Circuits, Systems, and Signal Processing (2021) 40:5468–5482
https://doi.org/10.1007/s00034-021-01727-4

Chirp Signal Denoising Based on Convolution Neural Network

Guangli Ben[1] · Xifeng Zheng[1] · Yongcheng Wang[1] · Xin Zhang[1] · Ning Zhang[1]

Received: 28 May 2020 / Revised: 19 April 2021 / Accepted: 22 April 2021 /
Published online: 10 May 2021

Abstract

Many classic chirp signal processing algorithms may show significant performance degradation when the signal-to-noise ratio (SNR) is low. To address this problem, this paper proposes a pre-filtering method in time-domain based on deep learning. Different from traditional signal filtering methods, the proposed denoising convolutional neural network (DCNN) is trained to recover the pure signal from the noisy signal as much as possible. Following denoising, we use two classic chirp signal parameter estimation algorithms to estimate the parameters of the DCNN output. The simulation results show that, compared with no DCNN processing, the parameter estimation accuracy is significantly improved. This is mainly due to the powerful pure signal extraction ability of DCNN, which can significantly improve the SNR and the accuracy of signal parameter estimation.

Keywords Deep leaning · Denoising · Chirp · Parameter estimation

✉ Guangli Ben
g2009051126@163.com

Xifeng Zheng
zhengxf@ccxida.com

Yongcheng Wang
wyc_dyy@sina.com

Xin Zhang
zhangxin162@mails.ucas.ac.cn

Ning Zhang
zhangning171@mails.ucas.ac.cn

[1] Changchun Institute of Optics, Fine Mechanics and Physics, Chinese Academy of Science, Changchun 130033, China

1 Introduction

Chirp signals have a broad range of applications, such as in radar, sonar, communication and medical imaging [9]. Chirp usage and estimation of their parameters is a significant part in the digital signal processing area [3]. At present, many methods have been proved to be efficient and practical. However, they all face a common problem, which is their performance is reduced a lot under the condition of low SNR.

Last few years, driven by the large amount of data, deep learning has been widely used in image denoising and has achieved good results [14, 18, 20, 21]. However, in the field of digital signal processing, the application of processing methods based on convolutional neural networks (CNN) is not as many as in image processing. Yuan Jiang proposed a deep learning denoising-based approach for line spectral estimation [10]. By using CNN to filter multi-component single-frequency signals, combined with the popular model order selection method and a subspace line spectral estimator, this method has achieved a substantial improvement compared with the line spectral estimation results obtained by directly applying the subspace estimator without denoising. In addition to using CNN for filtering in the time domain, Se Rim Park proposed a method to remove noise from speech signals in time–frequency domain [13] for enhancing the quality and intelligibility of speech. The training data and validation data of the network are the amplitude spectra of noisy speech signals and pure speech signals, respectively. Network's output is the magnitude spectrum of the denoised signal. And then converts back to the time domain by using inverse short-time Fourier transform of the output's magnitude spectrum and the noisy signal's phase spectrum. Xiaolong Chen applies CNN for replacing the Fourier transform and fractional Fourier transform (FrFT) and uses it for single-frequency signal and LFM signal detection and estimation [2]. It has proved that the CNN-based method can achieve good recognition performance at SNR above -2 dB, and above -10 dB combined with the wavelet denoising method. CNN also plays an active role in the field of signal detection and classification. Huyong Jin proposed a CNN-based framework to perform preamble detection for underwater acoustic communications application [11], which can learns features from the time–frequency spectrum, and can give an efficient solution for preamble detection under complicated underwater acoustic communications. For signal classification, Johan Brynolfsson uses Wigner–Ville distribution instead of the spectrogram as basic input into CNN to Classify One-Dimensional Non-Stationary Signals and has achieved good performance [1]. Inspired by the above ideas, this article applies deep learning to signal filtering processing. Compared with directly applying CNN to estimate the parameters of the signal as described in paper [2], the proposed method can extract pure signals from noisy signals and then can be combined with classic signal processing algorithms to improve algorithm performance further.

Assume a chirp signal $x(t)$ with additive Gaussian noise $n(t)$ is described as follows:

$$y(t) = x(t) + n(t) \tag{1}$$

where $x(t) = \sin\left(2\pi f_0 t + \pi k t^2\right)$, f_0 denotes the initial frequency, k denotes the chirp rate, the prior knowledge of their value are unknown. We employ DCNN to extract

denoised signal $\hat{y}(t)$ from observation $y(t)$. Then, the performance of DCNN is measured by comparing the SNR and parameter estimation accuracy of $y(t)$ and $\hat{y}(t)$.

This paper is organized as follows. Section 2 introduces the structure of DCNN and its associated inner layers. Section 3 introduces two typical parameter estimation methods used in the article. One is based on time–frequency analysis, and the other is based on fractional Fourier transform. In Sect. 4, simulation results are given to show the advantages of the DCNN. Finally, conclusions are drawn in Sect. 5.

2 The Structure of DCNN

The observation signal is one-dimensional data that changes with time. So the proposed DCNN is a one-dimensional network. Generally, DCNN has a layer structure that includes input layer, more hidden layers, and a regression output layer as shown in Fig. 1. The input of DCNN is a real mono-component chirp signal with N samples blurred by additive white Gaussian noises, denoted as $Y^0 \in R^{N\times 1}$.

Conv is convolutional layer which applies sliding convolutional filters to the input. The layer convolves the input by moving the filters along the input vertically and horizontally, and computing the dot product of the weights W^i and the input Y^{i-1}, then adding a bias term B^i. Convolution output G^i can be expressed as:

$$G^i = Y^{i-1} * W^i + B^i \tag{2}$$

where i denotes the layer number, W^i and B^i are learnable parameters, and Y^{i-1} is output of the previous activation layer. The dimension of G^i is $N \times 1 \times k$, as the convolution is operated with same padding.

BN is batch normalization layer. BN normalizes each input channel across a mini-batch before the nonlinearity layer to speed up training of convolutional neural networks and reduce the sensitivity to network initialization [8]. BN first calculates the mean μ and variance δ^2 across a mini-batch over each input G^i. Then, the normalized activations can be described as:

$$\hat{G}^i = \frac{G^i - \mu}{\sqrt{\delta^2 + \varepsilon}} \tag{3}$$

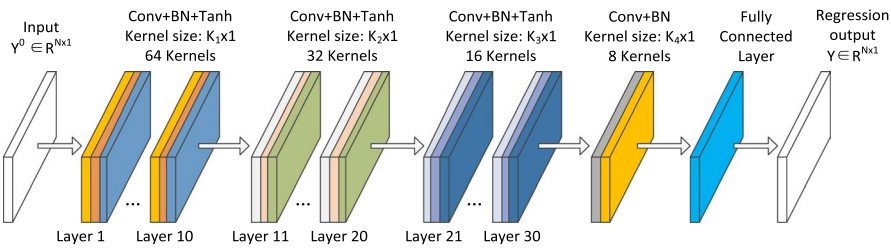

Fig. 1 Structure of DCNN, Conv is convolutional layer, BN is batch normalization layer, Tanh is hyperbolic tangent activation function of nonlinearity layer

where ε is a hyper parameter set to 0.00001 to improve numerical stability when mini-batch variance is very small. Then BN shift and scale the activations as:

$$Z^i = \lambda \hat{G}^i + \beta \tag{4}$$

where λ is scale factor, β is offset, and they are learnable parameters.

After BN normalization, a hyperbolic tangent function, denoted by Tanh, is used to increase nonlinearity properties of DnCNN, and then the output Y^i can be expressed as:

$$Y^i = \tanh\left(Z^i\right) = \frac{e^{z^i} - e^{-z^i}}{e^{z^i} + e^{-z^i}} \tag{5}$$

The regression output layer computes the half-mean-squared-error loss for regression problems. The optimization goal of training is to minimize the half-mean-squared-error loss between regression output $\hat{y}(n)$ and the clean labels $x(n)$, and then the loss function for training is:

$$\text{loss} = \frac{1}{2}\sum_{i=1}^{M}\left(\hat{y}_i - x_i\right)^2 \tag{6}$$

where M is mini-batch size, $\hat{y}_i \in R^{N\times 1}$, $x_i \in R^{N\times 1}$. During training process, we use mini-batch gradient descent with Adam optimization algorithm [4] to evaluate the gradient of the loss function and backpropagate to update DCNN weights.

Since the additive white Gaussian noise is randomly distributed, in order to train DCNN to effectively remove noises from observation Y^0, in the vicinity of the same SNR value, we need a lot of training data to enable the network to better adapt to the randomness of noise. The parameters of initial frequency f_0 and chirp rates k should also be selected reasonably so that they can cover the parameter range of the signal. In order to enable the network to better learn the data characteristics of the training data set and the verification data set, we need to train multiple epochs with the data. And we shuffle the data before each training epoch.

3 Parameter Estimation of Denoised Signal

After denoising, the initial frequency f_0 and chirp rate k of observation are estimated by denosied output $\hat{y}(n)$. There are many parameter estimation algorithms available, for example based on maximum likelihood principle [12, 16, 17], the gradient of the short-time Fourier transform complex phase [3] and the Fractional Order Cross Spectrum [5]. Here, we choose two classic parameter estimation methods, one is Radon–Wigner Transform (RW) [7, 19] based on time–frequency analysis, and the other is based on Fractional Fourier Transform (FrFT) [6, 15].

3.1 RW

For a chirp signal $y(t)$ with duration T, the Wigner Ville distribution (WVD) is:

$$\mathrm{WVD}(t,f) = \int_{-T/2}^{T/2} y(t+\tau/2)y^*(t-\tau/2)e^{-j2\pi f\tau}\mathrm{d}\tau \tag{7}$$

The $\mathrm{WVD}(t,f)$ can provide a high-resolution time–frequency representation of a mono-component chirp signal. In order to obtain signal parameters from time–frequency distribution information, we do Radon transform on $\mathrm{WVD}(t,f)$, which can be described as the following formula:

$$\mathrm{RW}(\rho,\theta) = \int \mathrm{WVD}(u\cos\theta - v\sin\theta, u\sin\theta + v\cos\theta)\mathrm{d}v \tag{8}$$

As shown in Fig. 2, the linear integration is performed along the line PQ at radius ρ and angle θ. Radon transform converts time–frequency domain distribution to $\rho-\theta$ domain distribution. Then we can extract linear component by searching the maximal amplitude of $\mathrm{RW}(\rho,\theta)$ at rotation angle θ' and integral radius ρ', which correspond to the initial frequency and chirp rate of $\hat{y}(n)$. Then we can get the estimation result as follows:

$$\{\rho',\theta'\} = \arg\max_{\rho,\theta}\{\mathrm{RW}(\rho,\theta)\} \tag{9}$$

$$\hat{k} = -\cot\theta' \tag{10}$$

$$\hat{f}_0 = \rho'/\sin\theta' \tag{11}$$

3.2 FrFT

The inverse FrFT of a finite signal $y(t)$ is defined as follows:

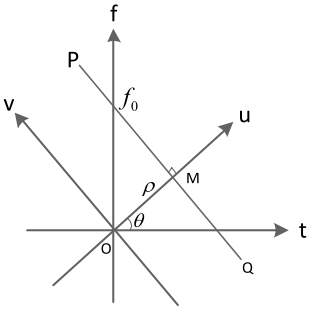

Fig. 2 The diagram of Radon transform

$$y(t) = \int_{-\infty}^{+\infty} Y_p(u) K_{-p}(t,u) \mathrm{d}t \tag{12}$$

where $Y_p(u)$ is the FrFT of $y(t)$, p is FrFT order, $K_p(t,u)$ is FrFT kernel function, and it is defined as:

$$K_p(t,u) = \begin{cases} \sqrt{\frac{1-j\cot\alpha}{2\pi}} \exp\left[j\left(\frac{u^2+t^2}{2}\cot\alpha - ut\csc\alpha\right)\right] & \alpha \neq n\pi \\ \delta(t-u) & \alpha = 2n\pi \\ \delta(t+u) & \alpha = (2n+1)\pi \end{cases} \tag{13}$$

where α is FrFT rotation angle, and $\alpha = p\pi/2$.

The above statement shows that $y(t)$ can be expressed as the expansion of inverse kernel $K_{-p}(t,u)$-based space. The expansion coefficients are $Y_p(u)$. And $K_{-p}(t,u)$ can be considered as a set of orthogonal basis of chirps. So the FrFT amplitude of a chirp signal at an appropriate order will be represented as a pulse. Gaussian noise does not have this property. In other words, FrFT have a good property of energy aggregation for chirp signal at corresponding order p. So, by searching the maximal amplitude of FrFT at u_0 and α_0, we can calculate chirp parameter as follows:

$$\{u_0, \alpha_0\} = \arg\max_{u,\alpha} \{abs(Y_p(u))\} \tag{14}$$

$$\hat{f}_0 = u_0 \csc\alpha_0 \tag{15}$$

$$\hat{k} = -\cot\alpha_0 \tag{16}$$

4 Simulation Results and Analysis

In this section, we present simulation results to demonstrate the performance of the proposed approach for denoising. The signal sampling frequency is 256 Hz, the duration is 2 s, so observation $Y^0 \in R^{513\times1}$. The midpoint of the signal represents the beginning of time. The initial frequency f_0 ranges from 15 to 20 Hz with 1 Hz interval. And the chirp rate k ranges from 5 to 10 Hz/s with 1 Hz/s interval. In order to make the experiment universal and comparable, we use the commonly used additive white Gaussian noise as the noise source. The SNR changes from − 13 to − 9 dB with approximately 1 dB intervals. Under the same SNR value, we randomly generate 800 training signals and 150 validation signals using the same signal parameters. So the training data set size is 144,000 signals and validation data set size is 27,000 signals.

We use test data set signals to verify the performance of the proposed method. The test signals have the same initial frequency and chirp rate as the training data set and validation data set. But the SNR changes from − 13 to − 3 dB with about

1 dB intervals. For any combination of initial frequency f_0 and chirp rate k, we randomly generate 10 signals under the same SNR condition. So the test data set size is 3960 signals.

In the process of training network, we use mini-batch gradient descent method to update DCNN parameters. The mini-batch size is 128. The maximum value of epoch is set to 8. So there are 8 full passes of the training algorithm over entire training set, and before each training epoch we shuffle the training data. This process is done by applying Matlab Deep Learning Toolbox.

We calculate the SNR of input observation signals and DCNN output signals separately, and the results are illustrated in Fig. 3. Obviously, it can be seen from the results that the SNR of denoised signal is greatly improved after DCNN filtering. Although we train DCNN under low SNR conditions, the network still has a good filtering performance at higher SNR with the range of -9 to -3 dB. We can also see that the SNR of denoised signal fluctuates greatly. This is because the strong noises are randomly distributed, which attenuate the signal structured features that DCNN can extract. Examples of using DCNN filtering are shown in Figs. 4 and 5, the RW and FrFT of the signal are shown in Figs. 6 and 7; from these figures, we can intuitively see that DCNN can effectively filter the unstructured noises and output highly structured signal.

Another issue we are concerned about after filtering is the parameters of the signal. We use RW and FrFT methods to estimate the parameters of observation and denoised signals, respectively. The performance metric is the root mean squared error (RMSE), which is defined as:

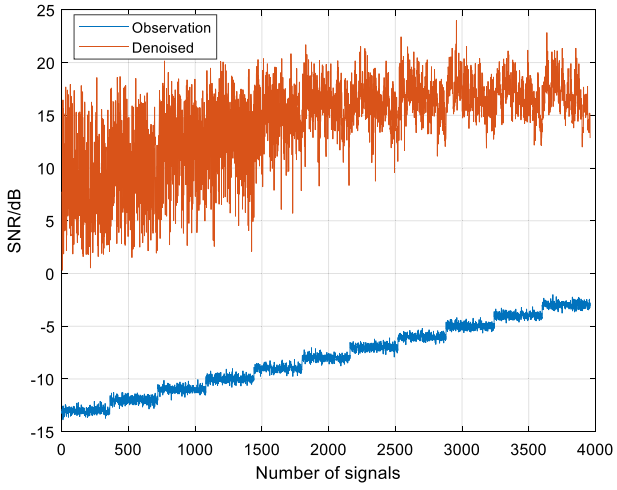

Fig. 3 Comparison of SNR between observation signals and denoised signals

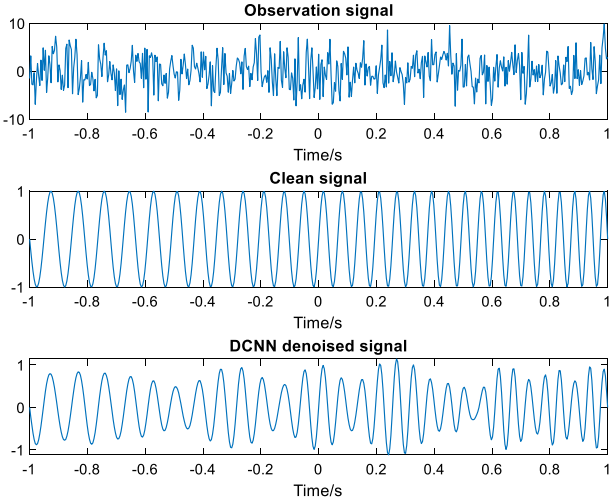

Fig. 4 DCNN denoise instance at SNR = − 13.3133 dB

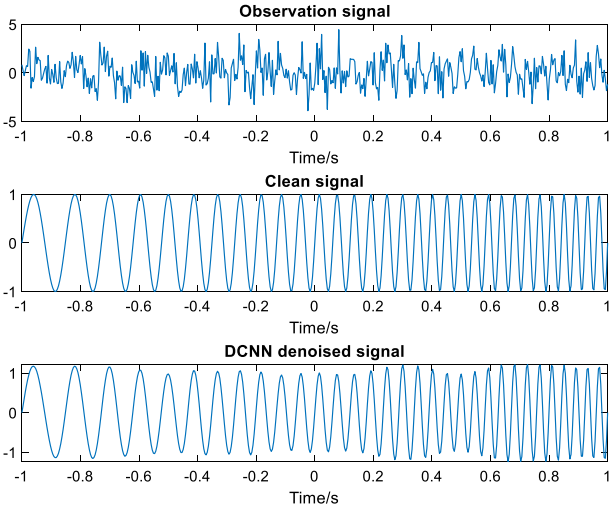

Fig. 5 DCNN denoise instance at SNR = − 4.8159 dB

$$\mathrm{RMSE}_{f_0} = \sqrt{\frac{1}{M}\sum_{i=1}^{M}\left(f'_{0_i} - f_{0_i}\right)^2} \tag{17}$$

$$\mathrm{RMSE}_{k} = \sqrt{\frac{1}{M}\sum_{i=1}^{M}\left(k'_{i} - k_{i}\right)^2} \tag{18}$$

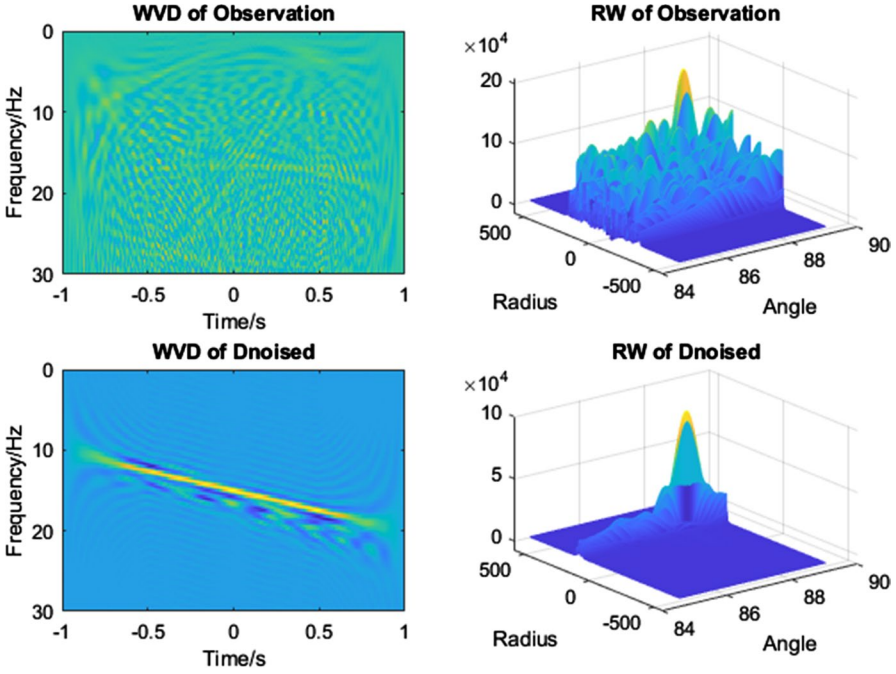

Fig. 6 WVD and RW of observation and denoised signal at SNR = − 13.3133 dB

where M is the number of signals under the same SNR in test set, f_{0_i} and k_i are the true value, and $f_{0_i}^{'}$ and $k_i^{'}$ are estimation results. The parameter estimation results by RW and FrFT are shown in Figs. 8 and 9. We can see that after DCNN processing, the parameter estimation accuracy has been significantly improved. One point to note is that different network structures and network parameters will have different effects on the filtering results. Due to the strong randomness of most noise, under the same SNR conditions, different training processes of the same network will have different results. But the trend of SNR and parameter estimation results of denoised signal will not change.

Parameter estimation error rate is defined by the following formula:

$$\mathrm{ER} = \frac{\text{number of estimation results greater than threshold}}{\text{number of total estimation results}} \times 100\% \qquad (19)$$

The threshold refers to the deviation between the estimation result and the true value. If the deviation is less than the threshold, it means that the estimation result is acceptable. We calculate error rate when threshold is equal to 0.1 and 0.25, respectively, and the result is shown in Figs. 10 and 11. But at low SNR circumstance, there are also considerable observation signals cannot be processed commendable and the error rate seems to be acceptable at SNR above − 10 dB.

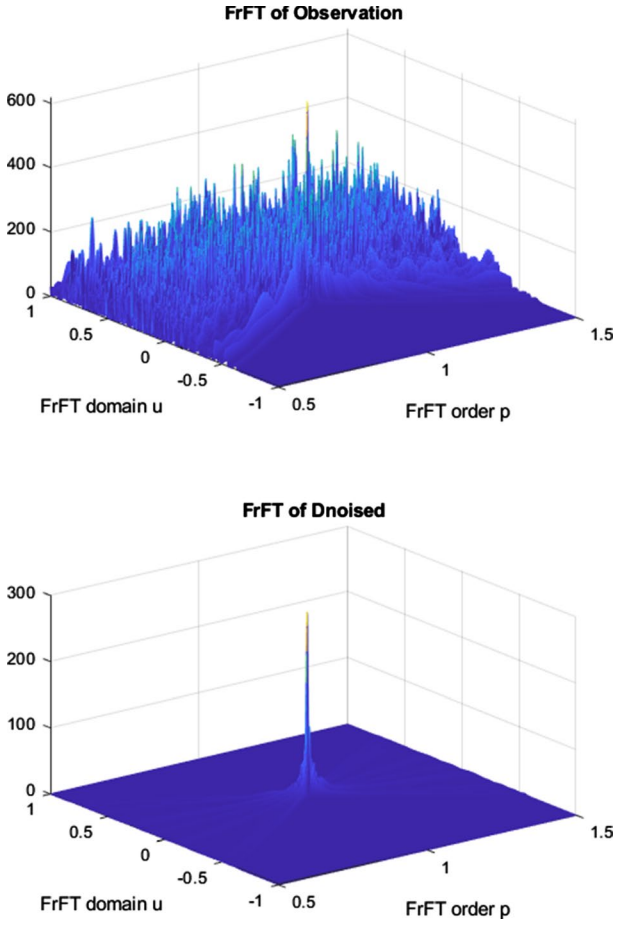

Fig. 7 FrFT of observation and denoised signal at SNR = − 13.3133 dB

5 Conclusion

In this paper, a DCNN-based method for extracting chirp signal in noise

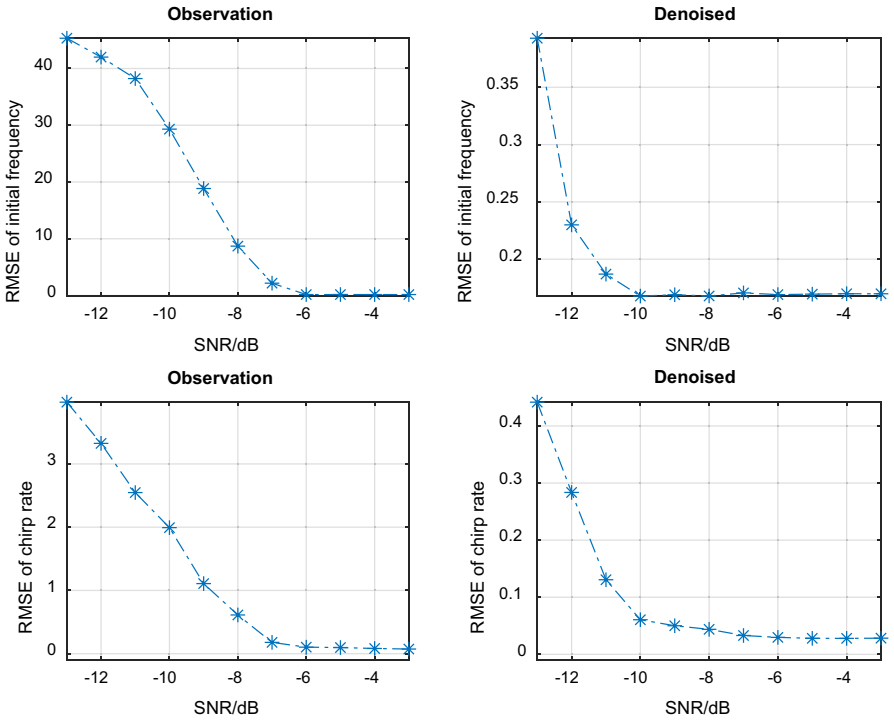

Fig. 8 RMSE of initial frequency and chirp rate estimated by RW

circumstances is proposed. It takes advantage of deep learning feature extraction ability to recover the clean chirp signals as far as possible. Simulation results indicate that DCNN have a good filtering performance at low SNR circumstances, which will help improve the accuracy of the parameter estimation algorithm. And although DCNN has learned the features of chirp signals at low SNR scenarios, it still works in high SNR conditions. But when the SNR is low enough, DCNN's performance seems to be unacceptable. In practice, in the various stages of the digital signal processing, whether the application of deep learning can play a role needs further research.

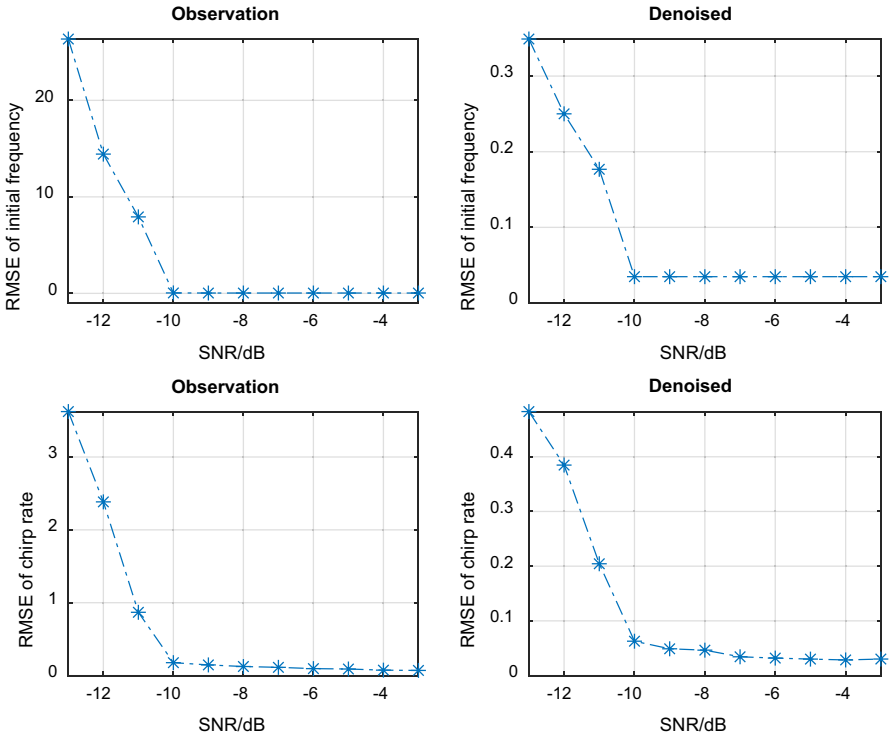

Fig. 9 RMSE of initial frequency and chirp rate estimated by FrFT

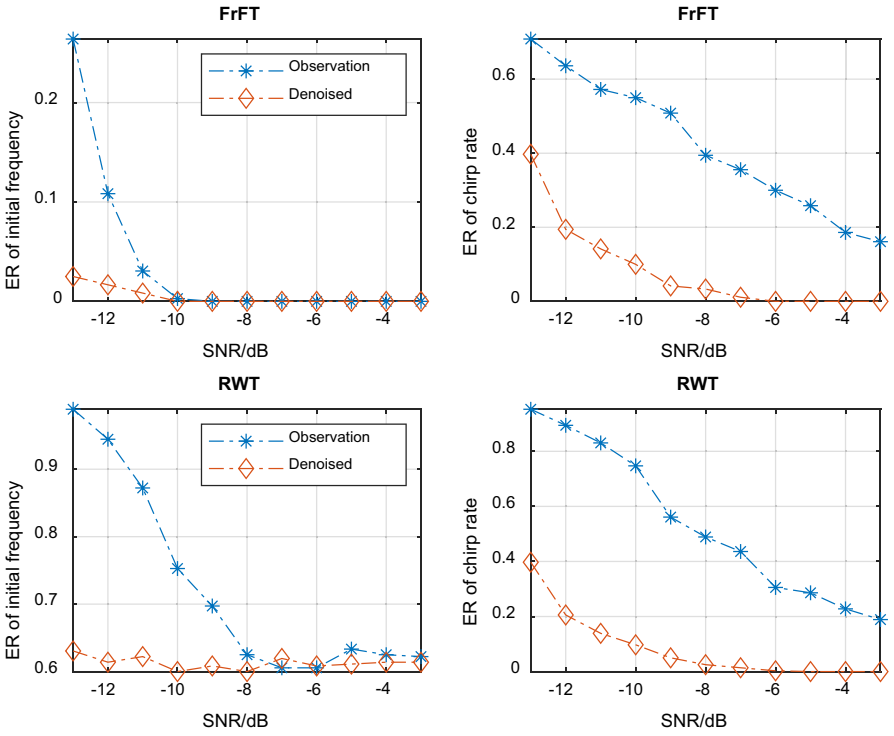

Fig. 10 Parameter estimation error rate at threshold = 0.1

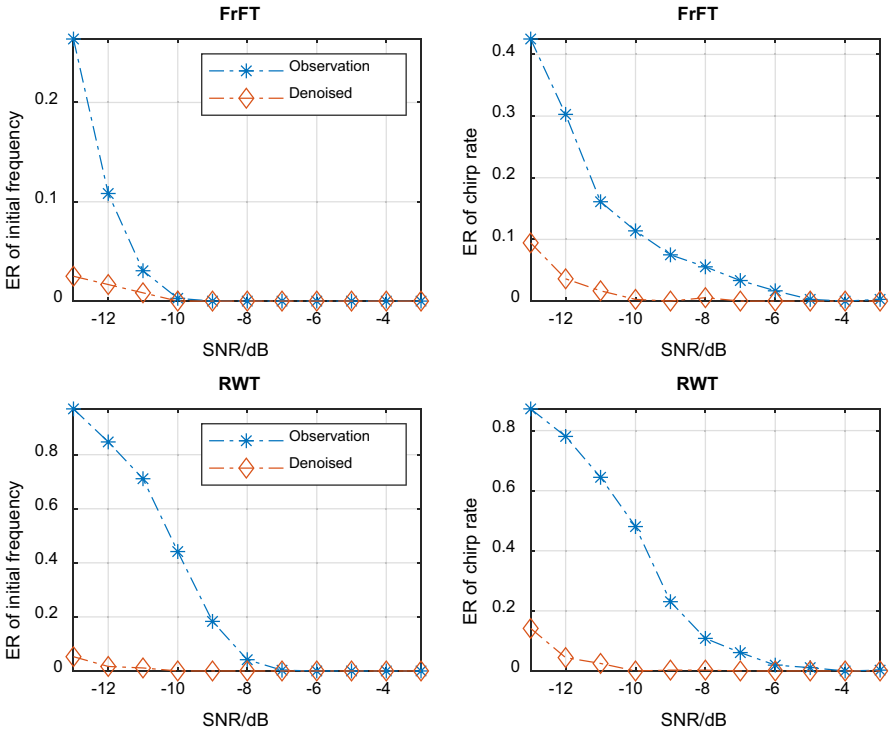

Fig. 11 Parameter estimation error rate at threshold = 0.25

Acknowledgements This manuscript has benefited greatly from the constructive comments and helpful suggestions of the anonymous referees; the authors would like to express their deep gratitude to them.

Funding Funding was provided by National Natural Science Foundation of China Grant 11703027.

Data Availability The datasets generated during and/or analyzed during the current study are available from the corresponding author on reasonable request.

Declarations

Conflict of interest The authors declare that they have no conflict of interest.

References

1. J. Brynolfsson, M. Sandsten, Classification of one-dimensional non-stationary signals using the Wigner–Ville distribution in convolutional neural networks, in *2017 25th European Signal Processing Conference (EUSIPCO)*, Kos, Greece (2017), pp. 326–330
2. X. Chen, Q. Jiang, N. Su, B. Chen, J. Guan, LFM signal detection and estimation based on deep convolutional neural network, in *2019 Asia-Pacific Signal and Information Processing Association Annual Summit and Conference (APSIPA ASC)*, Lanzhou, China (2019), pp. 753–758

3. K. Czarnecki, M. Moszyński, A novel method of local chirp-rate estimation of LFM chirp signals in the time-frequency domain, in *2013 36th International Conference on Telecommunications and Signal Processing (TSP),* Rome (2013), pp 704–708
4. K. Diederik, J. Ba, *Adam: A Method for Stochastic Optimization*. Computer Science (2014)
5. K. Ding, Q. Ding, R.Z. Lan, Parameters estimation of LFM signal based on fractional order cross spectrum, in *Proceedings of 2011 International Conference on Electronic & Mechanical Engineering and Information Technology*, Harbin (2011), pp. 654–656
6. Y. Ding, L. Sun, H. Zhang, B. Zheng, A multi-component LFM signal parameters estimation method using STFT and Zoom-FRFT, in *2016 8th IEEE International Conference on Communication Software and Networks (ICCSN)*, Beijing (2016), pp. 112–117
7. H. Hang, Time-frequency DOA estimation based on Radon–Wigner transform, in *2006 8th International Conference on Signal Processing*, Guilin, China (2006)
8. S. Ioffe, C. Szegedy, Batch normalization: accelerating deep network training by reducing internal covariate shift. JMLR.org (2015)
9. L.A.A. Irkhis, A.K. Shaw, Compressive chirp transform for estimation of chirp parameters, in *2019 27th European Signal Processing Conference (EUSIPCO)*, A Coruna, Spain (2019), pp. 1–5
10. Y. Jiang, H. Li, M. Rangaswamy, Deep learning denoising based line spectral estimation. IEEE Signal Process. Lett. **26**(99), 1573–1577 (2019)
11. H. Jin, W. Li, X. Wang, Y. Zhang, S. Yu, Q. Shi, Preamble detection for underwater acoustic communications based on convolutional neural networks, in *2018 OCEANS—MTS/IEEE Kobe Techno-Oceans (OTO)*, Kos, Greece (2018), pp. 1–6
12. Y. Lin, Y. Peng, X. Wang, Maximum likelihood parameter estimation of multiple chirp signals by a new Markov chain Monte Carlo approach, in *Proceedings of the 2004 IEEE Radar Conference (IEEE Cat. No.04CH37509)*, Philadelphia, PA, USA (2004), pp. 559–562
13. D. Liu, S. Paris, K. Minje, Experiments on deep learning for speech denoising, *in INTERSPEECH-2014* (2014), pp. 2685–2689
14. B. Liu, X. Tao, X. Qin, Y. Duan, J. Lu, Hyperspectral image denoising via nonnegative matrix factorization and convolutional neural networks, in *2018 IEEE International Geoscience and Remote Sensing Symposium(IGARSS 2018*) (IEEE, 2018)
15. V.A. Narayanan, K. Prabhu, The fractional Fourier transform: theory, implementation and error analysis. Microprocess. Microsyst. **27**(10), 511–521 (2003)
16. S. Saha, S. Kay, A noniterative maximum likelihood parameter estimator of superimposed chirp signals, in *2001 IEEE International Conference on Acoustics, Speech, and Signal Processing. Proceedings (Cat. No.01CH37221)*, Salt Lake City, UT, USA (2001), pp. 3109–3112
17. S. Saha, S.M. Kay, Maximum likelihood parameter estimation of superimposed chirps using Monte Carlo importance sampling. IEEE Trans. Signal Process. **50**(2), 224–230 (2002)
18. W. Shan, P. Liu, L. Mu, C. Cao, G. He, Hyperspectral image denoising with dual deep CNN. IEEE Access **7**, 171297–171312 (2019)
19. J.C. Wood, D.T. Barry, Radon transformation of time–frequency distributions for analysis of multicomponent signals. IEEE Trans. Signal Process. **42**(11), 3166–3177 (1994)
20. K. Zhang, W. Zuo, Y. Chen, D. Meng, L. Zhang, Beyond a Gaussian denoiser: residual learning of deep CNN for image denoising. IEEE Trans. Image Process. **26**(7), 3142–3155 (2016)
21. H. Zou, R. Lan, Y. Zhong, Z. Liu, X. Luo, EDCNN: a novel network for image denoising, *2019 IEEE International Conference on Image Processing (ICIP)*, Taipei, Taiwan (2019), pp. 1129–1133