Demonstrating superresolution in radar range estimation using a denoising autoencoder

Robert Czupryniak [1,2] Abhishek Chakraborty,[1,2] Andrew N. Jordan [2,3,4,1,*] and John C. Howell [2,3,†]
[1]*Department of Physics and Astronomy, University of Rochester, Rochester, New York 14627, USA*
[2]*Institute for Quantum Studies, Chapman University, Orange, California 92866, USA*
[3]*Schmid College of Science and Technology, Chapman University, Orange, California 92866, USA*
[4]*The Kennedy Chair in Physics, Chapman University, Orange, California 92866, USA*

(Received 21 June 2025; accepted 8 July 2026; published 10 August 2026)

We apply machine learning methods to demonstrate radar range superresolution using a denoising autoencoder trained without supervision. Focusing on the estimation of a single physical parameter, the separation between two scatterers in the subwavelength regime, we constrain the network to a one-dimensional bottleneck layer with its size matched to the parameter dimensionality. We find that the bottleneck layer forms a reproducible, monotonic mapping with the true separation, showing that the network learns a low-dimensional representation directly aligned with the underlying physical parameter. We further show that this representation preserves the Fisher information of the signal, indicating that the network recovers a physically meaningful representation for parameter estimation. We investigate the behavior of the bottleneck layer for the following types of pulses: a traditional sinc pulse, a bandlimited triangle-type pulse, and a theoretically near-optimal pulse created from a spherical Bessel function basis. Our results demonstrate that neural networks can discover physically meaningful compressed representations without explicit supervision, with performance ultimately governed by the information content of the signal.

DOI: 10.1103/w6m6-n6rk

I. INTRODUCTION

Recent discoveries in radar ranging [1–6] have shown that previous traditional bounds can be surpassed [7]. These previous bounds are based on the principle that in order to resolve two nearby scatterers, the temporal duration of the probing radar pulse must be smaller than the scatterer separation [5,8,9]. While many applications in radar could greatly benefit from breaking this trade-off—for example, when the medium imposes a bandlimit on EM wave propagation—to go beyond this limit was thought to be impossible.

The challenge is to make short distance range estimates with long-wavelength waves. However, it was recently shown that even in the context of classical wave theory, this bound could be beaten by orders of magnitude by using tailored pulses with a self-referencing estimation theory approach [7].

There are similarities to the breaking of the Rayleigh criterion in classical optical imaging systems using mode sorting techniques [10]; however, unlike those techniques, we deal with fully coherent wave physics and direct detection. Subsequent work developed bounds on this type of estimation technique using Fisher information methods by incorporating the technical noise added by the environment and amplifiers [11]. Optimal pulses were also derived using a bandlimit as a constraint and optimizing the Fisher information about the separation parameter, given prior knowledge of two scatterers [12].

While theoretical results and examples have also been worked out for a multiparameter scatterer system [11], it is of great interest to go to a more model-agnostic type of estimation method. As such, it is natural to consider machine learning (ML) approaches to this problem. It is of interest to see to what extent the ML methods can rank pulse shapes and even find new pulse shapes in order to estimate the scatterer separation under different types of constraints, as well as compare its performance with the previously mentioned fundamental bounds. While more general types of target recognition will be considered in the future, we focus here on the simplest case of balanced reflectors as a test bed for these methods.

It was recently shown that self-referencing waves, with local structure that can be distinguished using interferometric imaging, can provide subwavelength resolution below the Rayleigh limit [7,11]. This technique shows promise for subwavelength imaging, at the cost of a numerically intensive estimation algorithm due to the large dimensionality of the return signal. For complex signals, a linear dimensionality reduction technique such as principal component analysis [13] might be suboptimal for dimensionality reduction/data compression. Hence, this problem is well suited for machine-learning-based techniques.

Artificial neural networks (ANNs) and deep-learning-based approaches have shown great success in the field of image processing and computer vision, to the extent that most leading algorithms to solve the ImageNet Large Scale Visual Recognition Challenge [14] were based on deep-learning approaches. In particular, AlexNet [15], a deep convolutional neural network, surpassed previous heurisitic image-processing techniques based on manually designed

*Contact author: jordan@chapman.edu
†Contact author: johhowell@chapman.edu

Published by the American Physical Society under the terms of the Creative Commons Attribution 4.0 International license. Further distribution of this work must maintain attribution to the author(s) and the published article's title, journal citation, and DOI.

filters for detection of various features in images. Subsequently, a much deeper (152 layers) network (called ResNet [16]) featuring residual connections performed even better.

The success of these techniques can be attributed to the fact that the deep-learning techniques effectively offer an automated process to design feature-detection filters based on the training data. However, it becomes increasingly more difficult to attribute such a simple working principle to the success of more complex and deeper ANNs, thus making them seem like black boxes.

Representation learning [17], which aims to learn a lower dimensional representation of much higher dimensional data, has been used successfully for many tasks, including object detection [18], image segmentation [19], and superresolution imaging [20,21]. Such representations can be learned using a particular ANN architecture called autoencoders [22], which use an encoder-decoder structure with a bottleneck layer in between to learn a better representation of the data.

Autoencoders allow for a semisupervised/unsupervised learning approach to this problem, where the neural networks are trained to reconstruct the clean lower-dimensional representation of (potentially noisy) data, thus eliminating the need to manually assigned labels for classification/regression tasks. Instead, the representation learned in the bottleneck layer can be used as a label for further tasks downstream in the data-processing pipeline.

In this article, we examine/demonstrate the use of a denoising autoencoder [23] to solve the single-parameter radar range-resolution problem for a symmetric two-scatterer system. Leveraging the autoencoder, we evaluate the performance of three distinct radar pulse candidates within an unsupervised learning framework. Specifically, we analyze the autoencoder's denoising capabilities for various incoming, bandlimited signals. The signals we consider were designed to resolve separations below the inverse bandwidth of the signal; hence, we focus on that regime. For this problem, we show that the bottleneck layer activation is strongly correlated with the scatterer separation, thus demonstrating successful dimensionality reduction. Our technique is general and can be applied to other parameter estimation problems or candidate signals without any significant changes.

We show that the compressed representation learned in the bottleneck layer preserves the Fisher-information structure of the signal, indicating that the neural network acquires a physically meaningful internal representation. While earlier studies established that autoencoder latent variables can capture physically meaningful features of training data [24,25], here we connect this idea directly to the radar range-resolution problem. In addition, we identify an empirical law obeyed by the bottleneck-layer output.

In Sec. II, we review the radar range-resolution problem. Section III describes our deep-learning architecture and estimation procedure. Section IV reviews the three different candidate waves we compare for range resolution. Finally, in Sec. V, we describe our results.

II. RANGE RESOLUTION

Radar ranging is a special case of imaging using radio waves. In a one-dimensional setting (such as is valid in the far-field limit), given an object or scene represented by a function $O(x)$, which is illuminated using a point spread function $f(x)$, the image is given by a convolution of $O(x)$ and $f(x)$ [26,27]:

$$I(x) = O(x) * f(x) = \int_{-\infty}^{\infty} O(y) f(y - x) dy. \tag{1}$$

We consider the simple problem of estimating the separation between two equal-magnitude point scatterers in range resolution. The object function is represented by two delta-function spikes separated by a distance l. Let $f_0(x)$ be the outgoing signal to probe the scatterer separation l; the return signal has the form [26]

$$\begin{aligned} f^{(l)}(x) &= [f_0(x - l/2) + f_0(x + l/2)]/2 + \epsilon(x) \\ &= f_c(x) + \epsilon(x), \end{aligned} \tag{2}$$

where for convenience we set the origin at $l = 0$, $\epsilon(x)$ is the noise, and $f_c(x)$ denotes the noiseless return signal. $f^{(l)}(x)$ is then studied with the goal of estimating l. All of these signals are bandlimited and exist in finite range of frequencies $f \in [0, F]$, establishing a trade-off between the cutoff frequency and range-resolution abilities.

III. AUTOENCODERS

For this article, an autoencoder [22] is defined by a pair of functionals, $\mathcal{E} : \mathbb{R}^N \to \mathbb{R}^n$ and $\mathcal{D} : \mathbb{R}^n \to \mathbb{R}^N$, both represented by neural networks. The input and output of this architecture are both vectors $\mathbf{X} \in \mathbb{R}^N$. The output of the encoder is typically labeled $\mathbf{z} = \mathcal{E}[\mathbf{X}] \in \mathbb{R}^n$. For undercomplete autoencoders, $N > n$, related to a compression or dimensionality reduction of the data. This architecture can be trained in an unsupervised manner by minimizing the reconstruction error, given by the Euclidean distance (in $\mathbb{R}^N$) between the input and output, which is equal to the L_2 norm $\|\mathcal{D}[\mathcal{E}[\mathbf{X}]] - \mathbf{X}\|_2$. Once trained, the network is thought to have learned a compressed n-dimensional representation of the N-dimensional data, distilling the most important information about the input vector $\mathbf{X}$ into the compressed vector $\mathbf{z}$. Since artificial neural networks are universal function approximators, this can be thought of as a generalized nonlinear version of principal component analysis [13,27].

A. Denoising autoencoders

While regular autoencoders can effectively reconstruct noiseless signals for a variety of cases, the architecture can be adapted for denoising [23,28] tasks as well, leading to the so-called denoising autoencoder. In this variant, the encoder is fed a noisy input vector $\tilde{\mathbf{X}}$, obtained by adding a zero-mean noise vector $\boldsymbol{\xi} = \mathcal{N}_N(\mathbf{0}, \boldsymbol{\sigma}_\xi)$ to the "clean" vector $\mathbf{X}$. We use $\mathcal{N}_N(\mathbf{0}, \boldsymbol{\sigma}_\xi)$ to denote a vector with N elements, each of which is sampled from an independent, univariate normal distribution $\mathcal{N}(\mathbf{0}, \boldsymbol{\sigma}_\xi)$ with zero mean and $\boldsymbol{\sigma}_\xi$ standard deviation. The resulting noisy signal vector is [27]

$$\tilde{\mathbf{X}} = \mathbf{X} + \boldsymbol{\xi}. \tag{3}$$

The denoising autoencoder is then trained to reconstruct, from the noisy data, the clean vectors $\mathbf{X}$ by minimizing the modified reconstruction loss $\|\mathcal{D}[\mathcal{E}[\tilde{\mathbf{X}}]] - \mathbf{X}\|_2$. This architecture has

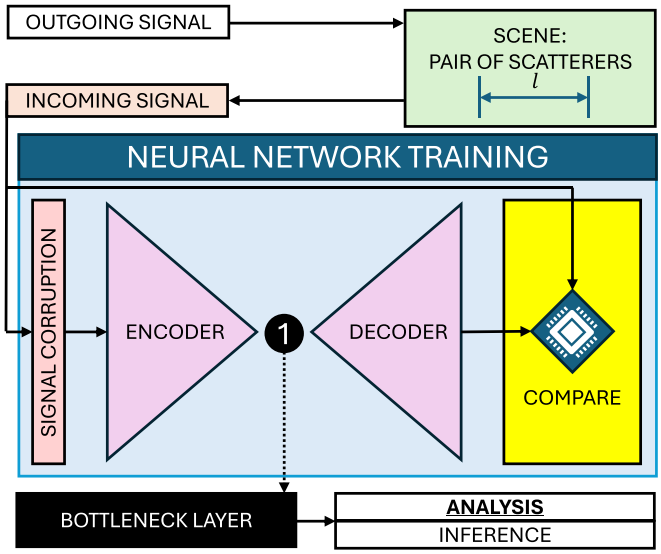


FIG. 1. Schematic overview of the autoencoder training process. The scene consists of a pair of scatterers, generating an incoming signal that is deliberately corrupted with white Gaussian noise before being processed by the neural network. The neural network's primary goal is to denoise the input and accurately reconstruct the original signal. The neural network under investigation is an autoencoder, which consists of two main components: an encoder and a decoder. The encoder learns a compressed representation of the input signal, referred to as the *bottleneck layer* (labeled, 1 neuron wide), while the decoder takes this compressed representation as input and attempts to reconstruct the original, noise-free signal. The objective of this paper is to examine the behavior of the bottleneck layer in trained autoencoders, with a focus on different types of outgoing signals.

been used for denoising images with good results [23]. While the techniques described below still apply for different kinds of noise, an extensive comparison of the effect of different noise spectra is beyond the scope of this article.

In the experimental setup, this noise model corresponds to adding white Gaussian detector noise to the incoming signal, similarly to the models considered in Refs. [7,12]. A more realistic model would include noise in the incoming signal arising from its propagation to the target scene and back to the detector. We leave this consideration beyond the scope of the present paper.

B. Autoencoder for radar-range resolution

In the remainder of this article, we will focus on the problem described in Sec. II. We consider a one-dimensional compressed representation ($n = 1$) since we are only concerned with a single-parameter estimation problem. For this setup, we expect that, once trained, the activation of the single bottleneck layer neuron z will be correlated with the separation parameter l. First, we examine this setup for noiseless signals to obtain the best-case scenario. Next, we corrupt the input signal with noise and train the denoising autoencoder to reconstruct the noiseless signal. We will compare the information learned by the bottleneck layer neuron in both cases. A schematic overview of this procedure is shown in Fig. 1.

For a given outgoing signal $f_0(x)$ and scatterer separation l, the noiseless incoming signal $f^{(l)}(x)$ is generated numerically using Eq. (2) and (optionally) corrupted by adding noise. The noiseless and noisy signals are then sampled and encoded in the vectors $\tilde{\mathbf{X}}^{(l)}, \mathbf{X}^{(l)} \in \mathbb{R}^N$, respectively. As a result, one can perform the estimation l using either the large vector $\mathbf{X}^{(l)}$ representing the return signal or the much smaller latent representation $\mathbf{z}^{(l)} = \mathcal{E}[\mathbf{X}^{(l)}]$. For instance, the trained encoder output can be used as input to another numerical or neural model to estimate l. Since the dimension of $\mathbf{z}^{(l)}$ is much smaller than the dimension of $\mathbf{X}^{(l)}$, the estimation process should be computationally less intensive when performed on the encoder output rather than the sampled signal vector. We focus solely on the analysis of bottleneck layer $\mathbf{z}^{(l)}$ activation, showing that it correlates with the separation l. An extensive analysis of parameter estimation by inverting this relationship is beyond the scope of this paper.

IV. OUTGOING SIGNALS

We will measure lengths in units of the inverse bandwidth $1/F$. We select three types of bandlimited outgoing signals for this analysis. First, we consider the sinc function raised to the mth power,

$$f_S(t) = \mathrm{sinc}\left(\frac{\omega t}{m\pi}\right)^m, \tag{4}$$

taken for $m = 10$ and $\omega = 2\pi$ and scaled to give bandwidth F. This function was considered as a *canvas* function to prepare superoscillating signals [29]. The second is the bandlimited *triangle signal* defined in Ref. [7]:

$$f_T(t) = p(t) f_S(t), \tag{5}$$

which takes $f_S(t)$ as an envelope and multiplies it by the polynomial

$$p(t) = 8t - 14.3984t^3 + 4.776\,12t^5 - 0.823\,15t^7. \tag{6}$$

While f_T should outperform f_S when used in single-parameter estimation [7], it is not the optimal signal for that purpose.

An approach to find such optimal signals based on orthogonal function theory was proposed in Ref. [30] and studied in Ref. [12], from which we take the following signal:

$$\begin{aligned} f_B(t) = {} & 0.259\,858\, j_0(\omega t) + 0.087\,993\,6\, j_2(\omega t) \\ & + 1.13614\, j_4(\omega t) - 0.136\,663\, j_6(\omega t) \\ & + 1.23652\, j_8(\omega t) - 0.185\,957\, j_{10}(\omega t) \\ & + 0.565\,418\, j_{12}(\omega t). \end{aligned} \tag{7}$$

$j_n(\cdot)$ denotes the nth spherical Bessel functions [31] taken for $\omega = 2\pi$. We plot the signals and their power spectrum in Fig. 2. We note that $f_B(t)$ is optimal, in principle, for an infinitely long waveform. Practically, we sample the incoming signal over a finite range and $f_B(t)$ will have a significant fraction of its power beyond it. Nevertheless, it is still expected to outperform the other two signals considered here. The incoming signals are studied for scatterer separations ranging from 0 to $1/F$ and sampled in the range $[-5, 5]1/F$, for the subresolved case. We downsample to $N = 1024$ samples from each signal and form the vector $\mathbf{X}^{(l)}$ out of it that is passed

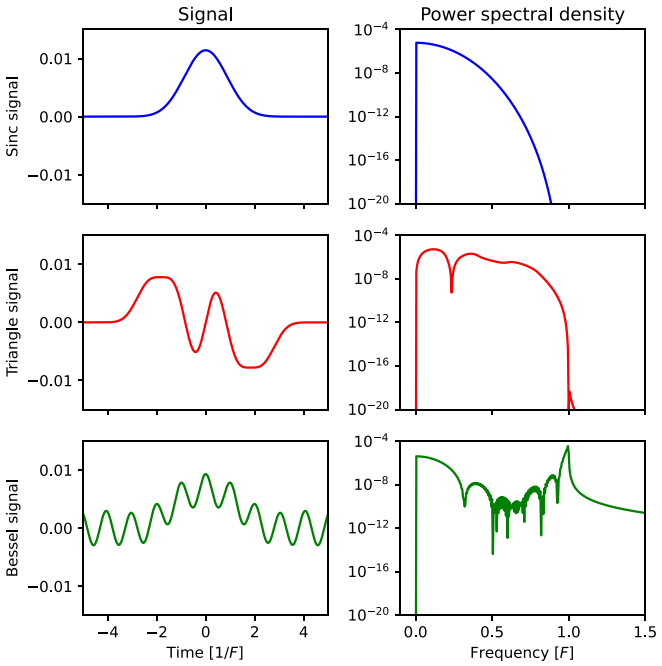


FIG. 2. Outgoing signals (left) and their corresponding power spectra (right), used to estimate scatterer separation. The three signals shown are defined in Eqs. (4), (5), and (7): the sinc pulse, the bandlimited triangle pulse, and the Bessel-based pulse. The high-frequency tail in the power spectrum of the Bessel signal arises from the finite sampling window, which is truncated at $t = \pm 5\,[1/F]$.

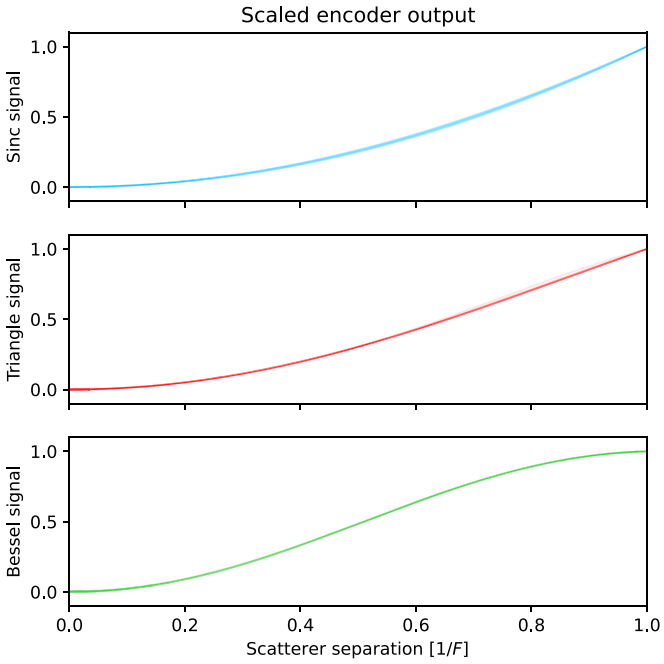


FIG. 3. Scaled encoder output [as defined in Eq. (8)] plotted as a function of scatterer separation for each outgoing signal type. Each panel shows results from 20 independently trained neural networks, represented by gray curves. The colored curve overlaid on each plot corresponds to a representative single network. The consistent, monotonically increasing behavior across networks indicates reliable and reproducible learning of scatterer separation by the autoencoder bottleneck layer. Note that the top plot contains one curve not matching the rest, indicating unsuccessful training.

through the neural networks. The uncorrupted signals are normalized to $\|\mathbf{X}^{(l)}\|_2 = 1$ before entering the neural networks.

The neural networks are trained separately for each signal type—sinc (f_S), triangle (f_T), and Bessel (f_B)—and for each considered signal-to-noise ratio (SNR): noiseless, 1/2, 2/3, 1, and 2. We calculate the SNR using the formula $\|\boldsymbol{\xi}\|_2/\|\mathbf{X}^{(0)}\|_2$, where $\|\cdot\|_2$ denotes the L_2 norm. To avoid the SNR's dependence on the separation parameter l, we use the norm of the outgoing signal $\|\mathbf{X}^{(0)}\|_2$, rather than that of the incoming signal $\|\mathbf{X}^{(l)}\|_2$.

All networks are initialized with random weights and biases. For networks trained on noisy signals, a new set of noise vectors $\boldsymbol{\xi}$ is generated in each training epoch and added to the signals. This procedure introduces inherent randomness into the optimization process. To ensure that our results are reproducible and not artifacts of random initialization or random noise generation, we independently train 20 identical autoencoder networks for each signal type and each signal-to-noise ratio. The details of the training procedure are provided in Appendix C.

V. RESULTS

A. Noiseless signal analysis

Due to inherent randomness of neural network training, a given signal can result in different bottleneck layer representations when forward passed through different instances of the trained neural networks, even when all networks have the the same architecture and we use the same training procedure. To make the results comparable across different networks, we first rescale the trained encoder output as follows.

We generate a test dataset of 256 incoming signals with scatterer separations between 0 and $1/F$. For each signal, we record the encoder output, resulting in the vector of encoder outputs $\mathbf{y}$ with entries $y_l = \mathcal{E}[\mathbf{X}^{(l)}]$. Let y_i and y_f be the initial and the final entry of $\mathbf{y}$. We obtain the scaled vector of encoder outputs from

$$\tilde{\mathbf{y}} = \frac{\mathbf{y} - y_i}{y_f - y_i}, \tag{8}$$

which fixes the end points at $\tilde{y}_i = 0$ and $\tilde{y}_f = 1$. This scales the output of the final layer in a similar manner as batch normalization layer [32]. The results are plotted in Fig. 3.

We observe that the curves representing the scaled encoder output as a function of scatterer separation have a roughly similar shape across all trained networks (see Fig. 3), indicating that the results are reproducible in different training sessions. All curves are monotonically increasing, making them suitable for parameter estimation. For each of them, one should be able to define a mapping from scaled encoder output to scatterer separation resulting in near-ideal estimation of l. We note that these statements hold even for a sinc signal (blue curves in Fig. 3) indicating that range resolution significantly below inverse bandwidth represents a simple task for neural networks in case of noiseless signals. Even though all three candidate signals are expected to perform similarly in the absence of noise, the advantage of f_T and f_B over f_S becomes evident for the corrupted signals.

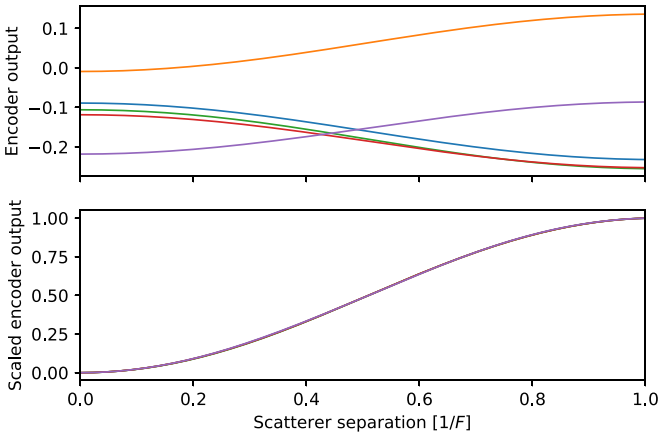


FIG. 4. Encoder outputs from five selected neural networks, all trained on Bessel signals. The top panel shows the raw (unscaled) encoder outputs as a function of scatterer separation, while the bottom panel presents the same outputs scaled according to Eq. (8). Each color corresponds to a different neural network and is consistent across both plots. The overlap of all curves in the bottom plot demonstrates the reproducibility and consistency achieved through scaling.

Remarkably, the autoencoders develop meaningful representations in the bottleneck layer despite not being provided with scatterer separation labels during training. Furthermore, assuming access to the outgoing signal, one can identify the label corresponding to $l = 0$ in the noiseless model via Eq. (2), since in this case the incoming and outgoing signals are identical. This enables qualitative comparisons across scenes, even without explicit knowledge of the underlying scatterer separations.

As an illustrative example, consider the task of comparing scatterer separations l_1 and l_2 in two distinct scenes. A forward pass of the outgoing signal through the encoder yields the reference output $y_0 = \mathcal{E}[\mathbf{X}^{(0)}]$, while the forward passes of the incoming signals yield $y_{l_{1(2)}} = \mathcal{E}[\mathbf{X}^{(l_{1(2)})}]$. Comparing the magnitudes $|y_{l_1} - y_0|$ and $|y_{l_2} - y_0|$ allows one to infer which scene corresponds to the greater separation.

Quantitative estimation of scatterer separation, however, is more challenging—if not infeasible—as it would require knowledge of the encoder output as an explicit function of separation.

B. Physical interpretation of the encoder output for clean signals

The curves shown in Fig. 3 demonstrate a stable and reproducible behavior of the bottleneck layer across different training sessions. Figure 3 demonstrates the curves after the scaling described by Eq. (8). The examples of raw encoder output before scaling are given in Fig. 4. Similar behavior is also observed for other encoder architectures, as discussed in the Appendix D.

This suggests that the autoencoders capture a fundamental feature of the signal and that an underlying physical relation governs the curves observed in Fig. 3. In this section, we aim to identify an empirical relation that describes these curves.

The sensitivity of a probability distribution to a change in an estimated parameter is quantified by the Fisher information (FI) [33,34]. Reference [11] adapted this concept to the subwavelength range-resolution problem considered in this article. For a discretized signal, the FI scales as

$$\mathcal{I}_X(l) \propto \sum_i \left(\frac{\partial [\mathcal{N}_l X_i(l)]}{\partial l} \right)^2 \propto \sum_i \left(\frac{\partial X_i(l)}{\partial l} \right)^2, \quad (9)$$

where the sum runs over all time samples of the signal, $X_i(l)$ denotes the noiseless incoming signal at time t_i, and $\mathcal{N}_l$ is a normalization factor for the incoming signal. The first proportionality follows from Ref. [11]. The second proportionality adapts this expression to the conditions of the present article. Here, the incoming signals are normalized by the norm of the outgoing signal, so that the noise added to the signal does not depend on the parameter l. As a result, $\mathcal{N}_l$ becomes a constant factor independent of l, and therefore does not affect the scaling of the Fisher information.

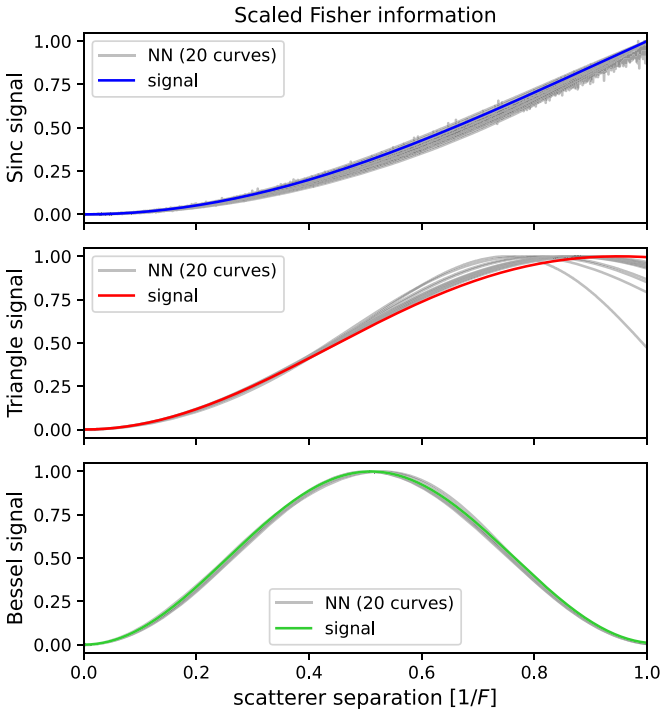


FIG. 5. Fisher information evaluated from Eq. (9) for the signals and from Eq. (10) for the trained neural networks. Each curve is normalized by its maximum value; i.e., we plot FI/ max(FI).

The encoder output serves as a compressed representation of the signal and therefore contains information about the parameter to be estimated. It can thus also be used to evaluate the Fisher information. Denote by $\mathcal{E}_l$ the encoder output produced by the signal $\mathbf{X}(l)$, i.e.,

$$\mathcal{E}_l = \mathcal{E}[\mathbf{X}(l)].$$

The corresponding Fisher information scales as

$$\mathcal{I}_\mathcal{E}(l) \propto \left(\frac{\partial \mathcal{E}_l}{\partial l} \right)^2. \quad (10)$$

The results obtained by computing Eq. (9) for the unprocessed signals and Eq. (10) for the trained neural networks are presented in Fig. 5. We emphasize that the curves are normalized by their maximum value; i.e., for each Fisher

information curve FI, we plot FI/ max(FI). This normalization allows the curves to be compared across different networks. For each neural network, the encoder outputs are effectively shifted and scaled versions of one another (see Fig. 4); without normalization, this scaling would also affect the corresponding Fisher-information curves.

Crucially, we observe excellent agreement between the Fisher-information curves computed from the encoder outputs and those computed from the raw signals. This indicates that the neural networks learn a physically meaningful representation of the signal described by Eqs. (9) and (10). We emphasize that this agreement constitutes a proof-of-principle demonstration for the signals considered in this article, while a more general analysis is left for future work.

C. Noisy signal analysis

In this section, we study the bottleneck layer of the denoising autoencoders trained to reconstruct a noiseless return signal from a noisy one. Note that each trained autoencoder corresponds to a specific candidate signal and SNR. For each network, we generate a dataset of 101 noiseless signals, each with a different scatterer separation ranging from $l = 0$ to $l = 1[1/F]$. For each value of scatterer separation, we generate 1000 corrupted signals, each of them with a different realizations of noise. These 1000 signals are then forward passed through the autoencoder to produce as many encoder outputs. We then compute the mean and standard deviation of these values. The procedure is repeated for all values of scatterer separation.

As a result, we obtain a vector of means $\mathbf{m}$ and standard deviations $\boldsymbol{\sigma}$ with entries corresponding to computation results for a given scatterer separation, i.e., $m_k = m(l_k)$, $\sigma_k = \sigma(l_k)$, in increasing order of scatterer separation l_k. Due to inherent randomness of deep learning, these vectors will be different for each neural network. Similar to noiseless signal analysis, we rescale the vectors $\mathbf{m}$ and $\boldsymbol{\sigma}$ to make the results comparable across different neural networks. Let us denote by i and f the initial and final elements of these vectors. We obtain the scaled vectors from

$$\tilde{\mathbf{m}} = \frac{\mathbf{m} - m_i}{m_f - m_i}, \qquad \tilde{\boldsymbol{\sigma}} = |(m_f - m_i)^{-1}|\boldsymbol{\sigma}. \tag{11}$$

$\tilde{\boldsymbol{\sigma}}$ can be understood as the spread of the scaled encoder output due to noise in the input, and it can be treated as a measure of how well the corruption process is reversed by the encoder. The results produced by the described analysis are given in Fig. 6.

D. Discussion—noisy signals

Observe that improving the quality of the outgoing signal leads to a substantial enhancement in the autoencoder's ability to reverse the corruption process. Across all considered signal-to-noise ratio values, the sinc signal, $f_S(t)$, exhibited the poorest performance, corresponding to the highest noise levels in the bottleneck layer. This result is expected, as $f_S(t)$ lacks distinguishing features that would enable it to resolve separations below the inverse bandwidth.

In contrast, the triangular signal, $f_T(t)$, introduced in Ref. [7] can be considered as an ansatz suitable for estimating

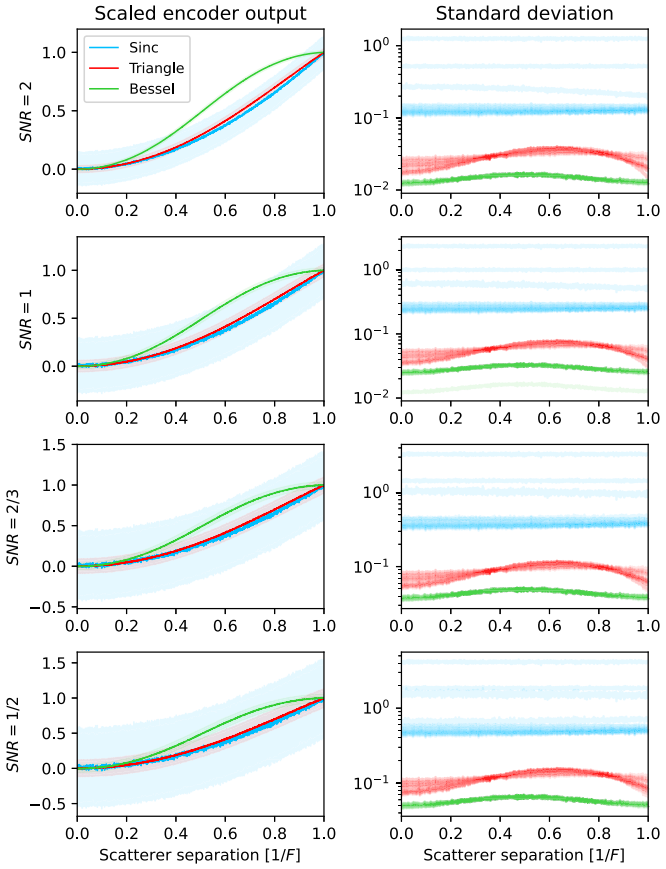


FIG. 6. Left column: Scaled mean encoder output (solid lines), along with ± one standard deviation (shaded region), plotted as a function of scatterer separation. The scaling is performed according to Eq. (11). For each separation value, 1000 noisy input signals were generated and passed through the autoencoder, and the mean and standard deviation of the scaled encoder output were computed. Each color represents a different type of incoming signal for a randomly selected autoencoder. Right column: Standard deviation of the scaled encoder output plotted for all trained networks (20 curves per signal type). The uncertainty is highest for the sinc signal, intermediate for the triangle signal, and lowest for the Bessel-based signal. Each row corresponds to autoencoders trained at a different SNR.

low scatterer separations. Indeed, autoencoders trained on incoming signals generated by $f_T(t)$ demonstrated significantly improved performance compared to those trained on sinc signals, as indicated by the red curves in Fig. 6.

The Bessel signal, $f_B(t)$, studied in Refs. [12,30], was explicitly designed for estimating small scatterer separations. Consequently, it was expected to outperform the other signals in this task. This expectation was confirmed, as autoencoders trained on incoming signals from $f_B(t)$ achieved the highest efficiency in noise reduction within the bottleneck layer.

The findings presented in this section suggest a hierarchical ranking of outgoing signals based on their effectiveness in estimating small scatterer separations. The sinc signal, $f_S(t)$, lacks any structural features beneficial for this task. The triangular signal, $f_T(t)$, serves as an ansatz that is anticipated to yield better results, while the Bessel signal, $f_B(t)$, is grounded in solid theoretical principles and is expected to deliver the best performance. This hierarchy is reflected in the perfor-

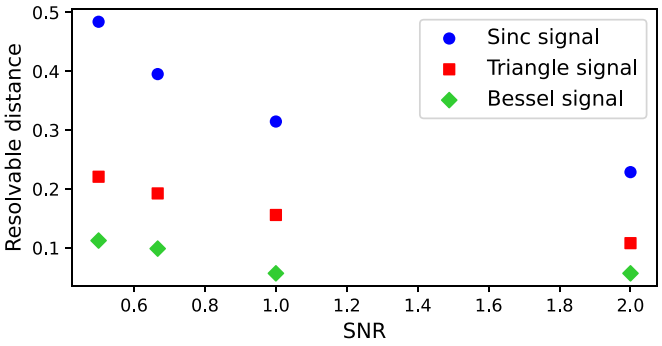


FIG. 7. Resolvable distance estimation as a function of the signal-to-noise ratio, obtained from the encoder-output statistics shown in Fig. 6 using Eq. (12).

mance of the trained autoencoders, as illustrated in the right column of Fig. 6.

It is crucial to emphasize that the only information provided to the autoencoders during training was the incoming signals. The autoencoders received no direct information about scatterer separation or signal-to-noise ratios. Remarkably, despite this limitation, the bottleneck layer consistently exhibited a structured response as a function of scatterer separation for both clean (Fig. 3) and corrupted signals (Fig. 6). Furthermore, the noise levels in the bottleneck layer remained consistent across different neural networks trained with the same outgoing signals and noise levels, indicating that the stochastic nature of neural network training does not compromise our conclusions. In the Appendix D, we demonstrate that these observations hold even when different encoder architectures are employed.

We now estimate the resolvable distance between scatterers based on the mean encoder output, i.e., the curves presented in the left column of Fig. 6. These curves represent the encoder output as a function of scatterer separation, denoted by $\mathcal{E}_l$, together with the corresponding standard deviation $\Delta\mathcal{E}_l$ arising from noise in the input.

To quantify resolution, we employ standard error propagation and define the uncertainty in the inferred separation as

$$\Delta l = \frac{1}{|d\mathcal{E}_l/dl|}\,\Delta\mathcal{E}_l. \tag{12}$$

We consider a separation l to be resolvable when $\Delta l = l$. While more stringent criteria could be adopted, this condition provides a consistent estimate of resolution across different signal types and noise levels.

A direct evaluation of Eq. (12) leads to an underestimation of the resolvable distance due to large numerical derivatives. This issue arises because the encoder-output curves in Fig. 6 exhibit small fluctuations originating from the noise in the incoming signal. To mitigate this effect, we smooth the function $\mathcal{E}_l$ prior to differentiation using a Savitzky-Golay filter [35] with window size 300 and polynomial order 3.

The resulting estimates of the resolvable distance are shown in Fig. 7. We observe a clear hierarchy consistent with previous findings: The sinc signal yields the poorest resolution, while the Bessel-function-based signal achieves the best

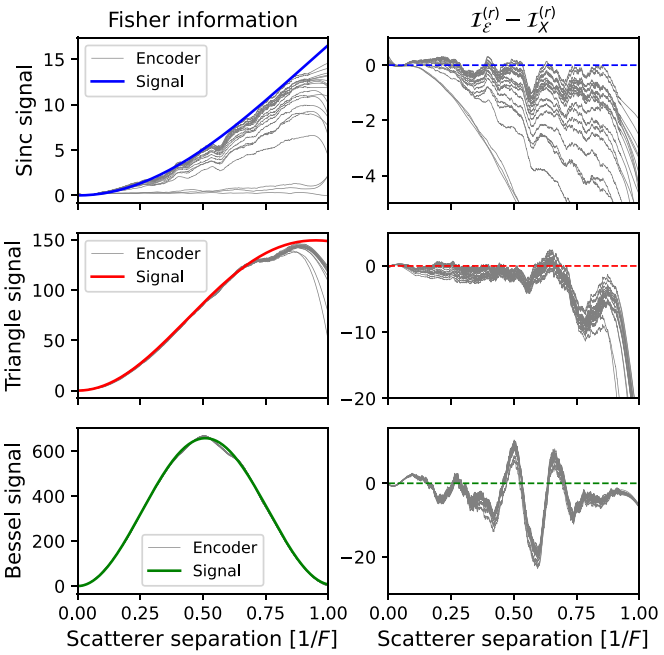


FIG. 8. Left column: Fisher information computed for the incoming signal (colored curves) and for the encoder output (gray curves). Right column: Difference between the encoder output and signal Fisher information, $\mathcal{I}_\mathcal{E}^{(r)} - \mathcal{I}_\mathbf{X}^{(r)}$. The dashed line indicates the threshold above which the encoder appears to outperform the raw signal. Each gray curve corresponds to one of the 20 independently trained neural networks. The encoder curves are obtained using the filtering procedure described in Sec. V E.

performance, enabling resolution beyond the Rayleigh limit even at relatively high noise levels.

E. Fisher information for noisy signals

We now compare the Fisher information of the raw, unprocessed incoming signal $\mathbf{X}^{(l)}$ with that inferred from the encoder output, based on the statistics shown in Fig. 6. We parametrize the noise strength by the ratio of the noise and signal norms,

$$r = \|\boldsymbol{\xi}\|_2/\|\mathbf{X}^{(0)}\|_2. \tag{13}$$

For a signal corrupted by noise characterized by r, the Fisher information associated with the incoming signal is given by [11]

$$\mathcal{I}_\mathbf{X}^{(r)}(l) = \frac{N}{r^2}\sum_i \left(\frac{\partial[\mathcal{N}_l\, X_i(l)]}{\partial l}\right)^2, \tag{14}$$

where N denotes the number of temporal samples, i.e., the dimensionality of the signal $\mathbf{X}^{(l)}$.

The encoder output $\mathcal{E}_l$ provides an implicit estimator of the parameter l. Its associated Fisher information can be expressed as

$$\mathcal{I}_\mathcal{E}(l) = \frac{1}{(\Delta\mathcal{E}_l)^2}\left(\frac{d\mathcal{E}_l}{dl}\right)^2, \tag{15}$$

where $\Delta\mathcal{E}_l$ denotes the standard deviation of the encoder output induced by noise.

Both the derivative $d\mathcal{E}_l/dl$ and the variance $\Delta\mathcal{E}_l$ are estimated from finite samples of noisy inputs, and therefore exhibit fluctuations. To obtain stable estimates, we follow the same procedure as in the previous section: The function $\mathcal{E}_l$ is first smoothed using a Savitzky-Golay filter prior to differentiation, and the resulting Fisher-information curves are subsequently smoothed to suppress residual irregularities. The resulting estimates are shown in Fig. 8.

We find that, in the vast majority of cases, the Fisher information of the raw signal exceeds that of the encoder output. This behavior is consistent with the expectation that a compression procedure cannot increase the information about the parameter encoded in the signal. In particular, it is in qualitative agreement with the data-processing inequality, which implies that any deterministic mapping (such as the encoder) cannot increase the Fisher information under appropriate regularity conditions.

At the same time, we observe localized regions in parameter space where the encoder Fisher information appears to exceed that of the raw signal. However, we do not consider these regions as conclusive evidence of a genuine information gain, as the precise shape of these regions is sensitive to the filtering protocol and the statistical noise in the estimates. A more detailed and rigorous analysis of these effects is left for future work, while here we aim only to present the main features of the autoencoder internal representation.

Overall, our results indicate that the encoder preserves the Fisher information of the incoming signal up to numerical effects, further supporting the conclusion that the neural network learns a physically meaningful compressed representation.

VI. CONCLUSIONS AND OUTLOOK

We have demonstrated a proof of concept that autoencoders (specifically denoising autoencoders) can be used for parameter estimation in the radar range-resolution problem.

The results of our ML simulations showed that the bottleneck layer of the autoencoder develops a useful representation for the parameter estimation problem. If the network is trained on clean signals and one has access to scatterer separations as labels, it is to possible establish a relationship of the encoder output as a function of scatterer separation, enabling quantitative estimates. Even without access to labels, the bottleneck layer still permits qualitative analysis of different scenes, for example, allowing comparison of scatterer separations between scenes.

Interestingly, different autoencoders trained on a given signal develop similar representations in the bottleneck layer, despite the inherent randomness of the ML method. We have shown that the bottleneck representations preserve the Fisher information of the incoming signal, indicating that the neural network learns its physically meaningful compressed representation. This similarity appears not only in the shape of the encoder output as a function of scatterer separation, but also in how the encoder output responds to noise in the incoming signal. These findings are consistent across different encoder architectures, as demonstrated in the Appendix D.

We have verified how the bottleneck layer is affected by noise depending on the incoming signal. As expected, the network exhibits less noise as we increase the quality of the incoming signal. To verify it, we compared three different signals: one based on a sinc wave to the *m*th power, a bandlimited *triangle wave*, and a signal constructed from Bessel functions, which is theoretically near-optimal for the single-parameter problem examined here. The Bessel signal performs best, followed by the triangle wave, with the sinc signal performing worst. Our results demonstrate that improving the quality of the outgoing signal enhances the correlation between bottleneck layer activations and the scatterer separation in autoencoder-based analysis. This suggests that in addressing range resolution beyond the inverse bandwidth, optimizing neural network architectures alone is insufficient; poorly designed outgoing signals can significantly degrade network performance.

ACKNOWLEDGMENTS

This work was supported by the Air Force Office of Scientific Research under Award No. FA9550-24-1-0329. We thank Achim Kempf, Barbara Šoda, Nick Vamivakas, Shunxing Zhang, and Derek White for helpful discussions.

DATA AVAILABILITY

The source code used to generate the results presented in this paper is available from the corresponding author upon reasonable request.

APPENDIX A: AUTOENCODER ARCHITECTURE

The architecture described in Table I is used for all neural networks responsible for generating the results presented in the main body of the paper. In the Appendix D, we explore two additional encoder variants for comparison.

The first alternative encoder consists of a sequence of convolutional layers, as detailed in Table II. This architecture preserves the same decoder structure used in the main model.

The second alternative encoder begins with a custom Fourier transform layer followed by a low-pass filter. The input is a 1024-element vector representing the sampled

TABLE I. Architecture of the autoencoder used with the linear encoder used to generate results in the main body of this article. The encoder is based on linear layers, and the decoder consists of two fully connected layers. B denotes the batch size. We take $B = 1$ in all cases in this article.

	Layer	Output size
Encoder	Flatten	$(B, 1024)$
	Linear + GELU	$(B, 256)$
	Linear + GELU	$(B, 64)$
	Linear + GELU	$(B, 8)$
	Linear	$(B, 1)$
Decoder	Linear + Tanh	$(B, 256)$
	Linear	$(B, 1024)$
	Reshape	$(B, 1, 1024)$

TABLE II. Architecture of the convolutional encoder. B indicates the batch size.

Layer	Output size
Conv1d, input channels = 32, output channels = 32, kernel size = 64	$(B, 1, 32, 961)$
MaxPool1d, kernel size = 4 + GELU	$(B, 1, 32, 240)$
Conv1d, input channels = 32, output channels = 32, kernel size = 32	$(B, 1, 32, 209)$
MaxPool1d, kernel size = 4 + GELU	$(B, 1, 32, 52)$
Conv1d, input channels = 32, output channels = 32, kernel size = 16	$(B, 1, 32, 37)$
MaxPool1d, kernel size = 4 + GELU	$(B, 1, 32, 9)$
Conv1d, input channels = 32, output channels = 32, kernel size = 9	$(B, 1, 32, 1)$
Flatten + GELU	$(B, 32)$
Linear	$(B, 1)$

signal. We compute its Fourier transform and retain only the frequency components below the inverse bandwidth. For the signal parameters considered in this work, this yields 11 frequency components. We extract both the real and imaginary parts of these components and concatenate them, producing a 22-element feature vector as the layer's output.

This 22-dimensional representation is then passed through a series of linear layers, forming the full encoder architecture described in Table III. The same decoder is used as in the main architecture.

All networks were implemented and trained using PyTorch v2.7.0.

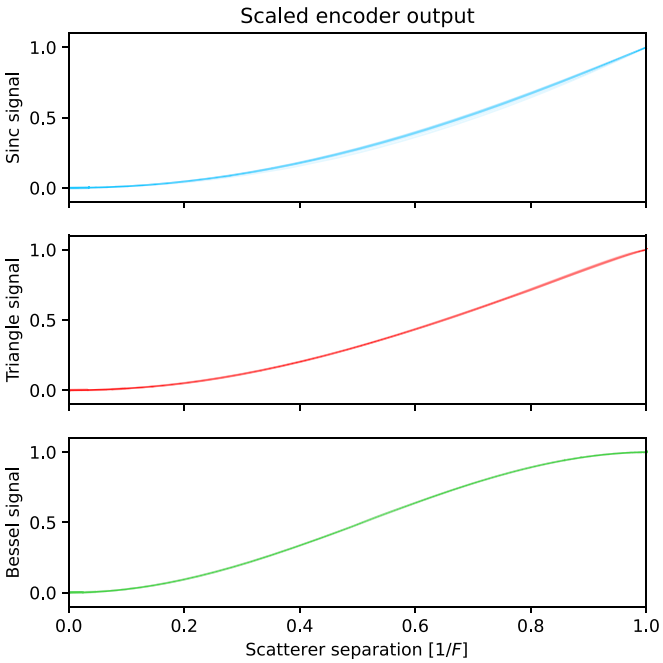


FIG. 9. Encoder output scaled according to Eq. (8) as a function of scatterer separation, plotted for the convolutional encoder architecture defined in Table II. Each panel corresponds to a different outgoing signal type. There are 20 curves plotted for each trained network.

APPENDIX B: ENCODER OUTPUT WITHOUT SCALING

Due to the inherent randomness of neural network training, different models—even with identical architectures and train-

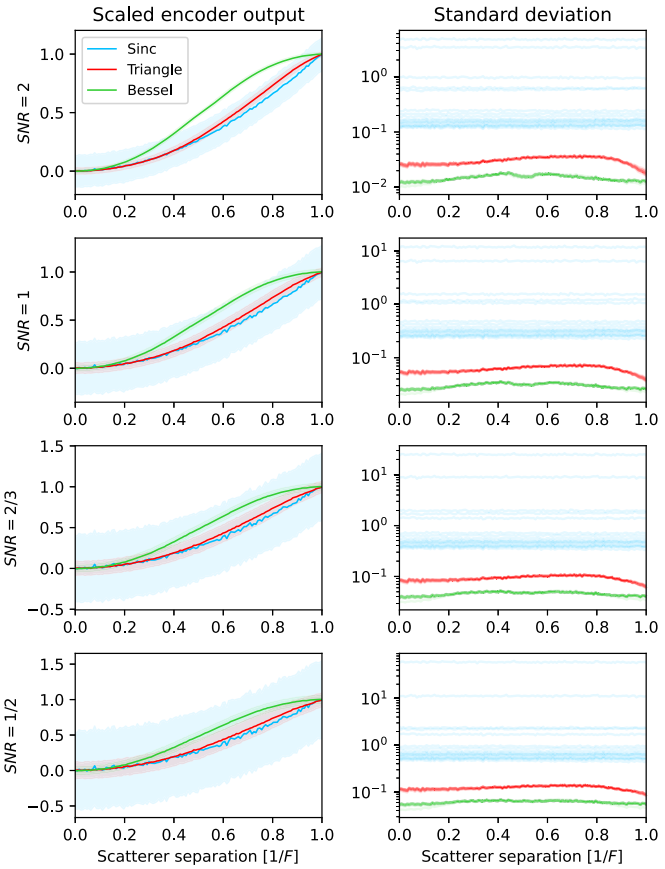


FIG. 10. Results for the convolutional encoder architecture defined in Table II, plotted in the same format as Fig. 6. Left column: Scaled mean encoder output (solid lines) ± one standard deviation (shaded region) as a function of scatterer separation, shown for a representative model. Each color represents a different type of outgoing signal. Right column: Standard deviation of the scaled encoder output plotted for all 20 independently trained networks, with each curve corresponding to a different training instance. Scaling was performed according to Eq. (11). Each row corresponds to a different SNR.

TABLE III. The architecture of the encoder starting with the Fourier transform layer. The same decoder architecture is used as in Table I. B indicates the batch size.

Layer	Output size
Fourier transform and low-pass filter	$(B, 1, 22)$
Flatten	$(B, 22)$
Linear + GELU	$(B, 22)$
Linear + GELU	$(B, 22)$
Linear + GELU	$(B, 22)$
Linear + GELU	$(B, 22)$
Linear	$(B, 1)$

ing procedures—can produce different representations in the bottleneck layer as a function of scatterer separation.

However, a reproducible structure in the encoder output emerges when scaling is applied. Figure 4 illustrates the difference in bottleneck behavior with and without scaling, showing that the rescaling procedure enables meaningful comparisons across independently trained networks.

APPENDIX C: TRAINING OF THE AUTOENCODERS

All considered models were trained for 5000 epochs. The training and validation datasets each consist of 1024 samples. The test dataset contains 2048 signals for networks described in Tables I and III, and 128 signals for networks described in Table II, where the smaller test size was chosen due to significantly longer inference time for convolutional networks. We deliberately chose datasets of comparable size because the training signals can be generated numerically with negligible cost, and using similarly sized datasets simplifies comparison of the loss curves across models. Minibatching was not used, as we did not observe any improvement in training performance after introducing it. The training and validation datasets were used to monitor the evolution of the training metrics.

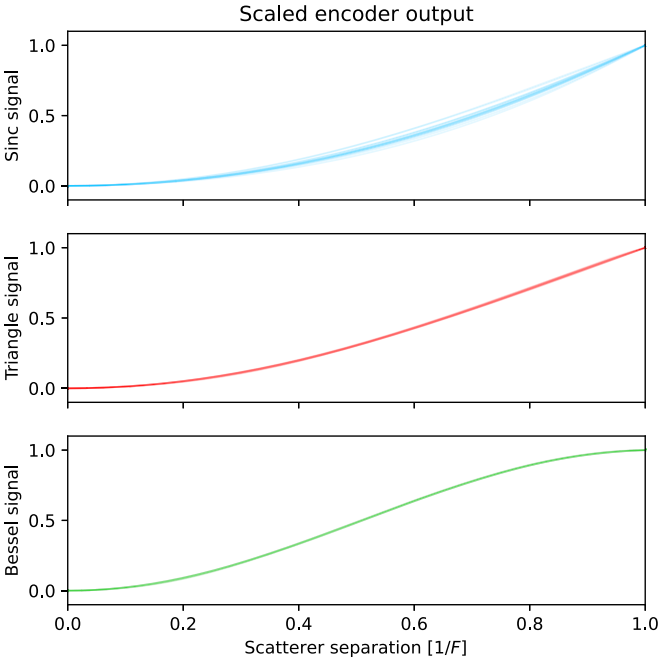


FIG. 11. Encoder output scaled by Eq. (8) as a function of scatterer separation, produced using the Fourier-based encoder architecture defined in Table III. Each panel corresponds to a different outgoing signal type. There are 20 curves plotted for each trained network.

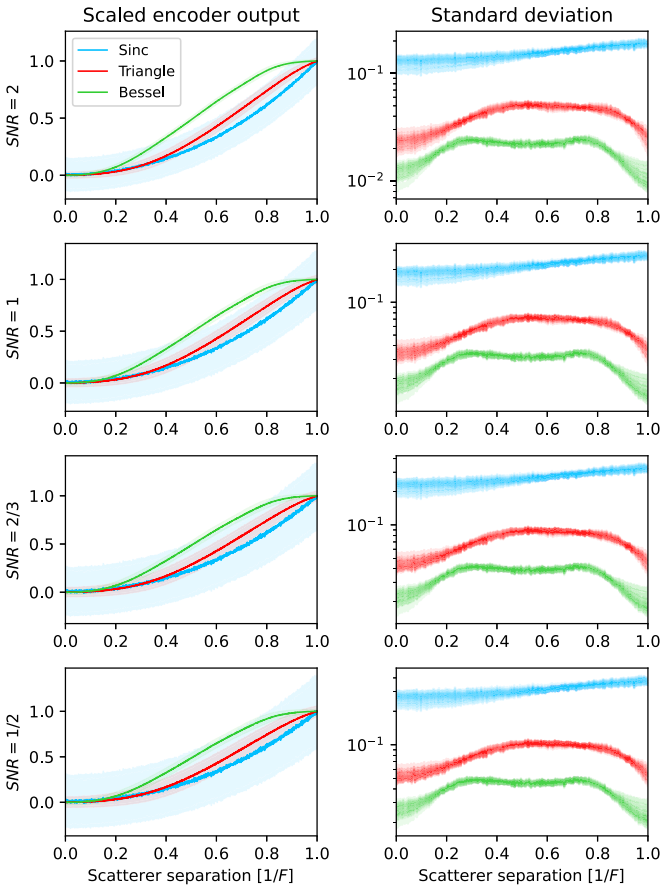


FIG. 12. Results for the Fourier-based encoder architecture defined in Table III, shown in the same format as Fig. 6. Left column: Scaled mean encoder output (solid lines) ± one standard deviation (shaded region) as a function of scatterer separation, for a representative model. Each color corresponds to a different outgoing signal type. Right column: Standard deviation of the scaled encoder output plotted for all 20 independently trained networks per signal type. Scaling is performed according to Eq. (11). Each row corresponds to a different SNR. The observed trends—including the performance hierarchy (Bessel, triangle, sinc)—are consistent with results from the larger architectures, confirming the effectiveness of the Fourier-based encoder.

While training networks on noiseless signals, we did not observe significant improvement after increasing the training set size to 10 000 or 100 000 samples. This behavior is expected because the signals effectively encode a single parameter—the scatterer separation. Once the allowed range of this parameter is sampled densely enough within the training dataset, the networks are able to capture all relevant signal features.

For networks trained to remove noise from corrupted signals, the corrupted signals in the validation and test datasets are kept fixed across epochs. During training, however, a

new noise realization is generated in each epoch and added to the clean signals. This procedure effectively increases the diversity of the training data. Over the course of training, the network is therefore exposed to about 5×10^6 distinct signals, corresponding to the number of epochs multiplied by the number of clean signals in the training set.

The training criterion is the mean squared error, and the optimization is performed using the Adam optimizer [36] with a learning rate of 0.001, keeping the default PyTorch parameters.

APPENDIX D: DEPENDENCE OF THE RESULTS ON NETWORK ARCHITECTURE

The results presented in the main text are based on a linear autoencoder architecture, as outlined in Table I. In this Appendix, we assess whether the key findings—specifically the reproducibility of bottleneck layer behavior—are robust to changes in the encoder architecture. We first examine a convolutional encoder, as detailed in Table II. The corresponding results are shown in Figs. 9 and 10. For clean signals, the scaled encoder output exhibits the same monotonic and reproducible trends as observed in Fig. 3. For noisy signals, a hierarchy in performance persists: The Bessel-based signal consistently yields the lowest standard deviation in the bottleneck output, followed by the triangle and sinc signals, mirroring the behavior reported in Fig. 10. Next, we consider a more compact architecture based on a Fourier transform and low-pass filtering, as described in Table III. The corresponding results are presented in Figs. 11 and 12. A similar behavior of bottleneck layer is reproduced to the one reported for other architectures.

[1] D. R. Wehner, *High Resolution Radar* (Artech House, Norwood, MA, 1987).

[2] J. Li, G. Liu, N. Jiang, and P. Stoica, Moving target feature extraction for airborne high-range resolution phased-array radar, IEEE Trans. Signal Process. **49**, 277 (2001).

[3] K. B. Cooper, R. J. Dengler, G. Chattopadhyay, E. Schlecht, J. Gill, A. Skalare, I. Mehdi, and P. H. Siegel, A high-resolution imaging radar at 580 GHz, IEEE Microwave Wireless Compon. Lett. **18**, 64 (2008).

[4] F. Zhang, Q. Guo, and S. Pan, Photonics-based real-time ultra-high-range-resolution radar with broadband signal generation and processing, Sci. Rep. **7**, 13848 (2017).

[5] M. I. Skolnik, *Introduction to Radar Systems*, 2nd ed. (McGraw-Hill, New York, 1980).

[6] N. Levanon and E. Mozeson, *Radar Signals*, 1st ed. (John Wiley & Sons, Hoboken, NJ, 2004).

[7] J. C. Howell, A. N. Jordan, B. Šoda, and A. Kempf, Super interferometric range resolution, Phys. Rev. Lett. **131**, 053803 (2023).

[8] R. E. Sheriff, Limitations on resolution of seismic reflections and geologic detail derivable from them, in *Seismic Stratigraphy—Applications to Hydrocarbon Exploration*, edited by C. E. Payton (American Association of Petroleum Geologists, 1977), Vol. 26, pp. 3–14.

[9] A. Neal, Ground-penetrating radar and its use in sedimentology: Principles, problems and progress, Earth Sci. Rev. **66**, 261 (2004).

[10] M. Tsang, R. Nair, and X.-M. Lu, Quantum theory of superresolution for two incoherent optical point sources, Phys. Rev. X **6**, 031033 (2016).

[11] A. N. Jordan and J. C. Howell, Fundamental limits on subwavelength range resolution, Phys. Rev. Appl. **20**, 064046 (2023).

[12] A. N. Jordan, J. C. Howell, A. Kempf, S. Zhang, and D. White, Optimal radar ranging pulse to resolve two reflectors, Phys. Rev. Res. **6**, 033341 (2024).

[13] I. T. Jolliffe, *Principal Component Analysis*, 2nd ed., Springer Series in Statistics (Springer, New York, 2002).

[14] O. Russakovsky *et al.*, ImageNet large scale visual recognition challenge, Int. J. Comput. Vision **115**, 211 (2015).

[15] A. Krizhevsky, I. Sutskever, and G. E. Hinton, ImageNet classification with deep convolutional neural networks, Commun. ACM **60**, 84 (2017).

[16] K. He, X. Zhang, S. Ren, and J. Sun, Deep residual learning for image recognition, in *2016 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)* (IEEE, Las Vegas, NV, 2016), pp. 770–778.

[17] Y. Bengio, A. Courville, and P. Vincent, Representation learning: A review and new perspectives, IEEE Trans. Pattern Anal. Mach. Intell. **35**, 1798 (2018).

[18] R. Kaur and S. Singh, A comprehensive review of object detection with deep learning, Digit. Signal Process. **132**, 103812 (2023).

[19] S. Minaee, Y. Boykov, F. Porikli, A. Plaza, N. Kehtarnavaz, and D. Terzopoulos, Image segmentation using deep learning: A survey, IEEE Trans. Pattern Anal. Mach. Intell. **44**, 3523 (2022).

[20] C. Ledig, L. Theis, F. Huszar, J. Caballero, A. Cunningham, A. Acosta, A. Aitken, A. Tejani, J. Totz, Z. Wang, and W. Shi, Photo-realistic single image super-resolution using a generative adversarial network, in *2017 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)* (IEEE Computer Society, Los Alamitos, CA, 2017), pp. 105–114.

[21] X. Wang, K. Yu, S. Wu, J. Gu, Y. Liu, C. Dong, Y. Qiao, and C. Loy, ESRGAN: Enhanced super-resolution generative adversarial networks, in *Computer Vision—ECCV 2018 Workshops*, edited by L. Leal-Taixé and S. Roth (Springer International Publishing, Cham, 2019), pp. 63–79.

[22] D. E. Rumelhart, G. E. Hinton, and R. J. Williams, Learning internal representations by error propagation, in *Parallel Distributed Processing: Explorations in the Microstructure of Cognition, Vol. 1: Foundations* (MIT Press, Cambridge, MA, 1986), pp. 318–362.

[23] P. Vincent, H. Larochelle, Y. Bengio, and P.-A. Manzagol, Extracting and composing robust features with denoising autoencoders, in *Proceedings of the 25th International Conference*

on Machine Learning, *ICML '08* (Association for Computing Machinery, New York, 2008), pp. 1096–1103.

[24] S. J. Wetzel, Unsupervised learning of phase transitions: From principal component analysis to variational autoencoders, Phys. Rev. E **96**, 022140 (2017).

[25] C. Alexandrou, A. Athenodorou, C. Chrysostomou, and S. Paul, The critical temperature of the 2D-Ising model through deep learning autoencoders, Eur. Phys. J. B **93**, 226 (2020).

[26] M. Cheney, B. Borden, Conference Board of the Mathematical Sciences, and National Science Foundation (U.S.), in *Fundamentals of Radar Imaging*, CBMS-NSF Regional Conference Series in Applied Mathematics, Vol. 79 (Society for Industrial and Applied Mathematics, Philadelphia, PA, 2009), Chaps. 2 and 3.

[27] P. Knee, *Sparse Representations for Radar with MATLAB Examples*, 1st ed. (Springer, Cham, 2012).

[28] I. Goodfellow, Y. Bengio, and A. Courville, *Deep Learning* (MIT Press, Cambridge, MA, 2016).

[29] B. Šoda and A. Kempf, Efficient method to create superoscillations with generic target behavior, Quantum Stud.: Math. Found. **7**, 347 (2020).

[30] T. Karmakar and A. N. Jordan, Beyond superoscillation: General theory of approximation with bandlimited functions, J. Phys. A: Math. Theor. **56**, 495204 (2023).

[31] Edited by G. B. Arfken, H.-J. Weber, and F. E. Harris, *Mathematical Methods for Physicists*, 7th ed. (Elsevier, Oxford, 2013).

[32] S. Ioffe and C. Szegedy, Batch normalization: Accelerating deep network training by reducing internal covariate shift, in *Proceedings of the 32nd International Conference on Machine Learning*, edited by F. Bach and D. Blei (PMLR, Lille, France, 2015), pp. 448–456.

[33] A. Ly, M. Marsman, J. Verhagen, R. P. Grasman, and E.-J. Wagenmakers, A tutorial on Fisher information, J. Math. Psychol. **80**, 40 (2017).

[34] M. G. Paris, Quantum estimation for quantum technology, Int. J. Quantum Inf. **07**, 125 (2009).

[35] A. Savitzky and M. J. Golay, Smoothing and differentiation of data by simplified least squares procedures, Anal. Chem. **36**, 1627 (1964).

[36] D. P. Kingma and J. Ba, Adam: A method for stochastic optimization, arXiv:1412.6980.