AN ADMM BASED NETWORK FOR HYPERSPECTRAL UNMIXING TASKS

Chao Zhou, Miguel R.D. Rodrigues

Dept. Electronic and Electrical Engineering, University College London
{chao.zhou.18, m.rodrigues}@ucl.ac.uk

ABSTRACT

In this paper, we use algorithm unrolling approaches in order to design a new neural network structure applicable to hyperspectral unmixing challenges. In particular, building upon a constrained sparse regression formulation of the underlying unmixing problem, we unroll an ADMM solver onto a neural network architecture that can be used to deliver the abundances of different (known) endmembers given a reflectance spectrum. Our proposed network – which can be readily trained using standard supervised learning procedures – is shown to possess a richer structure consisting of various skip connections and shortcuts than other competing architectures. Moreover, our proposed network also delivers state-of-the-art unmixing performance compared to competing methods.

Index Terms— HSI unmixing, Deep Neural Networks, Algorithm Unrolling, Algorithm Unfolding

1. INTRODUCTION

The hyperspectral image (HSI) unmixing problem involves decomposing a spectral signature onto its constituent spectrum, also known as endmembers, and their abundance. It arises in many applications where there is a need to understand materials of a scene/sample such as remote sensing [1, 2], art investigation [3], and many more.

The unmixing procedure [2] typically involves endmember extraction followed by abundance estimation. Many approaches [4, 5] have been proposed based on this procedure. In particular, lots of methods focus on abundance estimation for settings where the endmembers are known.

Model-based approaches leverage a mathematical formulation of the underlying mixing problem – such as a linear mixing model (LMM) [6] – in order to develop unmixing algorithms that can estimate the abundance of the various endmembers given a reflectance spectrum. Popular model-based methods leveraging sparsity driven techniques include constrained sparse regression (CSR) problems [7] – leveraging the fact that abundance vectors tend to be sparse –that can be solved by using iterative soft thresholding algorithms (ISTA) [8] or alternative direction method of multipliers (ADMM) [9] algorithms. The disadvantage of this class of approaches relates to computational complexity deriving from the fact that the underlying iterative process tends to be slow.

Recently, with the surge of machine learning, a number of data-driven approaches have also been proposed to tackle the unmixing challenge. In a supervised learning algorithm, one presents the machine learning model with a set of pairs of HSI reflectances along with the abundances of different endmembers in order to learn a function that is able to map a new unseen reflectance to corresponding abundances. For example, reference [10] proposed a two-stage approach based on the use of an auto-associative neural network for dimensionality reduction purposes followed by a new multi-layer perceptron for unmixing. Reference [11] instead proposed two end-to-end unmixing networks based on convolution neural networks (CNN), including a pixel-based CNN and cube-based CNN respectively. Gewali et al. [12] overviews various other supervised machine learning approaches – such as support vector machines – to the HSI unmixing challenge. However, the disadvantage of learning-based approaches derives from the need for substantial labelled datasets.

Notably, [8, 13] have proposed to use the emerging notion of algorithm unrolling or unfolding – which combines ideas deriving from model-based approaches with data-driven ones [14, 15] – for HSI unmixing purposes. In particular, based on a typical sparsity driven unmixing approach, these works have proposed to unfold an ISTA based solver to the CSR problem onto a deep network architecture that can be further trained to solve the unmixing challenge. However, it is well known that ISTA based solvers can underperform in relation to other solvers such as ADMM [16]. Therefore, in this paper, we propose instead to unfold an ADMM based solver to the CSR problem onto a network architecture to solve unmixing challenges.

This paper is organized as follows: In Section 2, we introduce the CSR model for abundance estimation. In Section 3, we propose to unfold ADMM into learning structure. In Section 4, we propose the new ADMM based network for abundance estimation along with its training and initialisation strategies. Section 5 instead illustrates that our proposed network outperforms existing ones in unmixing tasks. Conclusions are drawn in Section 6.

Thanks to China Scholarship Council for funding.

2. MODEL

We concentrate on LMM [6] given by:

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

where $\boldsymbol{y} = [y_1, \ldots, y_{N_B}]^T \in \mathbb{R}^{N_B \times 1}$ is a HSI reflectance vector across N_B bands for a given pixel, $\boldsymbol{x} = [x_1, \ldots, x_{N_E}]^T \in \mathbb{R}^{N_E \times 1}$ is corresponding abundance vector containing the proportion of each endmember, and $\boldsymbol{n} = [n_1, \ldots, n_{N_B}]^T \in \mathbb{R}^{N_B \times 1}$ is the additive noise. The matrix $\boldsymbol{A} = [\boldsymbol{a}_1, \ldots, \boldsymbol{a}_{N_E}] \in \mathbb{R}^{N_B \times N_E}$ – which is assumed to be known as in [9] –is an endmember signature matrix, i.e. $\boldsymbol{a}_m \in R^{N_B \times 1}$ models the signature of the m^{th} endmember ($m = 1, \ldots, N_E$). Note that the abundance vector must meet abundance nonnegative constraint (ANC), $\boldsymbol{x} \geq \boldsymbol{0}$, and abundance sum-to-one constraint (ASC), $\sum_{m=1}^{N_E} x_m = 1$.

Given that the mixing matrix $\mathbf{A}$ is often available in the form of a spectral library, and the abundance vector tends to be sparse, one can adopt a CSR based optimization problem in order to recover the abundance vector from the reflectance vector as follows [9]:

$$\min_{\boldsymbol{x}} \frac{1}{2}||\boldsymbol{y} - \boldsymbol{A}\boldsymbol{x}||_2 + \lambda||\boldsymbol{x}||_1, s.t., \boldsymbol{x} \geq \boldsymbol{0} \tag{2}$$

where $||\cdot||_2$ and $||\cdot||_1$ denote the ℓ_2-norm and ℓ_1-norm respectively, whereas $\lambda \geq 0$ is a regularization parameter that controls the sparsity of solutions.

The CSR optimization problem in (2) – which has led to a state-of-the-art performance in HSI unmixing over the years – can be solved using a range of solvers such as ISTA [8] and ADMM [9]. We next show that – by building upon the well-known ADMM solver – we can map this problem onto a network architecture that can be trained to improve HSI unmixing performance further.

3. ADMM-BASED UNFOLDING OF CONSTRAINED SPARSE REGRESSION

3.1. ADMM-Based CSR Solver

Our approach to HSI unmixing builds upon unfolding an ADMM solver to the CSR problem. In particular, by introducing an auxiliary variable $\boldsymbol{z}$ such that $\boldsymbol{x} = \boldsymbol{z}$ along with a dual variable $\boldsymbol{d}$, ADMM leads to an iterative scheme to compute the solution of the CSR problem appearing in (2) as follows:

$$\boldsymbol{x}_{k+1} = (\boldsymbol{A}^T\boldsymbol{A} + \mu\boldsymbol{I})^{-1}(\boldsymbol{A}^T\boldsymbol{y} + \mu(\boldsymbol{z}_k + \boldsymbol{d}_k)) \tag{3}$$

$$\boldsymbol{z}_{k+1} = \max\left(soft\left(\boldsymbol{x}_{k+1} - \boldsymbol{d}_k, \frac{\lambda}{\mu}\right), 0\right) \tag{4}$$

$$\boldsymbol{d}_{k+1} = \boldsymbol{d}_k - (\boldsymbol{x}_{k+1} - \boldsymbol{z}_{k+1}) \tag{5}$$

where, $\boldsymbol{x}_k$ (resp. $\boldsymbol{z}_k, \boldsymbol{d}_k$) is the value of variable $\boldsymbol{x}$ (resp. $\boldsymbol{z}, \boldsymbol{d}$) at k^{th} iteration, and $\mu \geq 0$ is a parameter that is usually chosen to be an upper bound to the largest eigenvalue of $\boldsymbol{A}^T\boldsymbol{A}$.

It is clear that this iterative scheme – involving different intertwined operations – can be mapped onto different components of a neural network architecture using algorithm unrolling techniques [14]. The different components of the neural network layers are described in the sequel.

3.2. Neural Network Layer Components

We first define each constituent of a neural network layer by unrolling the three different iterative operations appearing in (3), (4), and (5).

X-Update Component This component of the $(k+1)^{th}$ layer is obtained by unfolding (3). In particular, by noting that

$$\boldsymbol{x}_{k+1} = \boldsymbol{W}^T\boldsymbol{y} + \boldsymbol{B}^T(\boldsymbol{z}_k + \boldsymbol{d}_k) \tag{6}$$

where $\boldsymbol{W}^T = (\boldsymbol{A}^T\boldsymbol{A} + \mu\boldsymbol{I})^{-1}\boldsymbol{A}^T, \boldsymbol{B}^T = (\boldsymbol{A}^T\boldsymbol{A} + \mu\boldsymbol{I})^{-1}\mu$, we can then re-express (6) as follows:

$$\begin{aligned}\boldsymbol{x}_{k+1} &= \boldsymbol{W}_{k+1}^T\boldsymbol{y} + \boldsymbol{B}_{k+1}^T(\boldsymbol{z}_k + \boldsymbol{d}_k) \\ &:= f_X(\boldsymbol{z}_k, \boldsymbol{d}_k, \boldsymbol{y}; \boldsymbol{W}_{k+1}, \boldsymbol{B}_{k+1})\end{aligned} \tag{7}$$

where we have introduced the learnable parameters $\boldsymbol{W}_{k+1} \in \mathbb{R}^{N_E \times N_B}$ and $\boldsymbol{B}_{k+1} \in \mathbb{R}^{N_E \times N_E}$ that can differ from the original ones $\boldsymbol{W}$ and $\boldsymbol{B}$ in order to better adapt to the characteristics of the data.

Z-update component This component of $(k+1)^{th}$ layer is obtained by unfolding (4). In particular, we propose to re-express (4) as follows:

$$\begin{aligned}\boldsymbol{z}_{k+1} &= \max\left(soft\left(\boldsymbol{x}_{k+1} - \boldsymbol{d}_k, \theta_{k+1}\right), 0\right) \\ &:= f_Z(\boldsymbol{x}_{k+1}, \boldsymbol{d}_k; \theta_{k+1})\end{aligned} \tag{8}$$

where $\theta_{k+1} \in \mathbb{R}$ is a learnable parameter, that can also differ from the original parameter $\frac{\lambda}{\mu}$ in order to better adapt to the characteristics of the unmixing problem.

D-update component Finally, this component of the $(k+1)^{th}$ layer is obtained by unfolding (5). In particular, propose to re-express (5) as follows:

$$\begin{aligned}\boldsymbol{d}_{k+1} &= \boldsymbol{d}_k - \eta_{k+1}(\boldsymbol{x}_{k+1} - \boldsymbol{z}_{k+1}) \\ &:= f_D(\boldsymbol{x}_{k+1}, \boldsymbol{z}_{k+1}, \boldsymbol{d}_k; \eta_{k+1})\end{aligned} \tag{9}$$

where η_{k+1} is also a learnable parameter offering additional flexibility.

3.3. Overall Neural Network Layer

We can now define the network layer by intertwining the above components. See Fig. 1.

Such a layer corresponds to one iteration of the original ADMM algorithm. However, each layer is associated with learnable parameters $\boldsymbol{\Theta}_k = \{\boldsymbol{W}_k, \boldsymbol{B}_k, \theta_k, \eta_k\}$, whereas each iteration in the ADMM algorithm is associated with hand-coded fixed parameters $\{\boldsymbol{A}, \mu, \lambda\}$. One of the advantages of using such a learnable parameterization is that, the resulting

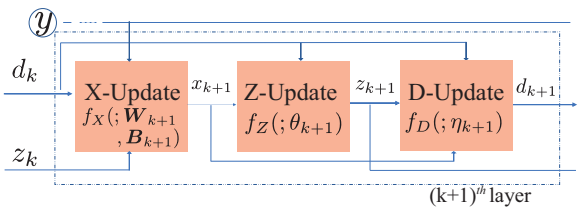


Fig. 1. Neural network layer unfolding from ADMM.

network has less number of layers than the number of iterations in the ADMM, thus lifting the computational burden. In addition, the resulting K-layers network has $(N_E^2 + N_E N_B + 2)K$ learnable parameters, less than millions of parameters in conventional HSI unmixing networks [11].

It is also instructive to comment on how different a network layer deriving from ADMM is compared to that from ISTA [13]. In particular, the proposed layer contains far more short cuts and skip connections [17] compared with ISTA based layers, including skip connections between different components of each layer (see Fig. 1). Such shortcuts have been shown to be beneficial during network training [18] and image processing tasks [19].

4. ADMM BASED ABUNDANCE ESTIMATION NETWORK

We next describe how to utilise the various components to construct a neural network to estimate abundances, as well as training and initialisation procedures.

4.1. Network Construction

The proposed network (designated U-ADMM-AENet) results from concatenating K layers – consisting of *X-Update*, *Z-Update*, and *D-Update* components – where the learnable parameters are untied across layers, that is, for layer k, we use layer specific parameters $\boldsymbol{\Theta}_k = \{\boldsymbol{W}_k, \boldsymbol{B}_k, \theta_k, \eta_k\}, \forall k \in [1, K])$. We set $K = 2$ as it has been experimentally verified that it has very little influence on the performance. This untied parameterisation is the key to increase the network capacity and flexibility, leading up to improved unmixing results.

In line with the standard ADMM algorithms, we let the network (pseudo) inputs be such that $\boldsymbol{z}_0 = 0$ and $\boldsymbol{d}_0 = 0$. We also let the network output to be derived from the *Z-Update* component in the last layer because this component guarantees compliance with the ANC constraint. We also further normalize the network output to meet the ASC constraint.

4.2. Training Procedure

The neural network is trained using a supervised learning approach, by leveraging access to a training set $D = \{\boldsymbol{y}_i, \boldsymbol{x}_i\}_{i=1}^N$ consisting of N reflectance spectra $\boldsymbol{y}_i$ and corresponding abundances $\boldsymbol{x}_i$. Instead of the commonly used MSE loss, we propose to use a composite loss given by:

$$L_{\boldsymbol{\Theta}} = \alpha_1 \cdot L_1 + \alpha_2 \cdot L_2 + \alpha_3 \cdot L_3 \tag{10}$$

where α_1, α_2, and α_3 are hyper-parameters controlling the contribution of corresponding loss terms.

The first component of the loss function derives from the standard mean-squared error (MSE) given by:

$$L_1 = \frac{1}{N} \sum_{\{\boldsymbol{y}_i, \boldsymbol{x}_i\} \in D} \|\boldsymbol{x}_i - \tilde{\boldsymbol{x}}_i(\boldsymbol{y}_i)\|_2^2 \tag{11}$$

where $\tilde{\boldsymbol{x}}_i(\boldsymbol{y}_i)$ corresponds to the network estimate of $\boldsymbol{x}_i$ given $\boldsymbol{y}_i$.

The second component of the loss function derives from the angle distance as follows:

$$L_2 = \frac{1}{N} \sum_{\{\boldsymbol{y}_i, \boldsymbol{x}_i\} \in D} \cos^{-1}\left(\frac{\boldsymbol{x}_i^T \tilde{\boldsymbol{x}}_i(\boldsymbol{y}_i)}{\|\boldsymbol{x}_i\|_2 \|\tilde{\boldsymbol{x}}_i(\boldsymbol{y}_i)\|_2}\right) \tag{12}$$

The third component of the loss function derives instead from the information divergence as follows:

$$L_3 = \frac{1}{N} \sum_{\{\boldsymbol{y}_i, \boldsymbol{x}_i\} \in D} KL(\boldsymbol{x}_i | \tilde{\boldsymbol{x}}_i(\boldsymbol{y}_i)) + KL(\tilde{\boldsymbol{x}}_i(\boldsymbol{y}_i) | \boldsymbol{x}_i) \tag{13}$$

where, $KL(\cdot|\cdot)$ is KL divergence, which is well-defined because abundance vectors can be seen as a distribution because of the ANC and ASC constraints.

The additional dissimilarity measures enable networks to gauge how different the recovered abundance is from the true one. By adopting stochastic gradient descent algorithms, the proposed network can be trained to minimize the proposed loss function $L_{\boldsymbol{\Theta}}$, where, via cross-validation techniques, we empirically set $\alpha_1 = 1.0, \alpha_2 = 1e-7, \alpha_3 = 1e-5$.

4.3. Initialisation Approach

The networks are also trained by adopting warm initialisation, in order to speed up the training procedure. In particular, as we assume knowledge of the true endmembers, the parameters $\mathbf{W}_k$ and $\mathbf{B}_k$ are initialised by using definitions of $\boldsymbol{W}$, $\boldsymbol{B}$ in Section 3. Similarly, θ_k is initialised by setting it to be equal to $\frac{\lambda}{\mu}$. Finally, η_k is initialised to one. We note with this initialisation strategy a K-layer U-ADMM-AENet corresponds exactly to K iterations of the ADMM algorithm.

5. EXPERIMENTS

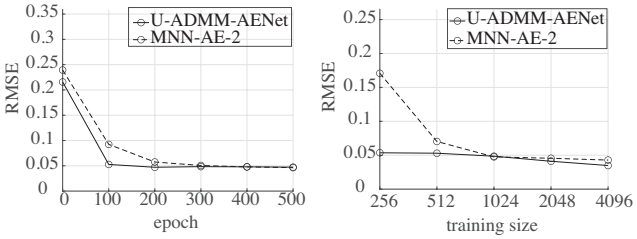
We now compare the performance of the proposed ADMM based unmixing network to the performance of the ISTA based unmixing network [8] – referred to as MNN-AE-2 – and the performance of SunSAL [9]. In particular, we also use the RMSE between the true abundance and the recovered one to measure the performance of the different approaches.

5.1. Evaluation on Synthetic Data

We adopt the procedure in [8] to generate the synthetic dataset. Six endmember signatures are randomly chosen from the USGS spectral library. We then divide a synthetic image of size 100×100 pixels into 100 disjoint patches, each of which we assign a spectrum mixed from two randomly selected endmembers with fractions $[0.8, 0.2]$. Finally, the abundance map is convolved with a Gaussian filter of size 11×11, followed by further re-scaling in order to meet the ASC constraint per pixel. In addition, the data is contaminated with additive white Gaussian noise (AWGN).

By default, the training data consists of 1000 randomly chosen pixels from the synthetic HSI data, which is polluted with AWGN leading to $SNR = 15$ dB, while the remaining data is used for evaluation. The network consists of two layers and is trained using ADAM optimizer [20] with learning rate set to $1e-4$, a batch size set to 64, and a number of epochs set to 300.

Performance *vs*. Training Epochs We first assess the convergence performance of the various approaches as a function of the number of training epochs. We use the default experiment settings but the number of epochs varies from 0 to 500. As shown in Fig. 2(a), the proposed method can achieve faster convergence than MNN-AE. We attribute this to the fact that ADMM based solvers typically converge faster than ISTA based ones. We also attribute this to the fact the weighted loss function adopted in our learning algorithms is more complex than that adopted in competing ones [13], allowing to promote additional dissimilarity.



(a) RMSE vs. Epoch (b) RMSE vs. Train size

Fig. 2. Performance versus various factors.

Performance *vs*. Training Data We now assess the performance of various approaches as a function of the number of training data. We use the default experiment settings but the training data size ranges from 256 up to 4096. Per Fig. 2(b) it is clear that our proposed methods can achieve better performance compared to competitors in the presence of smaller training datasets. Specifically, when the size of training dataset is very small (e.g. 256), U-ADMM-AENet outperforms MNN-AE a lot. This may be due to the fact that the network deriving from ADMM have much more residual connections in relation to networks deriving from ISTA.

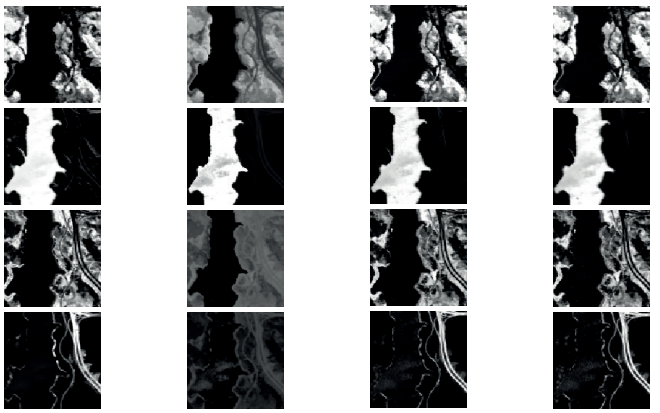
5.2. Evaluation on Real data

We also evaluate our method on real HSI datasets, Jasper Ridge [21]. There are four different endmembers in the scene: Road, Soil, Water, and Tree. We consider 100×100 pixels sub-images of the original images due to complexity considerations. After removing bands due to water vapour effects, 198 bands are used in the experiment. We use the default training settings used in synthetic data experiments, except that the training data size is set to be equal to 256.

Table 1. Averaged RMSE by Different Algorithms.

	SunSAL	MNN-AE-2	U-ADMM-AENet
RMSE	0.0612	0.1262	0.0214

The performance of various algorithms is reported in Table 1. It can be seen that the proposed U-ADMM-AENet achieves the best performance. A qualitative result – abundance map – is also shown in Fig. 3. It can also be seen that the proposed network leads to abundance maps closer to the ground truth in comparison to competing algorithms. We have also done more experiments to validate the superior of the proposed network, which are not shown due to limited space.



(a) SunSAL [9] (b) MNN-AE2 [8] (c) Proposed (d) Reference

Fig. 3. Abundance estimation maps. From top to bottom: Tree, Water, Soil and Road.

6. CONCLUSION

We have proposed a new HSI unmixing networks deriving from unfolding procedures. In particular, we have shown how ADMM leads to neural network architectures that possess very rich structures, thereby state-of-the-art unmixing performance in comparison to existing approaches. We have also proposed a new weighted loss function to better fit in HSI applications. We demonstrate the effectiveness of the proposed algorithm both on a synthetic and real dataset.

7. REFERENCES

[1] N. Keshava and J. F. Mustard, "Spectral unmixing," in *IEEE Signal Processing Magazine*, vol. 19, no. 1, pp. 44-57, Jan. 2002.

[2] G. Camps-Valls; D. Tuia; L. Gómez-Chova; S. Jiménez, "Remote Sensing Image Processing," in *Remote Sensing Image Processing*, Morgan & Claypool, 2011.

[3] N. Rohani, E. Pouyet, M. Walton, O. Cossairt and A. K. Katsaggelos, "Pigment Unmixing of Hyperspectral Images of Paintings Using Deep Neural Networks," in *2019 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, Brighton, United Kingdom, 2019, pp. 3217-3221.

[4] J. M. P. Nascimento and J. M. B. Dias, "Vertex component analysis: a fast algorithm to unmix hyperspectral data," in *IEEE Transactions on Geoscience and Remote Sensing*, vol. 43, no. 4, pp. 898-910, April 2005.

[5] D. C. Heinz and Chein-I-Chang, "Fully constrained least squares linear spectral mixture analysis method for material quantification in hyperspectral imagery," in *IEEE Transactions on Geoscience and Remote Sensing*, vol. 39, no. 3, pp. 529-545, March 2001.

[6] J. M. Bioucas-Dias, A. Plaza, G. Camps-Valls, P. Scheunders, N. Nasrabadi and J. Chanussot, "Hyperspectral Remote Sensing Data Analysis and Future Challenges," in *IEEE Geoscience and Remote Sensing Magazine*, vol. 1, no. 2, pp. 6-36, June 2013.

[7] M. Iordache, J. M. Bioucas-Dias and A. Plaza, "Sparse Unmixing of Hyperspectral Data," in *IEEE Transactions on Geoscience and Remote Sensing*, vol. 49, no. 6, pp. 2014-2039, June 2011.

[8] Q. Qian, F. Xiong and J. Zhou, "Deep Unfolded Iterative Shrinkage-Thresholding Model for Hyperspectral Unmixing," *IGARSS 2019 - 2019 IEEE International Geoscience and Remote Sensing Symposium*, Yokohama, Japan, 2019, pp. 2151-2154.

[9] J. M. Bioucas-Dias and M. A. T. Figueiredo, "Alternating direction algorithms for constrained sparse regression: Application to hyperspectral unmixing," *2010 2nd Workshop on Hyperspectral Image and Signal Processing: Evolution in Remote Sensing*, Reykjavik, 2010, pp. 1-4.

[10] G. A. Licciardi and F. D. Frate, "Pixel Unmixing in Hyperspectral Data by Means of Neural Networks," in *IEEE Transactions on Geoscience and Remote Sensing*, vol. 49, no. 11, pp. 4163-4172, Nov. 2011.

[11] X. Zhang, Y. Sun, J. Zhang, P. Wu and L. Jiao, "Hyperspectral Unmixing via Deep Convolutional Neural Networks," in *IEEE Geoscience and Remote Sensing Letters*, vol. 15, no. 11, pp. 1755-1759, Nov. 2018.

[12] U. B. Gewali, S. T. Monteiro, and E. Saber, "Machine learning based hyperspectral image analysis: a survey," *arXiv:1802.08701*, 2018.

[13] Y. Qian, F. Xiong, Q. Qian and J. Zhou, "Spectral Mixture Model Inspired Network Architectures for Hyperspectral Unmixing," in *IEEE Transactions on Geoscience and Remote Sensing*, vol. 58, no. 10, pp. 7418-7434, Oct. 2020.

[14] V. Monga, Y. Li, and Y. C. Eldar, "Algorithm unrolling: Interpretable, efficient deep learning for signal and image processing", *arXiv preprint arXiv:1912.10557*, Dec. 2019.

[15] K. Gregor and Y. LeCun, "Learning fast approximations of sparse coding," in *Proc. 27th Int. Conf. Machine Learning (ICML-10)*, 2010, pp. 399–406.

[16] S. Z. Tao, D. Boley, and S. Z. Zhang. "Convergence of common proximal methods for l1-regularized least squares," in *Proceedings of the 24th International Conference on Artificial Intelligence (IJCAI'15)*, AAAI Press, 2015, PP. 3849–3855.

[17] K. He, X. Zhang, S. Ren and J. Sun, "Deep Residual Learning for Image Recognition," *2016 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, Las Vegas, NV, 2016, pp. 770-778.

[18] B. Lim, S. Son, H. Kim, S. Nah and K. M. Lee, "Enhanced Deep Residual Networks for Single Image Super-Resolution," *2017 IEEE Conference on Computer Vision and Pattern Recognition Workshops (CVPRW)*, Honolulu, HI, 2017, pp. 1132-1140.

[19] K. Zhang, W. Zuo, Y. Chen, D. Meng and L. Zhang, "Beyond a Gaussian Denoiser: Residual Learning of Deep CNN for Image Denoising," in *IEEE Transactions on Image Processing*, vol. 26, no. 7, pp. 3142-3155, July 2017.

[20] D. P. Kingma and J. Ba, "Adam: A method for stochastic optimization," *arXiv preprint arXiv:1412.6980*, Jan. 2017.

[21] F. Zhu, Y. Wang, S. Xiang, B. Fan, and C. Pan, "Structured Sparse Method for Hyperspectral Unmixing," *ISPRS Journal of Photogrammetry and Remote Sensing*, vol. 88, pp. 101–118, 2014.