Machine Vision and Applications (2024) 35:39
https://doi.org/10.1007/s00138-024-01523-5

ORIGINAL PAPER

Hyperspectral image dynamic range reconstruction using deep neural network-based denoising methods

Loran Cheplanov[1,2] · Shai Avidan[1] · David J. Bonfil[3] · Iftach Klapp[2]

Received: 21 February 2023 / Revised: 21 February 2024 / Accepted: 21 February 2024

Abstract
Hyperspectral (HS) measurement is among the most useful tools in agriculture for early disease detection. However, the cost of HS cameras that can perform the desired detection tasks is prohibitive-typically fifty thousand to hundreds of thousands of dollars. In a previous study at the Agricultural Research Organization's Volcani Institute (Israel), a low-cost, high-performing HS system was developed which included a point spectrometer and optical components. Its main disadvantage was long shooting time for each image. Shooting time strongly depends on the predetermined integration time of the point spectrometer. While essential for performing monitoring tasks in a reasonable time, shortening integration time from a typical value in the range of 200 ms to the 10 ms range results in deterioration of the dynamic range of the captured scene. In this work, we suggest correcting this by learning the transformation from data measured with short integration time to that measured with long integration time. Reduction of the dynamic range and consequent low SNR were successfully overcome using three developed deep neural networks models based on a denoising auto-encoder, DnCNN and LambdaNetworks architectures as a backbone. The best model was based on DnCNN using a combined loss function of ℓ_2 and Kullback–Leibler divergence on images with 20 consecutive channels. The full spectrum of the model achieved a mean PSNR of 30.61 and mean SSIM of 0.9, showing total improvement relatively to the 10 ms measurements' mean PSNR and mean SSIM values by 60.43% and 94.51%, respectively.

Keywords Deep neural network · Dynamic-range reconstruction · Shortening shooting time · Hyperspectral · Image denoising · Kullback–Leibler divergence loss

1 Introduction

Modern agriculture presents diverse challenges, which can be dealt with using engineering resources–developing smart tools to enlarge process efficiency, building automated systems, developing control and monitoring systems, and more [1]. Plant diseases, one of the main problems in agricultural cultivation that can lead to large losses of whole crops, can be mitigated through early detection of disease development and an agrotechnical point-wise response. A common tool for monitoring changes in the chemical composition of plants is hyperspectral (HS) measurements; these can detect the typical response spectrum of a specific disease as well as general chemical changes indicating strain, insufficiency and disease in vegetative media [2, 3]. The appropriate treatment can then be given at the source, before the problem spreads.

HS measurements for agricultural needs are usually conducted in the range of 350–2300 nm [4]. HS sensors placed on satellites or airplanes can be used to capture the spectrometry

✉ Loran Cheplanov
loran cheplanov@gmail.com

Shai Avidan
avidan@tauex.tau.ac.il

David J. Bonfil
bonfil@volcani.agri.gov.il

Iftach Klapp
iftach@volcani.agri.gov.il

[1] School of Electrical Engineering, Tel Aviv University, Tel Aviv 69978, Israel

[2] Department of Sensing, Information and Mechanization Engineering, Volcani Institute, Agricultural Research Organization, Rishon LeZion 7505101, Israel

[3] Department of Vegetables and Field Crops, Gilat Research Center, Agricultural Research Organization, M.P. Negev 8531100, Israel

Published online: 22 March 2024

of wide areas [5]. HS cameras designed for the more local capture of spectrometric details are also available, but their current high cost prevents their integration into agricultural monitoring [6].

While shooting times for environmental scenery with a standard RGB digital camera are on the order of 10 ms or even 1 ms, HS cameras that rely on whiskbroom, pushbroom and spectral scanning techniques are often characterized by shooting times on the order of minutes, depending mostly on the set integration time. This long shooting time results in lengthy technical imaging procedures and limits the shooting of dynamic scenes. For the sake of shortening acquisition time, in this work, we aimed to shorten the integration time.

Shortening integration time becomes even more important when measuring outdoors, as the light source for spectral measurements is not stable mostly due to cloud motion changing sun illumination of the inspected field. Therefore, shortening the integration time increases the temporal uniformity of illumination during the measurements. From the other hand, shortening the integration time results in inefficient use of the camera's dynamic range, and a lowering of the signal-to-noise ratio (SNR). Since this results in a noisy image, it can be possibly corrected by means of a denoising-based algorithm.

Noise in an image can originate from many sources, denoising algorithms mitigate additive noises resulting from various sources such as acquisition noise, compression noise, and transmission noise [8]. A noisy image can be described as follows:

$$y = x + n \tag{1}$$

where y is the noisy image, x is the object and n is the additive noise.

The main challenge in denoising is preserving fine details in the image such as edges and textures that share similar fine details [8]. Many algorithms were previously suggested to confront the denoising problem, one may tackle it as an inverse problem, which generally requires descriptive information of the model and prior knowledge [9]. Benesty et al. used Wiener filtering for denoising in the frequency domain [10]. Tomasi et al. used bilateral filtering to estimate the pixel's value from its neighbors [8, 11].

Relying on prior knowledge of the object's characteristics, authors used various regularization forms for denoising. In one case, where the smoothness of natural objects is assumed, Rudin et al. suggested using a total variation (TV) prior for denoising [8, 12], while TV resulted in good edge preservation, textures tend to be over-smoothed [8].

Denoising methods based on using local kernels have disadvantages. These local kernels rely on nearby information and thus tend to fail when noise level arises—the local support becomes too noisy to grasp information from [8]. To meet that authors suggested incorporating non-local information such as a non-local means (NLM) [8, 13] for denoising. The approach of NLM was successfully extended to BM3D proposed by Dabov et al. [8, 14]. As mentioned above, the main difficulty in denoising is the similarity between the noise to the fine details in the object; therefore, authors often search for methods where noise and image are separable. Since the spatial characteristics of the object and the noise are different, another existing technique for filtering the additive noise is training a dictionary to decompose the object, and use it for extracting the original object [8]. Aharon et al. performed denoising of an image using sparse representation with a K-SVD [15]. In another work, authors show that wavelet transform is useful for denoising purposes by transforming the image to a wavelet domain, filtering it and transforming it back [8, 16].

In recent years, a number of denoising works using convolutional neural network schemes (CNN) have been introduced. Seung et al. [8, 17], suggested a 5 layers CNN schema for denoising natural images. Chen et al. suggested a Generative adversarial network for denoising [18, 19]. Zhang et al. introduced the DnCNN, a deep neural network (DNN) for denoising [20]. Zheng et al. suggested a denoising schema for HS data in the specific case where only part of the spectral bands is contaminated with noise, in that case, one can use information from nearby channels to recover the information in the noisy channel [18, 21].

In previous works, authors assumed that a noisy image is composed of a summation of the pure object and an additive noise, as described in (1). In this work we show that this assumption is not the case when extremely lowering the integration time to a level of an underutilized dynamic range of the imaging sensor. We pose a new problem in which in a very low integration time the signal suffers from a quantization problem. Following the success of deep learning methods for denoising in a wide variety of fields [8, 18, 19, 21–24], and specifically in the field of spectrometry [25, 26], our suggested solution is based on state-of-the-art DNN backbones, while extending the structures to support recovering the dynamic range of HS measurements, using a combined loss function of ℓ_2 and Kullback–Leibler divergence.

We will show that a drastic shortening in the integration time—by a factor of 20—can be compensated for by means of a neural network algorithm. We show that neural network algorithms can enhance HS data measured at a very short integration time in two critical aspects: restoring the signal's dynamic range and denoising using our unique captured HS images.

The rest of the paper is organized as follows: Sect. 2 describes the dataset-acquisition procedure using our HS scanner prototype; Sect. 3 presents the proposed method; Sect. 4 shows the experimental results; Sect. 5 presents the discussion, and Sect. 6 concludes.

2 Materials and methods

2.1 The low-cost HS system

In response to the HS cameras' cost problem, a low-cost HS system (Fig. 1a) was developed at the Agricultural Research Organization (ARO) Volcani Institute in Israel to accomplish desired missions while maintaining high performance [27, 28]. Replacing the typical pushbroom HS camera with a point spectrometer resulted in three important advantages for agricultural needs: (a) the spectral device's cost was reduced to the range of thousands of dollars; (b) the same system can be used with various point spectrometers, covering the spectral range from ultraviolet to shortwave infrared; (c) the spectral resolution of a point spectrometer is one order of magnitude more sensitive than that of a typical HS camera (e.g., 3000 spectral measures vs. 200), resulting in the detection of much narrower spectral responses associated with chemical changes in plants, and phenomena such as sun-induced fluorescence.

As shown in Fig. 1b, the system consists of optical elements (prisms, focusing lens and optical fibers) and mechanical elements (a motor for the scanning operation and a spectrometer). The resolution of the captured spatial HS image is in $H \times L \times W$, while each pixel's spectrum is measured independently and serially, which means that it takes $H \cdot L$ iterations to measure the overall scene's spectrum. In each iteration, the prisms rotate to the desired location, which is equivalent to the desired pixel to be captured in the final image (the scanning method is illustrated in Fig. 1c). Then, each pixel's spectrum is captured by propagation of the light rays through the prisms, focusing lens and finally, the spectrometer. The spectrum measurement is achieved by the spectrometer which takes the focused light ray, diffracts it to its wavelength components and focuses them onto a linear array of detectors [20]. In this work, we used an Ocean

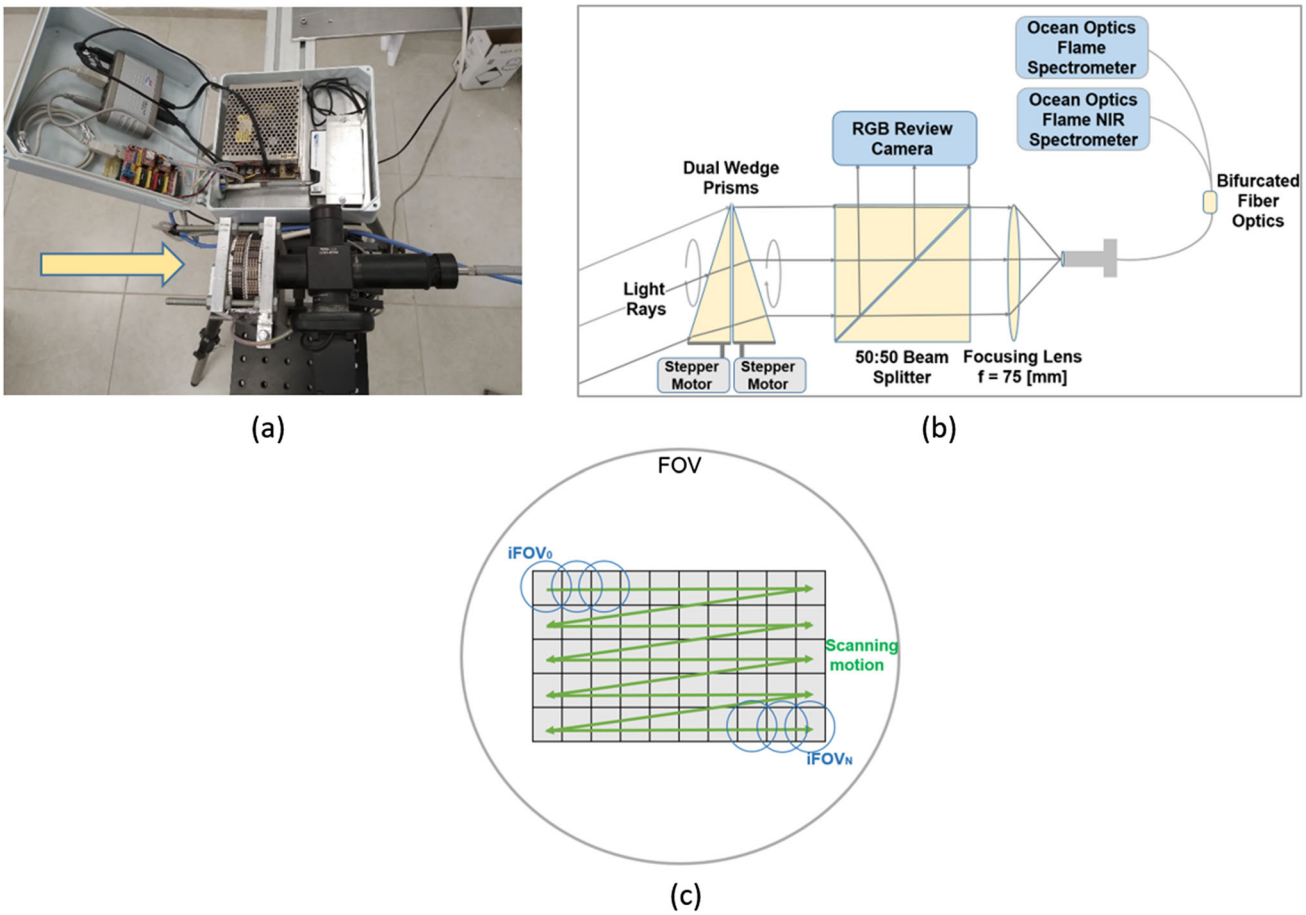

Fig. 1 HS system description. **a** The system's physical components. Yellow arrow indicates the direction of light entry. **b** Schematic drawing of the scanning system. Light rays pass through motorized prisms and are split between the focusing optics and RGB review camera; the light is transmitted into two optional silicon charge-coupled device (CCD) spectrometers via bifurcated fiber optics. **c** A selected region of interest is sampled by a sequence of field points, each with its unique instantaneous field of view (iFOV) which is space- and wavelength-variant, and therefore is determined individually depending on the instantaneous optical configuration of the scanner

Insight FLAME-T-XR1 CCD spectrometer which covers the desired spectrum. It works in the wavelength range of 195.5–1054.1 nm (3648 wavelengths per sample) and the A/D resolution of each pixel is 16 bit (pixel value ranges in $[0, 2^{16} - 1]$). The optic fibers chosen for this system were Ocean Insight type VIS–NIR, in accordance with the wavelength range. The RGB review camera was also part of the system but was only used in the initialization stage for prism-calibration check. The system was controlled through a laptop.

The system was extensively studied and tested. Various numerical simulations were developed that supported the prototype's design. An extensive lab comparison of the proposed system's spectral measurements to a point-spectrometer's measurements showed a normalized sum of absolute difference equal to 4.22% or less. The system was additionally tested on a few field study cases. For more details, the reader is referred to [7].

Overall, the HS system allows scanning a wide field of view while the obtained HS cubes are normalized in post-processing using a "white-balance" process [7, 29]. At the data-acquisition stage, for each set of scene images, a white target (WT) made of polystyrene, in the same orientation as that of the scanned field, was captured (Fig. 2a). Ideally, the final normalized images, which are the input images for our models, are obtained by dividing the HS image by the WT image. The rationale is that when a scene is captured by the HS system, the actual input into the spectrometer is the light rays reflected from the scene's object, which depend on both the light source spectrum and the reflection of the object. Our interest is in the reflection coefficient itself (R(λ)). The theoretical reflection from a white object is 1 for all wavelengths (λ), which means that in this case, the input to the spectrometer will be the light source. The mentioned division of the HS image by the WT image will produce the scene's pure reflection, as shown in (2). An example of a final result from the white-balancing process [29] is presented in Fig. 2b.

$$HS_{norm}(\lambda) = \frac{I_{HS}(\lambda)}{I_{WT}(\lambda)} \approx \frac{I_{sun}(\lambda) \cdot R(\lambda)}{I_{sun}(\lambda) \cdot 1} = R(\lambda) \tag{2}$$

2.2 Dynamic range and SNR

The total shooting time of our low-cost HS prototype is composed of two factors: the total rotation time of the prisms and the predetermined integration time (exposure time) of the spectrometer, which is the dominant contributor. For instance, for a 45 × 90 image at an ideal integration time of 200 ms, the shooting time is 15:41 min; of this, 13:30 min are wasted as a result of the long integration time. The scanning

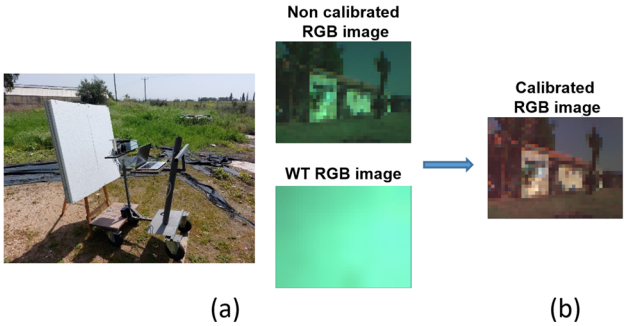

Fig. 2 Illustration of white-balance process (2). **a** Capturing white object. **b** Normalization process in which the outdoor scene's HS image is divided by the white object's HS image

time spent on the mechanical motion could be significantly shortened by adopting a well-optimized off-the-shelf double-wedge prism, thus our focus here is to find a solution based on DNN that compensates for shortening of the integration time.

The outdoor light source is the sun. The high spectral resolution, on the order of 1 nm, typical of HS measurements results in low illumination. Thus, the long integration time is unavoidable because a lower exposure time reduces utilization of the dynamic domain and significant noise is obtained in the resulting spectrum, characterized by the SNR. In general, the resultant HS measurements are a combination of the pure signal and the system noise: the pure signal can be thought of as the "real" and expected spectrum obtained from the measurement, while the system's noise mostly consists of detector noise, some parts of which are influenced by the set integration time. The main components of the system noise are photon shot noise (3), dark current shot noise (4), and read noise [30, 31]. Overall, the SNR of a measurement is the relation between the number of "pure signal" electrons generated and the total effective noise, as presented in (5) [32]. In (3)–(6), λ is the wavelength, QE is the quantum efficiency of the detector which depends on the device's structure and wavelength, Φ is the photon flux, t_I is the integration time, I_D is the average dark current and σ_R is the read noise.

$$\sigma_S(\lambda) = \sqrt{QE(\lambda) \cdot \Phi \cdot t_I} \tag{3}$$

$$\sigma_D = \sqrt{I_D \cdot t_I} \tag{4}$$

$$SNR(\lambda) = \frac{QE(\lambda) \cdot \Phi \cdot t_I}{\sqrt{QE(\lambda) \cdot \Phi \cdot t_I + I_D \cdot t_I + \sigma_R^2}} \tag{5}$$

$$S(\lambda) = QE(\lambda) \cdot \Phi \cdot t_I \tag{6}$$

Several inferences can be made from (5). First, the SNR increases with integration time. This relation also affects the resulting dynamic range. By definition, the signal (6) is the numerator of (5), and thus as the integration time increases, more photons are absorbed in the detector which yields

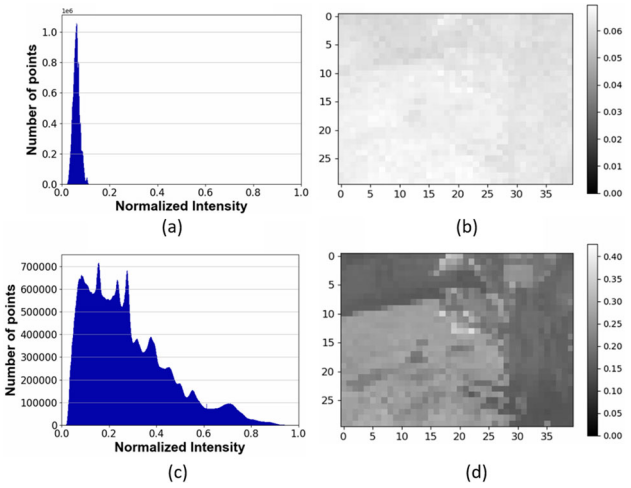

Fig. 3 Experimental demonstration of differences between used integration times. **a** Histogram of total normalized datapoints measured with t_I = 10 ms. **b** Spectral image example at wavelength 601.94 nm measured with t_I = 10 ms. **c** Histogram of total normalized datapoints measured with t_I = 200 ms. **d** Spectral image example at wavelength 601.94 nm measured with t_I = 200 ms

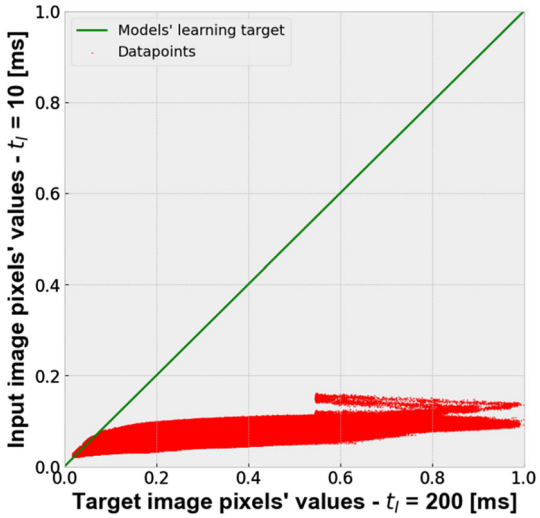

Fig. 4 Scatter plot of the total adjusted input-target pairs of datapoints

greater use of the dynamic range and a better distinction between the detected wavelengths (see Fig. 3a, c). However, it should be noted that the detector limits the number of electrons that a pixel can hold without becoming saturated [30]. Secondly, at short integration times, the dark current noise (4) is a more significant factor in the SNR value (5), whereas at long integration times, the noise measurements are expected to be dominated by photon shot noise.

Figure 3b, d give a visual demonstration of the resulting differences between the use of a long and short integration time in spatial measurements. Observing the figures, we see that the captured pixels' intensities for long t_I are ideally spread across the spectrometer's dynamic range, maximizing its utilization. On the other hand, the captured intensities of a short t_I are grouped in a much smaller dynamic range, directly causing loss of data. The loss in dynamic range results in a nonlinear relationship between the measurements with short and long integration times. Figure 4 shows a typical scatter plot of these measurements. Indeed, the gray levels of the two measurements are nonlinearly related because for a single t_I = 10 ms datapoint, there can be various corresponding t_I = 200 ms datapoint options. The mission of the models is a regression task, to learn successful manipulation of the noisy input data so that they will be equal to their corresponding target data. Thus, our challenge consists of restoring the information lost due to the system's use of a low dynamic range while overcoming the generated noise.

Following (6), the spectrometer's output is quantified and contains noise, thus the pixel's intensity can be formulated as:

$$I(\lambda) = Q(\alpha \cdot S(\lambda) + n(\lambda)), \tag{7}$$

where n is the additive noise, $Q()$ is the A/D quantization function of the spectrometer and $0 \leq \alpha \leq 1$ is the relative amplitude, so that:

$$\alpha = \frac{t_I}{t_{Imax}} \tag{8}$$

where t_{Imax} is the ideal maximal integration time for full utilization of the dynamic range without causing saturation of the measurements. Our goal is to minimize the difference of a measurement taken at $t_I < t_{Imax}$ compared to t_{Imax}. The relative error is:

$$\begin{aligned} err(\lambda) =& Q(S(\lambda)_{t_{Imax}} + n(\lambda)_{t_{Imax}}) \\ & - Q(\alpha \cdot S(\lambda)_{t_I} + n(\lambda)_{t_I}) \end{aligned} \tag{9}$$

There are two main contributors to the result of (9): the appearance of noise ($n(\lambda)$) and the utilization of dynamic range. In our problem, the low signals resulting in poor dynamic range are the significant factor in the equation, since they most influence the quantization error. Our goal is to find a $NN\{\}$ function to minimize this error. The loss function $\mathcal{L}()$ under general $\| \ \|_p$ norm is:

$$\begin{aligned} \mathcal{L}\left(I(\lambda)_{t_I}\right) =& argmin_\theta \| Q(S(\lambda)_{t_{Imax}} + n(\lambda)_{t_{Imax}}) \\ & - NN\{Q(\alpha \cdot S(\lambda)_{t_I} + n(\lambda)_{t_I}); \theta\} \|_p, \end{aligned} \tag{10}$$

where θ is the network parameters vector.

2.3 Dataset description

A unique dataset was collected through our HS system, where each measurement of HS cube was 30 × 40 × 3648 (H × L × W). Each measurement contains various outdoor scenes, including agricultural fields, trees, houses, grass and more. Previous work [27, 28] indicated that typical outdoor imagery with the low-cost HS scanner shows good dynamic and SNR performances with an integration time of 200 ms. To significantly reduce the integration time, we set our input images' integration time to 50 ms and 10 ms. Thus, for the model's training stage, each scene was sampled with three different integration times: 200 ms, 50 ms and 10 ms, with shooting times of 15:41, 5:34 and 2:52 min, respectively. The successive measurements resulted in a set of three HS images per captured scene.

The image of t_I = 200 ms served as the deep learning models' target to be learned, whereas the images of t_I = 50 ms and 10 ms served as the low-quality inputs to the model. The work focused on restoring the 10 ms dataset, which is the most challenging one. Thus, for the sake of algorithm development, we used sets that included pairs of HS cubes measured in integration times of 10 ms and 200 ms.

The dataset was divided into training, validation and test sets. A total of 56 HS cubes were used for the training (44) and validation (12) sets. The images were captured at the ARO Volcani Institute. A separate field experiment was conducted to create the test set and contained three HS cubes of agricultural fields. The experiment was intentionally performed in a different area of the country (at ARO's Gilat Research Center in the Negev desert, Israel) to ensure that our models are modular and that their performance is not influenced by a different location for the image capture. A stand was designed to hold the HS system in the back of a pickup truck, as shown in Fig. 5. To enrich the data, data augmentation was used on the training dataset, including random horizontal and vertical flips.

All of the data were normalized in a two-step preprocessing. First, the data were white-balanced, as formulated in (2). Then, the data were normalized to [0, 1], relatively to the dynamic range 2^{16}.

An important issue to consider was retaining the best possible alignment between the images within the sets, due to dynamic changes in the environment, such as light intensity, which vary with time. Therefore, the measuring method captured each pixel's spectrum for the three integration times serially, by performing an adaptation to the system's base code.

2.4 Computation considerations

Only partial use was made of the captured wavelength range due to lack of quantum efficiency and data measurements at the spectrum edges; we therefore used the spectral bands between 404 and 701.1 nm. The image spatial resolution chosen for the dataset creation was 30 × 40, where each pixel is composed of 1220 spectral measurements, so that the HS cube's total shape was 30 × 40 × 1220. This tensor size could not be input into the models all at once due to the random-access memory (RAM) restrictions, and we therefore decided to perform a separate training for each 20 adjacent channels; this was found experimentally to be the best number of channels in terms of final performance and memory usage, as will be further explained in Sect. 4. Thus, the input's shape was 30 × 40 × 20.

The experiments for the deep learning models' training were performed on a desktop computer using Windows 10 with Intel®Core™ i7-10700 CPU, 64 GB of RAM, and an Intel®UHD Graphics 630 graphics processing unit (GPU) card. The algorithms were implemented using PyTorch 1.11.0.

Fig. 5 Test dataset creation experiment at Gilat Research Center. The unique structure of the HS system was designed to be placed on a stand in a pickup truck to shoot the scene

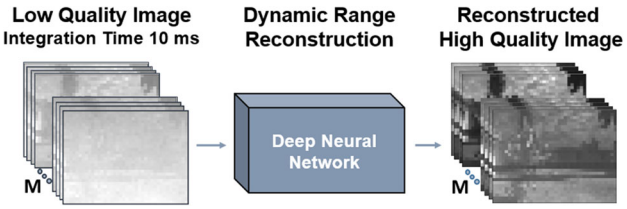

Fig. 6 The general DNN architecture. The DNN takes as input M adjacent channel layers of the low-quality HS cube that support each other's process, and outputs M reconstructed high-quality layers. The detailed DNN architectures are described in the following subsections and are illustrated in Figs. 7, 8 and 9

3 Proposed method

Whereas in the RGB images the contents of red, green and blue may differ spatially, HS data suggest a continuous change in spatial content between adjacent layers. In (9), we showed that a part of our problem is quantization of the noise contribution. Bearing that in mind, we can exploit the

deep learning denoising scheme designed for RGB images as a backbone for restoring the dynamic range of the HS data, while using adjacent layers of the HS cube to support each other's process. The general architecture of the proposed scheme is presented in Fig. 6, where M is the total number of input and output channel layers used and was chosen such that $M \cdot \Delta\lambda$ is much smaller than the spectrometer's bandwidth.

The section is organized as follows: Sect. 3.1 presents an introduction to the involved backbone models; Sects. 3.2–3.4 elaborate on the modifications and implementation of each backbone, the elaboration accompanied by a graphical figure and a table composed of the network layers, Sect. 3.5 discusses about the loss functions, and Sect. 3.6 provides the pseudo-code of the models' training phase.

3.1 Backbone DNN methods

To create a denoising scheme for the HS cube, three backbone models were used: denoising auto-encoder (DAE) [33], DnCNN [22] and LambdaNetworks [24]. As already noted, the models' inputs and outputs are of the same shape (30 $\times$ 40 $\times$ 20, H $\times$ L $\times$ W), where the inputs are the images of t_I = 10 ms and the outputs are learned from the ground-truth images of t_I = 200 ms. The models' final design was obtained through an extensive "pre-experiment" on each of the models' architectural parameters, such as number of layers and convolutional layer parameters (depth size, kernel size, etc.). The models performed a regression task under supervised learning. The following subsections detail our designed architectures.

3.2 Implementing the DAE algorithm on HS data

Our suggested HS-DAE model uses the concept of DAE [33] and is composed of two main parts: encoder and decoder. Each part consists of seven layers. As can be seen in Fig. 7, the encoder part takes the input matrix and reduces its spatial size while increasing its depth from layer to layer. Then the encoder's output (the "latent matrix") is entered as input to the decoder part that performs the opposite operation of increasing the input's spatial size while decreasing its depth till the matrix size reaches that of the original input matrix. The output uses a skip-connection of the input with the last layer. The HS-DAE model's layers are detailed in Table 1, all in all the model is composed of 78.7 $\times$ 10^6 parameters.

3.3 Implementing the DnCNN algorithm on HS data

The HS-DnCNN model is structured almost exactly as designed in the original article [22]. As shown in Fig. 8, the HS-DnCNN contains an overall 17 convolution layers which keep the input's spatial size constant along the layers. The first layer doubles the input channels' sizes, which are also kept constant till the last layer that reduces them back to the original size. The HS-DnCNN model's layers are detailed in Table 2, all in all the model is composed of 0.23×10^6 parameters. The structure of the output layer depends on the loss function used: if it is the combination of Kullback–Leibler (KL) divergence and some other function such as ℓ_1 or ℓ_2 (as will be further discussed in section 3.5), the output equals Sigmoid(BatchNorm2D(Conv2D(layer16))). Otherwise, the output equals Conv2D(layer16), as also appears in the original article [22].

Table 1 HS-DAE architecture overview

SN	Name	Input size	Output size
0	Input	–	30 × 40 × 20
1	ConvBlock1	30 × 40 × 20	28 × 38 × 40
2	ConvBlock2	28 × 38 × 40	26 × 36 × 80
3	ConvBlock3	26 × 36 × 80	24 × 34 × 160
4	ConvBlock4	24 × 34 × 160	22 × 32 × 320
5	ConvBlock5	22 × 32 × 320	20 × 30 × 640
6	ConvBlock6	20 × 30 × 640	18 × 28 × 1280
7	ConvBlock7	18 × 28 × 1280	16 × 26 × 2560
8	TrConvBlock1	16 × 26 × 2560	18 × 28 × 1280
9	TrConvBlock2	18 × 28 × 1280	20 × 30 × 640
10	TrConvBlock3	20 × 30 × 640	22 × 32 × 320
11	TrConvBlock4	22 × 32 × 320	24 × 34 × 160
12	TrConvBlock5	24 × 34 × 160	26 × 36 × 80
13	TrConvBlock6	26 × 36 × 80	28 × 38 × 40
14	TrConv2D	28 × 38 × 40	30 × 40 × 20
15	Output	30 × 40 × 20	30 × 40 × 20

All of the convolution layers use kernels of size 3 × 3 with stride = 1 and padding = 0
ConvBlock = ReLU(BatchNorm2D(Conv2D(input)))
TrConvBlock = ReLU(BatchNorm2D(TrConv2D(input)))
Output = Sigmoid(Input + layer14)

Table 2 HS-DnCNN architecture overview

SN	Name	Input size	Output size
0	Input	–	30 × 40 × 20
1	Conv2D + ReLU	30 × 40 × 20	30 × 40 × 40
2-16	ConvBlock	30 × 40 × 40	30 × 40 × 40
17	Output	30 × 40 × 40	30 × 40 × 20

The kernels are sized 3 × 3 with stride = 1 and padding = 1
ConvBlock = ReLU(BatchNorm2D(Conv2D(input)))

3.4 Implementing the LambdaNetworks algorithm on HS data

LambdaNetworks [24] is a new approach for self-attention mechanisms [34] that bypasses the expensive memory requirements of attention maps. While even local self-

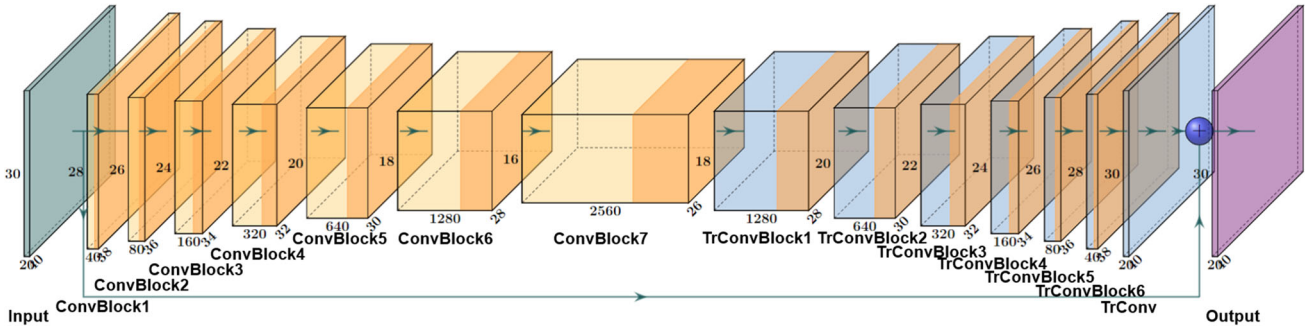

Fig. 7 HS-DAE architecture. Green—input, yellow—convolution layer, blue—transposed convolution layer, orange—batch normalization + activation function (ReLU), purple—output with added sigmoid function

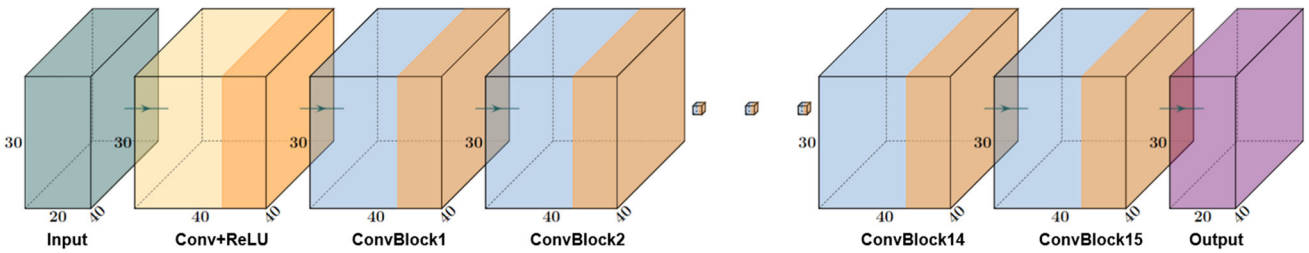

Fig. 8 HS-DnCNN architecture. Green—input, yellow—convolution layer, blue—convolution layer + batch normalization, orange—activation function (ReLU), purple—output whose design depends on the used loss function

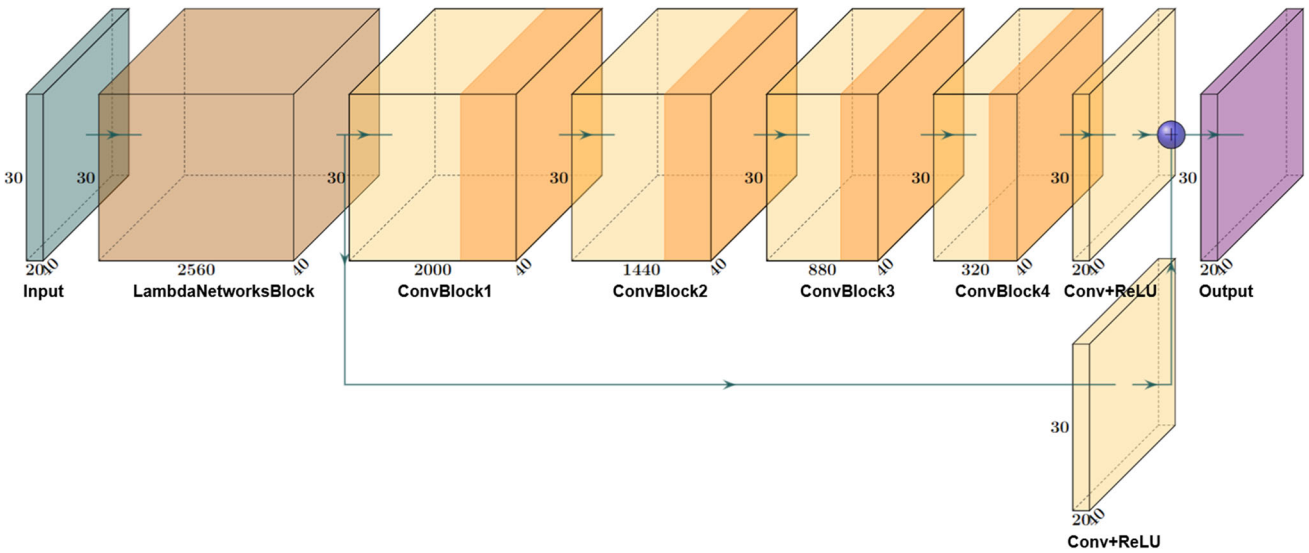

Fig. 9 HS-LambdaNetworks architecture. Green—input, brown—LambdaNetworks block, yellow—convolution layer + batch normalization, orange—activation function (ReLU), purple—output with added sigmoid function

attention is a highly demanding task for a standard computer, the LambdaNetworks model finds the solution by fitting the image with a linear transformation—the lambda layer. This layer needs to be computed only once and can be thought of as a reduced-size attention map. The lambda layer contains information about all of the pixels and the relationship between the pixels in the image. Originally, the model was produced for missions of classification, object detection and image segmentation. We assumed that use of the described concept of attention map might also be beneficial for dynamic range reconstruction and denoising missions.

The proposed HS-LambdaNetworks model uses the LambdaNetworks model as a main component in the total architecture (Fig. 9). The first layer is the LambdaNetworks block that takes the HS image input and produces a matrix with a larger channel size of 2560. This block is structured exactly as

Table 3 HS-LambdaNetworks architecture overview

SN	Name	Input size	Output size
0	Input	–	$30 \times 40 \times 20$
1	LN-Block	$30 \times 40 \times 20$	$30 \times 40 \times 2560$
2	ConvBlock1	$30 \times 40 \times 2560$	$30 \times 40 \times 2000$
3	ConvBlock2	$30 \times 40 \times 2000$	$30 \times 40 \times 1440$
4	ConvBlock3	$30 \times 40 \times 1440$	$30 \times 40 \times 880$
5	ConvBlock4	$30 \times 40 \times 880$	$30 \times 40 \times 320$
6a	BN(Conv(layer5))	$30 \times 40 \times 320$	$30 \times 40 \times 20$
6b	BN(Conv(layer1))	$30 \times 40 \times 2560$	$30 \times 40 \times 20$
7	Output	$30 \times 40 \times 20$	$30 \times 40 \times 20$

The kernels are sized 1×1 with stride = 1 and padding = 0
ConvBlock = ReLU(BatchNorm2D(Conv2D(input)))
COutput = Sigmoid(layer6a + layer6b)

presented in the original article [24]; however, we treated the context and input matrices mentioned in the original article as the same input matrix (our HS cube). Then, convolution layers are activated and reduce the image's channel sizes till the original channel size is obtained. Along the layers of our model, the spatial size of the features remains the same. In addition, as also can be seen in Fig. 9, a skip-connection was used at the output. The HS-LambdaNetworks model's layers are detailed in Table 3, all in all the model is composed of 4.6×10^6 parameters.

3.5 Loss functions

In general, in regression tasks such as image denoising, the commonly used function losses are ℓ_1-loss and ℓ_2-loss [35]. Equations (11) and (12) represent the formulas for ℓ_1 and ℓ_2, respectively:

$$\mathcal{L}^{\ell_1} = \frac{1}{N}\sum_{i=1}^{N} \mid y_i - f(x_i) \mid \tag{11}$$

$$\mathcal{L}^{\ell_2} = \frac{1}{N}\sum_{i=1}^{N} (y_i - f(x_i))^2 \tag{12}$$

where N is the total number of pixels, y is the target value, x is the input value and $f(x)$ is the model's output value.

In our work, we experimented with several combinations of loss functions, including the standard ones and a less conventional one-the KL divergence (D_{KL}) loss [36]. The D_{KL} loss (13) is a similarity function which consists of two parts: $\sum_{x\in X} p(x)\log p(x)$ is the negative of Shannon's entropy loss [37] and $-\sum_{x\in X} p(x)\log q(x)$ is the measure of inaccuracy proposed by Kerridge [38]. Overall, the loss function measures the discrepancy between the two distributions, where low values express high similarity.

$$\begin{aligned}\mathcal{D}_{KL}(\mathcal{P}\|\mathcal{Q}) &= -\sum_{x\in X} p(x)\log q(x) + \sum_{x\in X} p(x)\log p(x) \\ &= \sum_{x\in X} p(x)\log\left(\frac{p(x)}{q(x)}\right)\end{aligned} \tag{13}$$

where X is the probability space and $\{p(x), q(x)\}$ are the probability distributions. In our problem, we treat X as the total number of pixels and $\{p(x), q(x)\}$ as the target (ground-truth) and model output pixel values, respectively. The D_{KL} loss aims to quantify the similarity between each pair of $\{p(x), q(x)\}$ corresponding measurements. Before applying the D_{KL} component on the output, the pixels were Softmaxed along the channel's dimension.

The loss functions were applied on the last layer. In addition, they were applied pixel-wise and the results were averaged over a number of batches.

Algorithm 1 Training phase algorithm

Input: *Imgs* = Training HS Images
Output: *best_model* = Best Trained Model
preprocess_imgs ← Preprocess *Imgs* including white-balancing and normalization
train_imgs, *val_imgs* ← Split *preprocess_imgs* to training and validation datasets
best_mean_val_psnr ← 0
epochs ← Number of epochs
for *epoch* in range(*epochs*) **do**
  for *train_imgs_batch* in *train_imgs* **do**
    train_input_imgs, *train_target_imgs* ← *train_imgs_batch*
    train_output_imgs ← model(*train_input_imgs*)
    train_loss ← loss_func(*train_output_imgs* , *train_target_imgs*)
    Apply backpropagation based on *train_loss*
  end for
  mean_val_psnr ← 0
  for *val_imgs_batch* in *val_imgs* **do**
    val_input_imgs, *val_target_imgs* ← *val_imgs_batch*
    val_output_imgs ← model(*val_input_imgs*)
    val_loss ← loss_func(*val_output_imgs*, *val_target_imgs*)
    val_psnr ← convert_to_psnr_func(*val_loss*)
    mean_val_psnr ← *mean_val_psnr* + *val_psnr*
  end for
  mean_val_psnr ← *mean_val_psnr*/*len*(*val_imgs*)
  if *mean_val_psnr* > *best_mean_val_psnr* **then**
    best_mean_val_psnr ← *mean_val_psnr*
    best_model ← *model*
  end if
end for

3.6 Pseudo-code

As previously elaborated, our proposed method includes a use of a DNN with three different options of backbones, where their input is a low integration captured HS cube and their output is a reconstructed high-quality HS cube. The

training phase of these models could be generalized into the pseudo-code described in Algorithm (1).

4 Experimental results

4.1 Evaluation metrics

In image denoising and restoration tasks, a variety of evaluation metrics can be used. We chose to analyze our results according to the peak-signal-to-noise ratio (PSNR) and structural similarity index measure (SSIM), where each metrics describes a different facet of the measure. For the inference stage, the models' final weights were saved according to the highest achieved mean PSNR of the validation dataset.

The PSNR [39] is an interpretation of the mean squared error (MSE). A higher PSNR value indicates higher image quality and is defined by:

$$PSNR = 10 \cdot \log_{10}(P_{DR}^2/MSE) \tag{14}$$

where P_{DR} is the peak dynamic range value in the data and MSE is the exact $\mathcal{L}^{\ell_2}$ shown in (12). The SSIM [40] is a quality metric used to measure the similarity between two images (f and g in (15) and (16)). It is designed by modeling any image distortion as a combination of three factors: loss of correlation ($s(f,g)$), luminance distortion ($l(f,g)$) and contrast distortion ($c(f,g)$). Like the PSNR, a higher SSIM value indicates higher image quality and is defined by:

$$SSIM(f,g) = l(f,g) \cdot c(f,g) \cdot s(f,g) \tag{15}$$

$$\begin{cases} l(f,g) = \dfrac{2\mu_f\mu_g + C_1}{\mu_f^2 + \mu_g^2 + C_1} \\ c(f,g) = \dfrac{2\sigma_f\sigma_g + C_2}{\sigma_f^2 + \sigma_g^2 + C_2} \\ s(f,g) = \dfrac{\sigma_{fg} + C_3}{\sigma_f\sigma_g + C_3} \end{cases} \tag{16}$$

where C_1, C_2 and C_3 are small positive constants, μ_f and μ_g are the means of the compared images and σ_f and σ_g are the standard deviations of the compared images.

4.2 Training hyperparameters

The three proposed models were trained using the Adam optimizer [41] with its default PyTorch parameters. Table 4 presents the global hyperparameters used for training.

Table 4 Global hyperparameters

Batch size	Learning rate	Learning rate decay	Epochs
4	0.001	0.99995	4000

Table 5 Test of number of input channels

Number of channels	PSNR	SSIM	Parameters (M)
1	28.09	0.76	6.38×10^{-4}
4	29.89	0.88	0.001
10	30.05	0.89	0.06
20	30.63	0.89	0.23
50	30.47	0.9	1.44
100	30.67	0.88	5.76

4.3 Optimal input's channel size

Before testing our models with the various existing hyperparameters, we had to choose the number of channels that would be input to the models. We wanted to perform the training on the whole spectrum, which consists of 3648 channels. However, as mentioned above, it must be divided into parts since the hardware has a RAM limitation and cannot handle this channel size. The influence of channel number on PSNR and SSIM scores was determined with the DnCNN model, using a combined loss function of $\mathcal{L}^{\ell_2} + D_{KL}$ and weight decay of 10^{-4}. The channels were input as adjacent channels around the wavelength of 601.94 nm. Table 5 shows the results of the experiment.

Various factors were considered to choose the optimal number of channels. Performance and especially PSNR certainly topped the list, but the cost of the used models' number of parameters was also significant. As the chosen number of channels increases, so does the RAM used, as well as the training time. In addition, more parameters affect obtaining a generalized "solution" and gaining more overfitting. Altogether, it was decided to choose 20 channels as a decent trade-off between the above considerations.

4.4 Testing the proposed models

4.4.1 Comparisons of proposed models' performances

The models were tested with various selected combinations of the following loss functions and weight decays:

$$\textbf{Loss} \in \{\mathcal{L}^{\ell_1}, \mathcal{L}^{\ell_1} + \mathcal{D}_{KL}, \mathcal{L}^{\ell_2}, \mathcal{L}^{\ell_2} + \mathcal{D}_{KL}\}$$

$$\textbf{Weight decay} \in \{5 \cdot 10^{-4}, 10^{-4}, 5 \cdot 10^{-5}\}$$

where we can define the "outer product" of **Loss**⊗**Weight decay** as the total combination options examined. The final

Table 6 Average test results of the proposed models in the wavelength range of 599.5–604.1 nm

Model name	Parameters (M)	Loss function	Weight decay	Inference time (sec)	PSNR	SSIM
HS-DAE	78.7	ℓ_1	10^{-4}	0.84	29.25	0.87
HS-DAE	78.7	ℓ_1+D_{KL}	10^{-4}	0.84	28.95	**0.9**
HS-DAE	78.7	ℓ_2	5×10^{-4}	0.84	28.83	0.83
HS-DAE	78.7	ℓ_2+D_{KL}	5×10^{-4}	0.84	29.16	**0.89**
HS-LambdaNetworks	4.6	ℓ_2	5×10^{-5}	7.1	27.43	0.83
HS-LambdaNetworks	4.6	ℓ_2+D_{KL}	10^{-4}	7.1	28.64	0.88
HS-DnCNN	**0.23**	ℓ_1	10^{-4}	0.52	29.18	0.88
HS-DnCNN	**0.23**	ℓ_1+D_{KL}	10^{-4}	0.52	29.31	0.88
HS-DnCNN	**0.23**	ℓ_2	10^{-4}	0.52	29.7	0.88
HS-DnCNN	**0.23**	ℓ_2+D_{KL}	10^{-4}	0.52	**30.63**	**0.89**

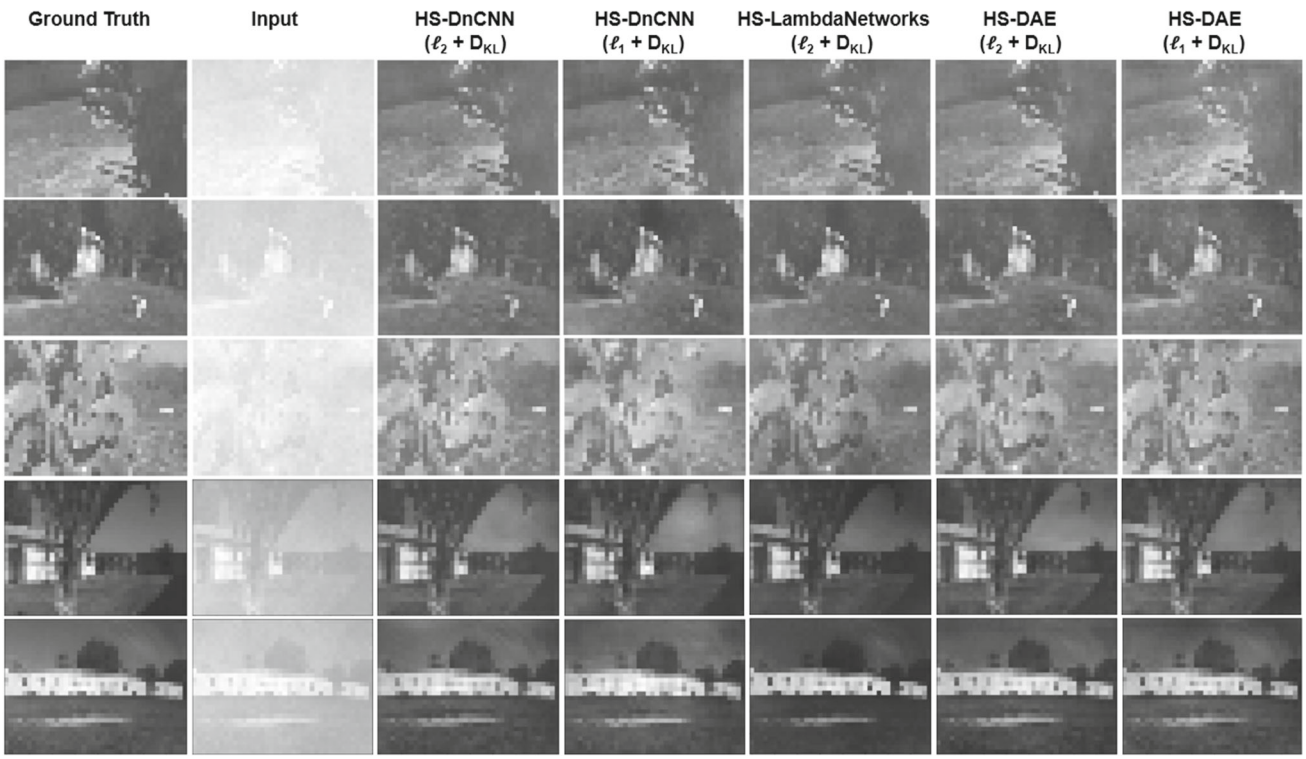

Fig. 10 Qualitative results of top-performing models at wavelength 601.94 nm

best performance results of our experimental models are described in Table 6 for the training wavelength range of 599.5–604.1 nm. The PSNR and SSIM values are the average of each channel's calculated PSNR and SSIM. For the reference, the input t_I = 10 ms images' PSNR and SSIM values were 18.71 and 0.41, respectively. According to the results, the three models significantly improved the t_I = 10 ms images' performances. The biggest improvement was achieved with the HS-DnCNN model using a combined loss function of $\mathcal{L}^{\ell_2} + D_{KL}$, which also has the shortest inference time of only 0.52 s. Moreover, addition of the D_{KL} loss seemed beneficial and improved the total results, except in the case of the HS-DAE model using the $\mathcal{L}^{\ell_1}$ loss function which indeed resulted in a worse PSNR but a better SSIM value (the best tested SSIM).

Figure 10 provides a visual demonstration of the top performance of the tested models. In general, the total qualitative results showed a significant improvement over the t_I = 10 ms data. In the small details, it can be noted that HS-DnCNN using $\mathcal{L}^{\ell_1} + D_{KL}$ and HS-DAE using $\mathcal{L}^{\ell_2} + D_{KL}$ are slightly better than the others, especially in the darker areas. Because our specific leading interest was to achieve an accurate model followed by a small MSE, we chose the HS-DnCNN model using $\mathcal{L}^{\ell_2} + D_{KL}$ as the leading model for further work due

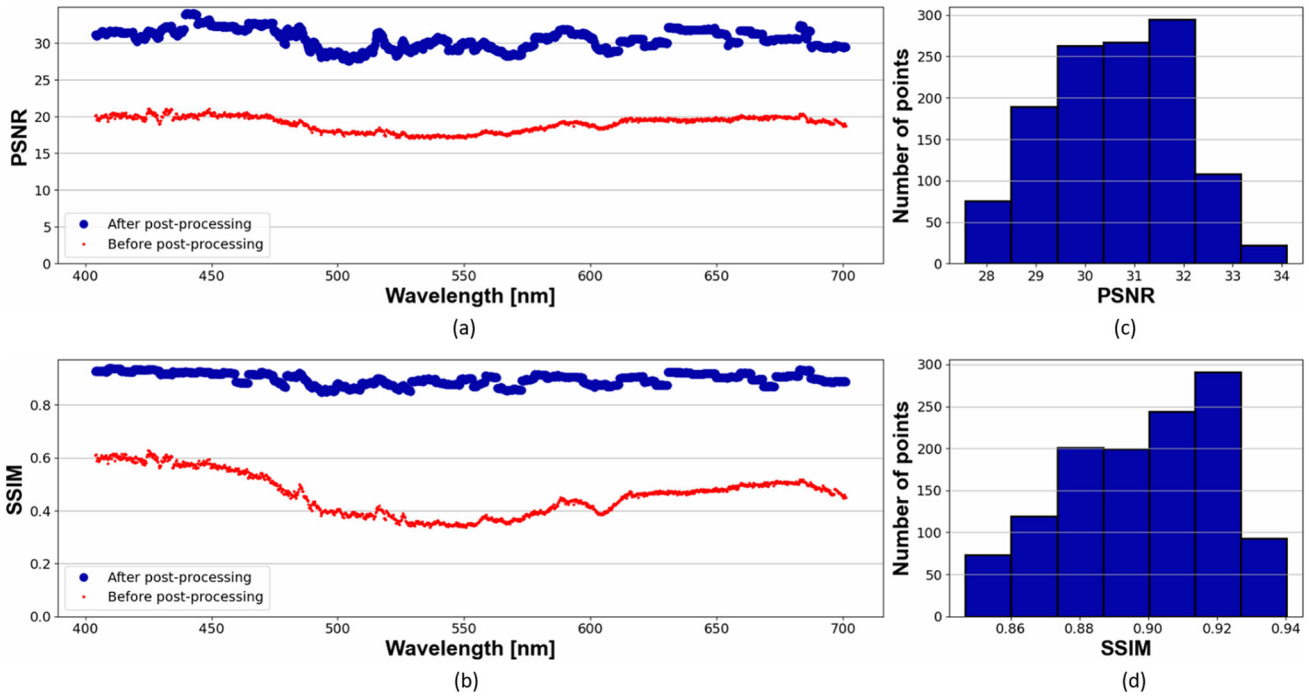

Fig. 11 Cross-wavelength performance analysis. **a** Full-spectrum PSNR performance graph. Red (thin) and blue (thick) curves represent the t_I = 10 ms input images and the network's output images, respectively. **b** Full-spectrum SSIM performance graph. Red (thin) and blue (thick) curves represent the t_I = 10 ms input images and the network's output images, respectively. **c** Full-spectrum PSNR performance histogram of the network's output images. **d** Full-spectrum SSIM performance histogram of the network's output images

Table 7 Test results for the HS-DnCNN model in the full wavelength range

Data type	Mean PSNR	Mean SSIM
Input	19.08	0.46
Output	30.61	0.9

to its best quantitative performance results while requiring a small number of parameters.

4.4.2 Full spectrum analysis

Further performance analysis was conducted based on the model that functioned best. The PSNR and SSIM values for the whole trained wavelength spectrum are shown in Fig. 11. As can be seen, the model's output statistics are not constant along the wavelength axis and their changes are highly correlated to the corresponding t_I = 10 ms input data statistics–the PSNR correlation coefficient is 0.735 and the SSIM correlation coefficient is 0.717. The model's PSNR and SSIM performance is detailed in Table 7, showing total improvement of the PSNR and SSIM values by 60.43% and 94.51%, respectively.

5 Discussion

The experiments were focused on achieving the best possible pixel-reconstruction accuracy. This was done by saving the models' weights according to the best PSNR scores during the training phase. The training could also be implemented differently for other goals, such as achieving the best visual results. In that case, the models' architectures may be preserved while adjusting the loss functions to the appropriate mission [35], e.g., using a combined loss function of $SSIM + D_{KL}$ and saving the model's weights accordingly to the best achieved SSIM scores.

The D_{KL} component was found to be a beneficial addition to the total loss function. Adding it forced the models to accomplish a depth-similarity between the output values and the ground-truth values.

As shown in the final test performance in Table 6, the three models yielded positive results but the HS-LambdaNetworks model functioned worse than others. Even though the performance expectations from this model were higher, it may be less suitable for restoration of the dynamic range of the signal and noise reduction.

6 Conclusion and future work

Many of previous works treat and define a noisy image as the summation of the pure image and an additive noise, and their suggestions of enhancing the image quality are based on extracting the ground truth image from the noisy image. In this work we show that when working with an HS system in a very low integration time the problem of denoising is more complicated. In this case, the dynamic range of the captured image occupies only a small portion of the system's dynamic range, which consequently results in deteriorating image's quality. A major part of the non-typical achieved noise in the resulted image appear due to a quantization problem that the captured signals suffer from.

Based on the partial similarity between our problem to classical denoising problems, this paper suggests three usable state-of-the-art denoising DNN backbones to obtain signal recovery: HS-DAE, HS-LambdaNetworks and HS-DnCNN. To boost the total signal recovery performances, we suggest to include an additional component of D_{KL} in the cost function. The proposed method was demonstrated on unique measurements taken by a previously suggested HS scanning system, showing our capabilities to accelerate its scanning time with a reduction of the integration time by factor of 20.

The results show a successful proof of concept for dynamic range reconstruction and noise reduction of HS images with low defined integration time. The three proposed models significantly improved the $t_I = 10$ ms images' performance, while showing rapid inference times ranging from 0.52 to 7.1 s, which provides a fast-paced image processing. Though the methods were implemented in the field of agriculture, the architectures are modular and may be trained over any HS datasets in various areas for shortening system shooting times.

Best results were obtained with HS-DnCNN while using the combined loss function of $\mathcal{L}^{\ell_2} + D_{KL}$. The HS-DnCNN requires the smallest number of parameters which is beneficial for small datasets such as available in real world HS imaging. Typical results present a very high recovery with mean PSNR of 30.61 and mean SSIM 0.9, showing total improvement relatively to the 10 ms measurements' mean PSNR and mean SSIM values by 60.43% and 94.51%, respectively. The model also achieved the shortest inference time among the three models of 0.52 s.

Our model's training phase requires large datasets; in our next phase we intend to collect more data to improve the total performances. In addition, our work shows that processing series of spectra together brings supportive information which improves signal recovery, the next phase of this research will deeper the investigation of supporting data.

Acknowledgements This work was supported by the Israeli Ministry of Agriculture and Rural Development under Grant 20-07-0061. The authors would like to thank Or Arad for his advice in operating the HS system, Adi Katzav and Aviram Dick for writing multiple integration time control scripts. We also thank Yiftach Afgin, Liad Reshef, and Lavi Rosenfeld for engineering support.

Funding Open access funding provided by The Agricultural Research Organization of Israel.

Declarations

Conflict of interest The author(s) declared no conflicts of interest with respect to the research, authorship, and publication of this paper.

References

1. Spiertz, H.: Challenges for crop production research in improving land use, productivity and sustainability. Sustainability. **5**, 1632–1644 (2013). https://doi.org/10.3390/su5041632
2. Pokhrel, A.: Role of individual components of disease triangle in disease development: a review. Plant Pathology & Microbiology. **12**(9) (2021).https://doi.org/10.35248/2157-7471.21.12.573
3. Elke, B., Heike, G., Xijuan, C., Ewald, S.: The potential of spectral measurements for identifying glyphosate application to agricultural fields. Agronomy. **10**, 1409 (2020). https://doi.org/10.3390/agronomy10091409
4. Du, L., Baoyuan, L., Xiaoping, Z., Xihua, Y., Liang, H., Jie, H., Jinwei, G., Jufeng, W., Qi, C.: An experimental study on field spectral measurements to determine appropriate daily time for distinguishing fractional vegetation cover. Remote Sensing. **12**(18), 2942 (2020). https://doi.org/10.3390/rs12182942
5. Lu, B., Dao, P.D., Liu, J., He, Y., Shang, J.: Recent advances of hyperspectral imaging technology and applications in agriculture. Remote Sensing. **12**(16), 2659 (2020). https://doi.org/10.3390/rs12162659
6. Stuart, M.B., Stanger, L.R., Hobbs, M.J., Pering, T.D., Thio, D., McGonigle, A.J.S., Willmott, J.R.: Low-cost hyperspectral imaging system: design and testing for laboratory-based environmental applications. Sensors. **20**(11), 3293 (2020). https://doi.org/10.3390/s20113293
7. Arad, O., Cheplanov, L., Afgin, Y., Reshef, L., Brikman, R., Elatrash, S., Stern, A., Tsror, L., Bonfil, D.J., Klapp, I.: Low-cost dispersive hyperspectral sampling scanner for agriculture imaging spectroscopy. IEEE Sensors. **23**(16), 18292–18303 (2023). https://doi.org/10.1109/JSEN.2023.3282835
8. Fan, L., Zhang, F., Fan, H., Zhang, C.: Brief review of image denoising techniques. Vis Comput Ind Biomed Art. **2**, 7 (2019). https://doi.org/10.1186/s42492-019-0016-7
9. Bertedo, M., Boccacci, P.: Introduction to inverse problems in imaging. Iop. (1998)

10. Chen, J., Benesty, J., Huang, Y., Doclo, S.: New insights into the noise reduction Wiener filter. In: IEEE Transactions on Audio, Speech, and Language Processing. **14**(4), 1218–1234 (2006). https://doi.org/10.1109/TSA.2005.860851
11. Tomasi, C., Manduchi, R.: Bilateral filtering for gray and color images. In: Abstracts of the sixth international conference on computer vision IEEE, Bombay, India, pp. 839–846 (1998).https://doi.org/10.1109/ICCV.1998.710815
12. Rudin, L.I., Osher, S., Fatemi, E.: Nonlinear total variation based noise removal algorithms. In: Paper presented at the eleventh annual international conference of the center for nonlinear studies on experimental mathematics: computational issues in nonlinear science, Elsevier North-Holland, Inc, New York, pp. 259–268 (1992).https://doi.org/10.1016/0167-2789(92)90242-F
13. Zibulevsky, M., Elad, M.: L1–L2 optimization in signal and image processing. IEEE Signal Processing Magazine. **27**(3), 76–88 (2010). https://doi.org/10.1109/MSP.2010.936023
14. Dabov, K., Foi, A., Katkovnik, V., Egiazarian, K.: Image denoising by sparse 3-D transform-domain collaborative filtering. IEEE Transactions on Image Processing. **16**(8), 2080–2095 (2007). https://doi.org/10.1109/TIP.2007.901238
15. Aharon, M., Elad, M., Bruckstein, A.: rmK-SVD: an algorithm for designin overcomplete dictionaries for sparse representation. IEEE Transactions on Signal Processing. **54**(11), 4311–4322 (2006). https://doi.org/10.1109/TSP.2006.881199
16. Khmag, A., Ramli, A.R., Hashim, S.J., Al-Haddad, S.A.R.: Review of Image Denoising Algorithms Based on the Wavelet Transformation. International Journal of Advanced Trends in Computer Science and Engineering (IJATCSE). **2**(5), 1–8 (2013)
17. Jain, V., Seung, H.S.: Natural image denoising with convolutional networks. In: Abstracts of the 21st international conference on neural information processing systems, ACM, Vancouver, pp. 769–776 (2008)
18. Goyal, B., Dogra, A., Agrawal, S., Sohi, B.S., Sharma, A.: Image denoising review: From classical to state-of-the-art approaches. Information Fusion. **55**, 220–244 (2020). https://doi.org/10.1016/j.inffus.2019.09.003
19. Shengjie, C., Shuo, C., Zhenhua, G., Yushen, Z.: Low-resolution palmprint image denoising by generative adversarial networks. Neurocomputing. **358**, 275–284 (2019). https://doi.org/10.1016/j.neucom.2019.05.046
20. Zhang, Y.V., Young, B., Gqamana, P.P., Anderson, W.B., Wu, A.H.B.: Mass spectrometry. In: Self-Assessment Q&A in Clinical Laboratory Science, III, ch. 16, pp. 195–206 (2021).https://doi.org/10.1016/B978-0-12-822093-1.00016-8
21. Zheng, X., Yuan, Y., Lu, X.: Hyperspectral image denoising by fusing the selected related bands. IEEE Transactions on Geoscience and Remote Sensing. **57**(5), 2596–2609 (2018). https://doi.org/10.1109/TGRS.2018.2875304
22. Zhang, K., Wangmeng, Z., Chen, Y., Meng, D., Zhang, L.: Beyond a gaussian denoiser: residual learning of deep CNN for image denoising. IEEE Transactions on Image Processing. **26**(7), 3142–3155 (2017). https://doi.org/10.1109/TIP.2017.2662206
23. Bank, D., Koenigstein, N., Giryes, R.: Autoencoders. (2020). https://doi.org/10.48550/arXiv.2003.05991 arXiv:2003.05991
24. Bello, I.: LambdaNetworks: modeling long-range interactions without attention. (2021). https://doi.org/10.48550/arXiv.2102.08602 arXiv:abs/2102.08602
25. Bjerrum, E.J., Glahder, M., Skov, T.: Data augmentation of spectral data for convolutional neural network (CNN) based deep chemometrics. (2017). arXiv:1710.01927
26. Liu, J., Osadchy, M., Ashton, L., Foster, M., Solomon, C.J., Gibson, S.J.: Deep convolutional neural networks for raman spectrum recognition: a unified solution. Analyst. **142**, 4067–4074 (2017). https://doi.org/10.1039/C7AN01371J
27. Arad, O., Klapp, I.: Towards, multi-purpose system for spatial and hyperspectral sampling of crop from a moving platform. Light, Energy and the Environment 2018 (E2, FTS, HISE, SOLAR, SSL). OSA Technical Digest (Optica Publishing Group, 2018), paper ET4A.5 (2018).https://doi.org/10.1364/EE.2018.ET4A.5
28. Arad, O., Klapp, I.: Dispersion analysis of a low cost hyper-spectral imaging system based on Risley prism scanner. Optical Sensors and Sensing Congress. OSA Technical Digest (Optica Publishing Group, 2020), paper EM2C.6 (2020).https://doi.org/10.1364/ES.2020.EM2C.6
29. Arad, O.: Multi-purpose system for spatial and spectral sampling of crop from a moving platform. M.S. thesis, Ben-Gurion University, Israel (2020)
30. Konnik, M., Welsh, J.: High-level numerical simulations of noise in CCD and CMOS photosensors: review and tutorial. (2014).https://doi.org/10.48550/arXiv.1412.4031 arXiv:1412.4031
31. Lesser, M.: Charge coupled device (CCD) image sensors. In: High Performance Silicon Imaging, ch. 3, pp. 78–97 (2014).https://doi.org/10.1533/9780857097521.1.78
32. Frischia, S.D., Chiuri, A., Angelini, F., Colao, F.: Optimization of signal-to-noise ratio in a CCD for spectroscopic applications. In: Proceedings of 15th European Conference on Advanced Control and Diagnosis (ACD 2019), pp. 439–452. Bologna, Italy (2022)
33. Vincent, P., Larochelle, H., Bengio, Y., Manzagol, P.A.: Extracting and composing robust features with denoising autoencoders. In: Proceedings of 25th International Conference on Machine Learning, pp. 1096–1103 (2008)
34. Vaswani, A., Shazeer, N., Parmar, N., Uszkoreit, J., Jones, L., Gomez, A.N., Kaiser, L., Polosukhin, I.: Attention is all you need. (2017). https://doi.org/10.48550/arXiv.1706.03762 arXiv:abs/1706.03762
35. Zhao, H., Gallo, O., Frosio, I., Kautz, J.: Loss functions for neural networks for image processing. (2018). https://doi.org/10.48550/arXiv.1511.08861 arXiv:1511.08861
36. Kullback, S., Leibler, R.A.: On information and sufficiency. In: The Annals of Mathematical Statistics **22**(1), 79–86 (1951)
37. Shannon, C.E.: A mathematical theory of communication. In: The Bell System Technical Journal, vol. 27, pp. 379–423, 623–656 (1948)
38. Kerridge, D.R.: Inaccuracy and inference. In: Journal of the Royal Statistical Society: Series B (Methodological) **23**, 184–194 (1961)
39. Horé, A., Ziou, D.: Image quality metrics: PSNR vs. SSIM. In: Proceedings 20th International Conference on Pattern Recognition, pp. 2366–2369 (2010)
40. Wang, Z., Bovik, A.C., Sheikh, H.R., Simoncelli, E.P.: Image quality assessment: from error visibility to structural similarity. IEEE Transactions on Image Processing. **13**(4), 600–612 (2004)
41. Kingma, D.P., Ba, J.: Adam: a method for stochastic optimization.(2014). arXiv:1412.6980