Transfer Learning for Segmenting Dimensionally Reduced Hyperspectral Images

Jakub Nalepa, *Member, IEEE*, Michal Myller, and Michal Kawulok, *Member, IEEE*

***Abstract*— Deep learning has established the state of the art in multiple fields, including hyperspectral image analysis. However, training large-capacity learners to segment such imagery requires representative training sets. Acquiring such data is human-dependent and time-consuming, especially in earth observation scenarios, where the hyperspectral data transfer is very costly and time-constrained. In this letter, we show how to effectively deal with a limited number and size of available hyperspectral ground-truth sets and apply transfer learning for building deep feature extractors. Also, we exploit spectral dimensionality reduction to make our technique applicable over hyperspectral data acquired using different sensors, which may capture different numbers of hyperspectral bands. The experiments, performed over several benchmarks and backed up with statistical tests, indicated that our approach allows us to effectively train well-generalizing deep convolutional neural nets even using significantly reduced data.**

***Index Terms*— Classification, deep learning, hyperspectral imaging, segmentation, transfer learning.**

I. Introduction

THE analysis of hyperspectral images (HSIs) has been gaining research attention due to the amount of useful information it can reveal. HSI is captured by an imaging spectrometer, which collects the reflectance values, being the portion of light reflected by the object at a certain wavelength, in hundreds of narrow spectral bands over a wide wavelength range, typically at least from visible through middle infrared. Such reflectance values, acquired within the adjacent hyperspectral bands for each pixel in the HSI, form three-dimensional hyperspectral cubes generated for a scanned spatial area (each band represents a single wavelength range of the electromagnetic spectrum; hence, a stack of such bands is created to form an HSI for this area). Importantly, remote sensors are being developed at an enormous speed, and acquisition of HSI with up to hundreds of spectral bands is more affordable nowadays. *Classification* and *segmentation*[1] of HSI help understand the underlying materials in the scene and can be exploited in multiple fields including chemistry, biology, medicine, food quality control, and more [1]. In earth observation applications, HSI can provide extremely detailed information on the earth peculiarities and may be utilized in an array of use cases, encompassing, among others, precision agriculture, managing environmental disasters, and military applications.

A. Related Work

HSI classification and segmentation algorithms can be divided into conventional machine learning [2], [3], and deep-learning-powered techniques [4]–[9]. Deep learning has established the state of the art in a variety of fields, consistently outperforming techniques which use hand-crafted features. To deploy such deep models in practice, we need representative ground-truth training sets. It is a significant obstacle in earth observation analysis, where transferring data back to earth is extremely costly. A problem of efficient hyperspectral data volume reduction (to enable its feasible storage and transmission) can be tackled, e.g., by reduction of digital precision from native (14/12-bit) down to 8/1-bit, elimination of low-variance components in principal component analysis, or reduction of spectral resolution by band/feature selection [10], feature extraction [11], [12], and fusion [13]. Annotating HSI by humans is error-prone and requires building a full understanding of the materials presented in a scanned region, therefore, involves acquiring observational ground-sensor data. These difficulties are reflected in a limited number of ground-truth sets [14].

B. Contribution

In this letter, we tackle both problems of 1) limited number of ground-truth hyperspectral sets and 2) large volumes of such data. We employ *transfer learning* to make convolutional neural networks (CNNs) easily applicable in supervised HSI segmentation with limited ground truth (Section II). First, we train the feature extraction part of a CNN over a source (larger) set, and then we fine-tune its classification part over the target (much smaller) set. Since different sensors acquire HSI with different spectral characteristics, we exploit our recent algorithm for simulating multispectral image (MSI) from its hyperspectral counterpart [15] and reduce the dimensionality of both source and target sets to the same number of bands, to make any spectral feature extractor trained over the source set straightforwardly applicable to the target set. This operation allows us to build extractors that are applicable to *any* HSI, once this HSI is reduced to the assumed number of bands. Also, it brings the possibilities of on-board data

Manuscript received June 23, 2019; revised August 14, 2019; accepted September 18, 2019. Date of publication October 10, 2019; date of current version June 24, 2020. This work was supported in part by the European Space Agency and in part by the Polish National Centre for Research and Development under Grant POIR.01.01.01-00-0356/17. The work of J. Nalepa and M. Kawulok was supported in part by the Silesian University of Technology, Poland, under Grant BKM-556/RAU2/2018 and Grant 02/020/BK_18/0128. *(Corresponding author: Jakub Nalepa.)*

The authors are with the Silesian University of Technology, 44-100 Gliwice, Poland and also with KP Labs, 44-100 Gliwice, Poland (e-mail: jnalepa@ieee.org).

Color versions of one or more of the figures in this letter are available online at http://ieeexplore.ieee.org.

This article has supplementary downloadable material available at http://ieeexplore.ieee.org, provided by the authors.

Digital Object Identifier 10.1109/LGRS.2019.2942832

[1] By *classification* we mean assigning a label to a pixel, and by *segmentation*—finding the boundaries of objects belonging to different classes in HSI.

TABLE I
RESULTS OBTAINED FOR ALL CONFIGURATIONS OF OUR 1-D-CNN ARCHITECTURE

CNN→	1 block									2 blocks									3 blocks								
Set→	Sa			PU			IP			Sa			PU			IP			Sa			PU			IP		
	OA	AA	κ	OA	AA	κ	OA	AA	κ	OA	AA	κ	OA	AA	κ	OA	AA	κ	OA	AA	κ	OA	AA	κ	OA	AA	κ
Var.↓	Full spectrum																										
B(E)	95.47	97.98	0.94	96.72	95.79	0.95	88.50	87.34	0.87	95.55	97.96	0.95	96.24	94.92	0.94	89.13	87.35	0.88	95.49	97.91	0.94	96.37	95.17	0.94	89.56	88.68	0.88
B	89.01	94.84	0.88	89.32	92.19	0.87	76.47	81.38	0.73	90.24	95.53	0.90	90.56	92.27	**0.88**	78.45	82.03	0.75	89.64	95.51	0.89	89.66	92.09	0.87	79.33	83.71	0.76
	100 bands																										
B(E)	95.55	97.97	0.95	95.32	93.89	0.86	90.42	89.74	0.92	94.37	97.34	0.94	95.29	93.24	0.93	84.61	81.79	0.85	93.63	96.75	0.93	95.28	93.37	0.94	81.17	79.44	0.81
B	90.81	95.80	**0.91**	91.02	93.06	0.85	81.06	85.42	0.77	91.11	96.15	0.90	91.03	92.73	0.87	81.68	85.97	0.80	90.04	95.76	0.89	91.38	92.42	0.87	82.57	86.17	0.82
Ex(Sa)	—	—	—	90.75	92.35	0.86	81.98	86.82	**0.79**	—	—	—	89.43	91.37	0.85	83.29	87.22	0.81	—	—	—	88.29	90.70	0.84	82.86	86.61	0.80
Ex(PU)	91.26	96.40	0.90	—	—	—	82.29	87.25	**0.79**	91.08	96.32	0.90	—	—	—	82.71	87.76	0.80	91.36	96.25	0.90	—	—	—	82.79	87.36	0.80
Ex(IP)	**91.81**	**96.63**	**0.91**	90.92	92.33	0.85	—	—	—	91.36	96.39	**0.91**	90.90	92.39	0.84	—	—	—	91.56	96.35	0.90	90.47	92.05	0.82	—	—	—
	75 bands																										
B(E)	96.21	98.29	0.96	96.95	95.95	0.94	90.93	90.84	0.92	96.53	98.42	0.96	96.69	95.49	0.95	91.84	91.39	0.92	95.15	97.71	0.94	96.40	95.16	0.94	89.51	87.13	0.88
B	90.34	95.70	0.90	91.38	93.21	**0.89**	80.21	85.11	0.75	90.87	96.18	0.90	92.11	**93.43**	0.86	82.70	87.12	0.80	89.88	95.67	0.89	90.81	92.82	**0.88**	82.26	86.28	0.80
Ex(Sa)	—	—	—	91.98	93.33	0.88	**83.05**	**87.49**	**0.79**	—	—	—	91.38	92.98	0.83	**84.74**	88.53	**0.83**	—	—	—	91.10	92.84	**0.88**	**85.07**	**89.27**	**0.83**
Ex(PU)	91.32	96.39	0.90	—	—	—	81.41	86.93	0.78	91.36	96.32	**0.91**	—	—	—	81.51	87.85	0.80	**91.83**	96.47	**0.91**	—	—	—	82.86	87.58	0.81
Ex(IP)	91.53	96.44	0.90	92.34	**93.49**	**0.89**	—	—	—	92.08	96.66	**0.91**	**92.16**	93.32	0.87	—	—	—	91.79	**96.57**	**0.91**	**91.47**	**92.99**	0.84	—	—	—
	50 bands																										
B(E)	95.87	98.16	0.95	96.78	95.60	0.94	90.31	89.45	0.90	95.82	98.06	0.95	96.57	95.42	0.94	91.88	91.04	0.93	94.56	97.08	0.94	95.84	94.43	0.94	85.93	81.59	0.85
B	89.96	95.48	0.90	90.91	92.92	0.87	78.66	84.87	0.74	90.51	95.87	0.90	91.77	93.10	0.87	82.67	86.68	0.80	89.21	94.80	0.89	90.37	92.02	0.84	78.81	83.31	0.75
Ex(Sa)	—	—	—	91.55	93.14	0.87	81.87	87.20	**0.79**	—	—	—	91.72	92.96	0.85	84.54	**88.75**	0.82	—	—	—	88.26	90.84	0.84	81.16	86.34	0.80
Ex(PU)	91.28	96.26	0.90	—	—	—	81.24	86.09	0.76	91.55	96.35	**0.91**	—	—	—	83.03	86.93	0.79	91.71	96.46	**0.91**	—	—	—	82.13	87.14	0.80
Ex(IP)	91.14	96.18	0.90	**92.54**	93.34	0.87	—	—	—	**92.25**	**96.72**	**0.91**	92.01	93.14	**0.88**	—	—	—	91.29	96.27	0.90	88.18	90.95	0.83	—	—	—
	25 bands																										
B(E)	94.91	97.68	0.94	96.44	95.21	0.94	88.56	89.67	0.87	94.93	97.61	0.94	96.26	94.89	0.95	89.30	88.35	0.88	—	—	—	—	—	—	—	—	—
B	90.05	95.24	0.90	89.69	91.99	0.85	76.42	82.54	0.71	90.35	95.60	0.90	91.07	92.63	0.87	82.94	87.02	0.80	—	—	—	—	—	—	—	—	—
Ex(Sa)	—	—	—	91.39	92.92	0.86	80.28	85.46	0.77	—	—	—	89.56	91.84	0.85	83.85	88.22	0.81	—	—	—	—	—	—	—	—	—
Ex(PU)	90.62	95.85	0.90	—	—	—	79.74	84.29	0.77	91.50	96.20	0.90	—	—	—	82.68	87.78	0.77	—	—	—	—	—	—	—	—	—
Ex(IP)	90.74	95.79	0.90	91.76	93.00	0.86	—	—	—	91.90	96.58	**0.91**	88.75	91.64	0.83	—	—	—	—	—	—	—	—	—	—	—	—

How to read this table: The *globally* best result (across HSI and simulated MSI), excluding B(E), in each column is boldfaced, and the background of the worst cell is grayed. For *each number* (100, 75, 50, and 25) of simulated multispectral bands, the background of the cell with the best result is colored—if the best result is obtained using transfer learning, the background is green. If the best result is obtained using a model trained over the B division of the target set (i.e., *without* transfer learning), the background is red.

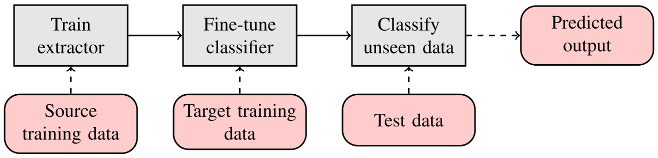

Fig. 1. We train the feature extractor over a source data and fine-tune the classifier over the target data. Note that a different number of classes in the source and target sets is not an issue, since we tune the classification part.

reduction executed before transferring the acquired data from an imaging satellite. Although there exist works which show the usefulness of transfer learning in HSI segmentation in various fields, they are focused on applying this technique to different deep architectures [16]–[19]. To the best of authors' knowledge, our approach is the first which comprehensively combines effective HSI data reduction and transfer learning. The experiments showed that the proposed algorithm leads to well-generalizing convolutional models, and the HSI reduction does not adversely affect their performance (Section III).

II. Transfer Learning for HSI Classification

Transfer learning helps us build large-capacity learners, e.g., deep neural networks, over *small* training data. In our approach (Fig. 1), we train the deep feature extractor over a source hyperspectral training data (containing $t_{\mathcal{S}}$ training examples) and fine-tune the classification part of a CNN over the target training data ($t_{\mathcal{T}}$ examples, where $t_{\mathcal{S}} \gg t_{\mathcal{T}}$). The fine-tuned CNN is used to classify the incoming test examples.

Since different sensors acquire HSI with different spectral characteristics, the number of bands in the source training data ($b_{\mathcal{S}}$) is likely to be different from the number of bands in the target set ($b_{\mathcal{T}}$). To make our method applicable to *any* HSI, we generate the simulated bands based on the original hyperspectral imagery by using nonoverlapping sliding windows of a size ℓ [15] and reduce the number of bands in both source and target sets to b_M (hence, ℓ depends on b_M). Therefore, the dimensionality of the source and target sets becomes the same which makes applying feature extractors trained over the source set to the target set straightforward. Note that if the source and target sets are of the same band characteristics (i.e., the number of bands is the same in both sets), the process of generating simulated bands may be omitted. However, in the case of high-dimensional and limited (in terms of the labeled examples) training sets, we may face the *curse of dimensionality* problem which can make obtaining well-generalizing deep models challenging [14].

Let us consider a source HSI with $b_{\mathcal{S}}$ bands b_i, $i = 0, 1, \ldots, b_{\mathcal{S}}-1$. Its simulated reduced counterpart will contain b_M bands, where $b_M = \lceil b_{\mathcal{S}}/\ell \rceil$. The corresponding simulated bands become $b'_i = f(b_{i\cdot\ell}, b_{i\cdot\ell+1}, \ldots, b_{i\cdot\ell+\ell-1})$, where f is a function which transforms ℓ consecutive HSI bands into simulated ones. Although f may be updated to any function that maps the neighboring signals into an aggregated signal [15], we perform the averaging across ℓ bands in a window. It can be seen as having wide bands covering the spectrum instead of more narrower bands. Sensors, which are sensitive at broader range of wavelengths gather more light, can increase the signal-to-noise ratio. Usually, the sensor sensitivity is wavelength-dependent—averaging the neighboring bands can be interpreted as an approximation of wider bands.

In Fig. 2, we present an example process of simulating bands from HSI ($\ell = 4$). The reduction ratio is dependent on the window size ℓ, and increasing ℓ will lead to a lower number of simulated bands. This reduction is crucial in

TABLE II

RESULTS OBTAINED FOR ALL CONFIGURATIONS OF THE STATE-OF-THE-ART PT-CNN. FOR HINTS ON HOW TO READ THIS TABLE, SEE TABLE I

CNN→	1 block									2 blocks									3 blocks								
Set→	Sa			PU			IP			Sa			PU			IP			Sa			PU			IP		
	OA	AA	κ	OA	AA	κ	OA	AA	κ	OA	AA	κ	OA	AA	κ	OA	AA	κ	OA	AA	κ	OA	AA	κ	OA	AA	κ
Var.↓	Full spectrum																										
B(E)	93.57	96.68	0.93	96.20	94.97	0.95	77.21	67.93	0.74	94.67	97.27	0.94	96.20	94.96	0.95	86.96	82.84	0.85	94.89	97.50	0.94	96.35	94.99	0.95	86.81	81.48	0.85
B	87.07	93.63	0.86	83.30	89.30	0.78	62.33	64.46	0.57	89.62	95.07	0.88	89.29	91.83	0.86	74.40	77.50	0.70	90.34	95.45	0.89	88.46	91.50	0.85	76.26	79.18	0.73
	100 bands																										
B(E)	94.51	97.39	0.94	96.38	95.11	0.93	81.88	75.61	0.81	95.45	97.83	0.95	96.56	95.41	0.95	89.11	88.15	0.90	95.32	97.76	0.95	96.43	95.22	0.96	88.67	86.05	0.90
B	88.85	94.46	0.87	88.29	91.40	0.82	65.21	68.16	0.59	90.93	95.97	**0.90**	90.40	92.21	0.83	79.45	84.37	0.77	90.46	95.77	0.89	89.32	91.67	0.84	79.08	82.89	0.76
Ex(Sa)	—	—	—	90.10	92.20	**0.85**	68.68	72.33	**0.64**	—	—	—	89.79	92.11	0.85	79.91	**84.56**	0.77	—	—	—	89.18	91.56	0.84	78.52	**83.94**	0.76
Ex(PU)	**89.24**	**94.91**	**0.88**	—	—	—	69.48	**73.11**	**0.64**	89.29	95.15	0.89	—	—	—	72.69	77.46	0.68	90.18	95.46	0.89	—	—	—	73.23	73.23	0.68
Ex(IP)	88.68	94.85	0.87	89.99	92.14	0.83	—	—	—	**91.44**	**96.32**	**0.90**	89.59	92.09	0.82	—	—	—	**91.70**	**96.51**	**0.91**	88.97	91.54	0.85	—	—	—
	75 bands																										
B(E)	94.23	97.15	0.94	96.47	95.33	0.93	82.07	74.59	0.78	95.48	97.89	0.95	96.64	95.54	0.95	89.66	86.37	0.89	95.15	97.72	0.95	96.61	95.44	0.94	89.43	86.95	0.89
B	87.97	94.03	0.86	88.61	91.70	0.84	76.35	66.63	0.59	90.55	95.86	**0.90**	**90.42**	**92.63**	0.84	**88.89**	84.42	**0.79**	90.67	95.73	0.89	**90.33**	**92.35**	0.85	**87.55**	83.00	**0.77**
Ex(Sa)	—	—	—	**90.64**	**92.53**	**0.85**	78.65	70.03	0.59	—	—	—	90.01	92.31	0.85	87.62	84.14	0.76	—	—	—	89.61	91.93	0.84	87.12	81.77	0.74
Ex(PU)	88.14	94.40	0.86	—	—	—	**79.65**	71.29	0.62	89.62	95.03	0.88	—	—	—	80.90	73.26	0.66	90.00	95.25	0.89	—	—	—	82.64	76.24	0.67
Ex(IP)	88.70	94.70	0.87	90.40	92.44	0.82	—	—	—	91.31	96.24	**0.90**	89.86	92.23	**0.87**	—	—	—	91.33	96.22	0.90	89.32	91.71	**0.86**	—	—	—
	50 bands																										
B(E)	94.10	97.09	0.93	96.18	95.01	0.95	80.28	71.55	0.77	95.24	97.73	0.95	96.49	95.28	0.95	87.99	85.10	0.87	94.32	97.13	0.94	96.38	95.18	0.96	87.23	81.81	0.86
B	87.49	93.87	0.86	86.60	90.82	0.80	63.16	63.51	0.57	90.42	95.64	0.89	89.47	92.06	0.82	76.97	81.96	0.73	90.49	95.79	0.90	88.87	91.53	0.85	75.45	79.64	0.72
Ex(Sa)	—	—	—	89.24	91.81	**0.85**	67.41	70.34	0.62	—	—	—	88.82	91.44	0.83	74.36	79.91	0.71	—	—	—	83.10	88.33	0.74	69.40	75.11	0.64
Ex(PU)	88.20	94.29	0.87	—	—	—	66.53	68.41	0.61	88.62	94.52	0.88	—	—	—	66.58	68.68	0.61	88.95	94.77	0.88	—	—	—	65.13	67.80	0.58
Ex(IP)	88.25	94.38	0.87	90.13	92.12	0.81	—	—	—	90.80	95.91	0.89	85.52	90.39	0.81	—	—	—	90.12	95.47	0.89	82.81	88.39	0.77	—	—	—
	25 bands																										
B(E)	93.09	96.50	0.92	95.90	94.55	0.94	78.08	67.60	0.75	—	—	—	—	—	—	—	—	—	—	—	—	—	—	—	—	—	—
B	87.18	93.42	0.86	84.84	89.70	0.76	61.58	62.17	0.55	—	—	—	—	—	—	—	—	—	—	—	—	—	—	—	—	—	—
Ex(Sa)	—	—	—	87.81	91.07	0.81	63.52	64.70	0.54	—	—	—	—	—	—	—	—	—	—	—	—	—	—	—	—	—	—
Ex(PU)	87.49	93.77	0.86	—	—	—	63.14	63.86	0.56	—	—	—	—	—	—	—	—	—	—	—	—	—	—	—	—	—	—
Ex(IP)	87.95	93.96	0.86	87.54	91.02	0.77	—	—	—	—	—	—	—	—	—	—	—	—	—	—	—	—	—	—	—	—	—

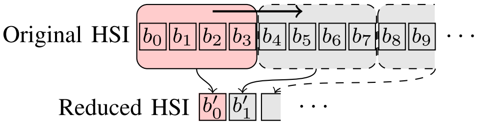
Original HSI
b_0 b_1 b_2 b_3 b_4 b_5 b_6 b_7 b_8 b_9 · · ·
Reduced HSI
b'_0 b'_1 · · ·

Fig. 2. Reducing the dimensionality of an input hyperspectral data by simulating bands using our sliding-window approach ($\ell = 4$).

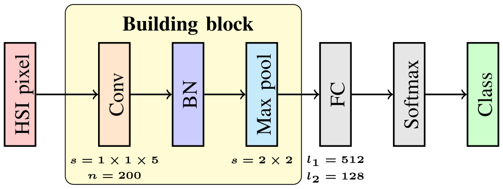

Fig. 3. 1-D-CNN with n kernels in the convolutional layer (s stride) and l_1 and l_2 neurons in the FC layers. BN is batch normalization.

earth observation scenarios to effectively transfer the acquired HSI from a satellite. Let us assume we want to capture a 2048 × 2048 12-bit HSI with 200 bands, and send it back to earth. This would give $2048 \cdot 2048 \cdot 200 \cdot 12 \approx 10$ gigabits for transmission. If we could use an X-band link with 3 Mbps nominal downlink speed, it would require 3355 s (56 min) for a single scene. Simulating MSI with 20 bands (the volume is reduced 10×), ideally without affecting the performance of a segmentation algorithm applied over this data, would greatly decrease this time and make it much more affordable.

III. EXPERIMENTS

We verify if transfer learning applied over reduced HSI can be used to get well-generalizing CNNs. We investigated two CNNs: our 1-D-CNN (Fig. 3) with different numbers of building blocks (one, two, and three) constituting the feature extractor and two fully connected (FC) layers followed by softmax in the classification part, alongside a state-of-the-art CNN [we call it pre-trained CNN (PT-CNN)] with one, two, and three convolutional layers acting as a feature extractor. Therefore, the main difference between 1-D-CNN and PT-CNN is the lack of pooling layers in the latter network [16]. For simplicity, we refer to the PT-CNN convolutional layers as building blocks too—they are followed by three FC layers and softmax. The deep nets were implemented[2] in `Python 3.6`, and the training (ADAM, learning rate of 10^{-4}, $\beta_1 = 0.9$, $\beta_2 = 0.999$) stops, if after 25 epochs the overall accuracy (OA) over the validation set (random subset of the training set) does not change. The experiments were run on NVIDIA GeForce GTX 1060.

TABLE III

AVERAGE RANKING (ACCORDING TO κ) OF ALL MODELS ACROSS ALL DATA SETS (HSI AND SIMULATED MSI). THE BEST RANKING IS IN BOLD

CNN↓	B	Ex(Sa)	Ex(PU)	Ex(IP)
1D-CNN	1.76	**1.41**	1.59	1.45
PT-CNN	1.80	1.65	1.95	**1.55**

In this letter, we exploited three most popular HSI benchmarks from the literature (see Section I): 1) Salinas Valley (Sa), USA (217 × 512 pixels, AVIRIS sensor; $\|\boldsymbol{T}\| = 4320$, $\|\boldsymbol{V}\| = 480$, $\|\Psi\| = 49329$, where $\|\cdot\|$ denotes the cardinality of the corresponding set) presenting different sorts of vegetation (16 classes, 224 bands, 3.7 m spatial resolution); 2) Indian Pines (IP), USA (145 × 145, AVIRIS; $\|\boldsymbol{T}\| = 2444$, $\|\boldsymbol{V}\| = 271$, $\|\Psi\| = 7534$) covering agriculture and forest (16 classes, 200 channels, 20 m); 3) Pavia University (PU),

[2] The full implementation (deep networks, simulating bands, and transfer learning) and the supplementary material (including various visualizations) are available at: https://gitlab.com/jnalepa/hsi_transfer_learning

TABLE IV

AVERAGE TIME OF FEATURE-EXTRACTOR TRAINING (IN s), FINE-TUNING THE CLASSIFIERS (s), AND THE INFERENCE OF THE FINAL MODELS (ms)

CNN→	1 block			2 blocks			3 blocks		
Set →	Sa	PU	IP	Sa	PU	IP	Sa	PU	IP
Scenario ↓	Full spectrum								
1D-CNN, classifier (training)	1562.42	404.74	365.41	1133.83	229.12	221.90	1069.54	268.64	213.91
PT-CNN, classifier (training)	651.59	390.51	139.79	483.84	242.10	152.51	536.22	294.72	129.24
1D-CNN, classifier (inference)	0.075	0.058	0.086	0.102	0.073	0.112	0.120	0.082	0.133
PT-CNN, classifier (inference)	0.032	0.031	0.040	0.044	0.039	0.055	0.046	0.041	0.059
	100 bands								
1D-CNN, extractor	1034.75	274.69	169.55	573.57	180.49	95.77	553.45	221.30	93.56
PT-CNN, extractor	1096.81	608.20	223.36	718.79	394.62	196.88	753.72	380.02	190.53
1D-CNN, classifier (fine tuning)	169.39	62.59	169.55	134.80	34.15	77.83	118.08	41.27	80.38
PT-CNN, classifier (fine tuning)	128.28	66.52	75.68	121.96	45.61	92.91	105.35	48.94	86.01
1D-CNN, classifier (inference)	0.057	0.057	0.066	0.072	0.072	0.085	0.082	0.082	0.096
PT-CNN, classifier (inference)	0.029	0.030	0.038	0.037	0.038	0.049	0.041	0.041	0.053
	75 bands								
1D-CNN, extractor	1347.01	420.51	274.61	998.79	293.05	211.10	638.42	312.06	179.85
PT-CNN, extractor	1089.25	575.43	222.08	780.75	394.83	195.71	691.52	448.30	212.69
1D-CNN, classifier (fine tuning)	143.41	50.26	94.15	110.17	32.22	68.59	109.64	34.12	64.56
PT-CNN, classifier (fine tuning)	124.44	70.17	90.18	127.59	52.28	195.71	100.21	55.21	88.21
1D-CNN, classifier (inference)	0.050	0.050	0.061	0.061	0.063	0.073	0.071	0.072	0.084
PT-CNN, classifier (inference)	0.030	0.030	0.039	0.038	0.039	0.051	0.041	0.040	0.056
	50 bands								
1D-CNN, extractor	1071.76	366.89	217.80	686.72	250.31	180.00	564.22	274.33	144.77
PT-CNN, extractor	1072.14	636.39	228.29	756.48	443.57	207.35	589.82	538.99	196.65
1D-CNN, classifier (fine tuning)	116.23	49.21	77.35	91.71	32.78	60.51	80.17	37.34	59.89
PT-CNN, classifier (fine tuning)	125.65	72.40	88.44	119.24	51.11	92.58	117.40	55.90	91.65
1D-CNN, classifier (inference)	0.047	0.046	0.054	0.051	0.059	0.069	0.060	0.066	0.081
PT-CNN, classifier (inference)	0.029	0.030	0.037	0.038	0.039	0.048	0.041	0.040	0.051
	25 bands								
1D-CNN, extractor	749.85	397.06	166.28	470.53	250.17	117.43	—	—	—
PT-CNN, extractor	1083.01	756.97	222.39	—	—	—	—	—	—
1D-CNN, classifier (fine tuning)	88.59	41.18	60.51	66.96	31.97	48.27	—	—	—
PT-CNN, classifier (fine tuning)	127.05	74.16	103.64	—	—	—	—	—	—
1D-CNN, classifier (inference)	0.037	0.039	0.046	0.051	0.052	0.060	—	—	—
PT-CNN, classifier (inference)	0.029	0.029	0.038	—	—	—	—	—	—

Italy (340 × 610, ROSIS; $\|\boldsymbol{T}\| = 2025$, $\|\boldsymbol{V}\| = 225$, $\|\Psi\| = 40526$) presenting urban scenery (9 classes, 103 channels, 1.3 m). For training extractors, we randomly split the source set into nonoverlapping training ($\boldsymbol{T}$; balanced), validation ($\boldsymbol{V}$), and test (Ψ) sets containing 80%, 10%, and 10% of all pixels, respectively, and refer to this division as B(E). For fine-tuning the classification part of CNNs over the target sets, we exploited much smaller balanced $\boldsymbol{T}$ and $\boldsymbol{V}$ sets with the number of pixels as reported in [9] (the B division). Finally, we simulated 100, 75, 50, and 25 bands. We report the average accuracy (AA) and OA, and the kappa scores (κ) [14], elaborated over the test sets, and averaged across 25 runs.

The results for all configurations of 1-D-CNN and PT-CNN are gathered in Tables I and II. For the simulated MSI with 25 bands, some of the models could not be trained due to significant dimensionality reduction performed by the network itself (see the corresponding kernel sizes and strides shown in Fig. 3 for 1-D-CNN, and reported in [16] for PT-CNN). The CNNs which were pretrained using different source data sets were consistently outperforming those learned over smaller target sets *without* any transfer learning. Therefore, the feature extractors trained over Salinas Valley, Ex(Sa), and IP, Ex(IP), for 1-D-CNN and PT-CNN, respectively, allowed for obtaining the best generalization over the target sets. Increasing the number of CNN building blocks does not bring significant improvement in the classification performance of the underlying models. It shows that even shallower CNNs with notably smaller capacity are able to build appropriate representations of the investigated HSI. Therefore, the most discriminant class features are likely manifested in specific parts of the spectrum, and these features can be automatically elaborated with shallow feature extractors. Finally, we can observe the impact of the data set split[3] on the obtained classification performance of our deep networks—for both 1-D-CNN and PT-CNN, the results for B(E) in the full-spectrum scenario were significantly better compared to the B split, where the training sets are much smaller. We report the measures over the unseen Ψ sets; thus, we can conclude that the models trained in the B(E) scheme did not overfit the training data and generalize well. However, this estimated performance was quantified over very limited (and likely not representative) Ψ's.

In Table III, we present the average ranking (according to κ) of all models trained with and without transfer learning. The dimensionality reduction by using our sliding-window approach allowed us to obtain statistically better performance of both 1-D-CNN and PT-CNN when compared with the full HSI practically in all architectural configurations for the simulated MSI with 100, 75, and 50 bands (two-tailed Wilcoxon test at $p < 0.005$). The best results were obtained for 100 simulated bands for both 1-D-CNN and PT-CNN with one building block, and for 75 simulated bands for 1-D-CNN and PT-CNN with two and three building blocks. On the other hand, the results obtained for the full HSI and the simulated MSI with 25 bands (for 1-D-CNN and PT-CNN with one building block) and 50 bands (for 1D-CNN and PT-CNN with three building blocks) are statistically the same. It shows that the HSI reduction not only does not deteriorate the performance of the models but can also improve their capabilities for the investigated data sets. Since the entire

[3] We exploit only *spectral* CNNs which operate exclusively on the spectral pixel information during its classification—for such networks, random training–validation–test division *does not* lead to the training–test information leak that makes the classification results over-optimistic and not reliable [14].

spectrum was downsampled, we intrinsically tackled the curse of hyperspectral dimensionality problem. Although simulating MSI from HSI was very beneficial for benchmark scenes, it must be carefully performed for real-life data, as too aggressive HSI reduction can lead to removing parts of the spectrum which convey discriminative information about very *specific* classes and to making them indistinguishable from other classes with similar spectral profiles.

To verify the significance of the obtained results, we executed two-tailed Wilcoxon tests for both CNNs with one, two, and three building blocks, and for all sets (for the detailed results, see the supplementary material). In the majority of cases, the differences are statistically important ($p < 0.005$); thus the models in which transfer learning has been applied significantly outperformed those trained over the B target data splits (Table III). We can observe the differences between the extractors trained over different source HSI sets—for PU (being the target set), the extractors trained over Sa, Ex(Sa), and IP, Ex(IP)—are statistically the same for both CNNs. Finally, our 1-D-CNN outperformed PT-CNN in all scenarios ($p < 0.005$). Ensuring the representation invariance with respect to small translation of the input feature maps by the pooling layers is pivotal to get well-generalizing models.

In Table IV, we collect the average training time of all deep feature extractors, the average time of fine-tuning the classifiers over the target data, and the average inference time of the trained models for a single example from the unseen test sets Ψ. Although all of the investigated models offered instant inference over all benchmarks, decreasing the spectral dimensionality led to accelerating the inference process. For both CNNs, adding more building blocks, hence increasing the number of trainable CNN parameters, allowed for obtaining faster training convergence, as the capacity of the models is enlarged. Interestingly, training feature extractors in 1-D-CNN over the simulated MSI with 75 bands were notably slower than over 100 bands. Since the networks were characterized by the same generalization abilities ($p > 0.2$), it indicates that higher-dimensional MSI appeared more challenging to learn from a fairly limited number of training samples. It could be mitigated by either introducing more training examples (i.e., generating more ground-truth data points) or—as presented in this letter—by reducing the dimensionality of the training data.

IV. Conclusion and Critical Discussion

In this letter, we tackled the problem of limited ground-truth hyperspectral data in the context of supervised HIS segmentation. We utilize transfer learning, train the deep models over a source set, and apply the learned feature extractors to the target data after fine-tuning the classification part of a CNN. We made our method applicable to *any* input HSI by incorporating effective dimensionality reduction, and simulated a constant number of bands for source and target sets. Our multifaceted experimental study showed that the models trained with transfer learning significantly (in the statistical sense) outperformed the other CNNs, and that our dimensionality reduction not only does not adversely affect the performance of the models but also improves their generalization. It brings new possibilities for onboard deep-learning-powered earth observation use cases, where transferring full HSI data is extremely costly, and the lack of ground-truth is an important real-life obstacle in deploying such learners in the wild.

Although we showed that our approach can be effectively applied to spectral deep models, it would be useful to verify its abilities over spectral–spatial networks which utilize both spectral and spatial pixel information for better classification. It still remains unclear if the developed techniques are robust against various types of noise that are present on orbit. Therefore, we work on the noise simulators to inject such noise into both trained models and HSI data, and to verify its impact on the model's performance. Also, our current HSI reduction scheme involves determining the desired number of simulated bands in a manual process, which may be difficult for challenging real-life sets—we work on its adaptive variant coupled with additional band selection. Finally, we work on the quantized versions of our CNNs to make them applicable in hardware-constrained satellite execution environments.

References

[1] M. J. Khan, H. S. Khan, A. Yousaf, K. Khurshid, and A. Abbas, "Modern trends in hyperspectral image analysis: A review," *IEEE Access*, vol. 6, pp. 14118–14129, 2018.

[2] G. Bilgin, S. Erturk, and T. Yildirim, "Segmentation of hyperspectral images via subtractive clustering and cluster validation using one-class support vector machines," *IEEE Trans. Geosci. Remote Sens.*, vol. 49, no. 8, pp. 2936–2944, Aug. 2011.

[3] T. Dundar and T. Ince, "Sparse representation-based hyperspectral image classification using multiscale superpixels and guided filter," *IEEE Geosci. Remote Sens. Lett.*, vol. 16, no. 2, pp. 246–250, Feb. 2019.

[4] Y. Chen, X. Zhao, and X. Jia, "Spectral–spatial classification of hyperspectral data based on deep belief network," *IEEE J. Sel. Topics Appl. Earth Observ. Remote Sens.*, vol. 8, no. 6, pp. 2381–2392, Jun. 2015.

[5] W. Zhao and S. Du, "Spectral–spatial feature extraction for hyperspectral image classification: A dimension reduction and deep learning approach," *IEEE Trans. Geosci. Remote Sens.*, vol. 54, no. 8, pp. 4544–4554, Aug. 2016.

[6] P. Zhong, Z. Gong, S. Li, and C.-B. Schönlieb, "Learning to diversify deep belief networks for hyperspectral image classification," *IEEE J. Sel. Topics Appl. Earth Observ. Remote Sens.*, vol. 55, no. 6, pp. 3516–3530, Jun. 2017.

[7] L. Mou, P. Ghamisi, and X. X. Zhu, "Deep recurrent neural networks for hyperspectral image classification," *IEEE Trans. Geosci. Remote Sens.*, vol. 55, no. 7, pp. 3639–3655, Jul. 2017.

[8] A. Santara *et al.*, "BASS net: Band-adaptive spectral-spatial feature learning neural network for hyperspectral image classification," *IEEE Trans. Geosci. Remote Sens.*, vol. 55, no. 9, pp. 5293–5301, Sep. 2017.

[9] Q. Gao, S. Lim, and X. Jia, "Hyperspectral image classification using convolutional neural networks and multiple feature learning," *Remote Sens.*, vol. 10, no. 2, p. 299, 2018.

[10] Q. Wang, J. Wan, F. Nie, B. Liu, C. Yan, and X. Li, "Hierarchical feature selection for random projection," *IEEE Trans. Neural Netw. Learn. Syst.*, vol. 30, no. 5, pp. 1581–1586, May 2019.

[11] Q. Wang, X. He, and X. Li, "Locality and structure regularized low rank representation for hyperspectral image classification," *IEEE Trans. Geosci. Remote Sens.*, vol. 57, no. 2, pp. 911–923, Feb. 2019.

[12] B. Taskesen, A. Koz, A. Alatan, and O. Weatherbee, "Change detection for hyperspectral images using extended mutual information and oversegmentation," in *Proc. WHISPERS*, Sep. 2018, pp. 1–5.

[13] S. Jia, K. Wu, J. Zhu, and X. Jia, "Spectral–Spatial Gabor surface feature fusion approach for hyperspectral imagery classification," *IEEE Trans. Geosci. Remote Sens.*, vol. 57, no. 2, pp. 1142–1154, Feb. 2019.

[14] J. Nalepa, M. Myller, and M. Kawulok, "Validating hyperspectral image segmentation," *IEEE Geosci. Remote Sens. Lett.*, vol. 16, no. 8, pp. 1264–1268, Aug. 2019.

[15] M. Marcinkiewicz, M. Kawulok, and J. Nalepa, "Segmentation of multispectral data simulated from hyperspectral imagery," in *Proc. IEEE IGARSS*, 2019, pp. 1–4.

[16] L. Windrim, A. Melkumyan, R. J. Murphy, A. Chlingaryan, and R. Ramakrishnan, "Pretraining for hyperspectral convolutional neural network classification," *IEEE Trans. Geosci. Remote Sens.*, vol. 56, no. 5, pp. 2798–2810, May 2018.

[17] H. Lee, S. Eum, and H. Kwon, "Is pretraining necessary for hyperspectral image classification?" *CoRR*, vol. abs/1901.08658, 2019. [Online]. Available: http://arxiv.org/abs/1901.08658

[18] J. Lin, R. Ward, and Z. J. Wang, "Deep transfer learning for hyperspectral image classification," in *Proc. IEEE MMSP*, Aug. 2018, pp. 1–5.

[19] B. Liu, X. Yu, A. Yu, and G. Wan, "Deep convolutional recurrent neural network with transfer learning for hyperspectral image classification," *J. Appl. Remote Sens.*, vol. 12, no. 2, Jun. 2018, Art. no. 026028.