Triplet-Watershed for Hyperspectral Image Classification

Aditya Challa, Sravan Danda, B S Daya Sagar, Laurent Najman

Triplet-Watershed for Hyperspectral Image Classification

Aditya Challa, Sravan Danda, *Member, IEEE,* B.S.Daya Sagar, *Senior Member, IEEE,* and Laurent Najman, *Senior Member, IEEE*

***Abstract*—Hyperspectral images (HSI) consist of rich spatial and spectral information, which can potentially be used for several applications. However, noise, band correlations and high dimensionality restrict the applicability of such data. This is recently addressed using creative deep learning network architectures such as ResNet, SSRN, and A2S2K. However, the last layer, i.e the classification layer, remains unchanged and is taken to be the softmax classifier. In this article, we propose to use a watershed classifier. Watershed classifier extends the watershed operator from Mathematical Morphology for classification. In its vanilla form, the watershed classifier does not have any trainable parameters. In this article, we propose a novel approach to train deep learning networks to obtain representations suitable for the watershed classifier. The watershed classifier exploits the connectivity patterns, a characteristic of HSI datasets, for better inference. We show that exploiting such characteristics allows the Triplet-Watershed to achieve state-of-art results. These results are validated on Indianpines (IP), University of Pavia (UP), and Kennedy Space Center (KSC) datasets, relying on simple convnet architecture using a quarter of parameters compared to previous state-of-the-art networks.**

***Index Terms*—Hyperspectral Imaging, Watershed, Triplet Loss, Deep Learning, Classification**

I. Introduction

HYPERSPECTRAL imaging has several applications ranging across different domains [1]. It has seen applications in earth observations [2] land cover classification [3] etc. Hyperspectral datasets have rich information both spatially and spectrally. However, spectral and spatial correlations make a lot of such information redundant. One can obtain efficient representations using techniques such as band selection [4], [5] subspace learning [6], [7] multi-modal learning [8] low-rank representation [9].

Large number of bands, spatial and spectral feature correlations and curse of dimensionality make Hyperspectral image classification challenging. Conventional approaches use hand crafted features with techniques such as scale-invariant feature transform (SIFT) [10] sparse representation [11] principal component analysis [12] independent component analysis [13]. Classic approaches to classification such as support vector machines (SVM) [2], neural networks [14] and logistic regression [15] aimed at exploiting the spectral signatures alone. Using spatial features have been extremely useful in obtaining better representations and higher classification accuracies [16]–[18]. Multiple kernel learning [19]–[21] use hand-designed kernels to exploit the spectral-spatial interactions. Deep learning approaches, especially CNNs, have been adapted to exploit the spectral-spatial information. [22] proposes a 3D-CNN feature extractor to obtain combined spectral-spatial features. [23] adapts CNN to a two-branch architecture to extract joint spectral-spatial features. [24] used 3D volumes to extract spectral-spatial features, which may be improved using multi-scale approaches [25]. Spectral-spatial residual network (SSRN) proposed in [26] uses residual networks to remove the declining accuracy phenomenon. Residual Spectral–Spatial Attention Networks (RSSAN) [27] exploit the concept of attention to improve on SSRNs. [28] proposes Attention-Based Adaptive Spectral-Spatial Kernel Residual networks (A2S2K) by exploiting adaptive kernels. [29] uses graph convolution networks and [30] uses capsule networks. Most of these approaches tackle the problem of Hyperspectral image classification by considering novel architectures. In this article, we take a different route to propose a novel classifier based on the watershed operator.

Watershed operator from Mathematical Morphology [31], [32] has been widely used for image segmentation, and, in particular, for Hyperspectral images [33], [34]. In [34], the authors combine (by majority voting) several watersheds computed on gradients of different bands. They observe that *class-specific accuracies were improved by using the spatial information in the classification for almost all the classes*, a result that we are going to confirm in the present paper.

In [35] the watershed operator is adapted to edge-weighted graphs. It is shown there that the watershed is closely related to the minimum spanning tree (MST) of the graph. Watersheds have also been highly successful as a post-processing tool for image segmentation [36]–[38]. In [39] the authors learn a representation suitable for MST-based classification. In [40] the authors learn a representation suitable to mutex-watershed, a different version of the watershed.

Departing from images, in our previous work [41] we have proposed to use the watershed operator as a generic classifier. We showed that it obtains a *maximum margin partition* similar to the support vector machine. We further showed that ensemble watersheds obtain comparable performance to other classifiers such as random forests. In this article we propose a

Aditya Challa is with the Department of Computer Science and Automation, Indian Institute of Science, Bengaluru, India, 5600012 e-mail:aditya.challa.20@gmail.com.

Sravan Danda is with APPCAIR, Department of Computer Science and Information Systems, BITS Pilani K K Birla Goa Campus, NH-17B, Zuarinagar, Goa 403726. email:sravan8809@gmail.com

B. S. Daya Sagar is with Systems Science and Informatics Unit, Indian Statistical Institute, Bengaluru, Karnataka, 560059 email: bsdsagar@yahoo.co.uk

Laurent Najman is with Université Gustave Eiffel, LIGM, Equipe A3SI, ESIEE, France. email:laurent.najman@esiee.fr

TABLE I
OVERALL ACCURACY (OA) VS NUMBER OF PARAMETERS. OBSERVE THAT THE PROPOSED METHOD HAS VERY LESS NUMBER OF PARAMETERS BUT OUTPERFORMS THE CURRENT STATE-OF-THE-ART APPROACHES. IP INDICATES INDIAN PINES DATASET. UP DENOTES UNIVERSITY OF PAVIA DATASET AND KSC INDICATES THE KENNEDY SPACE CENTRE DATASET.

	# params	IP	UP	KSC
A2S2K [28]	370.7K	98.66	99.85	99.34
SSRN [26]	364.1K	98.38	99.77	99.29
ENL-FCN [42]	113.9K	96.15	99.76	99.25
ResNet34 [43]	21.9M	92.44	97.38	79.73
Triplet-Watershed	**87.6K**	**99.57**	**99.98**	**99.72**

novel approach, simple and efficient, called *Triplet-Watershed* to learn representations (also known as embeddings) suitable for the watershed classifier.

Why watershed classifier? Previous work on hyperspectral image classification, as discussed above, establish that one must use both spatial and spectral aspects to obtain good classifiers. They achieve this with creative approaches to design neural networks such as adaptive kernels, attention mechanism, etc. However, most of these still use conventional softmax classifier. The watershed classifier naturally uses spatial information for inference. Thus, it allows us to use simpler networks for representation. Table I shows the overall accuracy scores obtained by our approach and other state of art methods. It also shows the number of parameters used. Observe that Triplet-Watershed parameters are just 25% of those of the current state-of-art (A2S2K) approach.

The main contributions of this article are the following.

(i) We propose a novel approach, namely *the Triplet-Watershed*, to learn a representation suitable to the watershed classifier. This representation is referred to as *watershed representations* in the rest of the article.
(ii) The Triplet-Watershed achieves state-of-art results on the hyperspectral datasets with very simple networks, using much fewer parameters than the previous state-of-the-art approaches.
(iii) The framework used here to obtain representations is not restricted to watershed classifiers. It can be extended to use with other classifiers such as random forest or k-nearest neighbours as well, although watershed results outperform other classifiers on our datasets.
(iv) The main insight of our paper is that enforcing spatial connectivity (achieved thanks to the watershed classifier) during the training is a relevant constraint for hyperspectral classification.

Overview: Section II reviews the watershed classifier and the required terminology for the rest of the article. In section III we design the neural net (NN) and the training procedure to learn watershed representations. Section IV provides empirical analysis.

II. WATERSHED CLASSIFIER

The watershed classifier is defined on an edge-weighted graph. We follow the exposition as given in [41]. $G = (V, E, W)$ denotes the edge-weighted graph. Here V denotes the set of vertices, E denotes the set of edges which is a subset of $V \times V$ and $W : E \to \mathbb{R}^+$ denotes the edge weight assigned to each edge. We assume that the edge weights are all positive in this article.

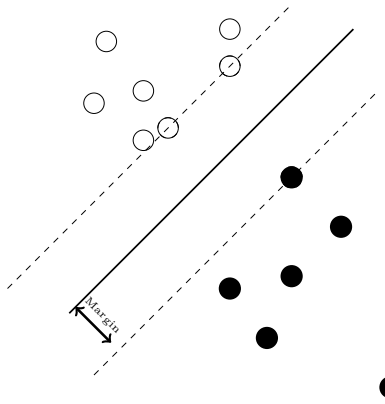


Fig. 1. Illustration of maximum margin for support vector machines (SVM) [41]. The key observation is - The margin is defined as the minimum distance between the training point labeled 0 and what would be labeled 1 after classification. And vice versa. The aim of the (linear) SVM classifier is to obtain a decision boundary that maximizes the margin. This can be extended to obtain a maximum-margin partition on an edge-weighted graph using (2), a solution of which is provided by the watershed classifier.

The (two-class) classification problem on the edge-weighted graph is stated as - Let $X_0, X_1 \subset V$ denote the labeled subset of vertices labeled 0 and 1 respectively. Classification problem requires a partition of $V = M_0 \cup M_1$ with $M_0 \cap M_1 = \emptyset$. With an additional constraint that $X_0 \subset M_0$ and $X_1 \subset M_1$. Here M_0 denotes all the vertices labeled 0 after classification and M_1 denotes all the vertices labeled 1. We also assume there exists a dissimilarity measure $\rho(x, y)$ between two vertices $x, y \in V$. This measure extends to subsets as

$$\rho(X, Y) = \min_{x \in X, y \in Y} \rho(x, y) \tag{1}$$

where X, Y are arbitrary subsets of V. Observe that there exist several partitions of $V = M_0 \cup M_1$ which satisfy the above conditions. Of these partitions, we only use the *Maximum Margin Partitions*, i.e the partitions which maximize

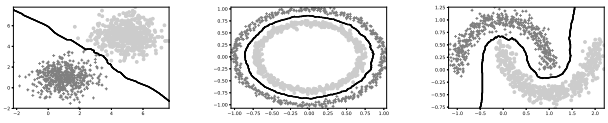
$$\min\{\rho(X_0, M_1), \rho(X_1, M_0)\} \tag{2}$$

This follows from the maximum margin principle of support vector machines (SVM). From figure 1, a key observation can be made - The margin for the SVM is the minimum distance between training point labeled 0 and what would be labeled 1 after classification. And vice versa. Linear SVM tries to obtain the boundary to maximize this margin. This can be extended to the edge-weighted graphs using (2).

The *Watershed Classifier* is defined by considering the dissimilarity measure to be

$$\rho(x, y) \coloneqq \rho_{max}(x, y) = \min_{\pi \in \Pi(x,y)} \max_{e \in \pi} W(e) \tag{3}$$

where π denotes a specific path between x, y. Π denotes the set of all possible paths. ρ_{max} is sometimes referred to as *pass value*.



(a) (b) (c)

Fig. 2. Figure illustrating the watershed boundaries [41]. Observe that in all these cases the boundary is in-between the classes. Also, it is in the middle of the zero density (no points exist) regions. This maximizes the margin between the boundaries and the classes. This is consistent with the maximum margin principle of SVM.

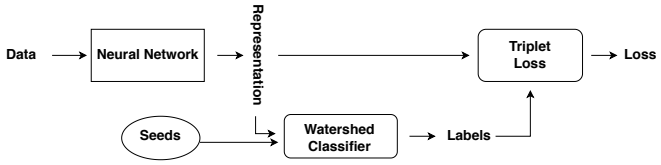


Fig. 3. Schematic of learning representations for the watershed classifier. Using a generic neural network we obtain the representation for the dataset. These representations are fed into the watershed classifier to obtain the labels using the seeds. Using the labels and the representation, we use triplet loss to compute the loss and also for obtaining the parameters for the neural network. Observe that the watershed classifier needs to be computed at every epoch.

If each edge-weight indicates the height of the corresponding edge, then $\rho_{max}(x, y)$ indicates the minimum height one has to climb to reach y from x. When the points belong to a Euclidean space, the edge weight is given by Euclidean distance. That is, the edge weight indicates the distance between the points. Hence $\rho_{max}(x, y)$ would indicate the minimum "jump" one has to make to reach y from x. Thus the boundaries (in cases where the classes are separable) would be along the low-density regions between classes. This is illustrated in Figure 2. In all the cases, the boundary is between the classes such that we have the maximum margin. This is consistent with the maximum margin principle of SVM.

Given the edge-weighted graph, the **Watershed algorithm** extends the Maximum Margin Partition principle to several classes and obtains the labels using the UNIONFIND data structure. This is described in algorithm 1.

Algorithm 1 Watershed clustering algorithm [41]

Input: edge-weighted graph $G = (V, E, W)$. A subset of labeled points $V_l \subset V$.

Output: Labels for each of the vertices L

1: Sort the edges E in increasing order w.r.t W.
2: Initialize the union-find data structure UF,
3: **for** $e = (e_x, e_y)$ in sorted edge set E **do**
4: **if** both e_x and e_y are labeled **then**
5: do nothing
6: **else**
7: UF.union(e_x, e_y)
8: **end if**
9: **end for**
10: Label each vertex of the connected component using labels V_l.
11: **return** Labels of the vertices.

Observe that step (10) is possible since each connected component would have exactly one unique label. One can see that watershed clustering is a semi-supervised algorithm, in the sense that it propagates the known labels to points with unknown label.

In practice, it has been observed that ensemble techniques improve the robustness of watershed classifier. This is achieved using only a subset of labeled points and only a subset of features and taking the weighted average. Details can be found in [41]. We refer to these two approaches as *single watershed classifier* and *ensemble-watershed classifier*.

III. LEARNING REPRESENTATIONS FOR THE WATERSHED CLASSIFIER

The previous section described how one can obtain the labels using the watershed classifier. In [41], it was shown that this compares reasonably well to other classifiers such as SVM, random forests, etc. However, observe that this classifier has *no trainable parameters*. In this section, we develop an approach to train a neural network for learning representations suitable to the watershed classifier.

A key observation is - Watershed classifier reduces the distances within each component and increases the distance across components. This leads to the schematic in figure 3. First, we use a generic neural network to obtain the representations for the dataset. These representations, along with a subset of labeled points, are used with the watershed classifier to obtain the labels. Using these labels, we obtain a metric-learning loss to decide if two pixels are either in the same component (same label) of the watershed or in two different components (different label). More precisely, we use triplet loss [44], [45] to learn the watershed representation. For training, this cost is minimized using standard autograd packages such as pytorch.

Why schematic in figure 3 learns watershed representations? Triplet loss function uses {(anchor, postive, negative)} triplets for computation of the cost. It compares an anchor-input to a positive-input and a negative-input. The distance from the anchor-input to the positive-input is minimized, and the distance from the anchor-input to the negative-input is maximized using the cost

$$\min\{d(\text{anchor}, \text{positive}) - d(\text{anchor}, \text{negative}) + \alpha\}_+ \quad (4)$$

where $\{*\}_+$ denotes the function $\max\{0, *\}$. By enforcing the order of distances, triplet loss models embed in the way that a pair of samples with the same label are smaller in distance than those with different labels. When watershed labels are used to obtain {(anchor, postive, negative)} triplets, this leads to representations that are compatible with the watershed classifier.

Remark (Supervised vs Semi-Supervised) : Recall that the watershed classifier uses a subset of training points (referred to as seeds) to obtain the labels of other training points. These labels are then used to the train the network with triplet loss. However, in the case of semi-supervised learning unlabeled data is also available at train time. These points can be labeled and be used to train the network. In this article we

use the semi-supervised approach, randomly choosing some seeds for the watershed classifier that iteratively propagates their labels to their most resembling neighbours, obtaining the connected components. Hence the combination of watershed clustering and triplet loss ensures that points with the most resembling representations are indeed clustered together, in the same connected component.

Training Dynamics

To summarize the entire training procedure of Triplet-Watershed, at each epoch

1) Obtain the representations for all the points using the neural network.
2) We consider a randomly chosen subset of labeled points as seeds
3) Propagate the labels to all points using the watershed classifier
4) Use the watershed labels to generate $\{$(anchor, positive, negative)$\}$ triplets
5) Use the triplet loss to train the neural network.

Few obvious questions follow - (a) When would the training converge? (b) What is the steady-state obtained?

Note that the training would converge when there would be no further improvement in the triplet-loss. At this stage, the out-of-box score[1] of the watershed classifier would not improve as well. This implies that - all pairs of points with the same labels and within the same component have similar representation. Hence, we obtain 100% out-of-box accuracy[2] with watershed classifier.

Remark (Overfitting): Traditional machine learning advices against reaching 100% training accuracy as the models might be overfitting. However, recent deep learning trends point to the contrary. Several deep learning models can indeed fit random data with 100% accuracy [46]. It is still an open question to understand the generalization ability of these models. However, few observations point to the *inductive bias* [47] as the reason behind good generalization. In our case, the inductive bias is dictated by the graph constructed from the data (union of 4-adjacency and EMST edges).

Also, note that during training we use a single watershed classifier. While, at inference, we use an ensemble-watershed classifier. This ensures robustness during inference.

IV. Empirical Analysis

In this section, we explore the application of the watershed classifier to the hyperspectral image classification task. We use the standard evaluation metrics for comparison:

(i) Overall Accuracy (OA): it measures the overall accuracy across all samples, not considering the class imbalance.
(ii) Average Accuracy (AA): it measures the average accuracy across the classes and
(iii) Kappa Coefficient (κ): it measures how well the estimates and groundtruth labels correspond, taking into account agreement by random chance.

[1]Accuracy on the training data excluding the seeds
[2]Here we assume that there exists at least one seed per component

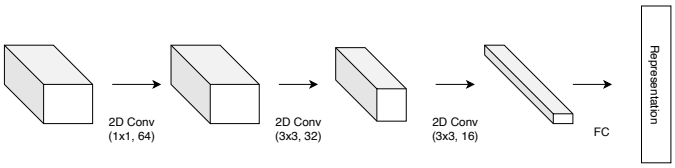


Fig. 4. Neural Network architecture used. The architecture is composed of 3 convolution layers followed by a fully connected layer to get the representation. Batch normalization is performed before each layer for efficient training. The number of parameters is 87K.

Three datasets are used for comparison.

- **Indian Pines (IP) :** Gathered by the Airborne Visible/Infrared Imaging Spectrometer (AVIRIS [48]) sensor over the test site in North-western Indiana. This data set contains 224 spectral bands within a wavelength range of 0.4 to 2.5×10^{-6} meters. The 24 bands covering region of water absorption are removed. The image spatial dimension is 145×145, and there are 16 classes not all mutually exclusive.
- **Kennedy Space Centre (KSC) :** The Kennedy Space Center (KSC) data set was gathered on March 23, 1996 by AVIRIS [48] with wavelengths ranging from 0.4 to 2.5×10^{-6} meters. 176 spectral bands are used for analysis after removal of some low signal-to-noise ratio (SNR) bands and water absorption bands. 13 classes representing the various land cover types that occur in this environment are defined for the site.
- **University of Pavia (UP) :** Acquired by the ROSIS [49] sensor during a flight campaign over Pavia, northern Italy. The number of spectral bands is 103 for Pavia University and is of size 610×610 pixels. The ground truth identifies 9 classes.

We preprocess the datasets using principal component analysis (PCA) [50] to obtain orthogonal components. We use 200 principle components for IP, 176 for KSC and 103 for UP datasets. The train/test split is obtained randomly using 10% for training and 90% for testing.

Graph Construction: Note that the watershed classifier is defined on edge-weighted graphs. This is constructed as follows

- The set of vertices V is taken to be the set of all the pixels in the dataset ignoring the $\{\text{labels} = 0\}$ class. Since, these points do not have any groundtruth labels.
- The edge set E is taken to be the union of 4-adjacency edges induced by the vertex set V (on the image) and EMST (Euclidean Minimum Spanning Tree) edges on the feature space. The EMST edges are obtained by considering the top 32 principal components and constructing the MST on V using the EMST algorithm [51].
- Given a representation obtained thanks to the neural network, the edge weights are computed using Euclidean distance. This representation (and hence the edge weights themselves) is updated at every epoch during training, while the edge set itself is never updated.

In all the experiments we use the neural net architecture as shown in figure 4. We consider a patch ($11 \times 11 \times$ #Bands) around each pixel of the input hyperspectral image, suitably

TABLE II
OA, AA, AND κ VALUES ON IP DATASET USING 10% OF SAMPLES FOR TRAINING

			Classic approaches			Deep-Learning approaches		
Class	Train	Test	RF [52]	SVM [2]	Ensemble-Watershed	SSRN [26]	A2S2K [28]	Triplet-Watershed
1	4	42	28.46 ± 0.061	51.22 ± 0.190	41.43 ± 0.2079	57.78 ± 0.423	97.56 ± 0.034	**100.00 ± 0.0000**
2	142	1286	56.63 ± 0.024	81.22 ± 0.037	81.07 ± 0.0202	98.37 ± 0.012	**98.62 ± 0.010**	**98.62 ± 0.0151**
3	83	747	48.42 ± 0.013	65.82 ± 0.013	71.49 ± 0.0250	97.47 ± 0.010	98.58 ± 0.006	**100.00 ± 0.0000**
4	23	214	33.49 ± 0.025	57.75 ± 0.041	45.70 ± 0.0327	99.12 ± 0.010	98.29 ± 0.014	**100.00 ± 0.0000**
5	48	435	85.21 ± 0.025	90.04 ± 0.014	92.78 ± 0.0286	97.79 ± 0.013	**99.02 ± 0.003**	97.98 ± 0.0254
6	73	657	92.64 ± 0.027	96.25 ± 0.006	98.57 ± 0.0033	98.50 ± 0.010	98.71 ± 0.010	**99.97 ± 0.0006**
7	2	26	2.67 ± 0.038	73.33 ± 0.019	99.17 ± 0.0167	66.67 ± 0.471	93.10 ± 0.097	**100.00 ± 0.0000**
8	47	431	97.67 ± 0.015	97.98 ± 0.006	98.14 ± 0.0075	96.45 ± 0.029	98.83 ± 0.016	**100.00 ± 0.0000**
9	2	18	9.26 ± 0.094	50.00 ± 0.045	37.50 ± 0.1854	56.25 ± 0.418	74.26 ± 0.038	**100.00 ± 0.0000**
10	97	875	60.91 ± 0.047	73.87 ± 0.018	85.81 ± 0.0227	98.33 ± 0.009	98.21 ± 0.016	**99.75 ± 0.0040**
11	245	2210	87.88 ± 0.019	82.90 ± 0.012	86.68 ± 0.0105	99.08 ± 0.005	99.09 ± 0.001	**99.61 ± 0.0054**
12	59	534	41.26 ± 0.030	74.91 ± 0.043	69.51 ± 0.0182	98.46 ± 0.009	98.37 ± 0.013	**99.89 ± 0.0022**
13	20	185	90.09 ± 0.040	96.94 ± 0.021	99.35 ± 0.0079	100.0 ± 0.000	99.80 ± 0.002	**100.00 ± 0.0000**
14	126	1139	95.46 ± 0.014	93.82 ± 0.010	92.59 ± 0.0085	98.63 ± 0.010	99.22 ± 0.007	**100.00 ± 0.0000**
15	38	348	41.11 ± 0.029	60.42 ± 0.044	54.48 ± 0.0396	99.24 ± 0.005	97.86 ± 0.013	**100.00 ± 0.0000**
16	9	84	79.37 ± 0.030	91.27 ± 0.054	79.29 ± 0.1163	95.63 ± 0.062	95.93 ± 0.057	**98.10 ± 0.0267**
OA	1018	9231	72.98 ± 0.006	82.00 ± 0.006	83.75 ± 0.0076	98.38 ± 0.004	98.66 ± 0.004	**99.57 ± 0.0026**
AA			59.41 ± 0.005	77.36 ± 0.019	77.10 ± 0.0228	91.11 ± 0.080	96.59 ± 0.003	**99.62 ± 0.0029**
κ			0.6862 ± 0.007	0.7941 ± 0.007	0.8143 ± 0.0086	0.9815 ± 0.005	0.9848 ± 0.005	**0.9951 ± 0.0030**

TABLE III
OA, AA, AND κ VALUES ON UP DATASET USING 10% OF SAMPLES FOR TRAINING

			Classic approaches			Deep-Learning approaches		
Class	Train	Test	RF [52]	SVM [2]	Ensemble-Watershed	SSRN [26]	A2S2K [28]	Triplet-Watershed
1	663	5968	91.11 ± 0.007	94.30 ± 0.008	94.34 ± 0.0032	99.85 ± 0.001	99.91 ± 0.000	**100.0 ± 0.000**
2	1864	16785	98.11 ± 0.003	97.65 ± 0.002	95.24 ± 0.0051	99.98 ± 0.000	99.99 ± 0.000	**100.0 ± 0.000**
3	209	1890	67.71 ± 0.014	81.26 ± 0.018	69.39 ± 0.0151	99.68 ± 0.003	99.88 ± 0.001	**99.8 ± 0.004**
4	306	2758	88.20 ± 0.006	94.63 ± 0.004	78.69 ± 0.0058	99.92 ± 0.001	99.95 ± 0.001	**99.96 ± 0.001**
5	134	1211	98.93 ± 0.002	99.20 ± 0.002	87.46 ± 0.0110	99.94 ± 0.000	**100.0 ± 0.000**	**100.0 ± 0.000**
6	502	4527	72.14 ± 0,022	90.58 ± 0,008	61.37 ± 0.0111	99.95 ± 0.001	99.91 ± 0.001	**99.99 ± 0.001**
7	133	1197	75.69 ± 0.017	85.71 ± 0.011	75.49 ± 0.0295	**100.0 ± 0.000**	**100.0 ± 0.000**	**100.0 ± 0.000**
8	368	3314	89.64 ± 0.013	88.20 ± 0.003	74.65 ± 0.0044	98.28 ± 0.015	98.88 ± 0.006	**99.97 ± 0.001**
9	94	853	99.77 ± 0.002	99.84 ± 0.001	99.77 ± 0.0015	99.39 ± 0.003	99.78 ± 0.003	**100.0 ± 0.000**
OA	4273	38503	90.41 ± 0.001	94.19 ± 0.002	86.13 ± 0.0023	99.77 ± 0.001	99.85 ± 0.001	**99.98 ± 0.001**
AA			86.81 ± 0.002	92.38 ± 0.003	81.82 ± 0.0039	99.66 ± 0.001	99.81 ± 0.001	**99.97 ± 0.001**
κ			0.8710 ± 0.002	0.9229 ± 0.002	0.8136 ± 0.0030	0.9969 ± 0.001	0.9981 ± 0.001	**0.9998 ± 0.001**

TABLE IV
OA, AA, AND κ VALUES ON KSC DATASET USING 10% OF SAMPLES FOR TRAINING

			Classic approaches			Deep-Learning approaches		
Class	Train	Test	RF [52]	SVM [2]	Ensemble-Watershed	SSRN [26]	A2S2K [28]	Triplet-Watershed
1	76	685	94.79 ± 0.012	95.43 ± 0.023	96.23 ± 0.0085	99.95 ± 0.001	99.95 ± 0.001	**100.0 ± 0.0000**
2	24	219	81.58 ± 0 047	83.71 ± 0.012	89.59 ± 0.0247	**100.0 ± 0.000**	98.68 ± 0.019	**100.0 ± 0.0000**
3	25	231	86.09 ± 0 020	78.41 ± 0.218	83.98 ± 0.0341	99.66 ± 0.005	98.72 ± 0.012	**100.0 ± 0.0000**
4	25	227	71.22 ± 0.061	27.17 ± 0.173	69.60 ± 0.0406	91.22 ± 0.047	94.27 ± 0.042	**96.56 ± 0.0423**
5	16	145	47.59 ± 0.060	22.99 ± 0.170	65.52 ± 0.0474	**100.0 ± 0.000**	94.46 ± 0.050	99.86 ± 0.0028
6	22	207	48.22 ± 0.014	36.89 ± 0.078	53.33 ± 0.0526	98.45 ± 0.022	99.82 ± 0.003	**99.52 ± 0.0000**
7	10	95	79.43 ± 0 096	87.94 ± 0.027	85.05 ± 0.0234	95.42 ± 0.050	99.61 ± 0.005	**100.0 ± 0.0000**
8	43	388	78.61 ± 0.054	70.19 ± 0.073	91.24 ± 0.0297	99.80 ± 0.003	**100.0 ± 0.000**	**99.90 ± 0.0000**
9	52	468	89.46 ± 0.011	85.33 ± 0.021	93.08 ± 0.0193	**100.0 ± 0.000**	**100.0 ± 0.000**	**100.0 ± 0.0000**
10	40	364	88.43 ± 0.034	78.88 ± 0.069	92.64 ± 0.0150	**100.0 ± 0.000**	**100.0 ± 0.000**	**100.0 ± 0.0000**
11	41	378	95.58 ± 0.014	93.81 ± 0.008	94.44 ± 0.0261	**100.0 ± 0.000**	**100.0 ± 0.000**	**100.0 ± 0.0000**
12	50	453	82.63 ± 0.032	86.98 ± 0.009	86.98 ± 0.0119	**100.0 ± 0.000**	**100.0 ± 0.000**	99.21 ± 0.0159
13	92	835	99.60 ± 0.002	**100.0 ± 0.000**	99.69 ± 0.0022	**100.0 ± 0.000**	**100.0 ± 0.000**	**100.0 ± 0.0000**
OA	516	4695	86.17 ± 0.004	81.27 ± 0.008	89.54 ± 0.0038	99.29 ± 0.004	99.34 ± 0.0008	**99.72 ± 0.0023**
AA			80.25 ± 0.004	72.90 ± 0.021	84.72 ± 0.0038	98.80 ± 0.008	98.88 ± 0.0018	**99.62 ± 0.0032**
κ			0.8459 ± 0.004	0.7909 ± 0.009	0.8834 ± 0.0042	0.9921 ± 0.004	0.9927 ± 0.001	**0.9969 ± 0.0026**

TABLE V
OA, AA, AND κ VALUES ON IP DATASET USING SEMI-SUPERVISED APPROACHES.

Class	Train	Test	S2GCN [53]	SSRN [26]	DC-GCN [54]	Triplet-Watershed
1	30	16	**100.0 ± 0.0000**	93.24 ± 0.0263	**100.00 ± 0.0000**	**100.00 ± 0.0000**
2	30	1398	84.43 ± 0.0250	76.63 ± 0.0596	91.28 ± 0.0360	**91.69 ± 0.0194**
3	30	800	82.87 ± 0.0553	68.78 ± 0.0753	92.88 ± 0.0396	**95.25 ± 0.0610**
4	30	207	93.08 ± 0.0195	87.64 ± 0.0249	98.11 ± 0.0151	**100.00 ± 0.0000**
5	30	453	97.13 ± 0.0134	86.72 ± 0.0154	95.54 ± 0.0339	**98.63 ± 0.0171**
6	30	700	97.29 ± 0.0127	92.05 ± 0.0182	98.67 ± 0.0104	**100.00 ± 0.0000**
7	15	13	92.31 ± 0.0000	95.66 ± 0.0051	**100.00 ± 0.0000**	**100.00 ± 0.0000**
8	30	448	99.03 ± 0.0093	95.90 ± 0.0297	**100.00 ± 0.0000**	**100.00 ± 0.0000**
9	15	5	**100.00 ± 0.0000**	**100.00 ± 0.0000**	**100.00 ± 0.0000**	**100.00 ± 0.0000**
10	30	942	93.77 ± 0.0373	82.42 ± 0.0324	91.91 ± 0.0378	**98.22 ± 0.0232**
11	30	2425	84.98 ± 0.0282	82.23 ± 0.0288	91.79 ± 0.0379	**94.43 ± 0.0229**
12	30	563	80.05 ± 0.0517	69.09 ± 0.0436	90.17 ± 0.0554	**99.08 ± 0.0185**
13	30	175	99.43 ± 0.0000	95.78 ± 0.0075	99.65 ± 0.0027	**100.00 ± 0.0000**
14	30	1235	96.73 ± 0.0092	86.52 ± 0.0243	99.73 ± 0.0066	**99.87 ± 0.0026**
15	30	356	86.80 ± 0.0342	73.12 ± 0.0528	99.94 ± 0.0016	**100.00 ± 0.0000**
16	30	63	**100.00 ± 0.0000**	86.21 ± 0.0130	**100.00 ± 0.0000**	99.37 ± 0.0078
OA			89.4 ± 0.0108	88.34 ± 0.0173	94.65 ± 0.1210	**96.74 ± 0.0194**
AA			92.9 ± 0.0104	85.75 ± 0.0069	96.85 ± 0.0040	**98.53 ± 0.0098**
κ			0.880 ± 0.012	0.866 ± 0.019	0.944 ± 0.014	**0.9627 ± 0.0221**

TABLE VI
OA, AA, AND κ VALUES ON UP DATASET USING SEMI-SUPERVISED APPROACHES.

Class	Train	Test	S2GCN [53]	SSRN [26]	DC-GCN [54]	Triplet-Watershed
1	30	6601	92.78 ± 0.0379	98.80 ± 0.0110	92.85 ± 0.0351	**99.56 ± 0.0088**
2	30	18619	87.06 ± 0.0447	98.45 ± 0.0054	97.53 ± 0.0140	**100.00 ± 0.0000**
3	30	2069	87.97 ± 0.0477	77.05 ± 0.1024	97.94 ± 0.0118	**99.85 ± 0.0084**
4	30	3034	90.85 ± 0.0094	83.02 ± 0.0907	94.57 ± 0.0109	**99.99 ± 0.0003**
5	30	1315	**100.00 ± 0.0000**	99.96 ± 0.0009	99.49 ± 0.0068	**100.00 ± 0.0000**
6	30	4999	88.69 ± 0.0264	87.03 ± 0.0626	98.57 ± 0.0278	**99.99 ± 0.0001**
7	30	1300	98.88 ± 0.0108	83.92 ± 0.0897	**100.00 ± 0.0000**	**100.00 ± 0.0000**
8	30	3652	89.97 ± 0.0328	88.41 ± 0.0463	**96.00 ± 0.0277**	92.15 ± 0.1560
9	30	917	98.89 ± 0.0053	99.97 ± 0.0004	97.51 ± 0.0140	**100.00 ± 0.0000**
OA			89.74 ± 0.0170	92.81 ± 0.0190	96.87 ± 0.0111	**99.20 ± 0.0129**
AA			92.80 ± 0.0047	90.73 ± 0.0226	97.16 ± 0.0076	**98.95 ± 0.0165**
κ			0.8665 ± 0.020	0.9059 ± 0.024	0.9677 ± 0.012	**0.9894 ± 0.0170**

TABLE VII
COMPARISON OF TRIPLET-WATERSHED WITH TRIPLET-RANDOM-FOREST AND TRIPLET-K-NEAREST-NEIGHBORS

	Triplet-Watershed	Triplet-RF	Triplet-KNN
IN	**99.57 ± 0.0026**	91.46 ± 0.011	90.86 ± 0.013
UP	**99.98 ± 0.001**	98.06 ± 0.007	99.62 ± 0.000
KSC	**99.72 ± 0.0023**	87.80 ± 0.039	82.38 ± 0.031

TABLE VIII
MEAN AVERAGE PRECISION (MAP) SCORES FOR THE REPRESENTATIONS.

	Triplet-Watershed	A2S2K [28]	SSRN [26]
IN	**0.9819**	0.9713	0.9135
UP	**0.9970**	0.9821	0.9703
KSC	**0.9822**	0.9837	0.9846

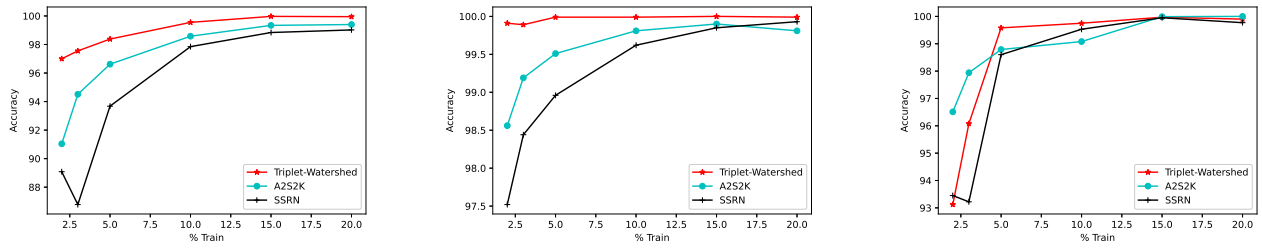
TABLE IX
TRIPLET-WATERSHED: ACCURACY VS EMBED DIMENSION

Dimension	KSC	IN	UP
16	99.53 ± 0.0031	99.45 ± 0.0025	99.95 ± 0.0002
32	99.70 ± 0.0029	99.72 ± 0.0010	99.97 ± 0.0003
64	99.54 ± 0.0017	99.67 ± 0.0011	99.98 ± 0.0001
128	99.72 ± 0.0004	99.84 ± 0.0009	99.97 ± 0.0001

padded with 0s. We use 3 conv2d layers and a fully-connected layer to obtain the representation. These representations are then used for watershed classification and training. All models are trained using stochastic gradient descent (SGD) with cyclic learning rates [55]. We use 40% of the training data as seeds for the watershed classifier. The default weight initialization by pytorch [56] is used. We use 64 as the dimension for the representations. All accuracies are reported in the format mean $\times\ 100\% \pm$ stdev to be consistent with [28]. The code is available at https://github.com/ac20/TripletWatershed_Code.

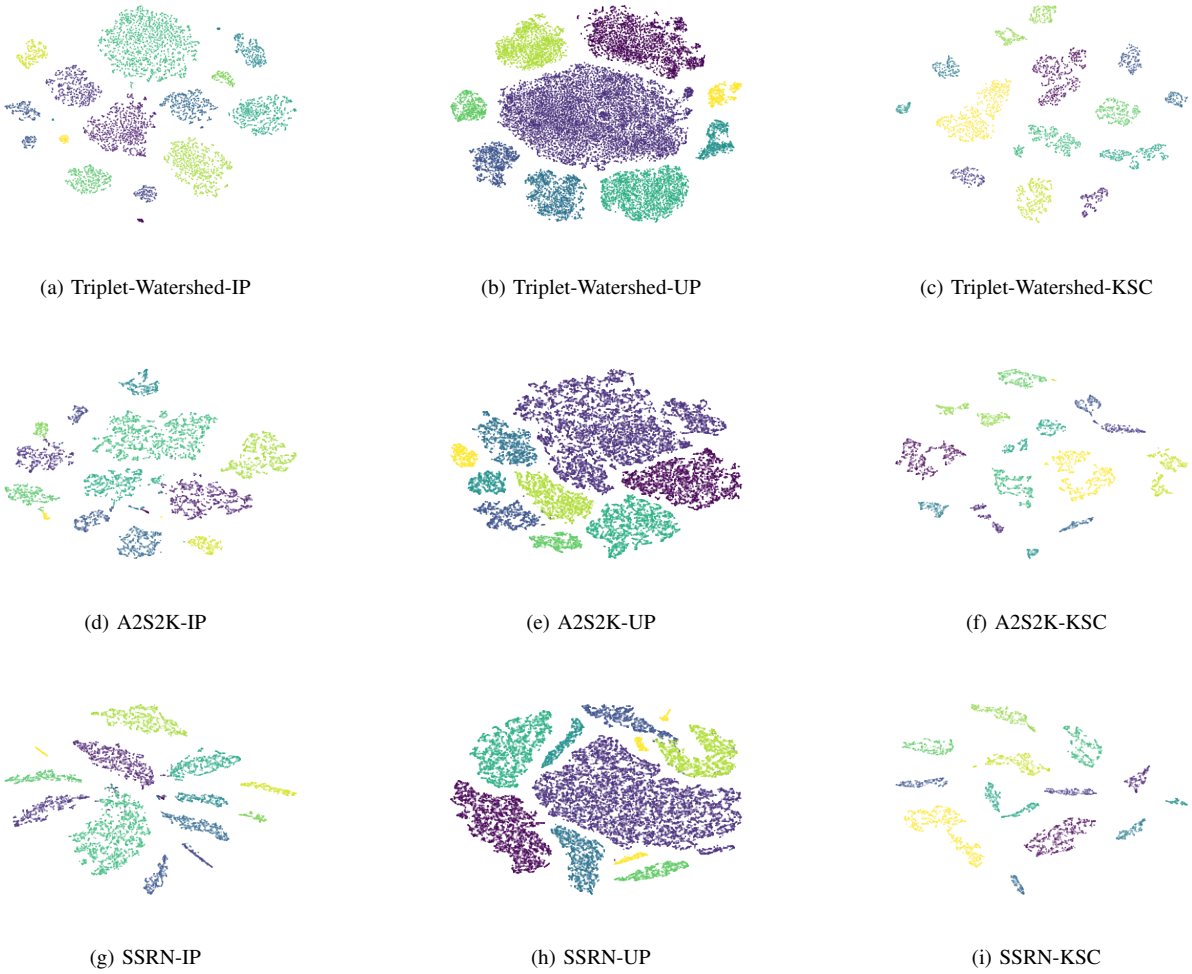
A. Supervised Classification

Firstly, we provide the results of Triplet-Watershed for supervised classification. We compare our approach with standard baselines (SVM [2] and Random Forest [52]), and



(a) IP (b) UP (c) KSC

Fig. 5. Accuracy (OA) vs % training data. We observe that Triplet-Watershed outperforms other approaches even at small sizes of training data.



(a) Triplet-Watershed-IP (b) Triplet-Watershed-UP (c) Triplet-Watershed-KSC

(d) A2S2K-IP (e) A2S2K-UP (f) A2S2K-KSC

(g) SSRN-IP (h) SSRN-UP (i) SSRN-KSC

Fig. 6. T-SNE Scatterplot of the various representations obtained. All approaches provide well-separated clusters, relatively compact. Table VIII however shows that triplet-watershed achieves a better precision (MAP score).

also with the two recent state-of-art methods SSRN [26] and A2S2K [28]. Tables II, III, IV show the results (OA, AA, κ) obtained. The train test splits per class are described in these tables. Note that Triplet-Watershed outperforms existing state-of-art A2S2KResNet [28] and other approaches in several aspects. This can be attributed to the fact that - Triplet Watershed exploits the connectivity patterns (4-adjacency and EMST edges) which exist in the datasets. Other approaches treat each pixel as a separate entity which would not exploit this observation. Simple Ensemble-Watershed results are shown in the tables as well.

B. *Semi-Supervised Classification*

We compare the Triplet-Watershed with three recent state-of-art semi-supervised approaches - S2GCN [53], SSRN [26] and DC-GCN (Dual Clustering GCN) [54]. We consider 30

samples for training if the class size is greater than 30 and 15 if the class size is less than 30. Tables V, VI show the results obtained. Observe that, once again, Triplet-Watershed obtains the state-of-art in several aspects.

C. Evaluation of Representation

Recall that accuracies in tables II-VI for Triplet-Watershed use ensemble watershed classifier. However, ensemble watershed exploits the connectivity patterns in the data. We now try to understand how well watershed representations compare with representations obtained by other approaches. Qualitatively, we use the TSNE [57] plots as in Figure 6. Note that there does not exist any major differences except that within a class, A2S2K and SSRN have "clumps" points while Triplet-Watershed has a uniform density. Quantitatively we use the mean average precision (MAP) over all points. The computation procedure is as follows:

1) Given a datapoint x_k, we order all other datapoints $\{y_i\}_i$ using an inverse function of distance, $\exp(-\text{distance})$.
2) Labels are assigned based on whether the points $\{y_i\}_i$ belong to the same class as x_k or not with class label 1 and 0 respectively.
3) Average precision (AP) computes the area under the precision-recall curve.
4) The AP scores are averages over all points $\{x_k\}_k$ to obtain the MAP score.

This procedure is as suggested in [58] to evaluate representations. The results are shown in Table VIII. Observe that the watershed outperforms the current state-of-art techniques.

D. Ablation Study

We now study the importance of various aspects of Triplet-Watershed for the accuracies.

1) Accuracy vs $\%$ training data: Figure 5 shows the plots of overall accuracy (OA) vs $\%$ training data. For IP and UP datasets, it can be seen that Triplet-Watershed outperforms other approaches even at small sizes of training data. This can be attributed to the fact that the watershed classifier propagates the information to unlabeled nodes, which is in turn used for training. (See Figure 3). For optimal performance, the watershed classifier requires at least one labeled node per component. In cases of very small training data and a large number of components, Triplet-Watershed does not perform well. This is the case for the KSC dataset at 2% and 3% training data, as shown in Figure 5.

2) Replacing Watershed With Other Classifiers: To illustrate the importance of the watershed classifier in the training pipeline (Figure 3), we replace it with Random Forest (RF) classifier and K-Nearest Neighbors (KNN) classifier with $k = 5$, referring to these as *Triplet-Random Forest* and *Triplet-K-Nearest-Neighbors*. The results are shown in Table VII. Firstly observe the dramatic improvement of accuracies with respect to vanilla classifiers (Tables II, III, IV). Also, observe that Triplet-Watershed outperforms the other techniques. This, once again, is attributed to the fact that watershed exploits the fact that classes in the groundtruth consist of connected components.

3) Accuracy vs embed dimension: Table IX shows the effect of embedding dimension on accuracy. Observe that there does not exist any significant trend with respect to the embedding dimension. We use 64 as the default embedding dimension.

V. Conclusion

In this article, we proposed a novel approach to train for the watershed classifier. We refer to this as Triplet-Watershed. We show that the watershed classifier exploits the connectivity patterns in the datasets. This leads to huge performance gains compared to other approaches which use simple softmax classifier. We prove this empirically by comparing Triplet-Watershed with existing state-of-art deep learning approaches such as A2S2K [28], SSRN [26] and also classic approaches - RF [52] and SVM [2]. We also compare the current technique with semi-supervised approaches such as S2GCN [53] and DC-GCN [54]. In each case, we achieve better accuracy while using a quarter of the parameters of the previous state-of-the-art approaches.

Acknowledgment

AC would like to thank Indian Institute of Science for the Raman Fellowship under which this work has been carried out. SD would like to acknowledge the funding received from BPGC/RIG/2020-21/11-2020/01 (Research Initiation Grant provided by BITS-Pilani K K Birla Goa Campus). The work of B. S. D. Sagar was supported by the DST-ITPAR-Phase-IV project and the Technology Innovation Hub on Data Science, Big Data Analytics and Data Curation project sanctioned under the National Mission for the Interdisciplinary Cyber-Physical Systems respectively under the Grant numbers INT/Italy/ITPAR-IV/Telecommunication/2018, and NMICPS/006/MD/2020-21. The work of Laurent Najman is supported by Programme d'Investissements d'Avenir (LabEx BEZOUT ANR-10-LABX-58).

References

[1] P. Ghamisi, N. Yokoya, J. Li, W. Liao, S. Liu, J. Plaza, B. Rasti, and A. Plaza, "Advances in hyperspectral image and signal processing: A comprehensive overview of the state of the art," *IEEE Geoscience and Remote Sensing Magazine*, vol. 5, no. 4, pp. 37–78, 2017.

[2] F. Melgani and L. Bruzzone, "Classification of hyperspectral remote sensing images with support vector machines," *IEEE Trans. Geosci. Remote. Sens.*, vol. 42, no. 8, pp. 1778–1790, 2004. [Online]. Available: https://doi.org/10.1109/TGRS.2004.831865

[3] P. O. Gislason, J. A. Benediktsson, and J. R. Sveinsson, "Random forests for land cover classification," *Pattern Recognit. Lett.*, vol. 27, no. 4, pp. 294–300, 2006. [Online]. Available: https://doi.org/10.1016/j.patrec.2005.08.011

[4] Y. Cai, X. Liu, and Z. Cai, "Bs-nets: An end-to-end framework for band selection of hyperspectral image," *IEEE Trans. Geosci. Remote. Sens.*, vol. 58, no. 3, pp. 1969–1984, 2020. [Online]. Available: https://doi.org/10.1109/TGRS.2019.2951433

[5] S. K. Roy, S. Das, T. Song, and B. Chanda, "Darecnet-bs: Unsupervised dual-attention reconstruction network for hyperspectral band selection," *IEEE Geoscience and Remote Sensing Letters*, pp. 1–5, 2020.

[6] D. Hong, N. Yokoya, J. Chanussot, J. Xu, and X. X. Zhu, "Joint and progressive subspace analysis (jpsa) with spatial-spectral manifold alignment for semi-supervised hyperspectral dimensionality reduction," 2020.

[7] D. Hong, N. Yokoya, J. Xu, and X. Zhu, "Joint and progressive learning from high-dimensional data for multi-label classification," in *Computer Vision – ECCV 2018*, V. Ferrari, M. Hebert, C. Sminchisescu, and Y. Weiss, Eds. Cham: Springer International Publishing, 2018, pp. 478–493.

[8] D. Hong, L. Gao, N. Yokoya, J. Yao, J. Chanussot, Q. Du, and B. Zhang, "More diverse means better: Multimodal deep learning meets remote-sensing imagery classification," *IEEE Transactions on Geoscience and Remote Sensing*, pp. 1–15, 2020.

[9] L. Gao, D. Yao, Q. Li, L. Zhuang, B. Zhang, and J. M. Bioucas-Dias, "A new low-rank representation based hyperspectral image denoising method for mineral mapping," *Remote Sensing*, vol. 9, no. 11, 2017. [Online]. Available: https://www.mdpi.com/2072-4292/9/11/1145

[10] Y. Li, Q. Li, Y. Liu, and W. Xie, "A spatial-spectral sift for hyperspectral image matching and classification," *Pattern Recognition Letters*, vol. 127, pp. 18–26, 2019, advances in Visual Correspondence: Models, Algorithms and Applications (AVC-MAA). [Online]. Available: https://www.sciencedirect.com/science/article/pii/S0167865518305117

[11] Y. Shao, N. Sang, C. Gao, and L. Ma, "Spatial and class structure regularized sparse representation graph for semi-supervised hyperspectral image classification," *Pattern Recognition*, vol. 81, pp. 81–94, 2018. [Online]. Available: https://www.sciencedirect.com/science/article/pii/S0031320318301171

[12] G. Licciardi, P. R. Marpu, J. Chanussot, and J. A. Benediktsson, "Linear versus nonlinear pca for the classification of hyperspectral data based on the extended morphological profiles," *IEEE Geoscience and Remote Sensing Letters*, vol. 9, no. 3, pp. 447–451, 2012.

[13] A. Villa, J. A. Benediktsson, J. Chanussot, and C. Jutten, "Hyperspectral image classification with independent component discriminant analysis," *IEEE Transactions on Geoscience and Remote Sensing*, vol. 49, no. 12, pp. 4865–4876, 2011.

[14] Y. Zhong and L. Zhang, "An adaptive artificial immune network for supervised classification of multi-/hyperspectral remote sensing imagery," *IEEE Transactions on Geoscience and Remote Sensing*, vol. 50, no. 3, pp. 894–909, 2012.

[15] J. Li, J. M. Bioucas-Dias, and A. Plaza, "Semisupervised hyperspectral image segmentation using multinomial logistic regression with active learning," *IEEE Transactions on Geoscience and Remote Sensing*, vol. 48, no. 11, pp. 4085–4098, 2010.

[16] P. Ghamisi, E. Maggiori, S. Li, R. Souza, Y. Tarablaka, G. Moser, A. De Giorgi, L. Fang, Y. Chen, M. Chi, S. B. Serpico, and J. A. Benediktsson, "New frontiers in spectral-spatial hyperspectral image classification: The latest advances based on mathematical morphology, markov random fields, segmentation, sparse representation, and deep learning," *IEEE Geoscience and Remote Sensing Magazine*, vol. 6, no. 3, pp. 10–43, 2018.

[17] L. He, J. Li, C. Liu, and S. Li, "Recent advances on spectral–spatial hyperspectral image classification: An overview and new guidelines," *IEEE Transactions on Geoscience and Remote Sensing*, vol. 56, no. 3, pp. 1579–1597, 2018.

[18] S. Li, W. Song, L. Fang, Y. Chen, P. Ghamisi, and J. A. Benediktsson, "Deep learning for hyperspectral image classification: An overview," *IEEE Transactions on Geoscience and Remote Sensing*, vol. 57, no. 9, pp. 6690–6709, 2019.

[19] G. Camps-Valls, L. Gomez-Chova, J. Munoz-Mari, J. Vila-Frances, and J. Calpe-Maravilla, "Composite kernels for hyperspectral image classification," *IEEE Geoscience and Remote Sensing Letters*, vol. 3, no. 1, pp. 93–97, 2006.

[20] M. Fauvel, J. Chanussot, and J. Benediktsson, "A spatial–spectral kernel-based approach for the classification of remote-sensing images," *Pattern Recognition*, vol. 45, no. 1, pp. 381–392, 2012. [Online]. Available: https://www.sciencedirect.com/science/article/pii/S0031320311002019

[21] L. Fang, S. Li, W. Duan, J. Ren, and J. A. Benediktsson, "Classification of hyperspectral images by exploiting spectral–spatial information of superpixel via multiple kernels," *IEEE Transactions on Geoscience and Remote Sensing*, vol. 53, no. 12, pp. 6663–6674, 2015.

[22] Y. Chen, H. Jiang, C. Li, X. Jia, and P. Ghamisi, "Deep feature extraction and classification of hyperspectral images based on convolutional neural networks," *IEEE Transactions on Geoscience and Remote Sensing*, vol. 54, no. 10, pp. 6232–6251, 2016.

[23] J. Yang, Y. Zhao, and J. C. Chan, "Learning and transferring deep joint spectral–spatial features for hyperspectral classification," *IEEE Transactions on Geoscience and Remote Sensing*, vol. 55, no. 8, pp. 4729–4742, 2017.

[24] A. Ben Hamida, A. Benoit, P. Lambert, and C. Ben Amar, "3-d deep learning approach for remote sensing image classification," *IEEE Transactions on Geoscience and Remote Sensing*, vol. 56, no. 8, pp. 4420–4434, 2018.

[25] M. He, B. Li, and H. Chen, "Multi-scale 3d deep convolutional neural network for hyperspectral image classification," in *2017 IEEE International Conference on Image Processing (ICIP)*, 2017, pp. 3904–3908.

[26] Z. Zhong, J. Li, Z. Luo, and M. Chapman, "Spectral–spatial residual network for hyperspectral image classification: A 3-d deep learning framework," *IEEE Transactions on Geoscience and Remote Sensing*, vol. 56, no. 2, pp. 847–858, 2018.

[27] M. Zhu, L. Jiao, F. Liu, S. Yang, and J. Wang, "Residual spectral–spatial attention network for hyperspectral image classification," *IEEE Transactions on Geoscience and Remote Sensing*, vol. 59, no. 1, pp. 449–462, 2021.

[28] S. K. Roy, S. Manna, T. Song, and L. Bruzzone, "Attention-based adaptive spectral-spatial kernel resnet for hyperspectral image classification," *IEEE Transactions on Geoscience and Remote Sensing*, pp. 1–13, 2020.

[29] D. Hong, L. Gao, J. Yao, B. Zhang, A. Plaza, and J. Chanussot, "Graph convolutional networks for hyperspectral image classification," *IEEE Transactions on Geoscience and Remote Sensing*, pp. 1–13, 2020.

[30] M. E. Paoletti, J. M. Haut, R. Fernandez-Beltran, J. Plaza, A. Plaza, J. Li, and F. Pla, "Capsule networks for hyperspectral image classification," *IEEE Transactions on Geoscience and Remote Sensing*, vol. 57, no. 4, pp. 2145–2160, 2019.

[31] L. Vincent and P. Soille, "Watersheds in digital spaces: An efficient algorithm based on immersion simulations," *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 13, no. 6, pp. 583–598, 1991. [Online]. Available: https://doi.org/10.1109/34.87344

[32] S. Beucher and F. Meyer, *The Morphological Approach to Segmentation: The Watershed Transformation*. CRC Press., 01 1993, vol. Vol. 34, p. 433–481.

[33] G. Noyel, J. Angulo, and D. Jeulin, "Morphological segmentation of hyperspectral images," *Image Analysis & Stereology*, vol. 26, no. 3, pp. 101–109, 2007.

[34] Y. Tarabalka, J. Chanussot, and J. A. Benediktsson, "Segmentation and classification of hyperspectral images using watershed transformation," *Pattern Recognition*, vol. 43, no. 7, pp. 2367–2379, 2010.

[35] J. Cousty, G. Bertrand, L. Najman, and M. Couprie, "Watershed cuts: Minimum spanning forests and the drop of water principle," *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 31, no. 8, pp. 1362–1374, 2009. [Online]. Available: https://doi.org/10.1109/TPAMI.2008.173

[36] S. C. Turaga, K. L. Briggman, M. Helmstaedter, W. Denk, and H. S. Seung, "Maximin affinity learning of image segmentation," in *Advances in Neural Information Processing Systems 22: 23rd Annual Conference on Neural Information Processing Systems 2009. Proceedings of a meeting held 7-10 December 2009, Vancouver, British Columbia, Canada*, Y. Bengio, D. Schuurmans, J. D. Lafferty, C. K. I. Williams, and A. Culotta, Eds. Curran Associates, Inc., 2009, pp. 1865–1873. [Online]. Available: https://proceedings.neurips.cc/paper/2009/hash/68d30a9594728bc39aa24be94b319d21-Abstract.html

[37] K. Maninis, J. Pont-Tuset, P. Arbeláez, and L. Van Gool, "Convolutional oriented boundaries: From image segmentation to high-level tasks," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 40, no. 4, pp. 819–833, 2018.

[38] S. Wolf, L. Schott, U. Köthe, and F. A. Hamprecht, "Learned watershed: End-to-end learning of seeded segmentation," in *IEEE International Conference on Computer Vision, ICCV 2017, Venice, Italy, October 22-29, 2017*. IEEE Computer Society, 2017, pp. 2030–2038. [Online]. Available: https://doi.org/10.1109/ICCV.2017.222

[39] J. Funke, F. Tschopp, W. Grisaitis, A. Sheridan, C. Singh, S. Saalfeld, and S. C. Turaga, "Large scale image segmentation with structured loss based deep learning for connectome reconstruction," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 41, no. 7, pp. 1669–1680, 2019.

[40] S. Wolf, A. Bailoni, C. Pape, N. Rahaman, A. Kreshuk, U. Köthe, and F. A. Hamprecht, "The mutex watershed and its objective: Efficient, parameter-free graph partitioning," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, pp. 1–1, 2020.

[41] A. Challa, S. Danda, B. S. D. Sagar, and L. Najman, "Watersheds for semi-supervised classification," *IEEE Signal Process. Lett.*, vol. 26, no. 5, pp. 720–724, 2019. [Online]. Available: https://doi.org/10.1109/LSP.2019.2905155

[42] Y. Shen, S. Zhu, C. Chen, Q. Du, L. Xiao, J. Chen, and D. Pan, "Efficient deep learning of nonlocal features for hyperspectral image classification," *IEEE Transactions on Geoscience and Remote Sensing*, pp. 1–15, 2020.

[43] K. He, X. Zhang, S. Ren, and J. Sun, "Deep residual learning for image recognition," in *2016 IEEE Conference on Computer Vision*

and Pattern Recognition, CVPR 2016, Las Vegas, NV, USA, June 27-30, 2016. IEEE Computer Society, 2016, pp. 770–778. [Online]. Available: https://doi.org/10.1109/CVPR.2016.90

[44] E. Hoffer and N. Ailon, "Deep metric learning using triplet network," in *3rd International Conference on Learning Representations, ICLR 2015, San Diego, CA, USA, May 7-9, 2015, Workshop Track Proceedings*, Y. Bengio and Y. LeCun, Eds., 2015. [Online]. Available: http://arxiv.org/abs/1412.6622

[45] M. Schultz and T. Joachims, "Learning a distance metric from relative comparisons," in *Advances in Neural Information Processing Systems 16 [Neural Information Processing Systems, NIPS 2003, December 8-13, 2003, Vancouver and Whistler, British Columbia, Canada]*, S. Thrun, L. K. Saul, and B. Schölkopf, Eds. MIT Press, 2003, pp. 41–48. [Online]. Available: https://proceedings.neurips.cc/paper/2003/hash/d3b1fb02964aa64e257f9f26a31f72cf-Abstract.html

[46] C. Zhang, S. Bengio, M. Hardt, B. Recht, and O. Vinyals, "Understanding deep learning requires rethinking generalization," in *5th International Conference on Learning Representations, ICLR 2017, Toulon, France, April 24-26, 2017, Conference Track Proceedings*. OpenReview.net, 2017. [Online]. Available: https://openreview.net/forum?id=Sy8gdB9xx

[47] P. W. Battaglia, J. B. Hamrick, V. Bapst, A. Sanchez-Gonzalez, V. F. Zambaldi, M. Malinowski, A. Tacchetti, D. Raposo, A. Santoro, R. Faulkner, Ç. Gülçehre, H. F. Song, A. J. Ballard, J. Gilmer, G. E. Dahl, A. Vaswani, K. R. Allen, C. Nash, V. Langston, C. Dyer, N. Heess, D. Wierstra, P. Kohli, M. Botvinick, O. Vinyals, Y. Li, and R. Pascanu, "Relational inductive biases, deep learning, and graph networks," *CoRR*, vol. abs/1806.01261, 2018. [Online]. Available: http://arxiv.org/abs/1806.01261

[48] R. O. Green, M. L. Eastwood, C. M. Sarture, T. G. Chrien, M. Aronsson, B. J. Chippendale, J. A. Faust, B. E. Pavri, C. J. Chovit, M. Solis, M. R. Olah, and O. Williams, "Imaging spectroscopy and the airborne visible/infrared imaging spectrometer (aviris)," *Remote Sensing of Environment*, vol. 65, no. 3, pp. 227–248, 1998. [Online]. Available: https://www.sciencedirect.com/science/article/pii/S0034425798000649

[49] B. Kunkel, F. Blechinger, R. Lutz, R. Doerffer, H. van der Piepen, and M. Schroder, "Rosis (reflective optics system imaging spectrometer) - a candidate instrument for polar platform missions," in *Optoelectronic Technologies for Remote Sensing from Space*, C. S. Bowyer and J. S. Seeley, Eds. SPIE, Apr 1988. [Online]. Available: http://dx.doi.org/10.1117/12.943611

[50] K. P. F.R.S., "On lines and planes of closest fit to systems of points in space," *The London, Edinburgh, and Dublin Philosophical Magazine and Journal of Science*, vol. 2, no. 11, pp. 559–572, 1901. [Online]. Available: https://doi.org/10.1080/14786440109462720

[51] W. B. March, P. Ram, and A. G. Gray, "Fast euclidean minimum spanning tree: algorithm, analysis, and applications," in *Proceedings of the 16th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining, Washington, DC, USA, July 25-28, 2010*, B. Rao, B. Krishnapuram, A. Tomkins, and Q. Yang, Eds. ACM, 2010, pp. 603–612. [Online]. Available: https://doi.org/10.1145/1835804.1835882

[52] J. Ham, Yangchi Chen, M. M. Crawford, and J. Ghosh, "Investigation of the random forest framework for classification of hyperspectral data," *IEEE Transactions on Geoscience and Remote Sensing*, vol. 43, no. 3, pp. 492–501, 2005.

[53] A. Qin, Z. Shang, J. Tian, Y. Wang, T. Zhang, and Y. Y. Tang, "Spectral–spatial graph convolutional networks for semisupervised hyperspectral image classification," *IEEE Geoscience and Remote Sensing Letters*, vol. 16, no. 2, pp. 241–245, 2019.

[54] H. Zeng, Q. Liu, M. Zhang, X. Han, and Y. Wang, "Semi-supervised hyperspectral image classification with graph clustering convolutional networks," 2020.

[55] L. N. Smith, "Cyclical learning rates for training neural networks," in *2017 IEEE Winter Conference on Applications of Computer Vision, WACV 2017, Santa Rosa, CA, USA, March 24-31, 2017*. IEEE Computer Society, 2017, pp. 464–472. [Online]. Available: https://doi.org/10.1109/WACV.2017.58

[56] A. Paszke, S. Gross, F. Massa, A. Lerer, J. Bradbury, G. Chanan, T. Killeen, Z. Lin, N. Gimelshein, L. Antiga, A. Desmaison, A. Kopf, E. Yang, Z. DeVito, M. Raison, A. Tejani, S. Chilamkurthy, B. Steiner, L. Fang, J. Bai, and S. Chintala, "Pytorch: An imperative style, high-performance deep learning library," in *Advances in Neural Information Processing Systems 32*, H. Wallach, H. Larochelle, A. Beygelzimer, F. dÁlché-Buc, E. Fox, and R. Garnett, Eds. Curran Associates, Inc., 2019, pp. 8024–8035. [Online]. Available: http://papers.neurips.cc/paper/9015-pytorch-an-imperative-style-high-performance-deep-learning-library.pdf

[57] L. Van der Maaten and G. Hinton, "Visualizing data using t-sne." *Journal of machine learning research*, vol. 9, no. 11, 2008.

[58] K. Musgrave, S. Belongie, and S.-N. Lim, "A metric learning reality check," in *European Conference on Computer Vision*. Springer, 2020, pp. 681–699.

Aditya Challa received the B.Math.(Hons.) degree in Mathematics from the Indian Statistical Institute - Bangalore, and Masters in Complex Systems from University of Warwick, UK - in 2010, and 2012, respectively. From 2012 to 2014, he worked as a Business Analyst at Tata Consultancy Services, Bangalore. He completed his PhD in computer science from Systems Science and Informatics Unit, Indian Statistical Institute - Bangalore. He is currently Raman PostDoc Fellow at Indian Institute of Science, Bangalore. His current research interests focus on using techniques from Mathematical Morphology in Machine Learning.

Sravan Danda received the B.Math.(Hons.) degree in Mathematics from the Indian Statistical Institute - Bangalore, and the M.Stat. degree in Mathematical Statistics from the Indian Statistical Institute - Kolkata, in 2009, and 2011, respectively. From 2011 to 2013, he worked as a Business Analyst at Genpact - Retail Analytics, Bangalore. He completed his PhD in computer science from Systems Science and Informatics Unit, Indian Statistical Institute - Bangalore under the joint supervision of B.S.Daya Sagar and Laurent Najman. He is currently working as a Assistant Professor at Department of Computer Science and Information Systems, BITS Pilani K K Birla Goa Campus. His current research interests are discrete mathematical morphology and discrete optimization.

B. S. Daya Sagar (M'03-SM'03) is a Full Professor of the Systems Science and Informatics Unit (SSIU) at the Indian Statistical Institute. Sagar received his MSc and Ph.D. degrees in Geoengineering and Remote Sensing from the Faculty of Engineering, Andhra University, Visakhapatnam, India, in 1991 and 1994 respectively. He is also the first Head of the SSIU. Earlier, he worked in the College of Engineering, Andhra University, and Centre for Remote Imaging Sensing and Processing (CRISP), The National University of Singapore in various positions during 1992-2001. He served as Associate Professor and Researcher in the Faculty of Engineering & Technology (FET), Multimedia University, Malaysia, during 2001-2007. Sagar has made significant contributions to the field of geosciences, with special emphasis on the development of spatial algorithms meant for geo-pattern retrieval, analysis, reasoning, modeling, and visualization by using concepts of mathematical morphology and fractal geometry. He has published over 85 papers in journals and has authored and/or guest-edited 11 books and/or special theme issues for journals. He recently authored a book entitled "Mathematical Morphology in Geomorphology and GISci," CRC Press: Boca Raton, 2013, p. 546. He recently co-edited two special issues on "Filtering and Segmentation with Mathematical Morphology" for IEEE Journal of Selected Topics in Signal Processing (v. 6, no. 7, p. 737-886, 2012), and "Applied Earth Observation and Remote Sensing in India" for IEEE Journal of Selected Topics in Applied Earth Observation and Remote Sensing (v. 10, no. 12, p. 5149-5328, 2017). His recent book "Handbook of Mathematical Geosciences", Springer Publishers, p. 942, 2018 reached 750000 downloads. He was elected as a member of the New York Academy of Sciences in 1995, as a Fellow of the Royal Geographical Society in 2000, as a Senior Member of the IEEE Geoscience and Remote Sensing Society in 2003, as a Fellow of the Indian Geophysical Union in 2011. He is also a member of the American Geophysical Union since 2004, and a life member of the International Association for Mathematical Geosciences (IAMG). He delivered the "Curzon & Co - Seshachalam Lecture - 2009" at Sarada Ranganathan Endowment Lectures (SRELS), Bangalore, and the "Frank Harary Endowment Lecture - 2019" at International Conference on Discrete Mathematics - 2019 (ICDM - 2019). He was awarded the 'Dr. Balakrishna Memorial Award' of the Andhra Pradesh Academy of Sciences in 1995, the Krishnan Medal of the Indian Geophysical Union in 2002, the 'Georges Matheron Award - 2011 with Lectureship' of the IAMG, and the Award of IAMG Certificate of Appreciation - 2018. He is the Founding Chairman of the Bangalore Section IEEE GRSS Chapter. He has been recently appointed as an IEEE Geoscience and Remote Sensing Society (GRSS) Distinguished Lecturer (DL) for a two-year period from 2020 to 2022. He is on the Editorial Boards of Computers & Geosciences, Frontiers: Environmental Informatics, and Mathematical Geosciences. He is also the Editor-In-Chief of the Springer Publishers' Encyclopedia of Mathematical Geosciences.

Laurent Najman (SM'17) received the Habilitation à Diriger les Recherches in 2006 from University the University of Marne-la-Vallée, a Ph.D. of applied mathematics from Paris-Dauphine University in 1994 with the highest honor (Félicitations du Jury) and an "Ingénieur" degree from the Ecole des Mines de Paris in 1991. After earning his engineering degree, he worked in the central research laboratories of Thomson-CSF for three years, working on some problems of infrared image segmentation using mathematical morphology. He then joined a start-up company named Animation Science in 1995, as director of research and development. The technology of particle systems for computer graphics and scientific visualization, developed by the company under his technical leadership received several awards, including the "European Information Technology Prize 1997" awarded by the European Commission (Esprit programme) and by the European Council for Applied Science and Engineering and the "Hottest Products of the Year 1996" awarded by the Computer Graphics World journal. In 1998, he joined OCÉ Print Logic Technologies, as senior scientist. He worked there on various problem of image analysis dedicated to scanning and printing. In 2002, he joined the Informatics Department of ESIEE, Paris, where he is professor and a member of the Institut Gaspard Monge, Université Gustave Eiffel. His current research interest is discrete mathematical morphology and discrete optimization.