Deep Learning for Regularly Missing Data Reconstruction

Xintao Chai, Genyang Tang, Shangxu Wang, Ronghua Peng, Wei Chen, and Jingnan Li

***Abstract*— Inspired by image-to-image translation, we applied deep learning (DL) to regularly missing data reconstruction, aimed at translating incomplete data into their corresponding complete data. With this purpose in mind, we first construct a network architecture based on an end-to-end U-Net convolutional network, which is a generic DL solution for various tasks. We then meticulously prepare the training data with both synthetic and field seismic data. This article is implemented in Python based on Keras (a high-level DL library). We described the network architecture, the training data, and the training settings in detail. For training the network, we employed a mean-squared-error loss function and an Adam optimization algorithm. Next, we tested the trained network with several typical data sets, achieving good performances (even in the presence of big gaps) and validating the feasibility, effectiveness, and generalization capability of the assessed framework. The feature maps for a sample going through the well-trained network are uncovered. Compared with the f-x prediction interpolation method, DL performs better and is capable of avoiding several assumptions (e.g., linearity, sparsity, etc.) associated with conventional interpolation methods. We demonstrated the influences of the network depth, the kernel size of the convolution window, and the pooling function on the DL results. We applied the trained network to dense data reconstruction successfully. The proposed method can overcome noise to some extent. We finally discussed some practical aspects and extensions of the evaluated framework.**

***Index Terms*— Convolutional neural networks (CNNs), deep learning (DL), missing data reconstruction.**

I. Introduction

DEEP learning (DL) [1] is one of the ten breakthrough technologies of 2013, according to the MIT technology review website. For a thorough textbook of DL, please refer to [2]. DL has enjoyed an explosive growth over past years, it shows great promise in a wide variety of fields (e.g., cell segmentation [3], self-driving cars [4], medical imaging [5], image recognition, and classification [6], [7]), and geoscience should also be included. DL is powerful for mining features or relationships from data. One of the most popular DL technologies is the convolutional neural network (CNN), first proposed by [8]. CNN is at the core of most state-of-the-art DL solutions for numerous tasks [9]. DL is ever more competing with traditional methodologies. In geophysics, or more specifically, in exploration seismology, DL or machine learning (ML) in general, has been intensively used in fault detection [10]–[14], fracture detection [15], structural interpretation [16], facies analysis [17], [18], inversion [19], [20], tomography [21], data interpolation [22]–[26], to name a few. Many problems in image processing and computer vision involve "translating" an input image into a corresponding output image. Taking inspiration from [27] and [28], we apply DL to regularly missing data reconstruction, aiming of translating an input regularly missing data set into a corresponding complete data set, the so-called super-resolution reconstruction technique.

In seismic data acquisition, physical or monetary constraints lead data to be undersampled [29]. However, many techniques rely on regularly sampled, unaliased, high-resolution input data (e.g., 2-D/3-D surface-related multiple elimination, velocity analysis, prestack migration/inversion, amplitude-variation-with-offset (AVO) analysis, amplitude-versus-azimuth analysis, time-lapse seismics, and reservoir characterization). The performance of multichannel data processing depends heavily on the sampling intervals. Too large an interval leads to aliasing, which adversely results in poor resolution. Therefore, missing data must be reconstructed from the available data [30].

The data missing problem can be classified into two categories: regular missing and irregular missing [31]. Regular missing means the data are equidistantly or periodically missing at a constant rate in uniform grids [32]. Irregular missing means the data are randomly missing on uniform grids [33]. We focus on tackling the data of regularly missing problem. We address the data that are regularly missing in both/either the space dimension and/or the time dimension (in the presence of big gaps), which is different from the work of [26].

So far, numerous methods have been proposed for seismic data reconstruction, based on different strategies. A complete discussion of this topic is out of the scope of this work.

Manuscript received July 21, 2019; revised November 16, 2019; accepted December 31, 2019. Date of publication January 21, 2020; date of current version May 21, 2020. This work was supported by the National Natural Science Foundation of China Program under Grant 41704129, Grant 41774143, Grant 41704133, and Grant 41804140. *(Corresponding author: Shangxu Wang.)*

Xintao Chai and Ronghua Peng are with the Center for Wave Propagation and Imaging (CWπ), DeepResearch Group, Institute of Geophysics and Geomatics, China University of Geosciences (Wuhan), Wuhan 430074, China (e-mail: xtchai@126.com; pengrh@cug.edu.cn).

Genyang Tang and Shangxu Wang are with the State Key Laboratory of Petroleum Resources and Prospecting, China University of Petroleum (Beijing), Beijing 102249, China, and also with the Key Laboratory of Geophysical Exploration, China National Petroleum Corporation (CNPC), Beijing 102249, China (e-mail: tanggenyang@163.com; wangsx@cup.edu.cn).

Wei Chen is with the Key Laboratory of Exploration Technology for Oil and Gas Resources of Ministry of Education, Yangtze University, Wuhan 434023, China (e-mail: chenwei2014@yangtzeu.edu.cn).

Jingnan Li is with the Sinopec Geophysical Research Institute, Nanjing 211103, China (e-mail: li.jnan@163.com).

Color versions of one or more of the figures in this article are available online at http://ieeexplore.ieee.org.

Digital Object Identifier 10.1109/TGRS.2020.2963928

We refer the reader to the Introduction section of [26] for a comprehensive overview on the topic. We only review some previous works closely related to the work presented here.

The prediction-error filter method [34] is commonly used for interpolating regularly sampled data, which assumes seismic data to be a superposition of plane waves and can be performed in the frequency-space (f-x) domain. The predictability of linear events in the f-x domain allows the missing data to be expressed as the output of a linear system, the input of which consists of the available data. The reconstruction operator is obtained by solving a set of linear equations whose coefficients depend on the spectrum of the spatial prediction filter defined by the available data. This makes the f-x prediction method a strong competition, which is used for comparison in our experiment.

The f-x prediction-based algorithms make use of the predictability of linear events, requiring that a small window should be utilized to assure the linearity for nonlinear events. Obtaining optimal window is not a trivial task [26].

Interpolation methods based on sparse transform (e.g., Fourier and curvelet transforms) had played an important role. Although sparsity-based methods can solve interpolation problems to some degree, their accuracy is sensitive to the sparsity-promoting parameters and optimization algorithms. For regular sampling, the Fourier spectrum of the sampled data exerts rhythmic replicas, which can only be removed when no overlap exists between the spectrum of the true data and the aliased components. Therefore, the success of the band-limited Fourier-based methods depends on the decimation factor and the band-limitation property of the true data [32]. In the frequency-wavenumber (f-k) domain, a mask function is generally designed to restrict the f-k energy distribution, influencing the interpolation precision.

Interpolation methods based on rank-reduction assume that seismic data organized in some way are of low-rank, and the missing data increases the rank. In general, the rank is estimated as the number of events in the analyzed local window. It is difficult to determine the optimal rank, leading to unsatisfactory results. Besides, the computationally expensive rank-reduction methods are mainly designed for irregularly sampling case.

In short, conventional interpolation methods involve different limitations and different parameter selections for different data applications to obtain the optimal result, requiring a mass of human-computer interaction far away from intelligence.

Of the numerous methods for interpolating aliased seismic data, few take advantage of the recent developments in DL or ML in general. Jia and Ma [22] proposed a method for reconstructing seismic data from undersampled traces based on a classic ML method of support vector regression, a state-of-the-art ML tool. For methods like [22] and [23], in the training stage, calculation of the initial preinterpolation data using a bicubic method or other methods is required, which affects the performance of ML methods. However, our method does not involve the preinterpolation using the bicubic method or other methods.

Wang *et al.* [26] proposed a DL-based approach for seismic antialiasing interpolation, extracting features of the training data in a nonlinear way. Eight-layers residual networks (ResNets) with a better back-propagation property for deep layers was designed for interpolation, based on CNN. Same as the works of [22] and [23], [26] also use the bicubic interpolation algorithm to generate the input of the network (i.e., preinterpolation).

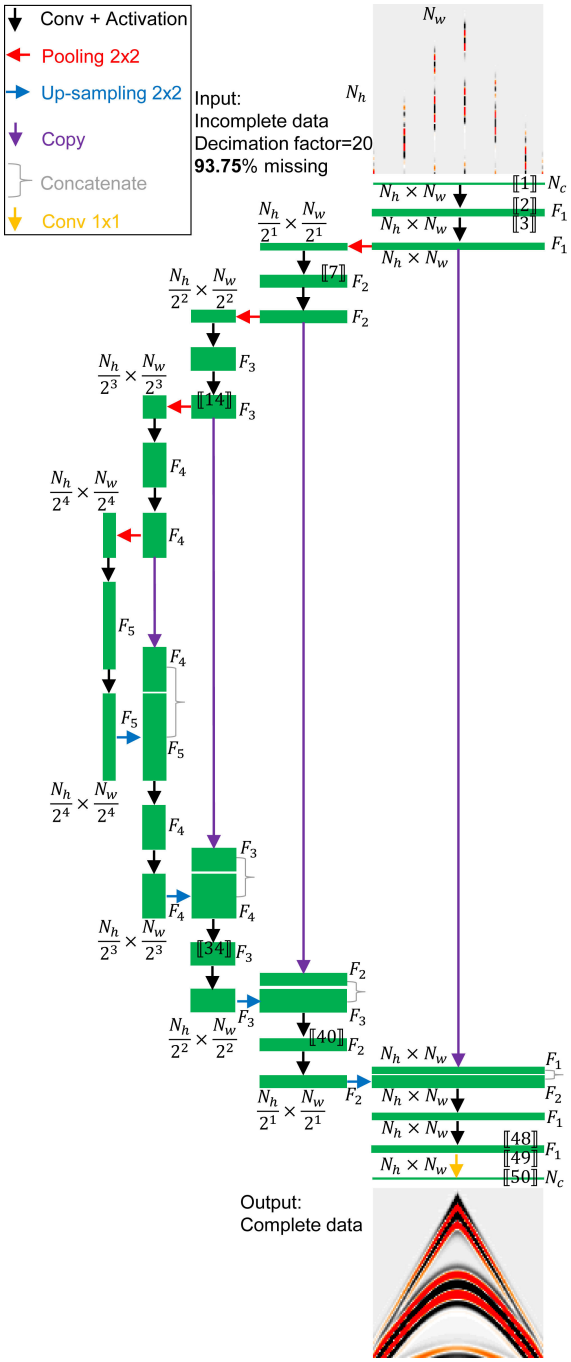

Fig. 1. Schematic of the network architecture (example for $\frac{N_h}{2^4} \times \frac{N_w}{2^4}$ pixels in the lowest resolution). N_h, N_w, and N_c are the height, width, and the number of channels of the input data, respectively. Each green box denotes multiple feature maps, and the number of feature maps (i.e., $F_{i,i\in[1,\ 5]}$) is denoted on the right of the box. The height-width-size of a feature map is provided around the box. The boxes with the same height have the same number of feature maps. The boxes with the same width indicate the same height-width-size of feature maps. The arrows and the right curly brace denote different operations. Numbers in [[ ]] are labeled in consistent with those in Fig. 2.

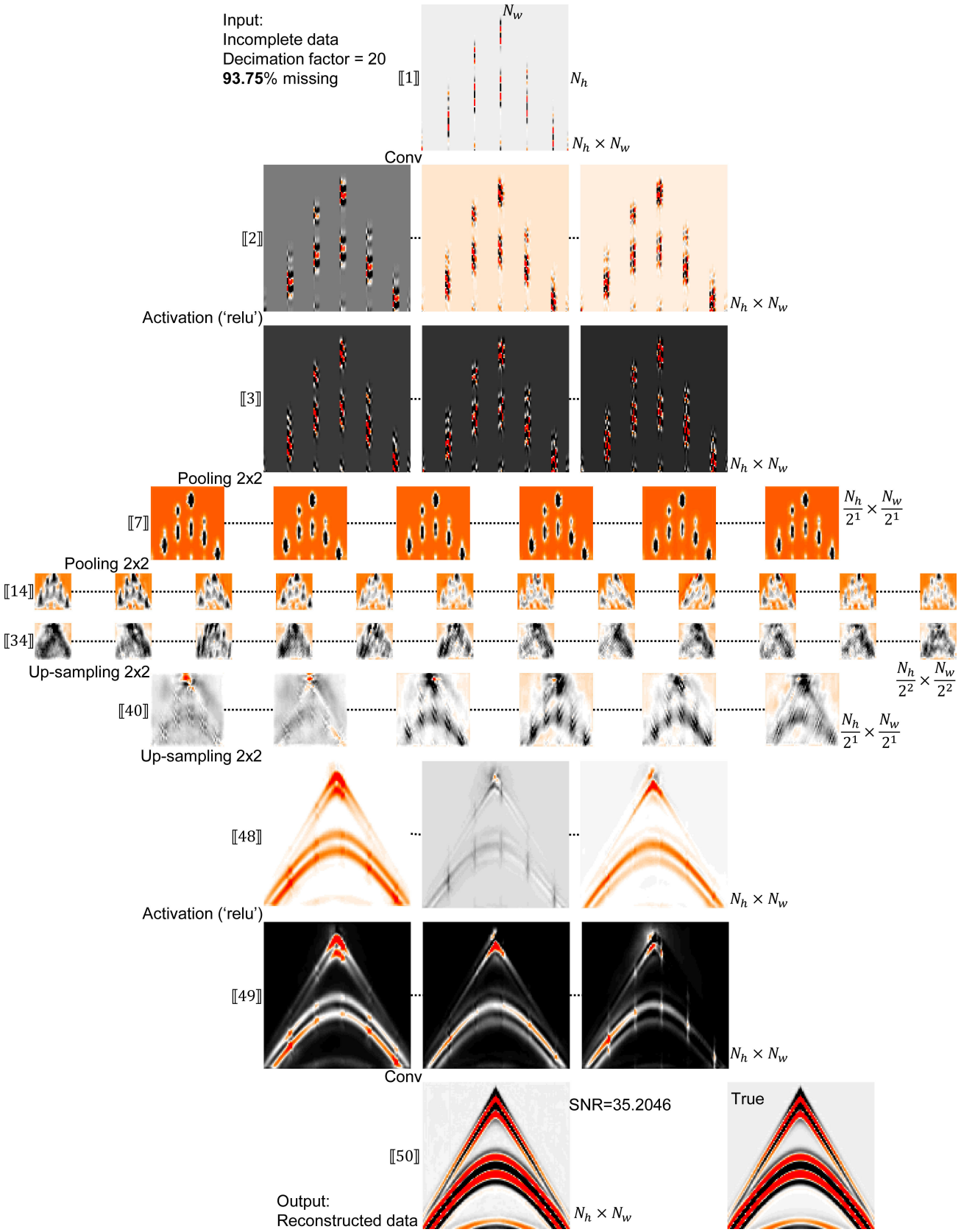

Fig. 2. Feature maps for a regularly missing sample going through a well-trained network.

Our initial motivation for applying DL to regularly missing data reconstruction is from the works of [27] and [28], where DL is investigated as a general-purpose solution to image-to-image translation problems. Many problems in geophysics can also be posed as translating an input into a corresponding output (e.g., denoising: noisy data to noise-free

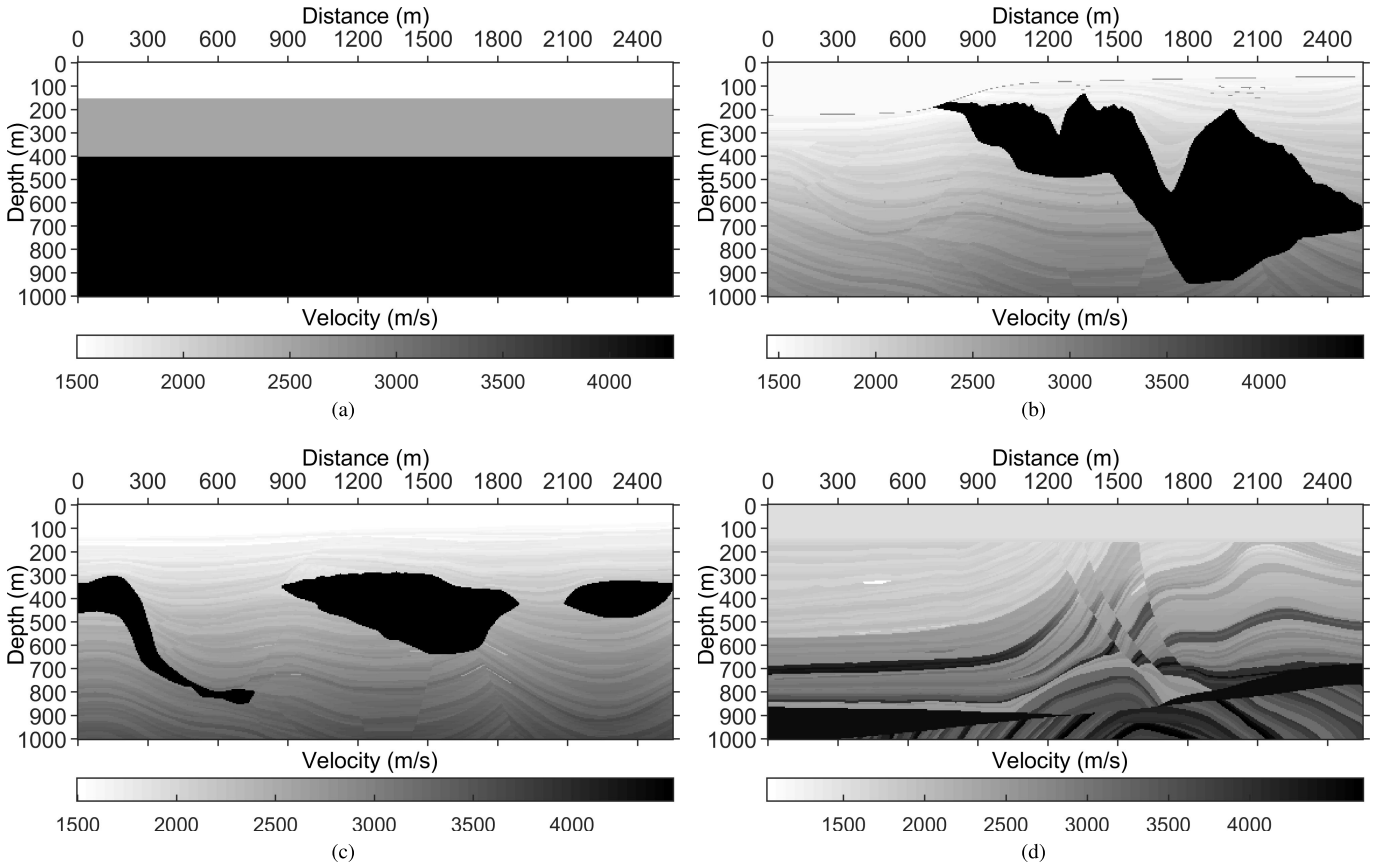

Fig. 3. Numerical geologic models used to generate the synthetic data. Each velocity model has its corresponding density model, however, only the velocity model is shown here. (a) Simple three-layer model. (b) Adapted Sigsbee2B model. (c) Adapted Pluto 1.5 model. (d) Downsampled Marmousi2 model [40]. Both the horizontal spacing dx and the depth spacing dz are 5 m.

data, data reconstruction: incomplete data to complete data, etc.). We define intelligent data-to-data translation as the task of translating one possible representation of data into another, for the given sufficient training data. DL allows trainable networks composed of multiple layers to learn representations of data with multiple levels of abstraction [1]. DL is representation-learning method with multiple levels of representation, obtained by composing non-linear modules that each transforms the representation at one level into a representation at a higher, slightly more abstract level [2]. With the composition of enough such layers, very complex functions can be learned. As a motivation, DL can theoretically avoid linearity, sparsity, and low-rank assumptions underlying the traditional interpolation methods [26].

With the DL theory in [2], we apply DL to regularly missing data reconstruction. In Section II, we first briefly transcribe some basic DL theory behind the evaluated approach; then we explain the constructed network architecture in detail, based on the U-Net CNNs [3], with incomplete data as the network input and the corresponding complete data as the output; next, we provide detailed training analysis including training data preparation, loss function definition, parameter setup. In Section III, we test the performances of the trained networks using several typical data sets (i.e., a synthetic training data set, a synthetic test data set, a physical modeling data set, the Mobil AVO Viking Graben line 12 data set, the F3 data set, a data set from the GeoFrame software, and a data set from the Hampson-Russell software); we further investigate the impact of the network depth, the kernel size of the convolution window, and the pooling layer on the DL results; moreover, the trained network is used to provide denser data with halved/quartered sampling intervals. After discussing some practical aspects and extensions of the framework, we summarize some concluding remarks.

II. Theory

For the sake of brevity, we refer the reader to [2] for a detailed explanation of the DL terminologies. For the nomenclature issue, e.g., "convolution," CNNs do not apply convolutions, in fact, they apply cross correlations. This indeed gives rise to distortions of traditional signal processing terms. In order to be consistent with other DL publications, we use the term "convolution" with no intention of ignoring their difference. CNNs can be viewed as bridges mapping the input space into some output space. For modeling the desired mapping, the network has to go through a training/learning process composed of iteratively changing the internal parameters (e.g., weights and biases of convolution kernels), through the presentation of many input patterns and their corresponding output patterns [19]. The learning process is accomplished

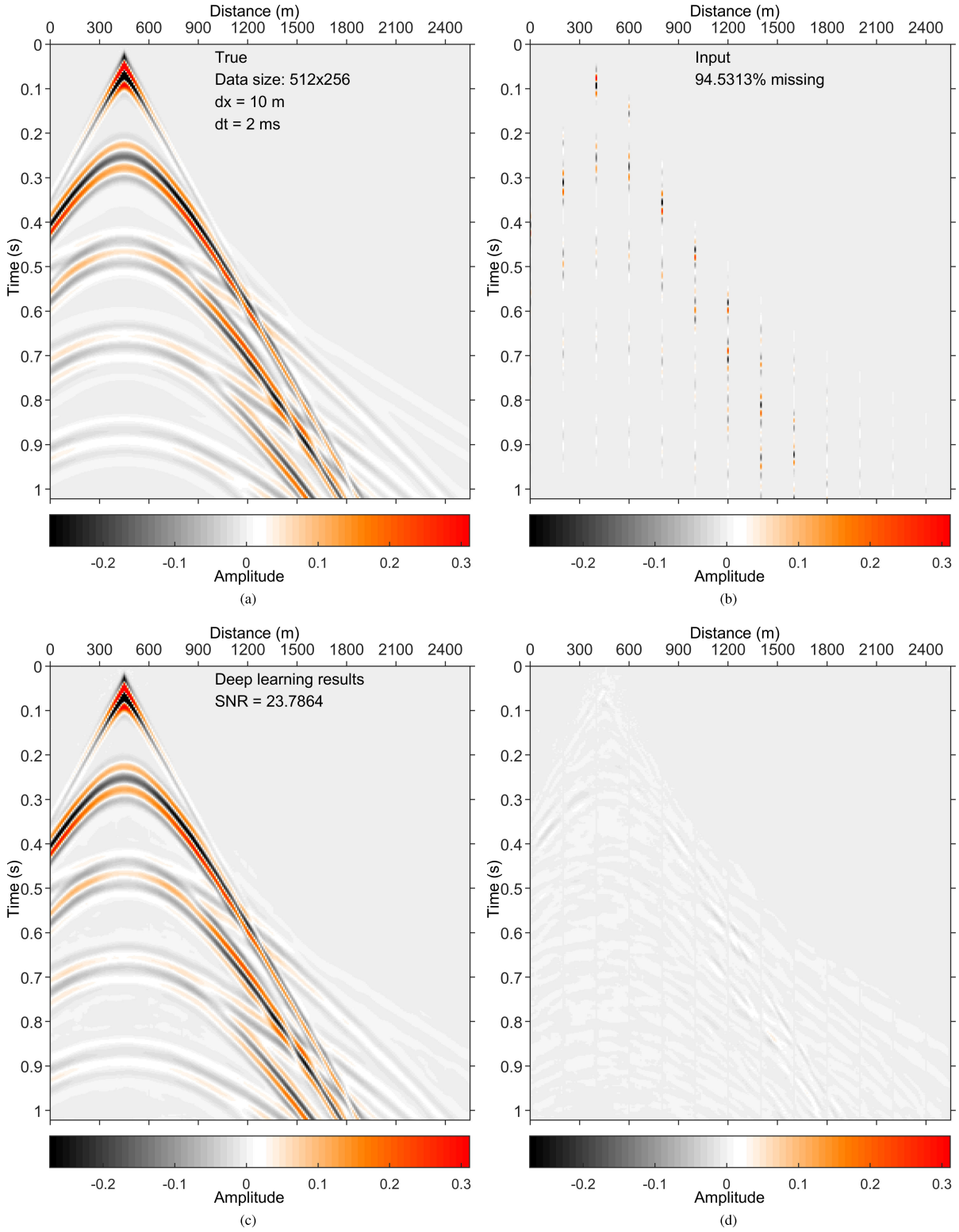

Fig. 4. Results on a synthetic training data set. (a) True data (a common-shot-point, CSP, gather). (b) Regularly sampled data with a decimation of factor 20 along the space dimension (approximately 95% missing). (c) DL reconstruction result. (d) Difference between (a) and (c).

if the error between the modeled output and the desired output (quantified by a loss/objective function) achieves an acceptable level for all samples in the training data [2]. Except the loss function, an optimizer (e.g., Adam [35], derived from adaptive moment estimation) is another argument required for training a network.

A. Network Architecture

Fig. 1 displays a schematic of the assessed network architecture, belonging to a specific family of neural network architectures known as U-Net (a generic DL solution for various tasks [3]). Specific to our mission, the network input data include one spatial dimension and one temporal dimension.

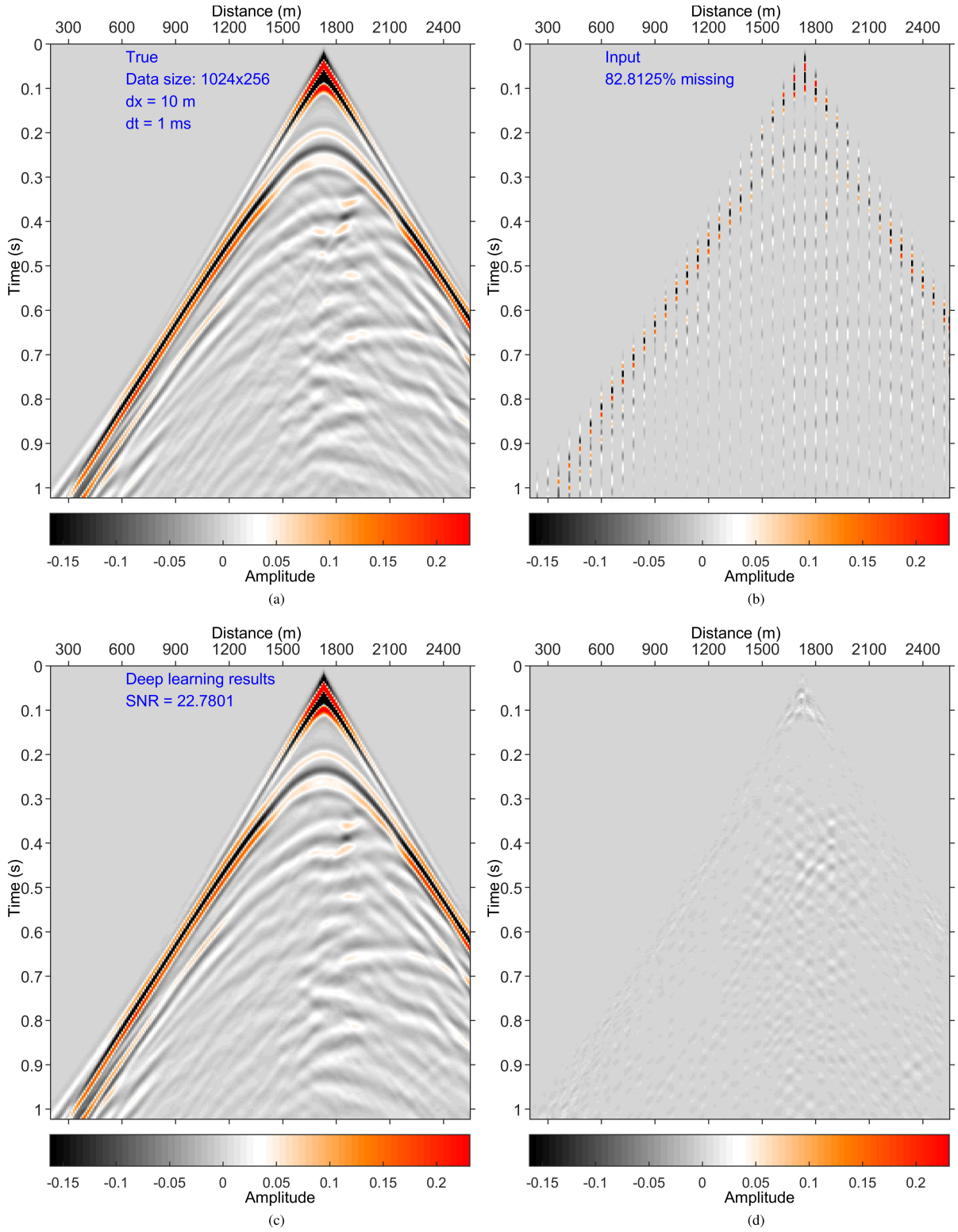

Fig. 5. Result on a test data set generated with the Marmousi2 model. (a) True data. (b) Regularly sampled data with a decimation of factor 6 along the space dimension (approximately 83% missing). (c) DL result. (d) Difference between (a) and (c).

In seismic terminology, N_h and N_w represent the sampling points of the input data along the time axis and the space axis, respectively. For seismic data, the number of channels N_c equals 1. We address a regular sampling problem, and the sampling ratio is $\frac{1}{K}$, which denotes sampling one point out of K points (i.e., the decimation factor is K). The network input data can be missing in both/either space dimension and/or time dimension. Interpolation along the time dimension is trivial. Aliasing generally does not exist along time; therefore, it can be done with very fast, precise, sinc interpolation, in negligible

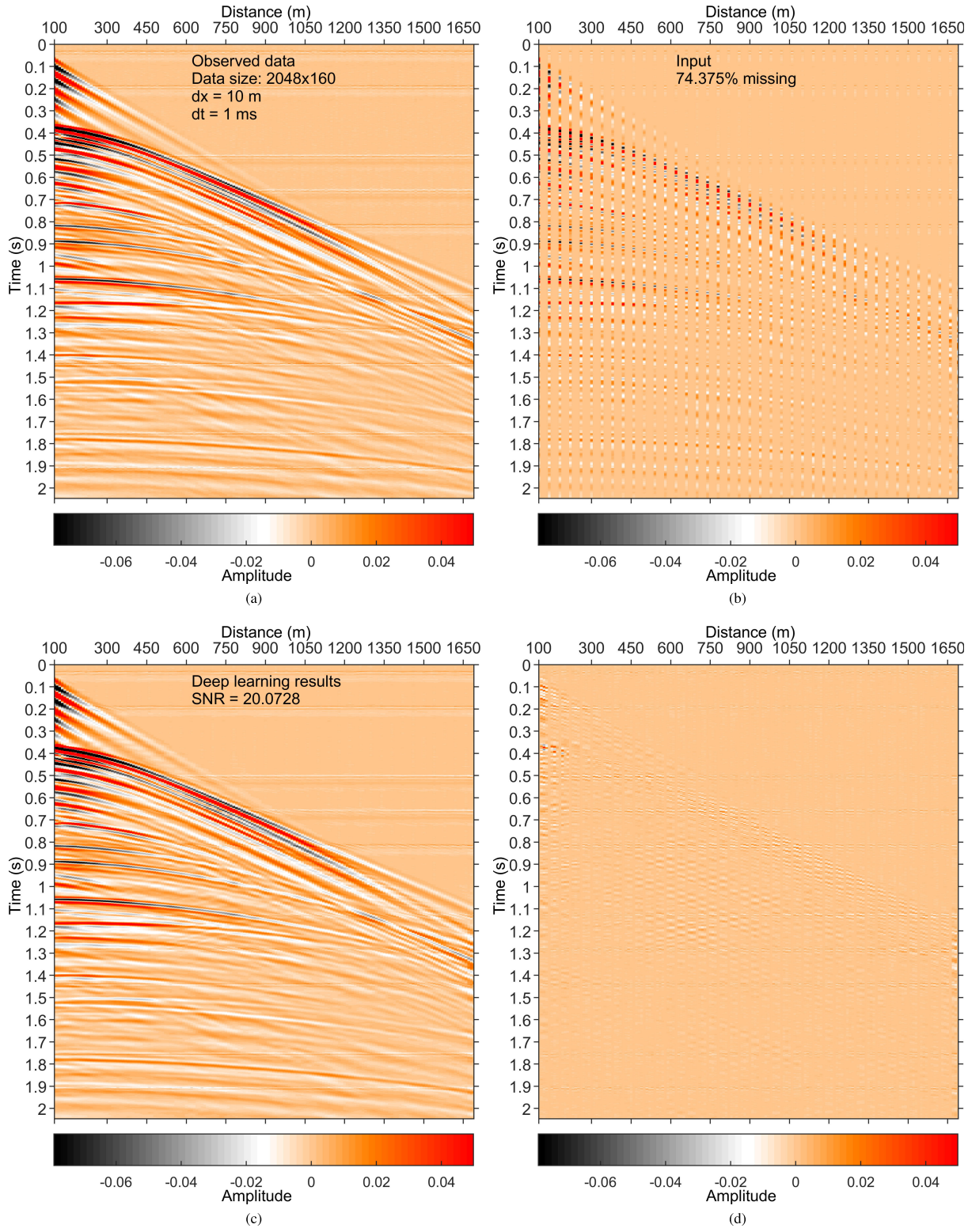

Fig. 6. Results on a physical modeling data set used in [41]. (a) Reference data. (b) Regularly sampled data with a decimation of factor 4 along the space dimension (approximately 74% missing). (c) DL result. (d) Difference between (a) and (c).

time, working trace by trace, on a single workstation without graphics processing units (GPUs). Therefore, we primarily concentrate on de-aliasing along the space dimension. The network output is the corresponding complete data, and no preinterpolation is required. K ranges from 2 to 20 during the training process.

The network architecture consists of a contracting path (upper side, interpreted as the encoder) and a more or less

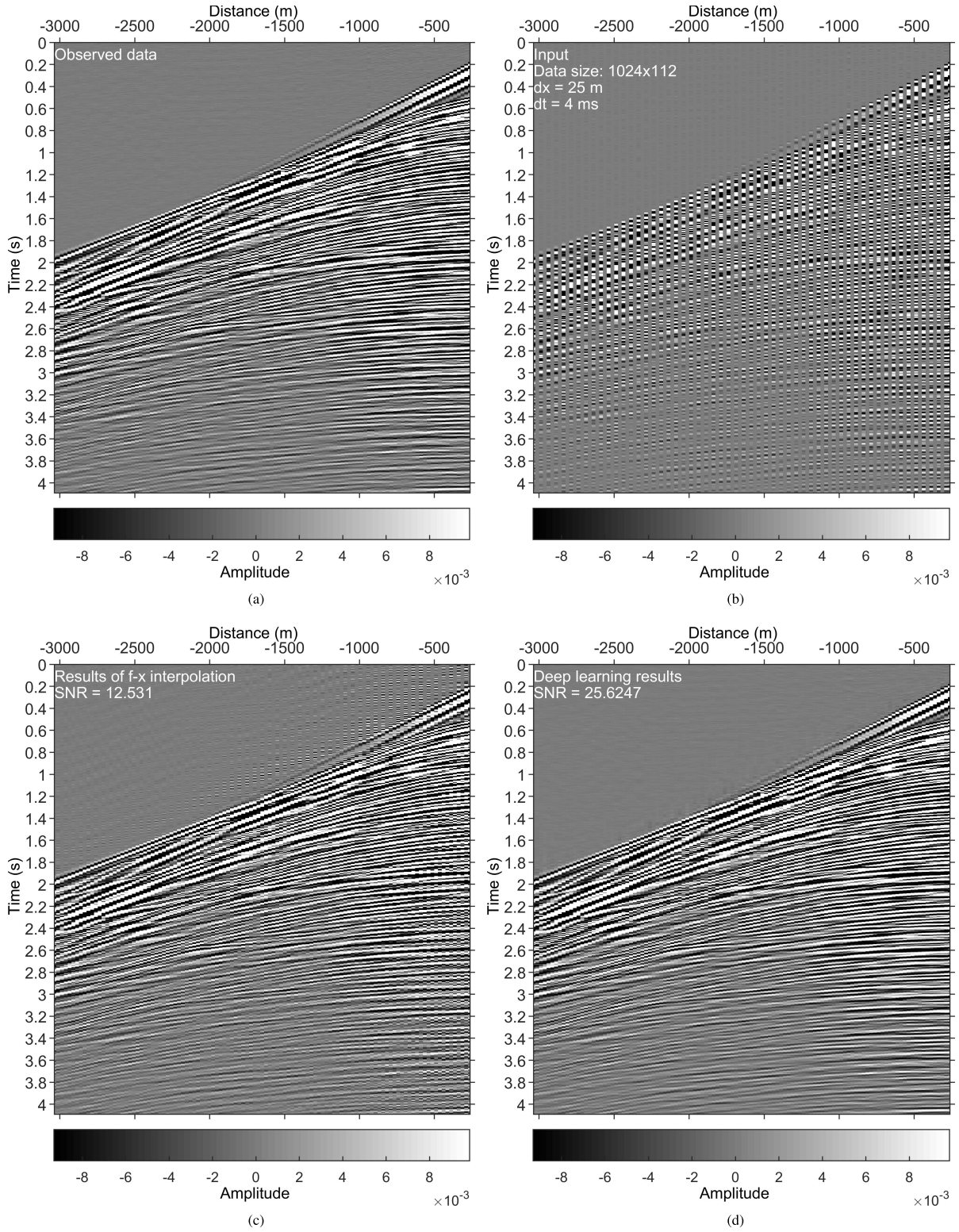

Fig. 7. Results on the Mobil Graben line 12 data set. (a) Reference data. (b) Regularly sampled data with a decimation of factor 2 along the space dimension. (c) Interpolated data using the f-x prediction method [34]. (d) DL reconstruction result.

symmetric expanding path (lower side, interpreted as the decoder). The contracting path follows the typical architecture of a convolutional network, consisting of the repeated application of two padded convolutions (padding to avoid the loss of border pixels in every convolution), each followed by an activation operation (black arrow) and a pooling operation with stride 2 (red arrow) for down-sampling. Each step in the expansive path is composed of an up-sampling of the feature maps (blue arrow) followed by a concatenation (right curly brace) with the correspondingly copied feature maps from the

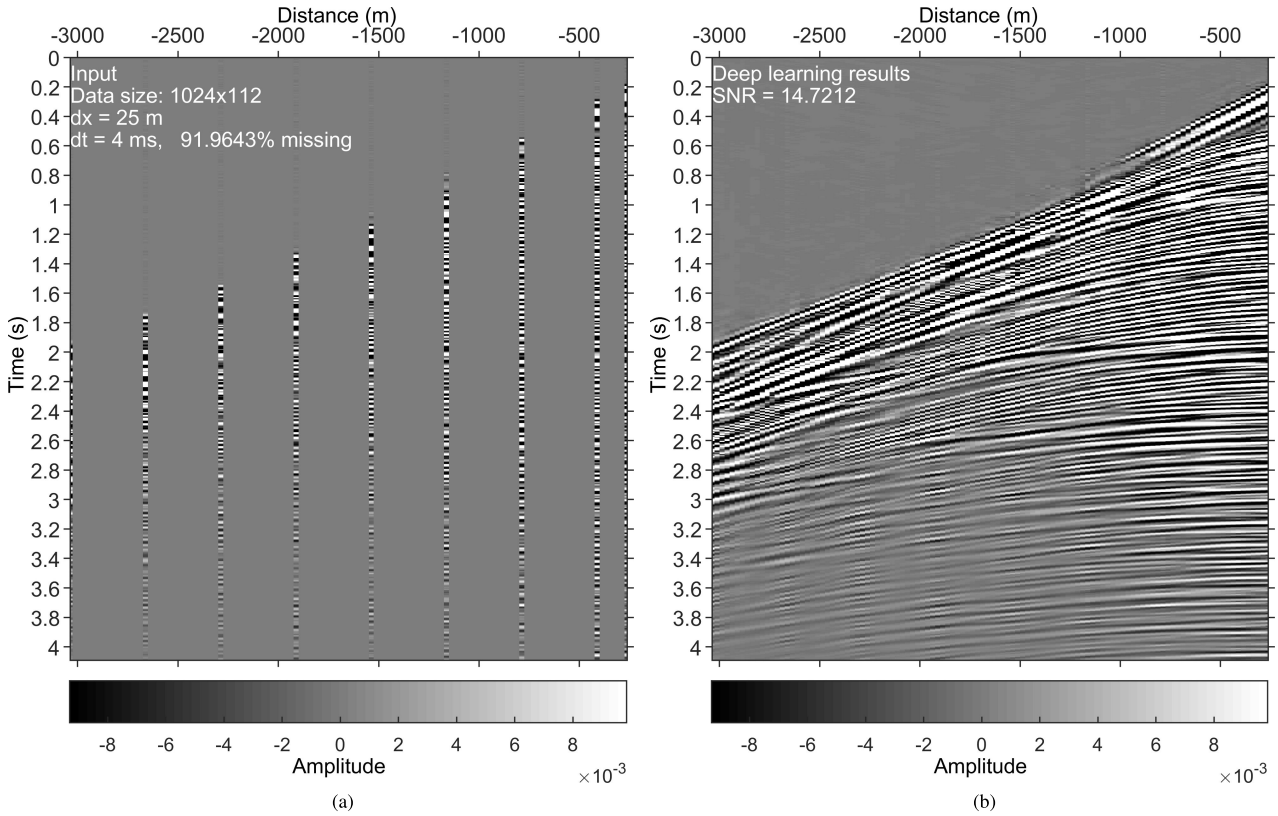

Fig. 8. Results on the Mobil Graben line 12 data set in the presence of big gaps. (a) Regularly sampled data with a decimation of factor 15 along the space dimension (approximately 92% missing). (b) DL reconstruction result.

TABLE I
SUMMARY OF FOUR DISCUSSED NETWORKS, WHERE THE SIZE OF THE CONVOLUTION KERNEL IS SPECIFIED TO BE 3×3

Network	# of downsampling layers	# of convolution layers	# of all layers	# of trainable parameters	Network size (MB)
Network 1	1	7	17	628,033	7.24
Network 2	2	11	28	1,881,985	21.60
Network 3	3	15	39	7,781,761	89.10
Network 4	4	19	50	31,377,793	359.00

contracting path (purple arrow), and two convolutions, each followed by an activation. The features from the contracting path are combined with the up-sampled output. Successive convolution layers can then learn to assemble a more precise output based on this. At the last layer a 1×1 convolution (green arrow) is utilized to map multiple feature maps to the desired output.

So far, the most popular nonlinear activation function is the rectified linear unit (ReLU), which returns element-wise $\max(\mathbf{x}, 0)$ with $\mathbf{x}$ being a tensor. The ReLU activation is adopted in the evaluated network architecture. There are several kinds of pooling operations (e.g., max pooling, average pooling, etc.). It is worth noting that the number of output filters in the convolution (i.e., $F_{i,i\in[1,\ 5]}$) increases (e.g., from 64 to 128, 256, 512, 1024) as we go deep in the network. At each down-sampling step we generally double the number of feature maps.

For better understanding, we provide a specified summary of the network architecture in the case of $N_c = 1$, $F_1 = 64$, $F_2 = 128$, $F_3 = 256$, $F_4 = 512$, $F_5 = 1024$, $K_h = K_w = 5$, where K_h and K_w represent the height and width of the 2D convolution kernel, also known as the kernel size. The total trainable parameters of this network are 87 149 953. There are 50 layers in total (including 19 convolutional layers). The trainable parameters concentrate upon convolutional layers, namely, there are no trainable parameters in the input, ReLU activation, pooling, up-sampling, and concatenate layers. The trainable parameters in a convolutional layer are calculated by

$$(K_h \times K_w \times F_{i-1} + 1) \times F_i \tag{1}$$

where F_{i-1} and F_i denote the number of feature maps in the previous and current layers, respectively. $+1$ means a bias is added. Fig. 2 displays the feature maps for a sample with big gaps going through a well-trained network. Once trained, the network is able to fill the big gaps in corrupted data by going through contracting and expanding stages.

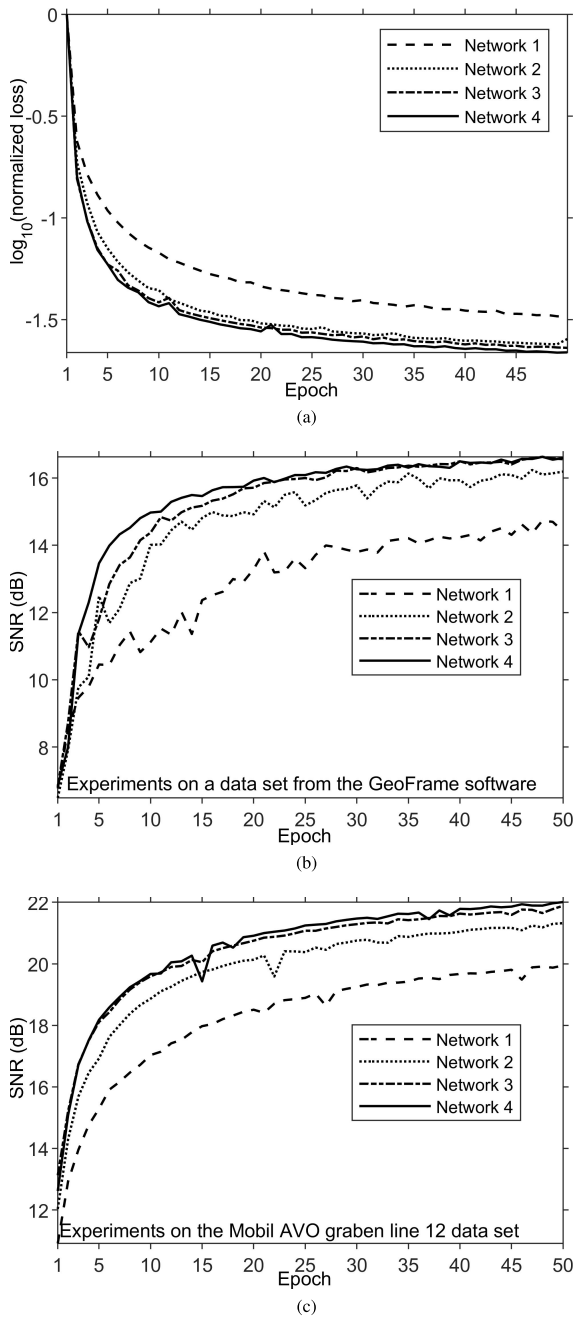

Fig. 9. Impact of the network depth. (a) During the training process, variation of $\log_{10}$(normalized loss) with epoch. (b) Variation of SNR with epoch while testing the discussed four networks on a data set from the GeoFrame software. (c) Variation of SNR with epoch for testing the discussed four networks on the Mobil AVO Viking Graben line 12 data set.

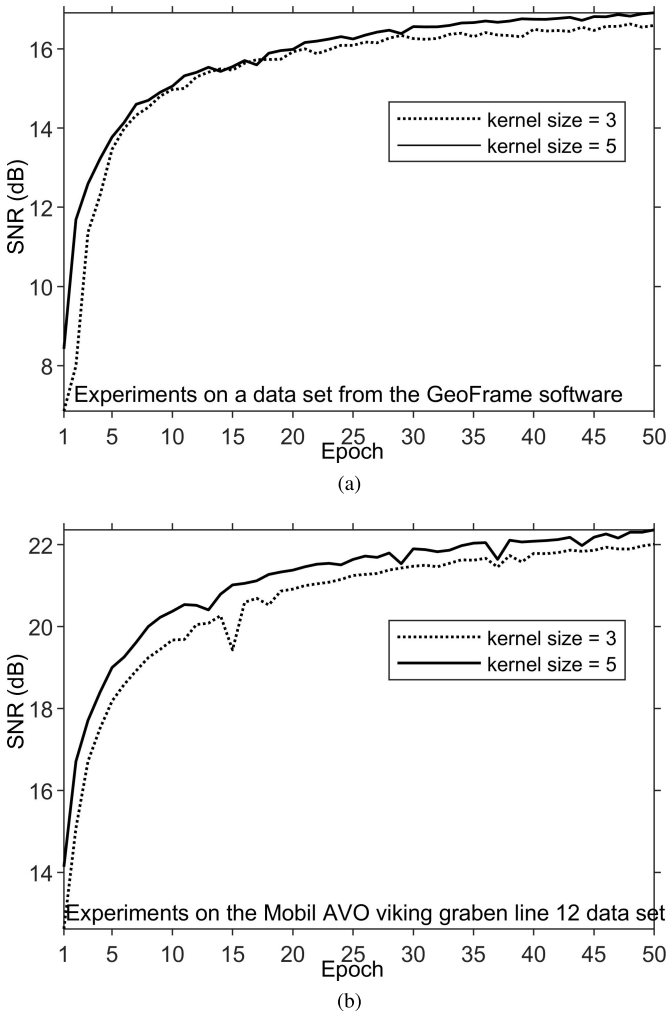

Fig. 10. Impact of the size of the convolution kernel. (a) Variation of SNR with epoch while testing the networks with different kernel size on a data set from the GeoFrame software. (b) Variation of SNR with epoch for testing the networks with different kernel size on the Mobil Graben line 12 data set.

B. Training Preparation

The training data are crucial for DL. To let the network learn the features of seismic data, we prepare the training data using not only the synthetic data simulated with the indoor computer, but also field data recorded from the outdoor industrial survey.

The synthetic training data are obtained using a finite-difference method [36], [37] based on several well-designed velocity-density earth models [see Fig. 3(a)–(c)]. The model in Fig. 3(d) is used to generate test data, which are unused in the training process. We can generate a large number of earth models and calculate their seismograms. The sources and receivers are equally distributed from 0 to 2550 m with 10-m spacing. The source is shifted from the location of the first shot to the last one. The source and receiver depths are varying for different earth model and different experiment. The initial synthetic data set includes 2048 shots (8 simulations, 256 shots per simulation), and 256 receivers per shot. There are 2048 samples along the time axis with a time interval dt of 0.5 ms. With our experience, dt = 0.5 ms is barely used in industry. Therefore, we revised the initial data set of size $2048 \times 256 \times 2048$ (in order of time axis, receiver axis, and shot axis) to three data sets: $1024 \times 256 \times 2048$ (dt = 1 ms), $512 \times 256 \times 2048$ (dt = 2 ms), $256 \times 256 \times 2048$ (dt = 4 ms), constituting the synthetic training data. Fig. 7(a) shows a sample.

We exploited the Mobil AVO Viking Graben line 12 data set [38] for generating the field training data. This data set consists of 1001 shot gathers. Each shot gather is of size 1024×120: 1024 rows represent time domain, sampled every

4 ms; 120 columns are in spatial domain with 25 m of sampling. Among 1001 shot gathers, we randomly selected 200 shot gathers as the field training data set. Fig. 7(a) shows a sample gather.

Before being fed into the network, each shot gather is normalized by dividing the maximum value of the absolute value of the corresponding shot gathers. Therefore, the amplitude finally ranges $[-1, 1]$. In order to ensure a sufficiently large number of data samples under learning, we work in a patch-wise fashion. There is a trade off between the patch-size (determining the receptive field) and the network depth. Larger patch-size requires more pooling/up-sampling layers, while small patches allow the network to see only little context. It is worthy to mention that we should select the input patch-size such that all 2×2 downsampling operations can be applied to a layer with an even height- and width-size. The patch-size and the batch-size are limited by the GPU memory. The batch size denotes the number of training examples in one forward/backward pass. The higher the batch size, the more GPU memory space we will need. To minimize the overhead and make maximum use of the GPU memory, we favor large input patch-size over a large batch-size.

After trial and error, the input patch size is chosen as 112×112, allowing four times downsampling operation for the field training data set with 120 receivers. Therefore, the training data are divided into small patches with the size of 112×112. To overlap adjacent patches, the patch-stride is of 80 pixels for the synthetic training data sets, 20 pixels for the field training data set. The training data in terms of patches is much larger than the number of training shot gathers. Patches with lower mean absolute value (e.g., less than 0.001) indicate that there are few events located within the patch or these pixel values are almost zeros. The patches below a threshold value are kicked out from the training data. We also pay special attention to the shallow first-arrival areas because their samples are fewer in comparison to other areas. Therefore, we increase the proportion of the samples coming from the shallow first-arrival areas, to some degree. The training process finally involves more than 1 316 608 patches. The batch size is set as 128. The steps-per-epoch is set as 10 286, denoting the total number of steps (batches of samples) before declaring one epoch finished and starting the next epoch. It is typically equal to the number of samples of the training data set (1 316 608) divided by the batch size (128).

The codes of this article are written in Python based on Keras [39]. We clarify that we wrote our own CNN interpolation code, rather than using/modifying the one provided with U-Net. The input incomplete data samples and their corresponding complete data samples are used to train the network. The training process can be regarded as an optimization problem, consisting of finding a set of the network weights and biases through the minimization of a distance metric defined between the modeled output and its desired output for all the samples fed into the network [19]. The distance metric is usually referred as the loss function, and its minimization is carried out using iterative optimization algorithms. We use a mean-squared-error loss function and an Adam optimization algorithm [35]. The learning rate is initialized at 0.0001, which is a critical parameter impacting the DL result. How to take optimal values of the learning rate is still an open issue.

The number of epochs should be specified to train the network. An epoch describes the number of times the algorithm sees the entire data set. Therefore, each time the algorithm has seen all the training samples, an epoch has completed. The number of iterations equals to the number of passes with each pass using "batch size" number of samples. An iteration describes the number of times a batch of data passed through the algorithm. Hence, every time we pass a batch of data through the CNNs, we completed an iteration. For example, if we have $128 \times 10\,286 = 1\,316\,608$ training samples, the batch size is 128, it takes 10 286 iterations to complete 1 epoch. Too few epochs produce a poor underfitting DL result, and too many epochs waste computation time, and possibly generate overfitting results. For our experiment, 50 epochs get sufficiently good results. In addition, the indexes of the input-output pairs are shuffled before starting the next epoch. Finally, to ensure the original data unchange, the live data from the original input are reinserted into their original positions in the DL result.

III. Experiments

All experiments are performed on a Dell precision tower 7810 workstation with windows 7, two Intel Xeon E5-2620 processors, 2.10 GHz CPU, 176 GB of RAM, and a NVIDIA GeForce RTX 2080 Ti GPU (11 GB). In the case of reference data available, we define the signal-to-noise ratio (SNR)

$$\mathrm{SNR(dB)} = -20 \log_{10} \frac{\|\mathbf{Y}_{\mathrm{ref}} - \mathbf{Y}_{\mathrm{pre}}\|_2}{\|\mathbf{Y}_{\mathrm{ref}}\|_2} \tag{2}$$

to assess the reconstruction quality, where $\mathbf{Y}_{\mathrm{ref}}$ and $\mathbf{Y}_{\mathrm{pre}}$ denote the true answer and the network prediction, respectively. In the training process, the network input is in a specified patch-size 112×112. However, while testing the trained network, the input is no longer restricted by the patch-size. That is, for network test, the input data need not to be divided into small patches (a shot gather can be fed into the network directly).

A. Feasibility and Effectiveness of the Evaluated Framework

For a well-trained network, it should produce reasonable output for new input that is never seen in the training process (the so-called "generalization" capacity of the network). To validate the generalization capability, we employ several data sets (e.g., a synthetic test data set shown in Fig. 5, a physical modeling data set shown in Fig. 6) that are never seen in the training process. Since we work in a local patch-wise fashion in training, the whole feature of a shot gather in the training data (see Fig. 4) is also not completely seen. Examples in Figs. 4–8 indicate that the trained network can reconstruct regularly missing data with high accuracy for different test data sets, even in the presence of big gaps (see Figs. 4 and 8). These results validate the feasibility, effectiveness, and generalization capacity of the evaluated approach.

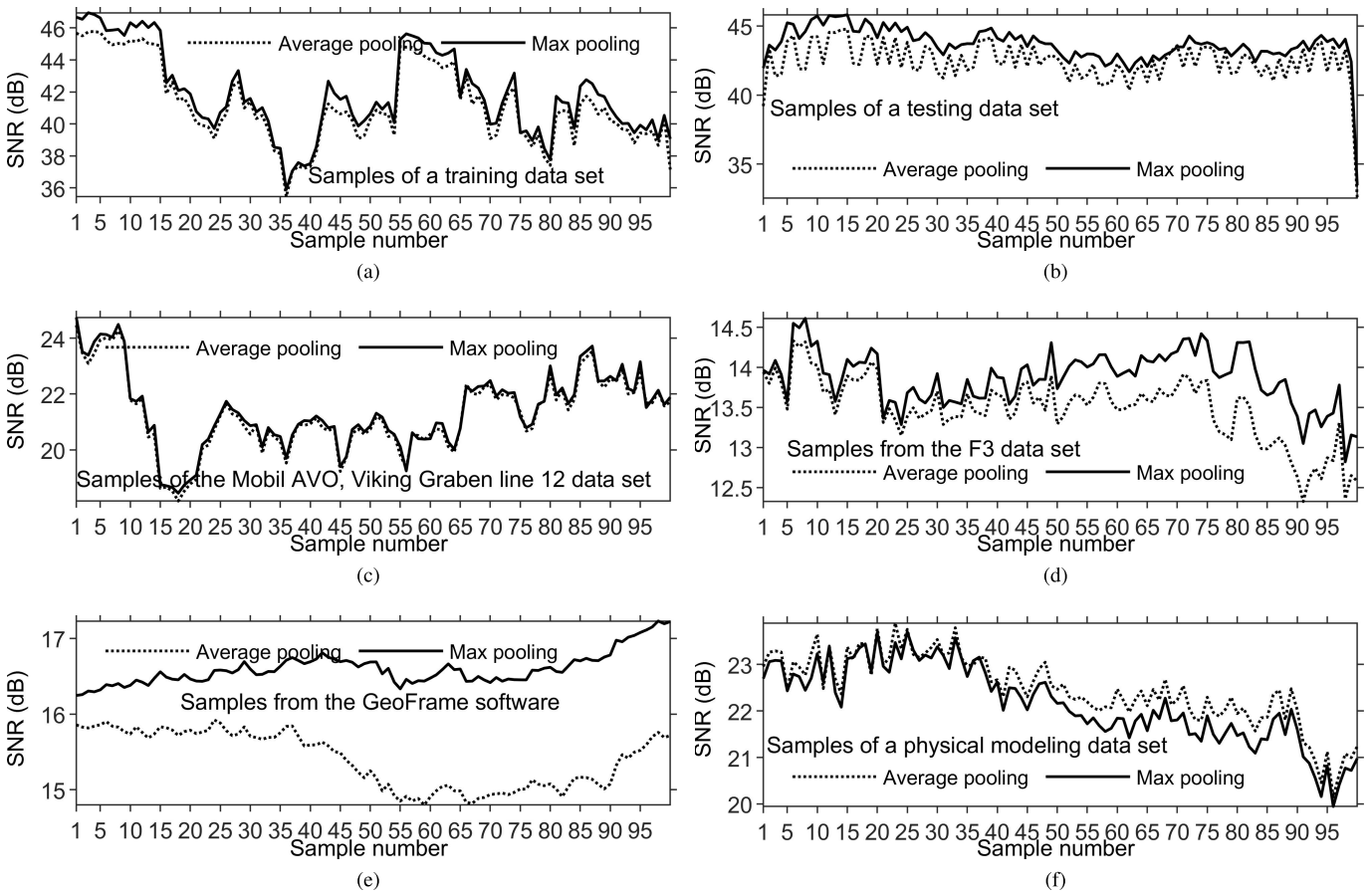

Fig. 11. Impact of the pooling way. Variations of SNR with the 100 test samples for (a) synthetic training data set, (b) synthetic test data set generated with the Marmousi2 model [see Fig. 3(d)], (c) Mobil Graben line 12 data set, (d) F3 data set, (e) data set from the GeoFrame software, and (f) physical modeling data set used in [41].

B. Compared with the f-x Prediction Interpolation Method

Fig. 7(a) shows a field shot gather from the Mobil AVO Viking Graben line 12 data set. To compare with the f-x prediction method [34], we apply a 2D regularly sampling operator with a decimation of factor 2 in the spatial-dimension only to Fig. 7(a), resulting in Fig. 7(b). The computational parameters of the f-x prediction method are taken empirically to achieve the best results in our tests. One main difficulty when applying the f-x prediction method to field data is the choice of the length of the prediction filter, which is set as 3 determined by trial and error. The result from the f-x prediction method [34] is shown in Fig. 7(c). Though the parameter may be not the optimal one, it can illustrate the shortcomings of the f-x interpolation method to some extent. Fig. 7(d) demonstrates the DL result. By comparing the DL results [see Fig. 7(d)] with those of Spitz's f-x method [see Fig. 7(c)], DL achieves less reconstruction error and higher SNR value (25.6247 versus 12.531). This is because the f-x interpolation method is suitable for linear events theoretically. This supports that DL-based methods can avoid linear events assumptions of the traditional interpolation methods.

C. Impact of the Network Depth

As an interesting topic, we investigate the impact of the network depth on the DL result. Table I summarized four networks with different network depth. From the network architecture shown in Fig. 1, we can see that the network depth is mostly determined by the downsampling operations. The more the downsampling layers, the deeper the network, the more the convolutional layers, the more the trainable parameters. Therefore, we classify the designed networks according to the number of downsampling layers. For instance, network 4 contains four downsampling operations, and correspondingly four upsampling operations, as shown in Fig. 1. The performances of the four networks are shown in Fig. 9. One of the most important issues is convergence of the training process. The training logs shown in Fig. 9(a) reveal convergence, which is getting better with the growing network depth. Fig. 9(b) and (c) shows the test performances of the four networks on two typical data sets. Comparison of SNR values obtained using different networks indicates that the results are getting better with the increasing network depth. However, the benefits derived from the growing network depth will reach their limitations. For example, the performance of network 3 is competitive to network 4, where both of them outperform networks 1 and 2. To some degree, the increased network depth also increases the difficulty to train the network.

D. Impact of the Kernel Size

As a curiosity, we study the impact of the size of the convolution kernel on the DL result. Fig. 10 shows the

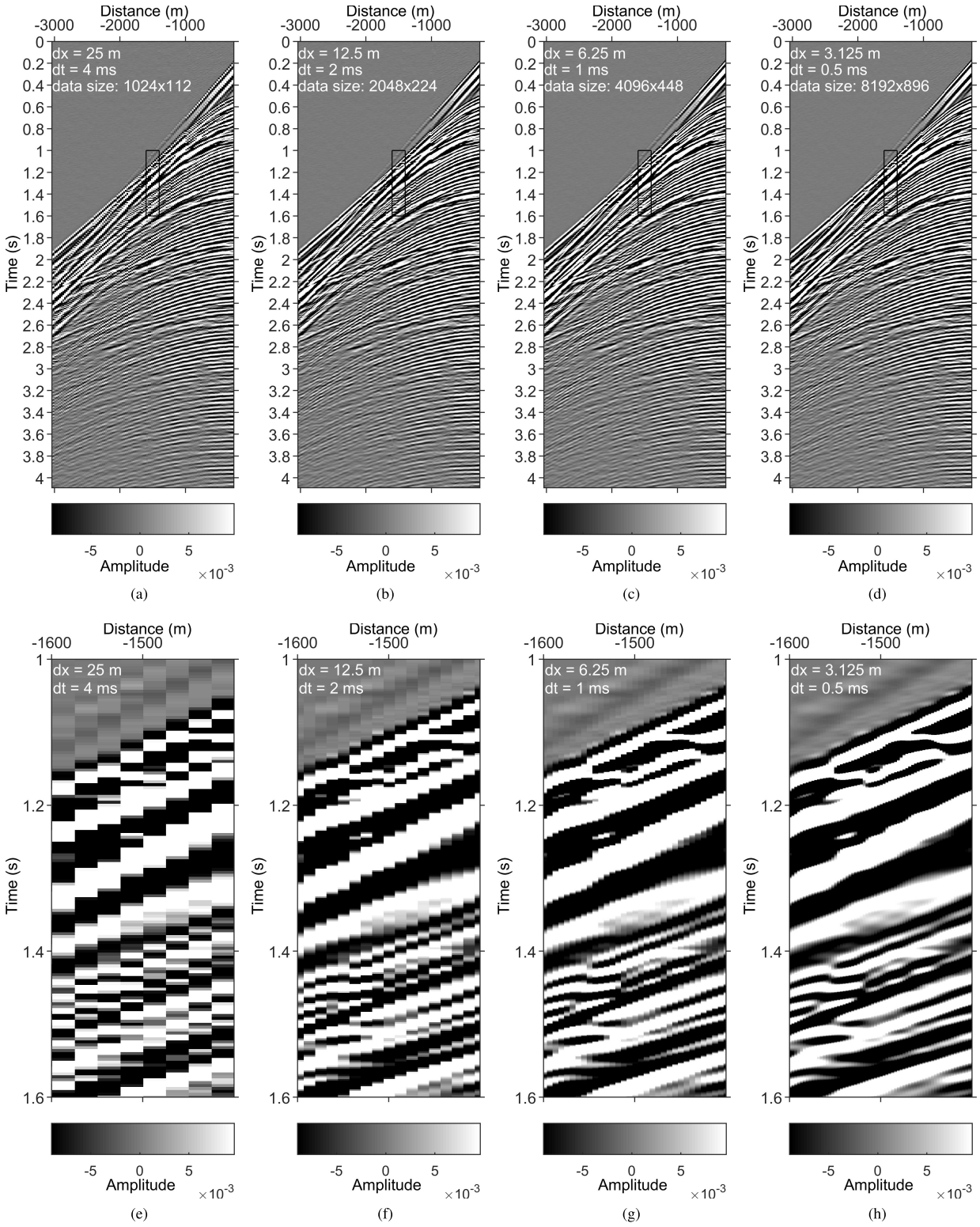

Fig. 12. Dense data reconstruction. (a) Original shot gather. Interpolated data using the trained network with both trace and time intervals (b) halved, (c) quartered, and (d) one-eighth. (e)–(h) show enlargement of marked rectangle in (a)–(d), respectively.

performances of two trained networks with different kernel size on two typical data sets. Comparison of SNR curves indicates that 5×5 kernel size outperforms 3×3 for this experiment. Once the network architecture is determined, the kernel size directly affects the number of the trainable parameters. The trainable parameters are 31 377 793 for kernel size 3×3 and 87 149 953 for 5×5. A larger kernel size partly expands the receptive field of the convolution kernel. The larger trainable parameters enhance the characterization capacity of the network.

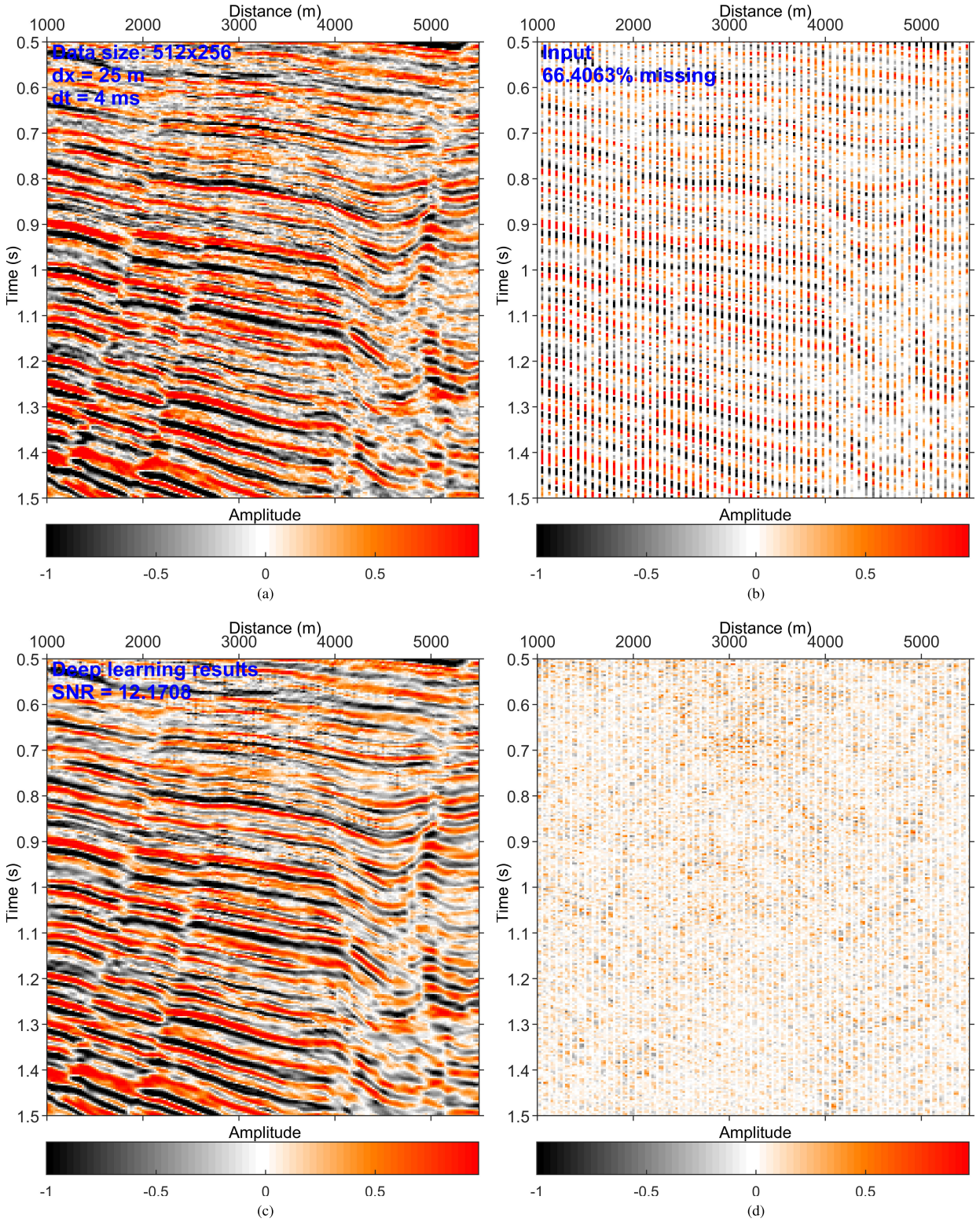

Fig. 13. Results on a data set from the GeoFrame software. (a) Reference data. (b) Regularly sampled data with a decimation of factor 3 along the space dimension (approximately 66% missing). (c) DL reconstruction result. (d) Difference between (a) and (c).

E. Impact of the Pooling Function

There are several kinds of pooling layers in the DL library (e.g., commonly used max pooling, average pooling, etc.). The max pooling operation reports the maximum output within a rectangular neighborhood (determined by the pool size) [2]. Accordingly, the average pooling function returns the average of a rectangular neighborhood. We intend to find out which pooling layer is suitable for our task. Fig. 11 shows the test performances of two trained networks with different pooling layer on several typical data sets. Comparison of the SNR

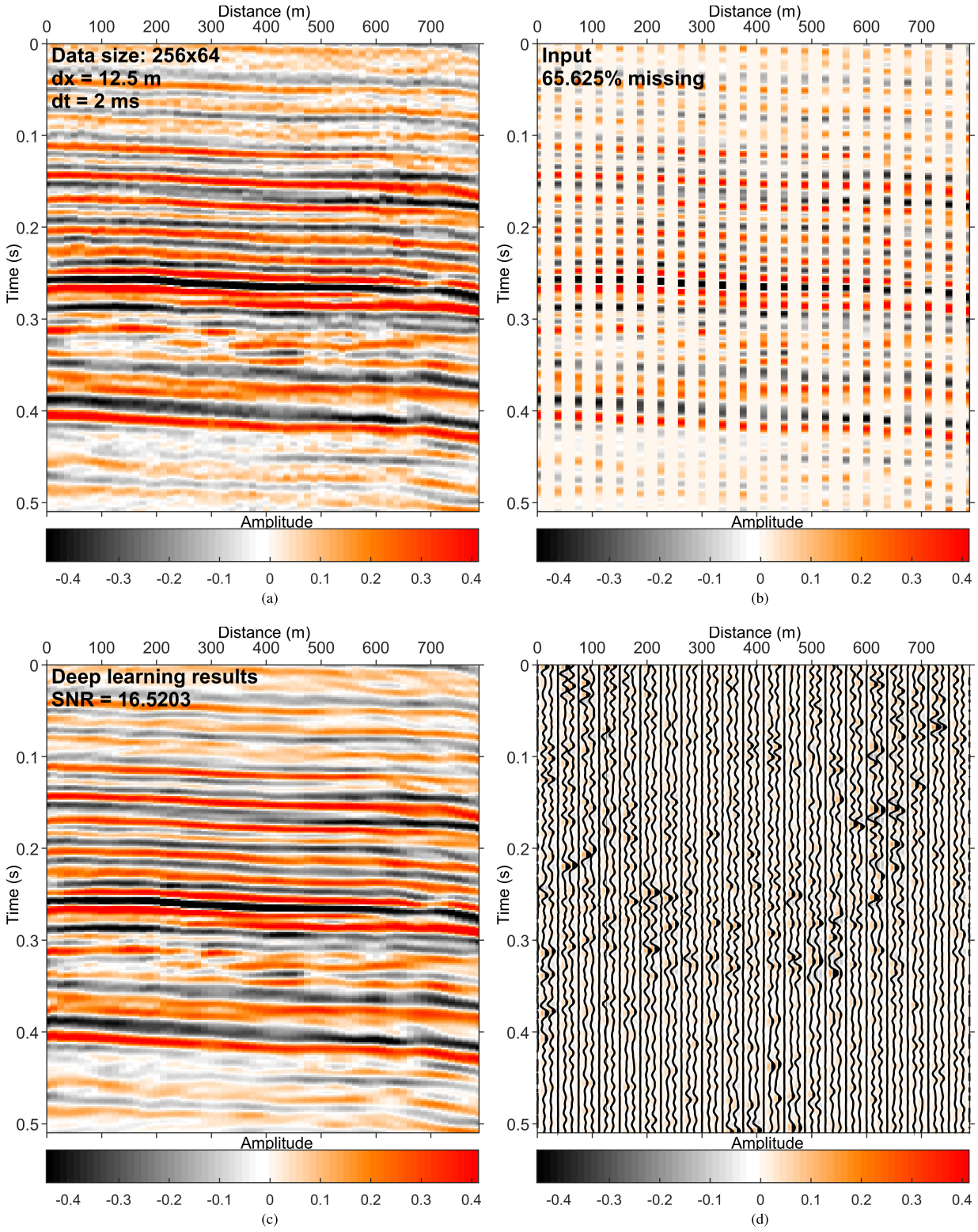

Fig. 14. Results on a data set from the Hampson-Russell software. (a) Reference data. (b) Regularly sampled data with a decimation of factor 3 along the space dimension (approximately 66% missing). (c) DL reconstruction result. (d) Difference between (a) and (c).

change of the reconstructions in Fig. 11 reveals that the max pooling operation (solid line) outperforms the average pooling operation (dotted line) under most situations, except for the physical modeling data set experiment [see Fig. 11(f)]. This partly explains why the max pooling layer is commonly used.

F. Dense Data Reconstruction

The trained network is ultimately used to reconstruct dense data (see Fig. 12). Visually, the four images in Fig. 12(a)–(d) are indistinguishable. The aggressive amplification makes all events aliased in Fig. 12(e). The DL results

shown in Fig. 12(f)–(h) are encouraging. Because the data in Fig. 12(d) are interpolated thrice as densely as the original data [see Fig. 12(a)] along both time points and receiver lines, the output data size (8192×896) is eight times the original data size (1024×112). The reconstructed dense data are more continuous along both the spatial and temporal dimensions, and the aliasing effects disappear.

IV. Discussion

The performances on synthetic and field data sets validate the feasibility and effectiveness of the evaluated framework. However, we underline that reduced performance appears if the features of the test data are significantly different from those of the training data. Figs. 13 and 14 serve to support this point, where the trained network is applied to two post-stack data sets (remember that the network is trained with pre-stack shot gathers only). The bias increases as the differences increase. Even though the precision decreases with the increasing feature difference between the test and training data, the network still generates reasonable results (see Figs. 13 and 14). This also demonstrates the generalization capacity of the trained network.

From Fig. 14 we find out that the trained network can overcome noise to some extent. The DL reconstruction result [see Fig. 14(c)] is cleaner than the reference data [see Fig. 14(a)]. Fig. 14(d) shows the removed noise. The main reason is that the training data sets are nearly noise-free, and the features of the random noise of the field data are not learned by the network.

The actual physical meaning of the spatial and temporal dimensions is not too relevant to the network. However, it is possible that the network fails if the spatial or temporal-sampling intervals of the test data are significantly different from those of the training data. Therefore, different kinds of reliable data should be better incorporated into the training set to further enhance the generalization capacity of the network.

The network as described in this article can have tens to hundreds of millions of trainable parameters, leading to a large (GPU) memory requirement. The main computational cost of the DL method arises from the training process, which happens once up front. To train the network used in Fig. 7(d), it takes a reasonable training time of roughly 5 h on our machine. After training, the prediction computational cost is relatively negligible. For instance, prediction of a 1024×112 data sample takes less than two seconds on our machine without using GPU. In other words, once training is done, the prediction is very efficient, and we generally do not need to re-train for every new data set (this is supported by Figs. 6, 13, and 14). Therefore, the overall computational cost is relatively efficient.

The framework assessed in this article can be generalized to multidimensional cases. Although we have focused on 2-D, extending to three/five dimensions is theoretically straightforward. A generalization to 3-D is trivial. This requires replacing the 2-D convolutional/pooling/up-sampling layers at the heart of the network architectures with 3-D versions, which is supported by most DL platforms (e.g., Keras, Tensorflow, etc). We are moving toward multi-dimensional reconstruction hoping to obtain better results by using more data. An efficient implementation is mandatory for practical multidimensional application because huge amounts of data are involved.

We have demonstrated the application of the evaluated framework for regularly missing data reconstruction, however, it can also handle irregularly missing data reconstruction, which will be reported soon. In this investigation, we primarily focus on missing data reconstruction but the insights presented here also suggest similar potentials of DL in other fields (e.g., super-resolution reconstruction of image/photo, and satellite map). Once a general network architecture is in place, the same principles can be applied to various problems. By carefully identifying the problems, we aim to overcome and setting up a network using existing best platforms, we can transfer this leading-edge technology to the wide geoscience world.

V. Conclusion

Taking inspiration from image-to-image translation, we evaluate a DL-based framework for regularly missing data reconstruction, aiming of translating incomplete data into their corresponding complete data. To accomplish this, we first establish a network architecture inspired by an end-to-end U-Net CNN. Then we meticulously prepare the training data with both synthetic and field seismic data. To train the network, we used a mean-squared-error loss function and an Adam optimization algorithm. The network architecture, the training data, and the training setups are explained in detail. Next, we test the trained network with several typical data sets, which achieves good performance (even in the presence of big gaps), validating the feasibility, effectiveness, and generalization capacity of the evaluated framework. We illustrate the feature maps for a sample going through the well-trained network. We demonstrate the impact of the network depth, the size of the convolution kernel, and the pooling function on the DL results. Comparison with the f-x prediction interpolation method indicates that the DL-based method performs better, and can avoid linear events assumptions of traditional interpolation methods. The trained network is successfully applied to dense data reconstruction. The proposed method can overcome noise to some degree. We finally discussed some practical aspects and extensions of the assessed framework. This article supports that DL manifests huge potential in advanced intelligent applications over traditional techniques.

Acknowledgment

The authors would like to thank Prof. Jianxin Wei at the China University of Petroleum (Beijing) for providing the used physical modeling data set and Dr. Hanming Chen for providing the seismic forward modeling codes used in this article. They would like to thank the data contributors and releasers of the Mobil AVO Viking Graben line 12 data set and the F3 data set that are obtained via an open data website (https://wiki.seg.org/wiki/Open_data) and Prof. Jianwei Ma for providing them the data set from the GeoFrame software.

They would also like to thank Prof. Mauricio D. Sacchi for releasing the SeismicLab toolbox, where they found the codes of the f-x prediction interpolation method, and SMAART JV for the used Sigsbee2B and Pluto 1.5 earth models. They would also like to thank Hampson-Russell for the used data set, Junyong Yu for help with proofreading, and Dr. Simon Yueh, Editor, Mauricio D. Sacchi, Associate Editor, and two anonymous reviewers for their helpful suggestions and comments that significantly improved the quality of this article.

REFERENCES

[1] Y. LeCun, Y. Bengio, and G. Hinton, "Deep learning," *Nature*, vol. 521, no. 7553, pp. 436–444, 2015, doi: 10.1038/nature14539.

[2] I. Goodfellow, Y. Bengio, and A. Courville, *Deep Learning*. Cambridge, MA, USA: MIT Press, 2016. [Online]. Available: http://www.deeplearningbook.org

[3] T. Falk *et al.*, "U-Net: Deep learning for cell counting, detection, and morphometry," *Nature Methods*, vol. 16, pp. 67–70, 2019, doi: 10.1038/s41592-018-0261-2.

[4] M. Bojarski *et al.*, "End to end learning for self-driving cars," 2016, *arXiv:1604.07316*. [Online]. Available: https://arxiv.org/abs/1604.07316

[5] H. Greenspan, B. Van Ginneken, and R. M. Summers, "Guest editorial deep learning in medical imaging: Overview and future promise of an exciting new technique," *IEEE Trans. Med. Imag.*, vol. 35, no. 5, pp. 1153–1159, May 2016, doi: 10.1109/tmi.2016.2553401.

[6] Y. Lecun *et al.*, "Backpropagation applied to handwritten zip code recognition," *Neural Comput.*, vol. 1, no. 4, pp. 541–551, Dec. 1989, doi: 10.1162/neco.1989.1.4.541.

[7] A. Krizhevsky, I. Sutskever, and G. E. Hinton, "ImageNet classification with deep convolutional neural networks," *Commun. ACM*, vol. 60, no. 6, pp. 84–90, May 2017, doi: 10.1145/3065386.

[8] Y. Lecun, L. Bottou, Y. Bengio, and P. Haffner, "Gradient-based learning applied to document recognition," *Proc. IEEE*, vol. 86, no. 11, pp. 2278–2324, Nov. 1998, doi: 10.1109/5.726791.

[9] J. Schmidhuber, "Deep learning in neural networks: An overview," *Neural Netw.*, vol. 61, pp. 85–117, Jan. 2015. [Online]. Available: http://www.sciencedirect.com/science/article/pii/S0893608014002135

[10] L. Huang, X. Dong, and T. E. Clee, "A scalable deep learning platform for identifying geologic features from seismic attributes," *Lead. Edge*, vol. 36, no. 3, pp. 249–256, Mar. 2017, doi: 10.1190/tle36030249.1.

[11] M. Araya-Polo, T. Dahlke, C. Frogner, C. Zhang, T. Poggio, and D. Hohl, "Automated fault detection without seismic processing," *Lead. Edge*, vol. 36, no. 3, pp. 208–214, Mar. 2017, doi: 10.1190/tle36030208.1.

[12] P. Lu, M. Morris, S. Brazell, C. Comiskey, and Y. Xiao, "Using generative adversarial networks to improve deep-learning fault interpretation networks," *Lead. Edge*, vol. 37, no. 8, pp. 578–583, Aug. 2018, doi: 10.1190/tle37080578.1.

[13] W. Xiong *et al.*, "Seismic fault detection with convolutional neural network," *Geophysics*, vol. 83, no. 5, pp. O97–O103, Sep. 2018, doi: 10.1190/geo2017-0666.1.

[14] X. Wu, L. Liang, Y. Shi, and S. Fomel, "FaultSeg3D: Using synthetic data sets to train an end-to-end convolutional neural network for 3D seismic fault segmentation," *Geophysics*, vol. 84, no. 3, pp. IM35–IM45, May 2019, doi: 10.1190/geo2018-0646.1.

[15] X. Tian and H. Daigle, "Machine-learning-based object detection in images for reservoir characterization: A case study of fracture detection in shales," *Lead. Edge*, vol. 37, no. 6, pp. 435–442, Jun. 2018, doi: 10.1190/tle37060435.1.

[16] Z. Wang, H. Di, M. A. Shafiq, Y. Alaudah, and G. Alregib, "Successful leveraging of image processing and machine learning in seismic structural interpretation: A review," *Lead. Edge*, vol. 37, no. 6, pp. 451–461, Jun. 2018, doi: 10.1190/tle37060451.1.

[17] T. Wrona, I. Pan, R. L. Gawthorpe, and H. Fossen, "Seismic facies analysis using machine learning," *Geophysics*, vol. 83, no. 5, pp. O83–O95, Sep. 2018, doi: 10.1190/geo2017-0595.1.

[18] F. Qian, M. Yin, X.-Y. Liu, Y.-J. Wang, C. Lu, and G.-M. Hu, "Unsupervised seismic facies analysis via deep convolutional autoencoders," *Geophysics*, vol. 83, no. 3, pp. A39–A43, May 2018, doi: 10.1190/geo2017-0524.1.

[19] G. Röth and A. Tarantola, "Neural networks and inversion of seismic data," *J. Geophys. Res.*, vol. 99, no. B4, pp. 6753–6768, Feb. 2004. [Online]. Available: https://agupubs.onlinelibrary.wiley.com/doi/abs/10.1029/93JB01563

[20] Y. Kim and N. Nakata, "Geophysical inversion versus machine learning in inverse problems," *Lead. Edge*, vol. 37, no. 12, pp. 894–901, Dec. 2018, doi: 10.1190/tle37120894.1.

[21] M. Araya-Polo, J. Jennings, A. Adler, and T. Dahlke, "Deep-learning tomography," *Lead. Edge*, vol. 37, no. 1, pp. 58–66, Jan. 2018, doi: 10.1190/tle37010058.1.

[22] Y. Jia and J. Ma, "What can machine learning do for seismic data processing? An interpolation application," *Geophysics*, vol. 82, no. 3, pp. V163–V177, May 2017, doi: 10.1190/geo2016-0300.1.

[23] Y. Jia, S. Yu, and J. Ma, "Intelligent interpolation by Monte Carlo machine learning," *Geophysics*, vol. 83, no. 2, pp. V83–V97, Mar. 2018, doi: 10.1190/geo2017-0294.1.

[24] S. Mandelli, F. Borra, V. Lipari, P. Bestagini, A. Sarti, and S. Tubaro, *Seismic Data Interpolation Through Convolutional Autoencoder*. Tulsa, OK, USA: Society of Exploration Geophysicists, 2018, pp. 4101–4105. [Online]. Available: https://library.seg.org/doi/abs/10.1190/segam2018-2995428.1

[25] S. Alwon, *Generative Adversarial Networks in Seismic Data Processing*, 2018, pp. 1991–1995. [Online]. Available: https://library.seg.org/doi/abs/10.1190/segam2018-2996002.1

[26] B. Wang, N. Zhang, W. Lu, and J. Wang, "Deep-learning-based seismic data interpolation: A preliminary result," *Geophysics*, vol. 84, no. 1, pp. V11–V20, Jan. 2019, doi: 10.1190/geo2017-0495.1.

[27] P. Isola, J.-Y. Zhu, T. Zhou, and A. A. Efros, "Image-to-image translation with conditional adversarial networks," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit. (CVPR)*, Jul. 2017, pp. 5967–5976, doi: 10.1109/cvpr.2017.632.

[28] J. Zhu, T. Park, P. Isola, and A. A. Efros, "Unpaired image-to-image translation using cycle-consistent adversarial networks," 2017, *arXiv:1703.10593*. [Online]. Available: http://arxiv.org/abs/1703.10593

[29] R. Abma and N. Kabir, "3D interpolation of irregular data with a POCS algorithm," *Geophysics*, vol. 71, no. 6, pp. E91–E97, Nov. 2006, doi: 10.1190/1.2356088.

[30] D. Trad, "Five-dimensional interpolation: Recovering from acquisition constraints," *Geophysics*, vol. 74, no. 6, pp. V123–V132, Nov. 2009, doi: 10.1190/1.3245216.

[31] P. M. Zwartjes and M. D. Sacchi, "Fourier reconstruction of nonuniformly sampled, aliased seismic data," *Geophysics*, vol. 72, no. 1, pp. V21–V32, Jan. 2007, doi: 10.1190/1.2399442.

[32] M. Naghizadeh and M. D. Sacchi, "On sampling functions and Fourier reconstruction methods," *Geophysics*, vol. 75, no. 6, pp. WB137–WB151, Nov. 2010, doi: 10.1190/1.3503577.

[33] J. Ma, "Three-dimensional irregular seismic data reconstruction via low-rank matrix completion," *Geophysics*, vol. 78, no. 5, pp. V181–V192, Sep. 2013, doi: 10.1190/geo2012-0465.1.

[34] S. Spitz, "Seismic trace interpolation in the F-X domain," *Geophysics*, vol. 56, no. 6, pp. 785–794, 1991, doi: 10.1190/1.1443096.

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

[36] H. Chen, H. Zhou, Q. Zhang, M. Xia, and Q. Li, "A k-space operator-based least-squares staggered-grid finite-difference method for modeling scalar wave propagation," *Geophysics*, vol. 81, no. 2, pp. T45–T61, Mar. 2016, doi: 10.1190/geo2015-0090.1.

[37] H. Chen, H. Zhou, Q. Zhang, and Y. Chen, "Modeling elastic wave propagation using *k*-space operator-based temporal high-order staggered-grid finite-difference method," *IEEE Trans. Geosci. Remote Sens.*, vol. 55, no. 2, pp. 801–815, Feb. 2017, doi: 10.1109/TGRS.2016.2615330.

[38] R. G. Keys and D. J. Foster, *Comparison of Seismic Inversion Methods on a Single Real Data Set*, 1998, doi: 10.1190/1.9781560802082.

[39] F. Chollet *et al. Keras*. 2015. [Online]. Available: https://github.com/keras-team/keras

[40] G. S. Martin, R. Wiley, and K. J. Marfurt, "Marmousi2: An elastic upgrade for Marmousi," *Lead. Edge*, vol. 25, no. 2, pp. 156–166, Feb. 2006, doi: 10.1190/1.2172306.

[41] X. Chai, S. Wang, J. Wei, J. Li, and H. Yin, "Reflectivity inversion for attenuated seismic data: Physical modeling and field data experiments," *Geophysics*, vol. 81, no. 1, pp. T11–T24, Jan. 2016, doi: 10.1190/geo2015-0250.1.

Xintao Chai received the B.S. degree in information and computing science and the M.S. degree in geophysics from the China University of Petroleum (Huadong), Dongying, China, in 2009 and 2012, respectively, and the Ph.D. degree in geophysics from the China University of Petroleum (Beijing), Beijing, China, in 2016.

He is currently with the Institute of Geophysics and Geomatics, China University of Geosciences (Wuhan), Wuhan, China, conducting research in the fields of exploration seismology. He is also the Founder and the Director of the Center for Wave Propagation and Imaging and DeepResearch Group. His research interests include seismic wave propagation theory, signal processing, estimation and imaging of primaries, surface-related multiples, and internal multiples, seismic forward modeling, imaging, and inversion, artificial intelligence, machine learning, and deep learning.

Genyang Tang received the B.Sc. degree in geophysics from the Department of Geophysics, Peking University, Beijing, China, in 2006, and the Ph.D. degree in marine geophysics from the Department of Earth Sciences, University of Cambridge, Cambridge, U.K., in 2012.

He is currently an Academic Fellow with the China University of Petroleum (Beijing), Beijing, teaching courses on seismic waves and their applications and conducting research in the field of seismic exploration. His research interests include seismological rock physics and seismic inversion.

Shangxu Wang received the B.S. degree in geophysics from the Wuhan College of Geoscience, Wuhan, China, in 1984, and the M.S. and the Ph.D. degrees in geophysics from the China University of Petroleum (Beijing), Beijing, China, in 1987 and 1990, respectively.

He has been with the China University of Petroleum (Beijing) since 1990. His research interests include seismic wave propagation theory, seismic forward modeling and inversion, and seismic involved rock physics. He is currently a Changjiang River Scholars Distinguished Professor, the Deputy Director of the State Key Laboratory of Petroleum Resource and Prospecting, and the Director of the Geophysics Key Laboratory Academic Committee, China National Petroleum Corporation (CNPC), Beijing.

Ronghua Peng received the B.S. and Ph.D. degrees in geophysics from the China University of Geosciences (Wuhan), Wuhan, China, in 2011 and 2016, respectively.

He is currently with the Institute of Geophysics and Geomatics, China University of Geosciences (Wuhan). His research interests mainly focus on geophysical electromagnetic forward modeling and inverse problems, as well as parallel computing.

Wei Chen received the B.S. and Ph.D. degrees in geophysics from the China University of Petroleum (Beijing), Beijing, China, in 2008 and 2016, respectively.

He is currently an Associate Professor with the College of Geophysics and Petroleum Resources, Yangtze University, Wuhan, China. His research interests include seismic signal analysis, time-frequency analysis, and seismic data processing.

Jingnan Li received the B.S. and Ph.D. degrees in geophysics from the China University of Petroleum (Beijing), Beijing, China, in 2011 and 2016, respectively.

He is currently with the Sinopec Geophysical Research Institute, Nanjing, China, where he is conducting research in the field of exploration geophysics. His research interests include seismic modeling, seismic inversion, reservoir characterization, and application of artificial intelligence to geophysics.