PIGMENT UNMIXING OF HYPERSPECTRAL IMAGES OF PAINTINGS USING DEEP NEURAL NETWORKS

Neda Rohani[⋆]*, Emeline Pouyet*[⋆⋆]*, Marc Walton*[⋆⋆]*, Oliver Cossairt*[⋆]*, Aggelos K. Katsaggelos*[⋆]

[⋆] Department of Electrical Engineering and Computer Science, Northwestern University, Evanston, IL, USA
[⋆⋆] Center for Scientific Studies in the Art, Northwestern University, Evanston, IL, USA

ABSTRACT

In this paper, the problem of automatic nonlinear unmixing of hyperspectral reflectance data using works of art as test cases is described. We use a deep neural network to decompose a given spectrum quantitatively to the abundance values of pure pigments. We show that adding another step to identify the constituent pigments of a given spectrum leads to more accurate unmixing results. Towards this, we use another deep neural network to identify pigments first and integrate this information to different layers of the network used for pigment unmixing. As a test set, the hyperspectral images of a set of mock-up paintings consisting of a broad palette of pigment mixtures, and pure pigment exemplars, were measured. The results of the algorithm on the mock-up test set are reported and analyzed.

Index Terms— Hyperspectral imaging, nonlinear unmixing, pigment identification, deep neural network, fusion.

1. INTRODUCTION

Pigment identification and mapping are used by conservation scientists to elucidate artist/workshop use of materials, to understand how a painted surface changed over time informing how an artwork is to be conserved, and, lastly, to identify anachronistic uses of materials that could be associated with past restoration. A primary tool for these tasks is hyperspectral imaging reflectance spectroscopy (HSI), a non-invasive and non-destructive method that has become common place in cultural heritage [1]. It can be applied *in-situ* to examine how pigments are distributed across a painted surface.

One of the primary challenges to use HSI for materials identification is that the reflectance response of a heterogeneous mixture of pigments is a nonlinear function of the responses of the individual pigments. Pigment identification refers to qualitatively decomposing a given spectrum to its pure pigments which can be modeled as a classification problem. Pigment unmixing refers to quantitatively decomposing a given spectrum into its basic elements and can be modeled as a regression problem. This fact has long been understood and described analytically by Kubelka-Munk (KM) theory [2]. Using this KM model, reflectance spectra collected from a painted surface are fitted to dictionaries of pure compounds [3]. While this approach is computationally straightforward for pigment identification/quantification, its application is limited to single-point analyses, rather than every pixel of a hyperspectral image cube, due to extended memory requirements and computation times necessary to undertake these KM calculations [1].

Deep neural networks (DNNs) have been proven to be successful in determining the nonlinear mappings between an input and the corresponding output automatically. In remote sensing and hyperspectral imaging, different architectures of neural networks have been used in many applications such as feature extraction [4, 5], classification [6, 7, 8] and unmixing [9]. Despite the numerous advantages of DNNs, they have not been applied widely to cultural heritage and only few works have utilized such advanced techniques, such as, [10, 11]. Both of these methods use DNNs for pigment identification. In addition to this, we propose to use a DNN for pigment unmixing as well. This network is trained to learn automatically the nonlinear pigment mixing function using the Kullback-Leibler divergence loss function and is used for the pigment unmixing task. This model can be applied to other nonlinear mixing functions as well, under the condition of having the training sets being built according to the mixing function.

2. THE PIGMENT UNMIXING PROBLEM

We assume that the painting under investigation I consists of N pixels and each pixel x is represented by P wavelengths, i.e., $I \in \mathbb{R}^{N\times P}$ and $x \in \mathbb{R}^P$. Furthermore, we assume that the painting is composed of M pure pigments . We refer to the set of M pure pigments as E. Let x be the hyperspectral value of a pixel in the image I with an unknown composition. We are interested in decomposing each pixel's spectrum as a function of the set of pure pigments E, according to

$$x = f(E, \alpha), \quad \text{s.t.} \quad \sum_{i=1}^{M} \alpha_i = 1 \quad \text{and} \quad \alpha_i \geq 0, \qquad (1)$$

where the function f can be linear or nonlinear and the vector $\alpha \in \mathbb{R}^M$ represents the concentration values of the pure pigments in the representation of the unknown spectrum x. It has been shown in the literature that the mixing model of the spectra of pigments is defined by a nonlinear function based on KM theory [12]. In [13, 11], we use sparse linear and nonlinear KM functions, respectively, to decompose any given

spectrum into its basic elements. We show that the sparse unmixing algorithm yields more accurate results than the linear unmixing algorithm using a fully constrained least square method [14]. In [11], we first detect the pigments in a given spectrum using a DNN and then by a using gradient-based algorithm [15], we estimate the coefficients of the detected pigments for the given spectrum. Here, we propose to utilize a DNN to learn the pigment nonlinear mixing function and find the coefficients of the pigments, α, in a given spectrum.

3. PROPOSED MODEL

As mentioned in section 2, to find the concentration values of the pigments in any spectrum, a DNN is trained. The proposed DNN is a feed-forward network, composed of four fully connected layers with 256, 128, 64 and 32 number of hidden nodes, with relu/sigmoid nonlinear activation functions (Fig. 1(a)). The softmax nonlinear function is used as the activation function of the last layer with $M = 11$ number of hidden nodes, so that the summing up to one constraint on the coefficients holds (Eq. 1). The input to the network is the spectrum vector of a pixel and the output is the estimated coefficient vector α of pigment concentrations for the given spectrum.

To obtain a more precise result, we can first identify the pigments for any unknown spectrum and use this information to decompose the given spectrum into the concentration values of the detected pigments. As shown previously in [11] (the supporting material, section F, table 4), the performance of the pigment unmixing algorithm improves when prior determination of expected pigments is used. To identify the pigments for a given spectrum, we use a deep feed-forward network [11]. Pigment identification can be modeled as a multi-label classification [16] problem. Here, each spectrum can have multiple binary labels due to the presence of multiple pigments in it. The input to the network is the spectrum and the output is a vector of zeros and ones, $y \in \{0,1\}^M$. In this network, in the last dense layer, the sigmoid nonlinear activation function is used and the objective loss function of the network is a *binary cross-entropy* based loss. The details of the network are given in [11] which is very similar to the network in Fig. 1 (a) (the same number of layers and hidden nodes in each layer with different output).

The detected labels, y, can be used as an extra input to the network, an "auxiliary input" in Fig. 1 (a) and can be merged with different layers of the network as shown in Figs. 1 (b), (c) and (d). In Fig. 1(b), *Early* fusion the labels (a vector of $M = 11$ zeros and ones) are concatenated to the spectrum input (of dimension 56) in the first layer resulting in an input with a dimension of 67 where the first 56 terms represent the spectrum and the last 11 terms the labels. In Fig. 1 (c), *Intermediate* fusion the labels are concatenated to the output of the third layer. Lastly, in Fig. 1 (d), *Late* fusion the output of the fifth layer is multiplied by the labels. Finally, another dense layer with $M = 11$ hidden nodes is added to the network in Fig.1 (d) to estimate the concentration values. It has been verified by our experiments that the performance of the network enhances when the labels are integrated into the outputs of the higher layers of the network (Fig. 2).

For training the networks proposed for pigment unmixing, we can use the Mean Squared Error (MSE), the cosine error or the *Kullback-Leibler Divergence* (KLD) as the loss function. Due to constraint on the coefficient values to sum up to one (Eq. 1), we consider the true and predicted coefficient values of the pigments as belonging to two probability distributions. Therefore, KLD would be an appropriate loss function for comparing the similarity of these two probability distributions. Based on our simulations, we have observed that the KLD loss function leads to a better performance than the MSE or the cosine loss functions. The KLD based loss function is defined as follows:

$$L = -\sum_{n=1}^{N}\sum_{i=1}^{M} \alpha_i^n \log(\frac{\hat{\alpha}_i^n}{\alpha_i^n}) \tag{2}$$

where $\hat{\alpha}_i^n$ and α_i^n are the *predicted* and *true* coefficient values of pigment i for training sample n, respectively. N and M again are the total numbers of the training samples and number of pure pigments, respectively. We use the Adam [17] optimizer with default configuration parameter values to optimize the objective loss function.

For all our deep learning implementations, we use Python utilizing the Keras library [18] with Tensorflow [19] as the backend. The number of epochs and the batch size are set to 200 and 64, respectively. To avoid overfitting, an *Early Stopping callback* is used and the number of patience is set to 10.

4. DATA SETS

In this section, the training (simulated) and test (mock-up) sets used in the experiments are described.

Mock-up Dataset: We have prepared a set of mock-up paintings composed of 11 pure colors and 41 mixtures. The average spectra of 11 pure pigments in the reflectance space are shown in Fig. 1 (e) top and the visible figure of five exemplar mock-up mixtures is provided in Fig. 1 (e) bottom. There are 30 mixtures of two colors composed of white pigment and a non-white pure pigment where three different proportions of white pigment (20%, 50% and 70%) have been used. There are 4 mixtures that are composed of two non-white colors with same proportions ($\frac{1}{2}$ of each color). The remaining 7 mixtures are composed of three colors with the same proportions ($\frac{1}{3}$ of each color). However, since no sophisticated unmixing and layering equipment were used, it is impossible to guarantee that in every position in the mock-up square the same concentration of a specific color is used. Unfortunately, the assumption of specific proportions of the colors represents the ground truth information we will be using. We build our hyperspectral image, which will be used as the test set, by concatenating 11 pure pigments and 41 mixtures in a rectangle.

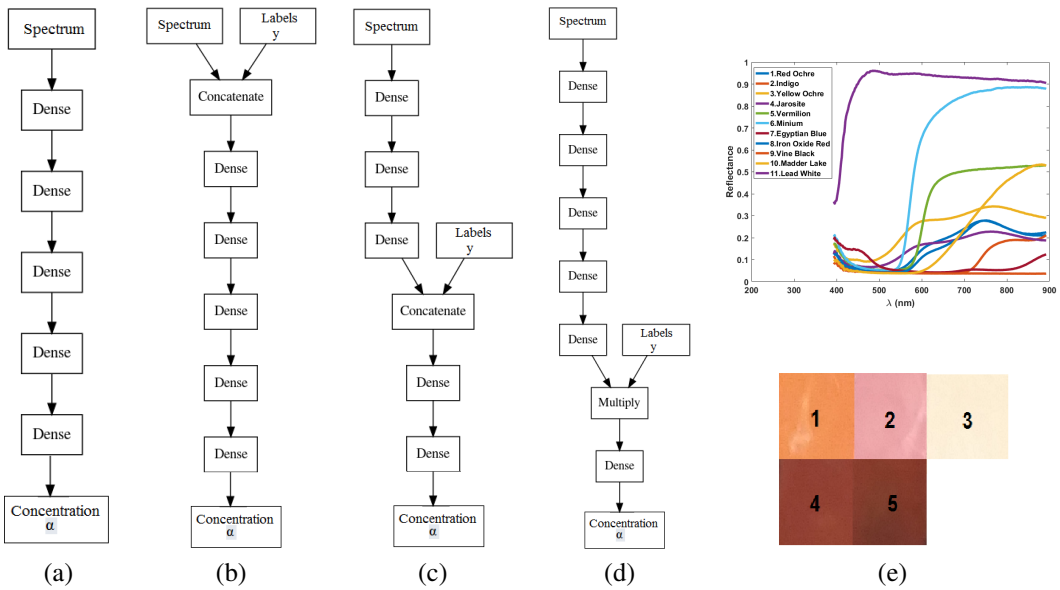

Fig. 1. Architectures of proposed networks (a) baseline network with the spectrum as the input, (b) a network with the spectrum and labels concatenated in the input layer (Early fusion), (c) a network with the spectrum as the input to the first layer and labels concatenated to the output of the third layer (Intermediate fusion), (d) a network with the spectrum as the input to the first layer and the output of the fifth layer multiplied by labels (Late fusion), (e) Top: Reflectance spectra of 11 colors used in dataset. Bottom: Visible picture of exemplar mock-up mixtures. The mixtures are : 1) Minium + Lead White 2) Madder Lake + Lead White 3) Jaoriste + Lead White 4) Red Ochre + Yellow Ochre and 5) Red Ochre + Yellow Ochre + Egyptian Blue

Simulated data: To train the network so that the mixing function is learned we need a dataset with known concentration values. Since for any experimental and real dataset, acquiring exact quantitative information is not possible as mentioned earlier, we resort to simulated data. A nonlinear KM mixing function [2, 12, 20] is used to generate simulated data. We use the 11 pure pigments in our set of mock-up paintings to build the mixtures. We consider mixtures of two or three colors and build all possible mixtures of two/three pure pigments. For each mixture, 500 random combinations of coefficients/concentration values summing up to one are selected and the KM function [12] is used to model random nonlinear mixtures (refer to Eq. 1 and replace mixing function with the KM nonlinear function, as shown in Eq. 2 in the supporting material of [11], section B). This dataset is used to train networks for the pigment identification and unmixing tasks.

Pre-processing: Our captured spectra consist of 240 bands with 2nm resolution from 383 to 893 nm. We remove the noisy channels and to reduce the noise in the signal further, a spectrally moving average filter is used and the number of channels is decreased by a factor of 4 leading to a smoother signal with 56 bands. Also, all pixel values are normalized by the reflectance scale factor of the hyperspectral camera (4095).

5. RESULTS AND DISCUSSION

As mentioned in section 4, we use the simulated data as our training dataset to find the optimal hyperparameters of the deep neural networks and mapping function between the spectrum and coefficient values. The set of mock-up paintings is used as the test sets here.

Pigment identification: As mentioned before to detect the constituent pigments in any given spectrum, a multi-label classifier [16] using a DNN is used where each unknown spectrum can have multiple labels. Here, we train 11 binary classifiers and each i-th classifier predicts if the i-th pure pigment is present in the given spectrum or not. The accuracy of the trained classifier on the mock-up test painting is $\mathbf{98.4}\%$.

Nonlinear unmixing: The network learns the mixing function of the pigments using the training set from the simulated data. Due to the space limitation, the average predicted coefficients by different network architectures (Fig. 1 (a)-(d)) for five exemplar mixtures from the mock-up (Fig. 1 (e) bottom) are given in Fig. 2. The studied mixtures are: Minium + Lead White $(50\%, 50\%)$, Madder Lake + Lead White $(80\%, 20\%)$, Jaoriste + Lead White $(30\%, 70\%)$, Red Ochre + Yellow Ochre $(50\%, 50\%)$ and Red Ochre + Yellow Ochre + Egyptian Blue $(\approx 33\%, 33\%, 33\%)$. In Fig. 2, each bar refers to a mixture, for which the dark blue, light blue and green show the average estimated coefficient values of the first, second, third (if present in the mixtures) pigments in the mixtures. In Fig. 2 yellow refers to the average estimated coefficient values of other pigments (not present in the mixture) that the model has predicted wrongly. In Fig. 2 (a), we observe that architecture (a) which does not use

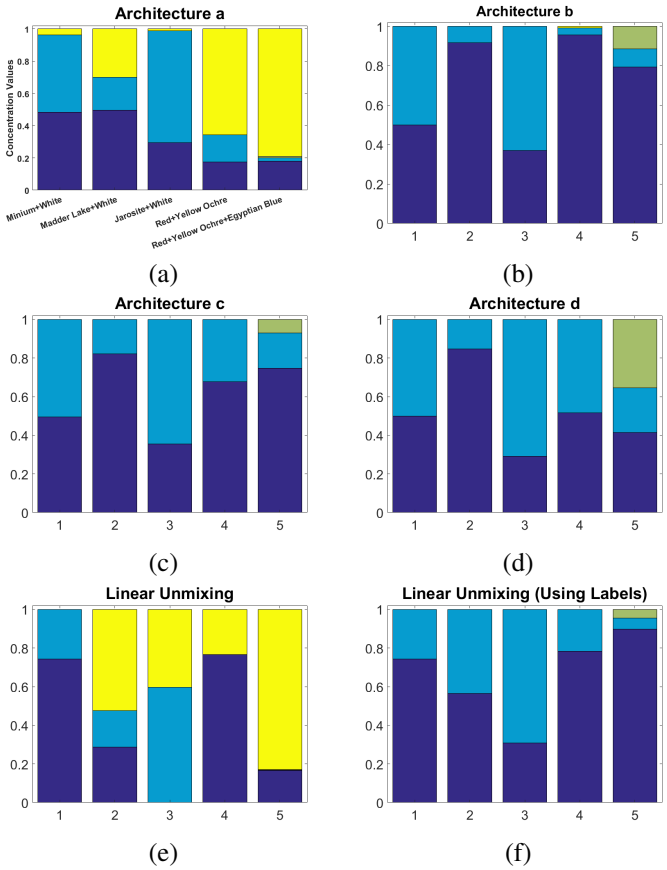

Fig. 2. Average estimated coefficient values by different networks in Fig. 1 (a) network architecture 1(a), (b) network architecture 1(b), (c) network architecture 1(c), (d) network architecture 1(d), (e) linear unmixing algorithm (Eq. 3), (f) linear unmixing algorithm using labels (applied to the subset of the spectra of the present pigments in any given spectrum).

the information of the labels has the worst performance and describes the mixtures by other pigments than the real constituent pigments (large yellow proportions in mixtures 2, 4 and 5). The predicted coefficients for mixtures 1 and 3 are fairly good. In Fig. 2 (b), we observe that the performance of the architecture (b) is better than architecture (a) and does not pick wrong pigments. However, the predicted coefficients for mixtures 2, 3 and 4 are still far from the expected values. The estimated coefficients of architecture (c) (Fig. 2 (c)) are closer to the expected values especially for mixtures 2, 4 and 5 compared to the architecture (b) and it seems that the network is learning the unmixing function better than the architectures (a) and (b). The best results is related to the last network (Fig. 2 (d)), where the labels are integrated into network in a higher layer. The estimated coefficients for two colors in mixtures 1 and 4 are close to 50 as expected. In mixture 2, the proportion of pigment 1 is close to 80 and in mixture 3 its proportion is close to 30 as expected. In mixture 5, the coefficients of three colors are similar. It seems that the mixtures with white pigment are easier for the network to decompose. On the other hand, the mixture with three non-white pigments seems to be the most challenging. This may be due to the fact that the nonlinear KM mixing function used in building the non-white mixtures may not be as accurate as for the mixtures with white pigments in our settings.

For the comparison with the algorithms used for pigment unmixing, we also studied the performance of linear unmixing algorithm using a fully constrained least square method according to

$$x = \alpha E, \quad \text{s.t.} \quad \sum_{i=1}^{M} \alpha_i = 1 \quad \text{and} \quad \alpha_i \geq 0, \qquad (3)$$

In Fig. 2 (e), we did not use the labels information and decomposed the given spectrum to a linear combination of spectra of all $M = 11$ pure pigments (Eq. 3). Similar to the architecture (a) (Fig. 1 (a)), the linear unmixing algorithm's performance in detecting the correct pigments is poor and we observe that there are large yellow portions in mixtures 2, 3, 4 and 5. In Fig. 2 (f), we used the labels information and decomposed the given spectrum to a linear combination of the spectra of a subset of E containing only the present pigments in a given spectrum. As can be seen from Fig. 2 (f), the linear unmixing performance is improved compared to Fig. 2 (e). However, it is much worse than architectures (c) and (d) and the estimated coefficients for all mixtures are still far from the expected values. As was expected, based on the results shown in Fig. 2, we observe that DNNs model the mixing function more accurately and yield better results than linear unmixing algorithm.

6. CONCLUSIONS

In this paper, we studied the automatic pigment identification/unmixing in hyperspectral paintings. Using deep neural networks and a supervised multi-label classification approach, the pigments present in any given spectrum are found. Using another deep neural network, the concentrations of the pigments are approximated. We studied a set of mock-up paintings and showed that the DNNs identify the pigments for any given spectrum with high accuracy. We integrated the label information to different layers of the network and showed that the pigment unmixing performance of the networks is better when late fusion is used and the estimated coefficient values are closer to what is expected. The next step of this research is to apply the improved version of the nonlinear unmixing algorithm to analyze the data acquired on historical objects. We will focus on fusion techniques to integrate the hyperspectral and XRF images to improve the techniques for pigment identification and unmixing problems.

7. ACKNOWLEDGEMENT

This work was supported by the National Science Foundation (NSF) Partnerships for International Research and Education (PIRE) program under grant number 1743748.

8. REFERENCES

[1] J. Delaney, P. Ricciardi, L. Glinsman, M. Facini, M. Thoury, M. Palmer, and E. R. de la Rie, "Use of imaging spectroscopy, fiber optic reflectance spectroscopy and x-ray fluorescence to map and identify pigments in illuminated manuscripts," vol. 59, 01 2013.

[2] R. S. Berns and M. Mohammadi, "Evaluating single- and two-constant kubelka-munk turbid media theory for instrumental-based inpainting," *Studies in Conservation*, vol. 52, no. 4, pp. 299–314, 2007.

[3] G. Dupuis and M. Menu, "Quantitative characterisation of pigment mixtures used in art by fibre-optics diffuse-reflectance spectroscopy," *Applied physics.*, vol. 83, no. 4, pp. 469–474, 2006.

[4] W. Zhao and S. Du, "Spectral-spatial feature extraction for hyperspectral image classification: A dimension reduction and deep learning approach," *IEEE Transactions on Geoscience and Remote Sensing*, vol. 54, no. 8, pp. 4544–4554, Aug 2016.

[5] Y. Chen, X. Zhao, and X. Jia, "Spectral-spatial classification of hyperspectral data based on deep belief network," *IEEE Journal of Selected Topics in Applied Earth Observations and Remote Sensing*, vol. 8, no. 6, pp. 2381–2392, June 2015.

[6] L. Wei F. Zhang W. Hu, Y. Huang and H. Li, "Deep convolutional neural networks for hyperspectral image classification," *Journal of Sensors*, vol. 2015, 2015.

[7] K. Makantasis, K. Karantzalos, A. Doulamis, and N. Doulamis, "Deep supervised learning for hyperspectral data classification through convolutional neural networks," in *2015 IEEE International Geoscience and Remote Sensing Symposium (IGARSS)*, July 2015, pp. 4959–4962.

[8] J. Yue, W. Zhao, S. Mao, and H. Liu, "Spectral-spatial classification of hyperspectral images using deep convolutional neural networks," *Remote Sensing Letters*, vol. 6, no. 6, pp. 468–477, 2015.

[9] R. Guo, W. Wang, and H. Qi, "Hyperspectral image unmixing using autoencoder cascade," in *2015 7th Workshop on Hyperspectral Image and Signal Processing: Evolution in Remote Sensing (WHISPERS)*, June 2015, pp. 1–4.

[10] B. Grabowski, W. Masarczyk, P. Glomb, and A. Mendys, "Automatic pigment identification from hyperspectral data," *Journal of Cultural Heritage*, 2018.

[11] Neda Rohani, Emeline Pouyet, Marc Walton, Oliver Cossairt, and Aggelos K. Katsaggelos, "Nonlinear unmixing of hyperspectral datasets for the study of painted works of art.," *Angewandte Chemie*, vol. 57 34, pp. 10910–10914, 2018.

[12] P. Kubelka, "New contributions to the optics of intensely light-scattering materials. part i," *J. Opt. Soc. Am.*, vol. 38, no. 5, pp. 448–457, May 1948.

[13] N. Rohani, J. Salvant, S. Bahaadini, O. Cossairt, M. Walton, and A. K. Katsaggelos, "Automatic pigment identification on roman egyptian paintings by using sparse modeling of hyperspectral images," in *2016 24th European Signal Processing Conference (EUSIPCO)*, Aug 2016, pp. 2111–2115.

[14] H. Abdi, "The method of least squares," *Encyclopedia of Measurement and Statistics*, 2007.

[15] J. Snyman, *Practical Mathematical Optimization: An Introduction to Basic Optimization Theory and Classical and New Gradient-Based Algorithms*, Applied Optimization. Springer, 2005.

[16] G. Tsoumakas and I. Katakis, "Multi-label classification: An overview," *International Journal of Data Warehousing and Mining*, vol. 3, pp. 1–13, 09 2009.

[17] D. P. Kingma and J. Ba, "Adam: A method for stochastic optimization," *CoRR*, vol. abs/1412.6980, 2014.

[18] F. Chollet, "keras," 2015.

[19] J. Bergstra, O. Breuleux, Fr. Bastien, P. Lamblin, . Pascanu, G. Desjardins, J. Turian, D. Warde-Farley, and Y. Bengio, "Theano: a CPU and GPU math expression compiler," in *Proceedings of the Python for Scientific Computing Conference (SciPy)*, June 2010, Oral Presentation.

[20] R. S. Berns and M. Mohammadi, "Single-constant simplification of kubelka-munk turbid-media theory for paint systems-a review," *Color Research and Application*, vol. 32, no. 3, pp. 201–207, 2007.