network, as it not only produces nontrivial inversion results on the 32 testing samples but can also be generalized to synthetic model data and field data. As a result, the initial weights are

critical to the neural network's performance, as they not only determine whether the deep neural network can be trained successfully but also affect the objective function's convergence speed and inversion performance.

Impacts of different objective functions on CNN-LSTM fusion deep neural network performance

We also compare the impacts of objective functions corresponding to equations (11) and (13) on the performance of the CNN-LSTM fusion deep neural network. The two experiments use Scheme 1 in Table 2 to initialize weights, and the same training parameter settings are used in the training process. The experimental results show that using SmoothL1 loss as the optimization goal for the proposed CNN-LSTM network architecture is not appropriate. The reason is that the gradient dispersion occurs during the training process, resulting in the loss curve no longer going down in the later training stage when establishing the mapping relationship between CMP gathers and RMS velocities (Figure 13), while the neural network does not converge to a decent solution at this stage and the network falls into local minimum when establishing the mapping relationship between CMP gathers and interval velocities. We use the network parameter models built with the two different objective functions to predict the RMS velocities and interval velocities of 32 testing samples separately, and the results in holistic receiving time are shown in Table 4. It can be seen that the MSE loss function with a freezing factor is better suited for the proposed CNN-LSTM fusion network to perform the velocity inversion task of subsurface media.

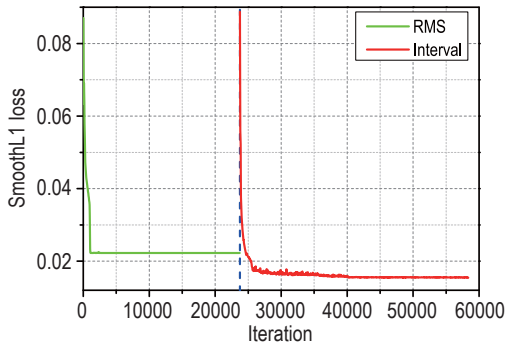

Figure 13. SmoothL1 loss curve.

Table 4 Comparison on the performance of CNN-LSTM fusion deep neural network with different objective functions

Objective function	Iteration	Mean MSE-RMS	Mean MSE-Interval	Mean SmoothL1-RMS	Mean SmoothL1-Interval
Equation (11)	**49480**	**0.00231**	**0.01282**	**0.00115**	**0.00641**
Equation (13)	58360	0.05068	0.03627	0.02534	0.01814

Conclusions

We propose a CNN-LSTM fusion deep neural network in this paper to perform end-to-end inversion of RMS velocity and interval velocity. This method overcomes the feature extraction limitations of the CNN and LSTM single neural network models and achieves better inversion accuracy than single neural network models. The CNN-LSTM fusion deep neural network's predictions can be used as the starting point for high-precision velocity model building methods like FWI. In the meantime, the proposed method significantly improves the efficiency of seismic data processing. Furthermore, the problem that deep neural networks are difficult to be trained successfully due to unstable gradient and easy to fall into local minimum is solved by jointly adopting Kaiming normal initialization with zero negative slopes of rectified units and MSE loss function introducing a freezing factor. This training approach can optimize specific neurons based on the network's learning task, allowing the training and fine-tuning processes to be synchronized. The proposed method not only produces notable predictions for the samples in the testing dataset, but it also produces promising inversion results on the complex structures, Marmousi model, and field data, demonstrating the method's effectiveness and strong generalization capability.

As the performance of neural networks is affected by diverse factors, it will be the next work to introduce more parameter tuning strategies into the proposed method to improve the accuracy of inversion.

Acknowledgments

The authors are grateful to the reviewers for their

valuable comments.

References

Alzahrani, H., and Shragge, J., 2021, Neural network seismic velocity model building: A frequency-stepping approach: First International Meeting for Applied Geoscience & Energy, Society of Exploration Geophysicists, 3370–3374.

Biswas, R., Arnulf, A. F., Sen, M. K., et al, 2020, Two-step velocity inversion using trans-dimensional tomography and elastic FWI: 90th Annual International Meeting, SEG, Expanded Abstracts, 3628–3633.

Bouvrie, J., 2006, Notes on convolution neural networks. [Online]. Available: http://cogprints.org/5869/1/cnn_tutorial.pdf

Chai, X., Tang, G., Peng, R., et al., 2018, The linearized bregman method for frugal full-waveform inversion with compressive sensing and sparsity-promoting: Pure and Applied Geophysics, **175**(3), 1085–1101.

Dix, C. H., 1955, Seismic velocities from surface measurements: Geophysics, **20**(1), 68–86.

Fabien-Ouellet, G., and Sarkar, R., 2020, Seismic velocity estimation: A deep recurrent neural-network approach: Geophysics, **85**(1), U21–U29.

Girshick, R., 2015, Fast R-CNN: 2015 IEEE International Conference on Computer Vision, 1440–1448.

Glorot, X., and Bengio, Y., 2010, Understanding the difficulty of training deep feedforward neural networks: Proceedings of the 13th international conference on artificial intelligence and statistics, 249–256.

Graves, A., Eck, D., Beringer, N., et al., 2004, Biologically plausible speech recognition with LSTM neural nets: International Workshop on Biologically Inspired Approaches to Advanced Information Technology, Springer, Berlin, Heidelberg, 127–136.

Graves, A., Mohamed, A. R., and Hinton, G., 2013, Speech recognition with deep recurrent neural networks: 2013 IEEE International Conference on Acoustics, Speech and Signal Processing, 6645–6649.

Guitton, A., 2012, Blocky regularization schemes for full-waveform inversion: Geophysical Prospecting, **60**(5), 870–884.

Gulordava, K., Bojanowski, P., Grave, E., et al., 2018, Colorless green recurrent networks dream hierarchically: arXiv: 1803.11138. [Online]. Available: https://arxiv.org/abs/1803.11138

Guo, X., Shi, Y., Wang, W., et al., 2019, A robust source-independent misfit function for time domain waveform inversion based on normalized convolved wavefield: Journal of Applied Geophysics, **166**, 129–146.

Han, J., and Moraga, C., 1995, The influence of the sigmoid function parameters on the speed of backpropagation learning: International Workshop on Artificial Neural, Networks (pp. 195-201). Springer, Berlin, Heidelberg.

He, K., Zhang, X., Ren, S., et al., 2015, Delving deep into rectifiers: Surpassing human-level performance on imagenet classification: 2015 IEEE International Conference on Computer Vision, 1026–1034.

Hochreiter, S., and Schmidhuber, J., 1997, Long short-term memory: Neural Computation, **9**(8), 1735–1780.

Hole, J. A., 1992, Nonlinear high-resolution three-dimensional seismic travel time tomography: Journal of Geophysical Research: Solid Earth, **97**(B5), 6553–6562.

Huang, P., 2020, The research of damage identification method for bridge based on CNN-LSTM architecture neural network: MS Thesis, Qinghai University, Xining.

Ioffe, S., and Szegedy, C., 2015, Batch normalization: Accelerating deep network training by reducing internal covariate shift: Proceedings of the 32nd International Conference on Machine Learning, Lille, France, **37**, 448–456.

Kazei, V., Ovcharenko, O., Plotnitskii, P., et al., 2021, Mapping full seismic waveforms to vertical velocity profiles by deep learning: Geophysics, **86**(5), 1–50.

Kingma, D. P., and Ba, J., 2014, Adam: A method for stochastic optimization: arXiv: 1412.6980. [Online]. Available: https://arxiv.org/abs/1412.6980

LeCun, Y., Bengio, Y., and Hinton, G., 2015, Deep learning: Nature, **521**(7553), 436–444.

LeCun, Y., Boser, B., Denker, J. S., et al., 1989, Backpropagation applied to handwritten zip code recognition: Neural Computation, **1**(4), 541–551.

Lian, S., Yuan, S., Wang, G., et al., 2018, Enhancing low-wavenumber components of full-waveform inversion using an improved wavefield decomposition method in the time-space domain: Journal of Applied Geophysics, **157**, 10–22.

Lin, Y., and Huang, L., 2014, Acoustic-and elastic-waveform inversion using a modified total-variation regularization scheme: Geophysical Journal

International, **200**(1), 489–502.
Liu, B., Yang, S., Ren, Y., et al., 2021, Deep-learning seismic full-waveform inversion for realistic structural models: Geophysics, **86**(1), R31–R44.
Liu, Z., and Bleistein, N., 1995, Migration velocity analysis: Theory and an iterative algorithm: Geophysics, **60**(1), 142–153.
Li, S., Liu, B., Ren, Y., et al., 2020, Deep-learning inversion of seismic data: IEEE Transactions on Geoscience and Remote Sensing, **58**(3), 2135–2149.
Li, X., Aravkin, A. Y., van Leeuwen, T., et al., 2012, Fast randomized full-waveform inversion with compressive sensing: Geophysics, **77**(3), A13–A17.
Maas, A. L., Hannun, A. Y., and Ng, A. Y., 2013, Rectifier nonlinearities improve neural network acoustic models: Proceedings of the 30th International Conference on Machine Learning, Atlanta, Georgia, USA, **30**(1), 3.
Mao, B., Han, L. G., Feng, Q., et al, 2019, Subsurface velocity inversion from deep learning-based data assimilation: Journal of Applied Geophysics, **167**, 172–179.
Meng, Z., and Scales, J. A., 1996, 2D tomography in multi-resolution analysis model space: 66th Annual International Meeting, SEG, Expanded Abstracts, 1126–1129.
Nair, V., and Hinton, G. E., 2010, Rectified linear units improve restricted Boltzmann machines: Proceedings of the 27th International Conference on Machine Learning, Haifa, Israel, 807–814.
Pratt, R. G., Shin, C., and Hick, G. J., 1998, Gauss-Newton and full Newton methods in frequency-space seismic waveform inversion: Geophysical Journal International, **133**(2), 341–362.
Pytorch Official Document. [Online]. Available: https://pytorch.org/docs/stable/nn.init.html#
Saxe, A. M., McClelland, J. L., and Ganguli, S., 2013, Exact solutions to the nonlinear dynamics of learning in deep linear neural networks: arXiv: 1312.6120. [Online]. Available: https://arxiv.org/abs/1312.6120
Sun, J., Innanen, K. A., and Huang, C., 2021, Physics-guided deep learning for seismic inversion with hybrid training and uncertainty analysis: Geophysics, **86**(3), R303–R317.
Sun, J., Niu, Z., Innanen, K. A., et al., 2020, A theory-guided deep-learning formulation and optimization of seismic waveform inversion: Geophysics, **85**(2), R87–R99.
Woodward, M. J., 1992, Wave-equation tomography: Geophysics, **57**(1), 15–26.

Cao Wei received a B.S. degree in information management and information system from Northeast Petroleum University, Daqing, China, in 2018. She is currently pursuing her Ph.D. degree at Northeast Petroleum University, Daqing, majoring in geological resources and geological engineering. Her research interest is seismic velocity inversion based on artificial intelligence.
E-mail: caowei202007@163.com

Guo Xue-Bao (Corresponding author) received his B.S. and M.S. degrees in exploration technique and engineering from Northeast Petroleum University, Daqing, China, in 2012 and 2015, respectively, and a Ph.D. degree in solid geophysics from the Institute of Geology and Geophysics, Chinese Academy of Sciences, Beijing, China, in 2018. His research interests include reverse time migration and full waveform inversion.
E-mail: guoxuebao1108@163.com