Journal of Big Data

RESEARCH Open Access

CEU-Net: ensemble semantic segmentation of hyperspectral images using clustering

Nicholas Soucy* and Salimeh Yasaei Sekeh

*Correspondence:
nicholas.soucy@maine.edu
Computer Science, SCIS, University of Maine, Orono, ME, USA

Abstract
Most semantic segmentation approaches of big data hyperspectral images use and require preprocessing steps in the form of patching to accurately classify diversified land cover in remotely sensed images. These approaches use patching to incorporate the rich spatial neighborhood information in images and exploit the simplicity and segmentability of the most common datasets. In contrast, most landmasses in the world consist of overlapping and diffused classes, making neighborhood information weaker than what is seen in common datasets. To combat this common issue and generalize the segmentation models to more complex and diverse hyperspectral datasets, in this work, we propose a novel flagship model: Clustering Ensemble U-Net. Our model uses the ensemble method to combine spectral information extracted from convolutional neural network training on a cluster of landscape pixels. Our model outperforms existing state-of-the-art hyperspectral semantic segmentation methods and gets competitive performance with and without patching when compared to baseline models. We highlight our model's high performance across six popular hyperspectral datasets including Kennedy Space Center, Houston, and Indian Pines, then compare them to current top-performing models.

Keywords: Hyperspectral images, Big data, Semantic segmentation, UNet, Convolutional neural network, Ensemble methods, Clustering, Patching

Introduction

Between climate change, invasive species, and logging enterprises, it is important to know which ground types are where on a large scale. Recently, due to the widespread use of satellite imagery, big data hyperspectral images (HSI) are available to be utilized on a grand scale in ground-type semantic segmentation [1–4].

Ground-type semantic segmentation is a challenging problem in HSI analysis and the remote sensing domain. Ground types in a natural forest environment are overlapping, diverse, similar, and diffused. In contrast, the two most common datasets, Indian pines, and Salinas [5] datasets are small and land-separated. Due to the already segmented nature of farmland and small sample size, the techniques that apply to these datasets do not translate well to large complex natural forests. In contrast, recent advancements in remote sensing imaging have increased spectral resolution exponentially which affects the segmentation models' performance significantly [6]. Therefore, models that exploit the rich spectral information more efficiently can see higher accuracy without a large performance cost.

Patching

Patching is a practical preprocessing technique that often increases the overall test accuracy of a semantic segmentation model by using spatial neighborhood information via overlapping patches [6–10]. Patching is implemented by three approaches: exclusive, majority, and center pixel classification. Examples of these patching techniques are described in Fig. 1. Despite patching improving the performance of segmentation models with particular datasets like Indian Pines and other farmland datasets [10], it is often not as useful in datasets that have diverse overlapping classes as shown in Tables 5, 6, 7, and 8. Due to the limited number of labeled samples and the nature of individual pixel classification, exclusive and majority patching are rarely used in hyperspectral semantic segmentation models because these techniques would further reduce the dataset size. In addition, exclusive and majority patching would not be possible in datasets with diverse

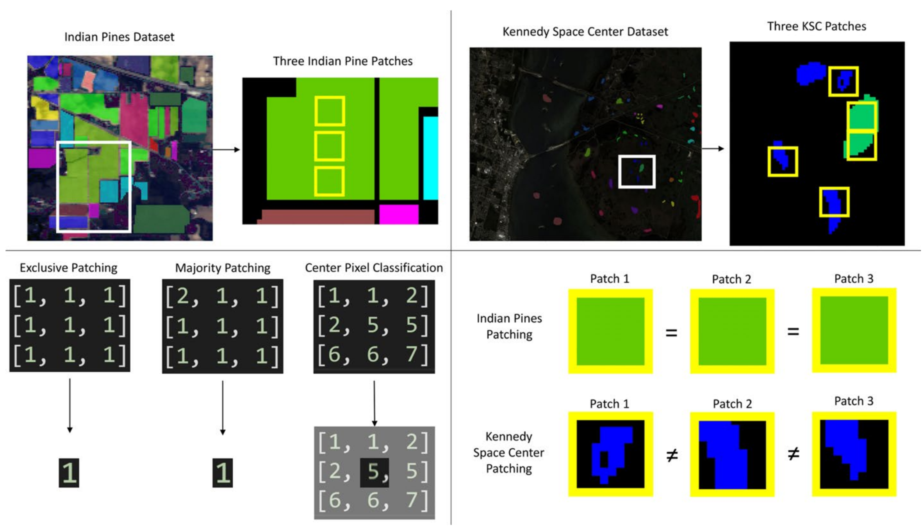

Fig. 1 The top left shows a zoomed-in area of the Indian Pines dataset with three example patches created during the patching process with the same center pixel class. The top right shows a zoomed-in area of the Kennedy Space Center dataset with five example patches. The bottom right shows the three types of patching (1) Exclusive patching takes a patch of n x n pixels and reduces the size of the dataset by downsizing the patch into one pixel if all classes in the patch match, similar to convolution. (2) Similar to exclusive patching, majority patching will downsize the patch into one pixel based on the most popular class in that patch. Both exclusive and majority patching are not used in our experiments and other works due to the already small number of labeled pixels. However, we include them as they could be used in future datasets which have potentially millions of labeled pixels. (3) Center pixel creates a n x n patch for each pixel that contains all the neighborhood information of that pixel as input into the CNN. Then it classifies the center pixel in each patch. Farmland datasets like Indian Pines have better neighborhood information than a diffused forest and therefore benefit more heavily from center pixel classification. Datasets like Kennedy Space Center have less useful neighborhood information and CPC has little impact on overall test accuracy [11] as shown in Tables 5 and 7. The bottom right shows how the three patches with the same center pixel class from the Indian Pines have identical neighbors, this shows the high value of the neighborhood information, therefore patching would be a useful step to improve semantic segmentation accuracy. However, in contrast, the patches in the Kennedy Space Center dataset do not have similar-looking neighbors and therefore the neighborhood information is not as useful

and overlapping classes. Therefore, we focus on center pixel classification in the following sections.

Center pixel classification (CPC) is used in more recent works including one of our baseline models HybridSN [6]. The CPC method is implemented by taking a patch of size n x n for each pixel in the dataset as input to the model to capture the spatial neighbors of the pixel. This exponentially increases the time complexity of training and testing due to each sample being an n x n x w patch, where w is the number of spectral bands, instead of just a single pixel with spectral bands w. This technique can work for many datasets where other techniques like exclusive and majority patching will not because the size of the dataset is not reduced, and datasets with overlapping classes can still be classified. This can lead to a dramatic increase in time complexity with diminishing returns to test accuracy if the neighborhood information is not as useful. Figure 1 bottom left visually demonstrates CPC.

There has been an effort in recent works that focus on neighborhood information instead of spectral information to further increase semantic segmentation accuracy in the popular HSI datasets Indian Pines, Salinas, and Pavia University [8, 9]. For example, due to industrial farming techniques, corn is grown in a single patch, therefore a pixel of corn will be accompanied by other corn pixels. This ensures that the neighbors of each pixel in a single class are all similar. In addition, other datasets like Pavia University and Houston focus on areas of man-made structures that are also easily segmentable. This information can be used in the classification network to much success. However, once most of the land types HSI researchers are interested in have diverse overlapping classes, neighborhood information is weak. The dataset Kennedy Space Center (KSC) is the closest example of this phenomenon and contains classes that are more spectrally similar like different tree types (see Fig. 1). The KSC dataset is often left out of many works due to its small labeled sample size. However, we include KSC to compare and contrast the performance of existing patching-based methods and highlight the weakness in their assumptions.

In this paper, we provide an extensive and systematic discussion on both the benefits and drawbacks of patching and validate our analysis with experimental results.

Feature reduction

The uniqueness of HSI data in remote sensing is the rich spectral features for each pixel. Due to a large number of features, a reduction is often necessary to reduce training runtimes [12, 13]. This is a common practice within HSI semantic segmentation and classification in general.

In HSI Semantic Segmentation, papers [10, 14] focuses on semantic segmentation and/or feature reduction while using neighborhood information. In addition, the top feature reduction methods often use random forest or support vector machine classifiers instead of neural network-based semantic segmentation methods in their research [7]. In this paper, we explore dimensionality reduction techniques that can select pertinent spectral features in the data for later classification in traditional and cutting-edge neural network-based classifiers without neighborhood information; thereby reducing runtime complexity and storage size for classification, while maximizing overall classification accuracy.

In this paper, we experimentally determine that deep neural network feature reduction techniques, like autoencoders, do not beat projection-based feature reduction techniques when using spectral information only.

Semantic segmentation

In remote sensing semantic segmentation, techniques have been focusing on higher dimensional convolutional neural networks (CNNs) to better incorporate neighborhood information. Older techniques used 1D CNNs to use only the spectral information in a given pixel, however, 2D and 3D CNNs have seen greater success with only spectral information included in the training process [15]. Recent works in HSI semantic segmentation have been focusing on using these 2D and 3D CNNs to incorporate neighborhood information in the form of patching [6, 10, 16]. This recent research has mostly ignored the development of spectral-only semantic segmentation, which has notably faster runtimes.

In recent works outside of remote sensing, semantic segmentation has seen great strides in the medical field with the introduction of a novel deep neural network (DNN) architecture called U-Net [17–19]. The idea to use U-Net for semantic segmentation in HSI has to our knowledge been done only once from the paper AeroRIT [10]. The novelty in their U-Net architecture adds complexity via a custom squeeze and excitation block. However, with a high number of features, their custom layer increases the time complexity exponentially. In addition, AeroRIT did not include studies on other datasets and they used neighborhood information in the form of patching as a preprocessing step.

To combat these issues in HSI semantic segmentation, we increase the effectiveness of U-Net with our novel Clustering Ensemble U-Net (CEU-Net) by using an ensemble method to create separate parallel models that are trained in subsets of pixels for better overall classification accuracy.

Our goal in this paper is to develop a semantic segmentation model that is more dataset independent and provides competitive performance versus baselines with and without implementing patching as a preprocessing step.

Ensemble methods

Ensemble learning aims to create a collection of individual classifiers to increase the accuracy of classification/semantic segmentation models. There are three general approaches to ensemble learning: Bagging, Boosting, and Stacking [20, 21].

1 Bagging: Bagging is an ensemble technique that extracts a subset of the dataset to train sub-classifiers. Each sub-classifier and subset are independent of one another and are therefore parallel. The results of the overall bagging method can be determined through a voted majority or a concatenation of the sub-classifier outputs [20].
2 Boosting: Boosting was first developed by the famous algorithm AdaBoost [22]. In boosting the complete dataset is used to train each sub-classifier, then after each iteration, the weights are adjusted for the overall ensemble network to improve classification accuracy [20].

3 Stacking: Stacking is the most unique of the ensemble methods because instead of paralleling the networks like Bagging and Boosting, sub-classifiers are stacked on top of each other in a linear fashion. Therefore, making the output of one sub-classifier the input for another to create a whole ensemble stacking model [21].

For the HSI domain, large data is common creating exponentially increasing running times. In contrast, methods like boosting and stacking can be incredibly costly to running time. Methods like bagging, however, could be implemented to increase accuracy while decreasing runtime. Therefore, in this paper, we aim to create a bagging ensemble method to increase classification accuracy and reduce runtime complexity.

To summarize, our contributions in this paper are,

1 Debuting Clustering Ensemble U-Net, CEU-Net, for HSI semantic segmentation to get more competitive accuracies with and without neighborhood information.
2 Empirical analysis on the common preprocessing technique of patching and focusing more on spectral information instead of neighborhood information to make our model, CEU-Net, more data independent.
3 Experimental analysis on deep neural network-based feature reduction techniques while using only spectral information.

Related works

Current machine learning (ML) based solutions employing neural networks focus on semantic segmentation. Due to the lack of sufficient labeled samples in popular HSI datasets, this is often treated as a pixel classification problem.

Recent works have been using 2D and 3D CNN in both feature reduction and semantic segmentation techniques to implement neighborhood information in addition to spectral information [6, 16]. The works that focus on 2D CNN architectures [23] are older, however, more recent works have focused on 3D CNN architectures or 2D-3D hybrids with greater success [6, 16].

Several works including [6–8] employ a combination of three datasets: Indian Pines, Salinas, and Pavia University due to their well-labeled nature and easy access. We will be focusing on these datasets in addition to Kennedy Space Center, Botswana [5] and Houston [24].

Neighborhood information

The use of neighborhood information is not new in HSI semantic segmentation, almost all of the CNN models for HSI semantic segmentation use neighborhood information in the form of patching as a preprocessing step [6, 23, 25]. Models use neighborhood information due to the nature of the most popular HSI datasets: Indian Pines, Salinas, and Pavia University. These flagship datasets are popular due to their consistent use and the number of labeled pixels. However, the vast majority of HSI images are of dense forest areas with diverse ground types but are not labeled [7, 26].

In [11], the authors discuss patching and its shortcomings by demonstrating how patching only exploits the local spatial information and results in high noise in the data when classes overlap frequently. They propose a full patching network called SPNet with an end-to-end deep learning architecture to do the spectral patching instead of manual analysis. However, SPNet is still a network-based approach that adds significant runtime to semantic segmentation over the common patching method CPC. Further, this work shows how patching is not always the best approach to semantic segmentation. Therefore, we do not include this in this paper, as we focus on improving solely spectral information in our semantic segmentation network for datasets that are more complex like tree species data.

Feature reduction

Certain Bands of light in hyperspectral images might not be as important for classification based on the labeled ground types. Once deep neural network algorithms are quite computationally expensive, reducing the number of input features would increase runtime dramatically. In addition to runtimes, fewer input features often correspond to fewer parameters in the classification model. A model with too many parameters is prone to overfitting issues. Our goal is to improve training runtime and overcome overfitting challenges in semantic segmentation models for HIS by reducing the feature size.

One paper [27] uses feature selection to reduce HSI feature size. The top-performing feature selection method in the paper was a Sequential feature selector (SFS). SFSs work by removing or adding one feature at a time, then performing classification on that feature subset until the feature subset is of the desired size. A drawback of SFSs is that they are supervised and are a greedy search algorithm. Also in [27] different feature selection algorithms were explored like Random Forest and Support Vector Machines (SVMs). However, most of these methods are outperformed by neural network approaches [14]. The optimal feature selector from [27]: SFS, guarantees that we get the optimal feature subset as it goes through each permutation of the feature space, but it is prohibitively computationally expensive. It has been shown that Principal Component Analysis (PCA) can reduce the size and incorporates information on the original features all while being unsupervised and computationally less expensive than SFSs and other feature selectors [28].

To increase the selection of pertinent features, many works now focus on neural network-based feature reduction techniques. Self-Organizing Maps (SOMs) [29] are similar to neural networks as they employ neurons, but their architecture is quite different. Rather than a series of connected layers, SOMs are composed of a single-layer linear 2D grid of neurons. Each node on the grid is connected to the input vector, but not one another. None of these nodes knows the weights of the other nodes. The grid acts as the map that organizes itself at each iteration based on the input data. Each node has its 2D coordinate that allows the calculation of the Euclidean distance between each node. In [30], the authors propose an unsupervised method for the dimensionality reduction of hyperspectral images based on Kohonen's self-organized maps. However, SOMs have dramatically increased runtime when compared to projection-based methods like PCA.

In addition to semantic segmentation, one can learn the feature representations using convolutional networks, for example, in [31] the authors proposed a model called CNNiN that has two parts, a feature learning and a semantic segmentation section that are attached linearly. In the feature learning part, they use a general convolutional network that acts like a U-Net or autoencoder. They have a contracting path that embeds the features into a smaller space, then an expansion path that embeds the desired feature size for classification. Once they have their feature reduction and classification connected in one network, this feature reduction approach is supervised. In addition, the CNNiN method uses neighborhood spacial information in the form of patching and does not get competitive results when compared to other deep learning feature reduction approaches like autoencoders [31, 32].

Recurrent Neural Networks (RNNs) were developed to tackle many recursive machine learning problems including feature reduction. One of the RNN based HSI feature reduction is long short term memory network in which cells solve the vanishing/exploding gradient problem in the backpropagation and can effectively capture contextual information of adjacent data. However, like most deep learning approaches in hyperspectral feature reduction, spacial information is the focus. In [33], they focus on a solely spatial LSTM feature reduction approach. The work in [34] unifies spatial and spectral information by combining spectral LSTM and spatial LSTM networks for feature reduction. However, RNNs appear to be outperformed by other deep learning approaches like convolutional autoencoders [32–34].

A more popular deep learning technique in current literature that uses unsupervised approaches for feature reduction is convolutional autoencoders (CAEs). However, autoencoders are being used more recently to exploit the spatial information of the data rather than the spectral image. 2-Dimensional Convolutional Autoencoders (2D-CAEs) are developed to exploit the spatial information, while 3-Dimensional Convolutional Autoencoders (3D-CAEs) are developed to exploit both the spatial and spectral information available. Current research shows greater semantic segmentation accuracy among 3D-CAE results when incorporating spectral information rather than 2D-CAEs that only use spatial information [14, 32, 35, 36]. The work by [14] introduces an unsupervised spatial-spectral feature learning strategy for HSIs using a 3D-CAE. 3D-CAEs consist of 3D or element-wise operations only, 3D convolution, 3D pooling, and 3D batch normalization, to maximally explore spatial-spectral structure information for feature reduction, rather than spatial only. A companion 3D convolutional decoder network is also designed to reconstruct the input patterns to the 3D-CAE method for full unsupervised learning. Papers [32, 35, 36] create a more complex autoencoder architecture that uses variational autoencoders in their feature reduction structure. Variational autoencoders are similar to autoencoders except their latent space vector is calculated based on the mean and standard deviation of the previous layer. In traditional autoencoders, the latent space vector is simply a layer in the network. Furthermore, the work in [14, 32, 35, 36] rely heavily on spatial for their feature extraction and therefore uses patching as a preprocessing technique. The features are selected due to spacial and spectral instead of solely spectral information. In addition, these papers often use PCA as a preprocessing step before their deep learning feature reduction, making the success of their feature reduction method dependent on PCA. In this paper analysis of autoencoders is provided without patching to determine their effectiveness in selecting pertinent features in the spectral domain only.

In this paper, our main goal is to focus on semantic segmentation without patching, however, feature extraction is a necessary step in the process to improve accuracy and

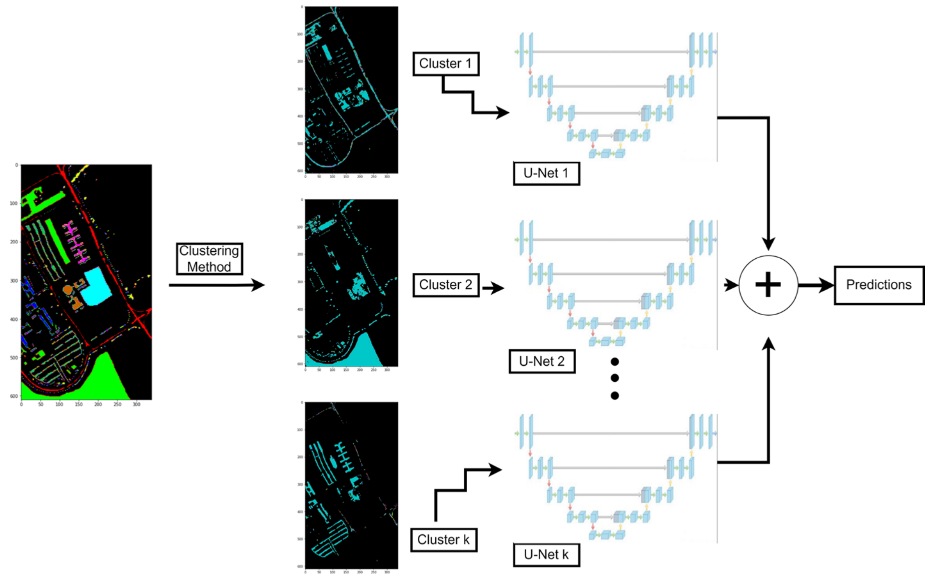

Fig. 2 Holistic overview of our CEU-Net model. We first choose a clustering method and *k* cluster number that is tuned for each dataset based on preliminary experiments shown in Fig. 3. After the unsupervised clustering method separates our training data into *k* clusters, we train the *k* sub-U-Nets for each cluster in parallel. Then we cluster our test data using the same clustering model and send each cluster into their respective sub-U-Nets. Then we concatenate the *k* sub-U-Net predictions on the test data pixels as the overall model accuracy

computational complexity. Therefore, we provide brief experiments on feature reduction techniques to determine the effectiveness of deep learning feature reduction techniques without patching. For this experiment, we choose autoencoders and compare them to PCA to determine if the success of deep learning feature reduction methods in HSI is dependent on spatial information.

Semantic segmentation

HybridSN model: Among several segmentation models [11, 14], to our knowledge the most successful CNN-based model for popular HSI datasets is HybridSN [6]. Instead of using exclusively 3D-CNNs and sacrificing runtime, or using exclusively 2D-CNNs and sacrificing accuracy, they propose a hybrid spectral CNN (HybridSN) for hyperspectral semantic segmentation [6].

HybridSN is a spectral-spatial 3D-CNN model followed by a spatial 2D-CNN. The 3D-CNN facilitates the joint spatial-spectral feature representation from a stack of spectral bands. The 2D-CNN on top of the 3D-CNN further learns more abstract-level spatial representation via neighborhood information. Moreover, the use of hybrid CNNs reduces the number of parameters in the model compared to the use of 3D-CNN alone. This creates a faster semantic segmentation technique while getting state-of-the-art accuracy scores.

However, once HybridSN relies heavily on neighborhood information for its semantic segmentation network, it is unknown if it is a strong spectral classifier. Some other networks, including CEU-Net, work better for classifying solely via spectral information, and/or with smaller patch sizes.

U-Net model: AeroRIT included a U-Net architecture that added complexity via a custom squeeze and excitation block. This is a common practice in the RGB image domain. It works by scaling network responses by modeling channel-wise attention weights, similar to the residual layer in ResNet [19]. The authors use this on large-scale hyperspectral data, however, with the number of channels (bands) that are in hyperspectral compared to RGB, the time complexity increases exponentially. In addition, AeroRIT did not include studies on other datasets, and neighborhood information is used in the form of patching as a preprocessing step.

Ensemble methods

Ensemble methods in HSI semantic segmentation are not new. However, most ensemble methods either do not focus on the bagging ensemble method or do not use CNN architectures.

In the paper, [37] an ensemble boosting method is performed to increase the overall accuracy of a rotation forest (RoF) classifier. However, ensemble method boosting is a costly method that requires multiple training sessions to perform. In addition, the RoF classifier has been shown to be an underperforming classifier compared to CNN techniques.

Deep neural network techniques in HSI semantic segmentation include [38] and [39]. The authors in [39] do a boosting ensemble method called Deep CNN Ensemble where they take the top performing models, HybridSN and ResNet, for their submodels. However, the boosting method increases the running time exponentially because of training multiple models on the same pixels. In addition, they use patching as a preprocessing step to use neighborhood information in their model, which leads to a further increase in running time.

In [38], a bagging ensemble method is used called EECNN, but this method applies a random sampling technique on the feature space to obtain the data subsets for each submodel. However, there are clustering models that cluster data in an unsupervised fashion and outperform random sampling. Moreover, random sampling can lead to too much class disparity between sub-models leading to a reduction in classification accuracy due to the small number of pixels for training. The work in [40] also uses a bagging ensemble method called TCNN-E-ILS. However, they do not have any intelligent way of discriminating what data goes to which network and they have a large number of ensemble classifiers. In this paper, we compare our ensemble method against EECNN and TCNN-E-ILS baselines.

In summary, the ensemble techniques in HSI semantic segmentation use the ensemble method to integrate multiple successful networks/techniques together so they can work together and get higher performance. However, as discussed earlier in this section, boosting is a very costly method that results in higher runtime. The papers that leverage the bagging ensemble technique to reduce computational complexity while increasing classification accuracy do not use an intelligent sampling system, such as clustering, to determine which samples are input to which sub-classifier [38, 40]. Therefore we propose the CEU-Net model to improve the bagging ensemble semantic segmentation technique by focusing on bagging with an intelligent sampling technique for subset division.

Clustering ensemble U-Net (CEU-Net)

One single U-Net is a strong architecture for semantic segmentation, however, without neighborhood information, it is difficult to get competitive accuracy versus models that use it. Our solution to this challenge is the proposed CEU-Net model. In machine learning, the ensemble technique is used to improve the accuracy and stability of learning models, especially for the generalization ability on complex datasets. The overview of our Clustering Ensemble U-Net model is demonstrated in Fig. 2. We propose to separate dissimilar pixels by performing unsupervised clustering on pixels via their spectral signature.

Previous ensemble works, like [38, 40], use a parameter called ensemble size, often denoted as T. In our work, once we use clustering as our intelligent method for determining the number of ensemble networks, we refer to this ensemble size as 'cluster number' and denote it as k.

We formalize our model as follows:

Notation

For an HSI semantic segmentation problem, conditioned on an observed image $\mathbf{x} \in \mathbb{R}^{N\times w}$ with N pixels and w spectral range. The objective is to learn the true posterior distribution $p(\mathbf{y}|\mathbf{x})$, where $\mathbf{y} \in \{1,\ldots m\}^N$, and $1,\ldots,m$ are land type labels. Throughout the paper we use the notations below:

- $\{x_i, y_i\}_{i=1}^N$: Training data where x_i is a pixel and y_i is label, $y_i \in \{1,\ldots,m\}$.
- Classifier F: A function mapping the input space $\mathcal{X}$ to a set of labels $\mathcal{Y}$, i.e. $F: \mathcal{X} \mapsto \mathcal{Y}$. In this paper, this map function is a U-Net model, $F = F^{U-Net}$.
- $\mathcal{L}_{\theta_j}(F_j(\mathbf{x}), y)$: Loss function with parameter set θ_j. In the U-Net model, F_j^{U-Net}, the parameter set θ_j is the network's weight matrix and offsets.
- ω: Ensemble weight vector, $\omega = [\omega_1,\ldots,\omega_k]^T$, where k is the number of clusters.

Methodology

Training a classifier is performed by minimizing a loss function:

$$\mathbf{\Theta} = \arg\min_{\theta} \mathcal{L}_\theta(F(\mathbf{x}), y). \tag{1}$$

Table 1 Information on the more popular datasets in HSI semantic segmentation used in this paper, SP stands for Spectral Bands [5]

Dataset	Sensor	SB	# of Classes	# of Pixels	# of Labeled Pixels	% of Labeled Pixels
Indian Pines	AVIRIS	200	16	21,025	10,249	48.75
Salinas	AVIRIS	204	16	111,104	54,129	48.71
Pavia University	ROSIS	103	9	207,400	42,776	20.62
KSC	AVIRIS	176	13	314,368	5211	0.017
Botswana	NASA EO-1	145	14	377,856	3248	0.009
Houston	ITRES CASI 1500	144	15	664,845	17,270	2.60

In ensemble approach with k classifiers, $F_1(\mathbf{x}), \ldots F_k(\mathbf{x})$ and weight vector $\omega = [\omega_1, \ldots, \omega_k]^T$, where $\omega_k \geq 0$, satisfying $\sum\limits_{j=1}^{k} \omega_j = 1$, we find the optimal parameter set $\boldsymbol{\Theta}$ as follows:

$$\boldsymbol{\Theta} = \arg\min_{\theta,k,\omega} \sum_{j=1}^{k} \omega_j \mathcal{L}_{\theta_j}(F_j(\mathbf{x}), y), \tag{2}$$

where θ_j is the parameter set of classifier F_j. Our proposed CEU-Net architecture extends (2) by utilizing clustering method: Let $C_1(\mathbf{x}), \ldots, C_k(\mathbf{x})$ be the results of partitioning the training data $\{x_i, y_i\}_{i=1}^N$ with label sets $y_{C_1}, \ldots, y_{C_k}$, respectively, into k-clusters. CEU-Net optimizes parameter set θ by

$$\boldsymbol{\Theta} = \arg\min_{\theta,k,\omega} \sum_{j=1}^{k} \omega_j \mathcal{L}_{\theta_j}(F_j(C_j(\mathbf{x})), y_{C_j}). \tag{3}$$

Note that in CEU-Net model F_j is a single U-Net model i.e. $F_j = F_j^{U-Net}$. In this work, we consider k as a hyperparameter and do not learn it under optimization problem 2. The pseudocode of our CEU-Net model is illustrated in Algorithm 1.

Algorithm 1: CEU-Net Algorithm

```
Input: HSI Data {x_i, y_i}_{i=1}^N
Output: Overall Test Accuracy
Set k to be the number of clusters
Set T to be the number of trials
Determine ω_1, ..., ω_k to be the ensemble weights
for t = 1, ..., T do
    Cluster training data {x_i}_{i=1}^N as C_1(x), ..., C_k(x) and store their corresponding label sets
     y_{C_1}, ..., y_{C_k}
    for j = 1, ..., k do
        Train F_j^{U-Net} using data points in jth cluster {C_j(x), y_{C_j}} and loss function ω_j L_{θ_j}
        Store test accuracy AC_{tj}
    end
    Sum AC_{tj} i.e AC_t = Σ_{j=1}^{k} AC_{tj}
end
Compute the average of {AC_1, AC_2, ..., AC_T} i.e.
                    (1/T) Σ_{t=1}^{T} AC_t → AC.
Report AC
```

In the practical implementation of Algorithm 1 the value of weights $\omega = [\omega_1, \omega_2, \ldots, \omega_k]^T$ is determined experimentally. We then take the training data and use an unsupervised clustering method that separates the pixels into k clusters. Both k and the clustering method will be tuned for each dataset. We then send the training pixels from each cluster into k separate sub-U-Nets for separate training in a supervised fashion with categorical cross entropy as the loss function. This way, each sub-U-Net becomes an expert in its given cluster and is trained in parallel with its corresponding pixel cluster. After each sub-U-Net is trained, we use the same clustering method to cluster the testing data into k clusters. Then we predict the labels for each cluster using the corresponding trained sub-U-Net for each cluster. Finally, the sub-U-Nets' predicted

Table 2 The layer-wise summary of the single U-Net and the sub-classifiers used in the CEU-Net architecture. *n* x *n* is the patch size, *w* is the input spectral dimension, and *m* is the class size for the given dataset

Layer #	Layer Name	Layer Details	Inputs	Output Shape
0	Input Layer		N/A	(n,n,*w*)
1A	Conv2D_1	Kernel = (3,3), strides = (1,1)	0	(n,n,64)
1B	BatchNormalization_1		1A	(n,n,64)
1C	LeakyReLU_1		1B	(n,n,64)
1D	Dropout_1	0.2 Dropped	1C	(n,n,64)
2A	Conv2D_2	Kernel = (3,3), strides = (1,1)	1D	(n,n,128)
2B	BatchNormalization_2		2A	(n,n,128)
2C	LeakyReLU_2		2B	(n,n,128)
2D	Dropout_2	0.2 Dropped	2C	(n,n,128)
3A	Conv2D_3	Kernel = (3,3), strides = (1,1)	2D	(n,n,256)
3B	BatchNormalization_3		3A	(n,n,256)
3C	LeakyReLU_3		3B	(n,n,256)
3D	Dropout_3	0.2 Dropped	3C	(n,n,256)
4A	Conv2DTranspose_1	Kernel = (3,3), strides = (1,1)	3D	(n,n,256)
4B	BatchNormalization_4		4A	(n,n,256)
4C	LeakyReLU_4		4B	(n,n,256)
4D	Dropout_4	0.2 Dropped	4C	(n,n,256)
4E	Concatenate_1	2D + 4D	2D,4D	(n,n,384)
5A	Conv2DTranspose_2	Kernel = (3,3), strides = (1,1)	4E	(n,n,128)
5B	BatchNormalization_5		5A	(n,n,128)
5C	LeakyReLU_5		5B	(n,n,128)
5D	Dropout_5	0.2 Dropped	5C	(n,n,128)
5E	Concatenate_2	1D + 5D	1D, 5D	(n,n,192)
6A	Conv2DTranspose_3	Kernel = (3,3), strides = (n,n)	5E	(1,1,*m*)
6B	Reshape		6A	(1,*m*)
6C	PixelSoftmax		6B	(1,*m*)

labels are concatenated and we compare them to the ground truths for overall testing accuracy. Each sub-U-Net is identical to the single U-Net architecture using the configuration presented in Table 2.

Experimental results

The experimental results section is divided into two main parts, the first discusses the performance of CEU-Net in the context of the state-of-the-art semantic segmentation algorithm and illustrates key insights into the expected behavior of CEU-Net. The second part emphasizes the efficiency improvement of CEU-Net and hyper-parameter tuning.

We briefly outline the datasets, feature reduction, U-Net architecture, configuration, clustering methods, and metrics used across our experiments.

Datasets

In this experiment, we choose six datasets: Indian Pines, Salinas, Pavia University, Kennedy Space Center, Botswana, and Houston [5]. HSI data is infamously difficult to label due to the professional and time requirements necessary to label ground-types [41]. These well-known HSI datasets are well labeled and will provide good testing data for our semantic segmentation techniques.

These datasets while used profusely in the ML hyperspectral community, have quite a few flaws.

1 *Easily Segmentable*: Indian Pines and Salinas are farmland datasets while Pavia University and Houston are man-made structures. These land areas are quite easily separable spatially. This means grass is often next to other grass and tar is next to other tar etc. This makes training an easy task in just the pixel domain.
2 *Not Representative of Most Land Areas*: A vast majority of land in the world is forest regions and most hyperspectral remote sensing is done in these areas [26]. Therefore, the existing semantic segmentation models for HSI in remote sensing are not transferable to other landscapes due to the unavailability of labeled samples. Kennedy Space Center is the closest dataset to represent these more difficult datasets, however, once it is in a desert biome, the ground-truth labels are still easily spatially clustered.
3 *Small Amount of Labeled Samples*: Due to the difficulty of labeling HSI data, the amount of pixels in a dataset is often not a good description of its entire size. All the datasets we use here have a labeled pixel percentage under 50% as shown in Table 1. This could lead to over-fitting when presented with complex architectures.

It is clear why the first three datasets are picked more often, they have a larger amount of labeled pixels. All of these datasets have large, separated regions for their ground truths and not more pixel-specific classes like tree species, making neighborhood information a smart choice to increase semantic segmentation accuracy for these datasets. A new dataset called AeroRIT [10] is introduced that has more labeled pixels, however, because it (1) does not have diverse classes, (2) has a small number of classes, (3) is similar in scope and classes to the Houston dataset, and (4) is not practical for forest remote sensing, we did not include it in our study.

Feature reduction

In this paper, we use PCA as our baseline feature reduction technique to compare our other two customized CNN-based techniques. We apply autoencoder models using customized 2D and 3D convolutional autoencoder architectures for feature dimensionality reduction.

Many related works have shown that 2D and 3D CNN structures have had success when compared to traditional feature reduction techniques [6, 7]. Therefore, to start off our first autoencoder architecture we decided to use a 2D convolutional autoencoder. This way, if the accuracy produced by the 2D autoencoder is sufficient, we do not have to apply a customized 3D autoencoder which would be more computationally expensive to train.

To customize both the 2D and 3D convolutional autoencoders, we vary the kernel/pooling size and strides to determine the most efficient feature size for each dataset to train our classifiers. However, the operations, input shapes, and activation functions are kept constant. The autoencoders have the exact same layers except each 2D layer is its 3D equivalent in the 3D autoencoder. At the end of the decoder network, we have an upscaled image of the same size as the original to compare to for unsupervised learning. The loss function used is Mean Squared Error. A layer-wise summary of both networks can be found in Table 11.

For our feature reduction experiments, we chose to reduce the features to 40, 35, 30, 25, and 20 for each dataset. Once feature reduction reduces the computational complexity of network training, any feature size over 40 increases runtime while diminishing returns to classification accuracy. Any feature size under 20 will result in too little information for the models to distinguish different classes within the data. Therefore, feature sizes between 40 and 20 are explored. More experimental detail on feature reduction is provided in "Performance comparison" section.

Single U-Net architecture

For our main clustering ensemble model contribution, we develop a CNN-based model for semantic segmentation that is lightweight to deal with a large number of features per sample. Based on this strategy, we propose a custom CNN that focuses on the rich spectral data available for each pixel, therefore customized CNN under a U-Net backbone was our first choice among various architectures.

A general U-Net consists of two parts: a contracting path (left side of 'U') and an expansive path (right side of 'U'). U-Net's novelty is in supplementing a usual contracting network by successive layers where the typical pooling layers are replaced by upsampling. This technique increases the resolution for each pixel. The successive convolutional layer can then learn to assemble a precise output based on this information. In addition, U-Net has a large number of feature channels in the upsampling part, which allow the network to propagate context information to higher-resolution layers. This makes the expansive path symmetric to the contracting path yielding the famous 'U'-shaped architecture.

Table 3 Experimental results of our feature reduction techniques between PCA, 2DCAE, and 3DCAE. An explanation of the metrics can be found in "Experiment configurations" section

Methods	OA	AA	Kappa	OA	AA	Kappa
	IP (k = 2)			Salinas (k = 3)		
PCA	**90.01 ± 0.1**	**90.52 ± 0.2**	**88.67 ± 0.1**	**96.44 ± 0.1**	**98.36 ± 0.1**	**96.34 ± 0.1**
2D-CAE	65.39 ± 0.2	51.39 ± 0.2	60.04 ± 0.2	85.97 ± 0.2	91.65 ± 0.2	84.36 ± 0.2
3D-CAE	70.50 ± 0.2	55.84 ± 0.3	61.21 ± 0.2	87.02 ± 0.2	91.71 ± 0.2	85.94 ± 0.2
	KSC (k = 2)			Botswana (k = 3)		
PCA	**95.25 ± 0.1**	**93.05 ± 0.1**	**94.98 ± 0.1**	**96.43 ± 0.2**	**97.1 ± 0.2**	**96.13 ± 0.2**
2D-CAE	90.02 ± 0.2	84.39 ± 0.2	88.87 ± 0.2	91.26 ± 0.2	91.84 ± 0.2	90.52 ± 0.2
3D-CAE	91.10 ± 0.2	85.62 ± 0.2	89.46 ± 0.2	93.12 ± 0.2	93.44 ± 0.2	91.45 ± 0.2
	PU (k=2)			Houston (k=2)		
PCA	**96.18 ± 0.1**	**95.10 ± 0.1**	**95.00 ± 0.1**	**98.49 ± 0.1**	**98.38 ± 0.1**	**98.36 ± 0.1**
2D-CAE	80.94 ± 0.2	75.85 ± 0.2	73.85 ± 0.1	51.57 ± 0.4	54.00 ± 0.4	47.76 ± 0.4
3D-CAE	81.47 ± 0.2	78.45 ± 0.1	74.99 ± 0.2	74.35 ± 0.3	73.81 ± 0.4	72.25 ± 0.3

Highest performing values are highlighted in bold

Table 4 Experimental results exploring the favorable feature size without patching for each dataset using the spectral feature reduction method PCA. An explanation of the metrics can be found in "Experiment configurations" section

Methods	OA	AA	Kappa	OA	AA	Kappa
	IP (k = 2)			Salinas (k = 3)		
PCA 45	89.91 ± 0.1	88.66 ± 0.1	87.99 ± 0.1	96.06 ± 0.1	98.25 ± 0.1	95.61 ± 0.1
PCA 40	**90.2 ± 0.1**	**90.99 ± 0.2**	88.76 ± 0.1	96.40 ± 0.1	98.34 ± 0.1	96.12 ± 0.1
PCA 35	90.15 ± 0.1	90.79 ± 0.2	**88.79 ± 0.1**	96.36 ± 0.1	98.32 ± 0.1	95.96 ± 0.1
PCA 30	90.01 ± 0.1	90.52 ± 0.2	88.67 ± 0.1	**96.44 ± 0.1**	**98.36 ± 0.1**	**96.34 ± 0.1**
PCA 25	89.01 ± 0.1	87.77 ± 0.2	87.56 ± 0.1	96.33 ± 0.1	98.32 ± 0.1	95.97 ± 0.1
PCA 20	88.00 ± 0.1	86.89 ± 0.2	86.41 ± 0.1	96.37 ± 0.1	98.24 ± 0.1	95.95 ± 0.1
PCA 15	82.18 ± 0.2	79.59 ± 0.2	79.69 ± 0.2	96.31 ± 0.2	98.11 ± 0.2	95.81 ± 0.2
	KSC (k = 2)			Botswana (k = 3)		
PCA 45	88.49 ± 0.1	82.49 ± 0.1	87.14 ± 0.1	95.69 ± 0.1	95.82 ± 0.1	95.33 ± 0.1
PCA 40	**96.21 ± 0.1**	**95.01 ± 0.1**	**96.11 ± 0.1**	96.43 ± 0.2	96.89 ± 0.2	96.01 ± 0.2
PCA 35	95.91 ± 0.1	94.12 ± 0.1	95.75 ± 0.1	96.34 ± 0.2	97.04 ± 0.2	96.00 ± 0.2
PCA 30	95.25 ± 0.1	93.05 ± 0.1	94.98 ± 0.1	**96.43 ± 0.2**	**97.10 ± 0.2**	**96.13 ± 0.2**
PCA 25	93.75 ± 0.1	90.11 ± 0.1	93.01 ± 0.1	96.40 ± 0.2	96.84 ± 0.2	95.89 ± 0.2
PCA 20	92.40 ± 0.1	87.58 ± 0.1	91.69 ± 0.1	96.41 ± 0.2	96.82 ± 0.2	95.78 ± 0.2
PCA 15	82.87 ± 0.2	76.57 ± 0.2	80.85 ± 0.2	96.32 ± 0.2	96.55 ± 0.2	96.77 ± 0.2
	PU (k = 2)			Houston (k = 2)		
PCA 45	96.01 ± 0.1	94.98 ± 0.1	94.88 ± 0.1	96.56 ± 0.1	97.00 ± 0.1	96.50 ± 0.1
PCA 40	95.88 ± 0.1	94.16 ± 0.1	94.54 ± 0.1	96.55 ± 0.1	96.89 ± 0.1	96.01 ± 0.1
PCA 35	96.13 ± 0.1	94.64 ± 0.1	94.87 ± 0.1	97.01 ± 0.1	96.99 ± 0.1	96.84 ± 0.1
PCA 30	**96.18 ± 0.1**	**95.1 ± 0.1**	**95.00 ± 0.1**	**96.43 ± 0.1**	**97.1 ± 0.1**	**96.89 ± 0.1**
PCA 25	96.01 ± 0.1	94.56 ± 0.1	94.78 ± 0.1	96.4 ± 0.1	96.84 ± 0.1	95.89 ± 0.1
PCA 20	95.8 ± 0.1	94.57 ± 0.1	94.58 ± 0.1	96.41 ± 0.1	96.82 ± 0.1	95.78 ± 0.1
PCA 15	95.89 ± 0.1	94.51 ± 0.2	94.47 ± 0.1	96.01 ± 0.1	96.44 ± 0.1	95.79 ± 0.1

Highest performing values are highlighted in bold

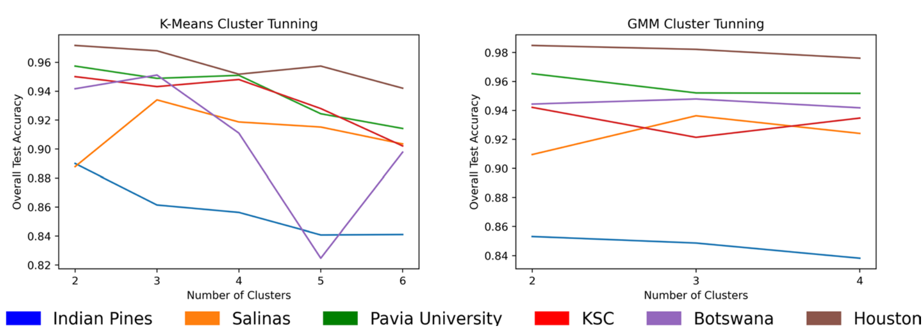

Fig. 3 Results of our cluster tuning. We explored both K-Means and Gaussian Mixture Models (GMM) for our clustering methods along with a wide spread of cluster numbers. Any cluster larger than 4 for GMM or 6 for K-Means resulted in clusters with too little data for semantic segmentation in specific sub-U-Nets. The number of clusters cannot equal 1, as this would result in the entire dataset being the only cluster and therefore an ensemble CEU-Net approach would not be possible. The relatively small number of clusters in each dataset shows how easily segmentable these datasets are

For our specific U-Net architecture, the contracting path consists of three 3x3 2D convolutions followed by a leaky rectified linear unit (LReLU) and then a Dropout layer with a 20% rate to prevent over-fitting. For our expansive path, we have three 3x3 2D convolution transposes with the last layer outputting a logit array of size equal to the number of classes in the dataset. We then do a softmax layer for calculating semantic segmentation accuracy. This architecture is based on the original U-Net [19, 42].

Table 2 shows an example of the layer-wise summary of our single U-Net. When patching is applied, the patch size replaces the n x n output shapes in each layer where n is the patch size. When patching is not used n x n becomes 1 x 1. D is the input spectral dimension size and m is the output class size for any dataset. Therefore our single U-Net and CEU-Net can work with various patching techniques and datasets with ease, increasing the flexibility and applicability of our algorithm.

Experiment configurations

In this section, we implement feature reduction and semantic segmentation techniques.

Our first experiment is to determine the empirically optimal feature reduction technique for our datasets. We start by reducing our feature size to 30 using PCA, 2D CAE, and 3D CAE to determine the empirically optimal feature reduction method for spectral-only data for each dataset. The classifier used is the same for each feature reduction method: CEU-Net no patching with a 75%/25% training/testing split. Afterward, we take the best-performing feature reduction method and determine the empirically optimal feature size for each dataset. For this experiment, we reduce the spectral feature space to 40, 35, 30, 25, and 20. We do this for each of our feature reduction methods, PCA, 2D CAE, and 3D CAE.

For our semantic segmentation validation, we perform a 5-fold cross-validation by shuffling the dataset randomly and splitting the dataset into five different training and testing sets using a test size of 25%. This shows the stability of our results by reporting an average of the test metric along with the standard deviation. In addition, it shows that we did not choose a training and testing set to strategically give us the best results. Therefore, we gain larger stability in our test results when compared to a single data point and show that our test metrics are consistent among other random training/testing splits [43].

To show that the results of each feature reduction method and semantic segmentation method are statistically significant with respect to each other, we perform One-Way Analysis of Variance (ANOVA) tests. We use the accepted $\alpha = 0.05$ value for the p-value null hypothesis rejection criteria [44], which indicates there is a 5% risk of concluding that a difference exists when there is no actual difference. Therefore, during our ANOVA testing, if the p-value: P is $\leq \alpha$, then our results are statistically significant between different methods. For each ANOVA calculation, we use the full results of the 5 trials from the 5-fold cross-validation from each experiment.

For each dataset: 2D CAE trains for 100 epochs, 3D CAE trains for 150 epochs, HybridSN, Single U-Net, and AeroRIT U-Net train for 150 epochs then CEU-Net trains for 200 epochs for each sub-U-Net. Each semantic segmentation technique uses categorical cross entropy for the loss function and has a learning rate of 0.0001. All tests are run via Google Colab using Nvidia Tesla K80 GPU with 24GB of memory.

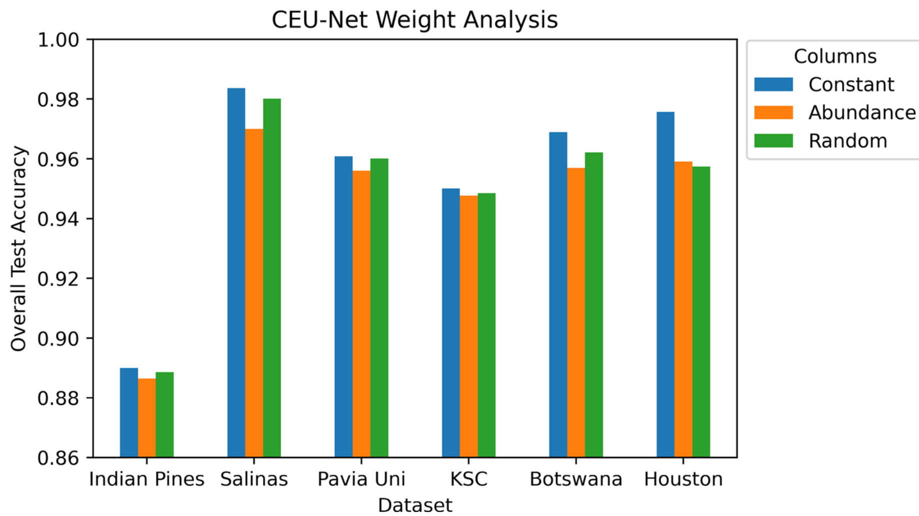

Fig. 4 Results of the loss weight $\omega = [\omega_1, \ldots, \omega_k]^T$ tuning for our ensemble CEU-Net method. Three different weight types are explored: 1) Constant weights in each sub-model, 2) Weights equaling the abundance of data given to each sub-model, and 3) Random weights assigned. All weights have to sum to equal 1 as explained in Eq. 2. All tests were run with 5-fold cross-validation. We observed that the constant weights outperform other methods, therefore, we use constant weights in all of our CEU-Net experiments

Table 5 Test metric results for each semantic segmentation method for each dataset without patching. An explanation of the metrics can be found in "Experiment configurations" section

Methods	OA	AA	Kappa	OA	AA	Kappa
	IP (k = 2)			Salinas (k = 3)		
HybridSN	86.99 ± 0.1	86.5 ± 0.2	85.69 ± 0.1	96.74 ± 0.1	97.62 ± 0.1	96.37 ± 0.1
AeroRIT	74.44 ± 0.2	65.5 ± 0.3	70.97 ± 0.2	95.06 ± 0.1	97.7 ± 0.1	94.49 ± 0.1
U-Net	87.25 ± 0.1	87.8 ± 0.2	85.64 ± 0.1	**96.78 ± 0.1**	98.24 ± 0.1	**96.37 ± 0.1**
CEU-Net	**90.01 ± 0.1**	**90.52 ± 0.2**	**88.67 ± 0.1**	96.44 ± 0.1	**98.36 ± 0.1**	96.34 ± 0.1
	KSC (k = 2)			Botswana (k = 3)		
HybridSN	94.85 ± 0.1	91.9 ± 0.1	94.27 ± 0.1	96.29 ± 0.2	96.5 ± 0.2	96.92 ± 0.2
AeroRIT	93.94 ± 0.1	91.15 ± 0.2	93.23 ± 0.1	89.7 ± 0.3	89.95 ± 0.3	88.77 ± 0.2
U-Net	95.00 ± 0.1	92.73 ± 0.1	94.93 ± 0.1	**96.77 ± 0.2**	96.45 ± 0.2	**96.94 ± 0.2**
CEU-Net	**95.25 ± 0.1**	**93.05 ± 0.1**	**94.98 ± 0.1**	96.43 ± 0.2	**97.1 ± 0.2**	96.13 ± 0.2
	PU (k = 2)			Houston (k = 2)		
HybridSN	95.99 ± 0.1	94.59 ± 0.1	94.71 ± 0.1	98.3 ± 0.1	98.2 ± 0.1	98.17 ± 0.1
AeroRIT	93.89 ± 0.1	91.56 ± 0.2	91.9 ± 0.2	93.98 ± 0.1	93.99 ± 0.2	93.49 ± 0.1
U-Net	96.02 ± 0.1	94.95 ± 0.1	94.86 ± 0.1	98.38 ± 0.1	98.21 ± 0.1	98.25 ± 0.1
CEU-Net	**96.18 ± 0.1**	**95.1 ± 0.1**	**95.00 ± 0.1**	**98.49 ± 0.1**	**98.38 ± 0.1**	**98.36 ± 0.1**

Highest performing values are highlighted in bold

To determine the effectiveness of all techniques, three evaluation metrics are used: Overall Accuracy (OA), Average Accuracy (AA), and Kappa Coefficient (Kappa) [45].

1 Overall Accuracy: OA represents the total correctly classified samples out of the testing data.
2 Average Accuracy: AA represents the average of the class-wise classification accuracies.

3 Kappa Coefficient: Kappa is a statistical metric that represents the mutual information between the ground-truth map and the classification map [45].

Clustering methods

There were two clustering methods explored, K-Means++ [46] and Gaussian Mixture Models (GMM) [47, 48] clustering. K-Means uses the mean to calculate the centroid for each cluster, while GMM takes into account the variance of the data in addition to the mean. Therefore, based on the distribution for each dataset, one method may work better than the other.

K-Means++ and GMM were chosen due to their unsupervised, simple, fast, and historically good performance. Once we wish to decrease the overall time that CEU-Net takes to train, and get competitive accuracy, K-Means++ and GMM were good first choices for our clustering algorithms. In addition to the clustering method, the number of clusters will also be varied in the preliminary experiment to determine the most effective number of clusters for each dataset.

Performance comparison

Here, our main goal is to compare the performance of CEU-Net on the original testing set to mini-batch SGD training and highlight how we can improve performance without using neighborhood information. We first briefly demonstrate that among various feature reduction approaches PCA provides the best performance.

Table 6 Test metric results for each semantic segmentation method for the Indian Pines and Salinas datasets while employing patching for different patch sizes. For Deep CNN Ensemble T = 10, for EECNN and TCNN-E-ILS T = 20. An explanation of the metrics can be found in "Experiment configurations" section

Patch	Methods	OA	AA	Kappa	OA	AA	Kappa
		IP (k=2)			Salinas (k=3)		
	HybridSN	**98.55 ± 0.1**	**98.19 ± 0.1**	**98.35 ± 0.1**	99.80 ± 0.05	99.76 ± 0.05	99.78 ± 0.05
5 x 5	U-Net	95.10 ± 0.1	94.85 ± 0.1	94.82 ± 0.1	**99.81 ± 0.05**	**99.80 ± 0.05**	**99.87 ± 0.05**
	CEU-Net	94.50 ± 0.1	93.50 ± 0.1	93.75 ± 0.1	98.78 ± 0.1	99.31 ± 0.05	98.64 ± 0.05
	HybridSN	97.03 ± 0.1	95.57 ± 0.1	96.62 ± 0.1	99.80 ± 0.05	99.74 ± 0.05	99.73 ± 0.05
10 x 10	U-Net	**97.35 ± 0.1**	**96.64 ± 0.1**	**96.99 ± 0.1**	99.78 ± 0.05	99.76 ± 0.05	99.75 ± 0.05
	CEU-Net	96.34 ± 0.1	95.29 ± 0.1	95.37 ± 0.1	**99.85 ± 0.05**	**99.78 ± 0.05**	**99.78 ± 0.05**
	HybridSN	97.23 ± 0.1	94.08 ± 0.1	95.84 ± 0.1	99.79 ± 0.05	99.75 ± 0.05	99.77 ± 0.05
15 x 15	U-Net	95.70 ± 0.1	89.78 ± 0.1	95.10 ± 0.1	99.80 ± 0.05	99.76 ± 0.05	99.76 ± 0.05
	CEU-Net	**97.36 ± 0.1**	**94.68 ± 0.1**	**95.98 ± 0.1**	**99.86 ± 0.05**	**99.77 ± 0.05**	**99.77 ± 0.05**
	EECNN	N/A	N/A	N/A	98.48 ± 0.03	98.37 ± 0.03	97.58 ± 0.04
25 x 25	EECNN	97.57 ± 0.07	96.23 ± 0.02	97.23 ± 0.08	N/A	N/A	N/A
27 x 27	CNN Ensemble	92.54	N/A	90.94	96.05	N/A	95.93
33 x 33	TCNN-E-ILS	91.88 ± 1.13	77.37 ± 4.04	90.28 ± 1.34	N/A	N/A	N/A

Highest performing values are highlighted in bold

Table 7 Test metric results for each semantic segmentation method for the Pavia University and Kennedy Space Center datasets while employing patching for different patch sizes. For Deep CNN Ensemble T = 10, for EECNN and TCNN-E-ILS T = 20. An explanation of the metrics can be found in "Experiment configurations" section

Patch	Methods	OA	AA	Kappa	OA	AA	Kappa
		PU (k=2)			KSC (k=2)		
	HybridSN	99.59 ± 0.05	99.37 ± 0.05	99.50 ± 0.05	96.62 ± 0.1	96.10 ± 0.1	96.46 ± 0.1
5 x 5	U-Net	**99.60 ± 0.05**	**99.40 ± 0.05**	**99.52 ± 0.1**	96.90 ± 0.1	96.30 ± 0.1	96.52 ± 0.1
	CEU-Net	98.61 ± 0.1	97.9 ± 0.1	98.00 ± 0.1	**96.97 ± 0.1**	**96.29 ± 0.1**	**96.54 ± 0.1**
	HybridSN	99.58 ± 0.05	**99.56 ± 0.05**	99.38 ± 0.05	97.31 ± 0.1	96.54 ± 0.1	97.00 ± 0.1
10 x 10	U-Net	99.54 ± 0.05	99.10 ± 0.05	**99.40 ± 0.05**	95.24 ± 0.1	94.72 ± 0.1	94.69 ± 0.1
	CEU-Net	**99.59 ± 0.05**	99.12 ± 0.05	98.00 ± 0.1	**99.10 ± 0.1**	**98.57 ± 0.1**	**98.97 ± 0.1**
	HybridSN	99.57 ± 0.05	99.45 ± 0.05	**99.47 ± 0.05**	95.32 ± 0.1	93.81 ± 0.1	94.78 ± 0.1
15 x 15	U-Net	99.34 ± 0.05	**99.50 ± 0.05**	99.10 ± 0.05	92.1 ± 0.1	90.94 ± 0.1	91.19 ± 0.1
	CEU-Net	**99.59 ± 0.05**	99.12 ± 0.05	98.00 ± 0.1	**97.7 ± 0.1**	**96.57 ± 0.1**	**97.43 ± 0.1**
	EECNN	99.34 ± 0.06	99.30 ± 0.04	99.27 ± 0.07	N/A	N/A	N/A
25 x 25	EECNN	N/A	N/A	N/A	N/A	N/A	N/A
27 x 27	CNN Ensemble	94.98	N/A	92.04	N/A	N/A	N/A
33 x 33	TCNN-E-ILS	89.62	85.14	86.51	**99.27 ± 0.36**	**98.87 ± 0.64**	**99.19 ± 0.41**

Highest performing values are highlighted in bold

Table 8 Test metric results for each semantic segmentation method for the Botswana and Houston datasets while employing patching for different patch sizes. For TCNN-E-ILS T = 20. An explanation of the metrics can be found in "Experiment configurations" section

Patch	Methods	OA	AA	Kappa	OA	AA	Kappa
		Botswana (k=3)			Houston (k=2)		
	HybridSN	**99.88 ± 0.15**	**99.89 ± 0.15**	**99.87 ± 0.15**	98.28 ± 0.1	98.18 ± 0.1	98.19 ± 0.1
5 x 5	U-Net	98.65 ± 0.15	98.97 ± 0.15	98.65 ± 0.15	**98.30 ± 0.1**	**98.34 ± 0.1**	**98.21 ± 0.1**
	CEU-Net	97.54 ± 0.15	97.81 ± 0.15	97.34 ± 0.15	94.47 ± 0.1	95.02 ± 0.1	94.02 ± 0.1
	HybridSN	**98.89 ± 0.15**	**98.97 ± 0.15**	**98.79 ± 0.15**	97.5 ± 0.1	97.30 ± 0.1	96.30 ± 0.1
10 x 10	U-Net	97.92 ± 0.15	98.02 ± 0.15	97.97 ± 0.15	97.78 ± 0.1	**97.69 ± 0.1**	**96.52 ± 0.1**
	CEU-Net	96.57 ± 0.15	96.45 ± 0.15	96.44 ± 0.15	**97.92 ± 0.1**	96.54 ± 0.1	96.34 ± 0.1
	HybridSN	**97.41 ± 0.15**	**97.55 ± 0.15**	**97.19 ± 0.15**	**98.05 ± 0.1**	**98.11 ± 0.1**	**97.89 ± 0.1**
15 x 15	U-Net	90.88 ± 0.15	91.34 ± 0.15	90.1 ± 0.15	94.95 ± 0.1	95.51 ± 0.1	94.54 ± 0.1
	CEU-Net	91.38 ± 0.15	91.55 ± 0.2	90.66 ± 0.2	93.94 ± 0.1	94.01 ± 0.1	92.97 ± 0.1
33 x 33	TCNN-E-ILS	N/A	N/A	N/A	88.33 ± 0.68	88.10 ± 0.86	87.39 ± 0.74

Highest performing values are highlighted in bold

Feature reduction

The results in Table 3 show that PCA is a superior feature reduction technique versus 2D and 3D CAE. PCA has higher testing metrics for all datasets. In addition, the feature reduction runtime for 2D and 3D CAE is high as they require training of neural networks, while PCA is an unsupervised mathematical technique. Results for each feature reduction technique can be seen in Table 3. All reported results are determined by using our tuned CEU-Net for the classifier. Statistical significance testing between each of the feature reduction methods from the data shown in Table 3 are shown in Appendix

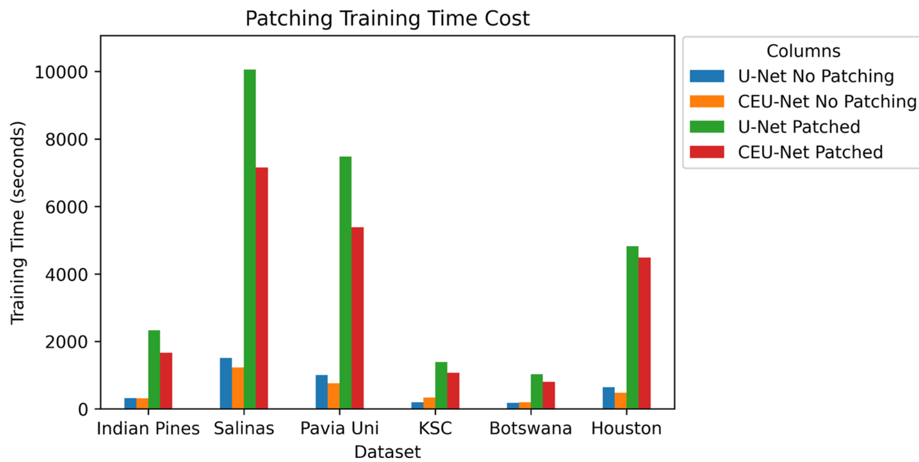

Fig. 5 Runtime analysis for single U-Net and CEU-Net for each dataset. The empirically optimal feature reduction technique was used before each semantic segmentation: PCA. Bands were reduced to 30. This figure shows the dramatic difference in runtime when employing patching. Patching greatly increases the runtime of semantic segmentation models. CPC was used with a patch size of 10 x 10

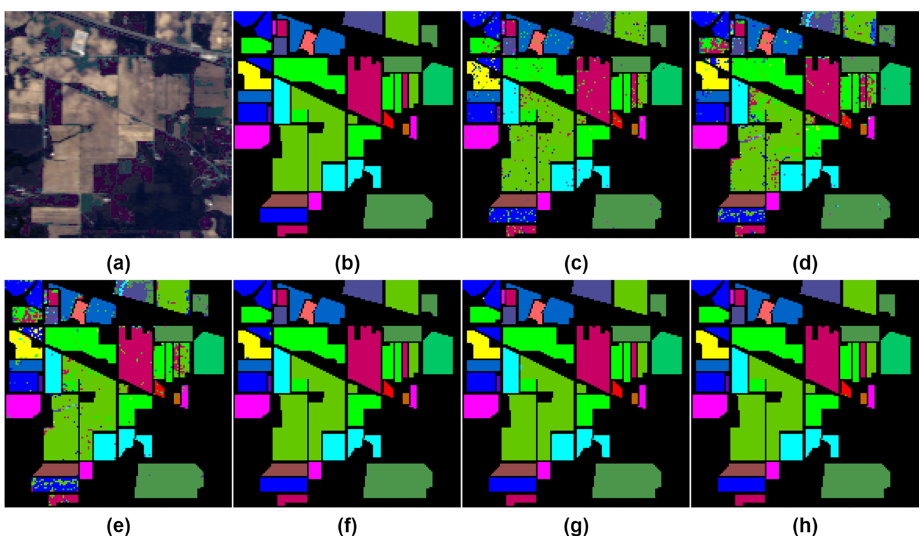

Fig. 6 The classification map for the Indian Pines dataset. **a** RBG Image **b** Ground Truth **c**–**e** Predicted classification maps for HybridSN, single U-Net, and CEU-Net with no patching respectively. **f**–**h** Predicted classification maps for HybridSN, single U-Net, and CEU-Net with patching with a patch size of 10x10 pixels

Table 9. Once all p-values for each test are well below $\alpha = 0.05$, we can conclude that all results are statistically significant.

Feature size

The results in Table 4 show that PCA 30 is the favorable feature size for most datasets. For the two datasets that have an increased accuracy at a feature size of 40 (Indian Pines and Kennedy Space Center), 30 seems to be a tipping point where there is an exponential decline in performance starting at 30. Therefore, once we keep runtime in mind due to computational complexity and performance trade-off, 30 features are used for all data-sets for our classification data. Results for each feature reduction size can be seen in

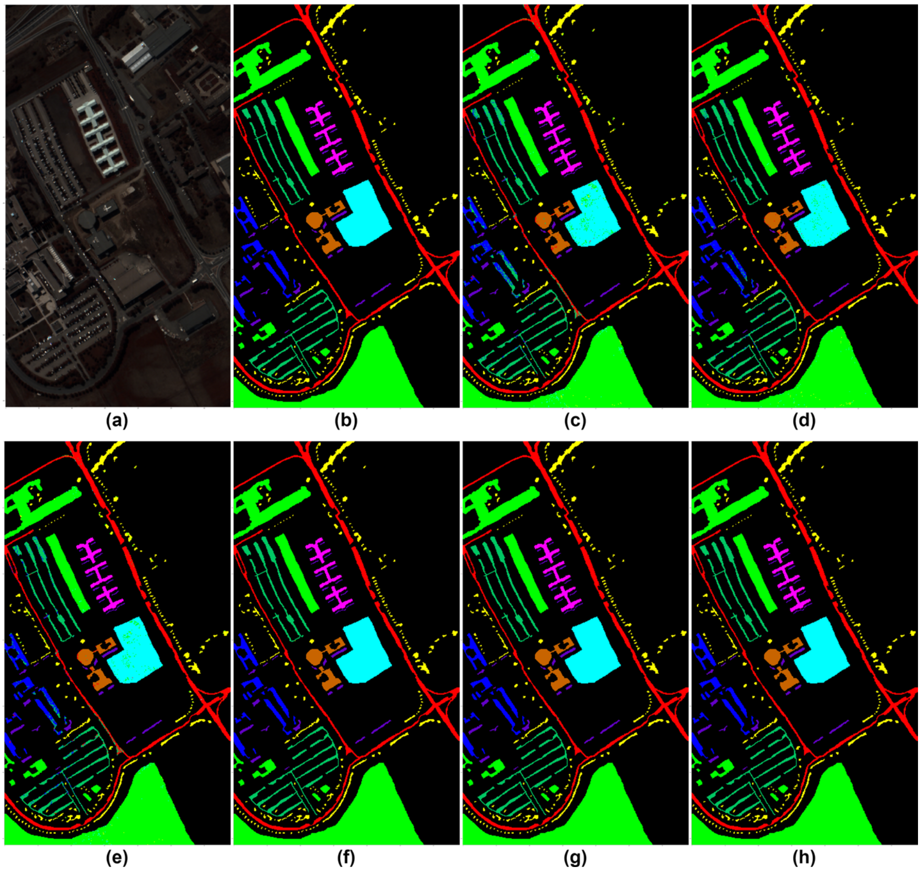

Fig. 7 The classification map for the Pavia University dataset. **a** RBG Image **b** Ground Truth **c**–**e** Predicted classification maps for HybridSN, single U-Net, and CEU-Net with no patching respectively. **f**–**h** Predicted classification maps for HybridSN, single U-Net, and CEU-Net with patching with a patch size of 10x10 pixels

Table 4. All reported results are determined by using our no-patching CEU-Net for the classifier.

Cluster hyperparameters

The number of clusters k and the clustering method was considered a hyperparameter for each dataset for our CEU-Net. The results are shown in Fig. 3. Therefore when the results show X% overall accuracy in CEU-Net for Indian Pines, that was achieved through K-means with cluster size determined in Table 5, ($k = 2$).

The preliminary result for these parameters involved implementing our CEU-Net with 5-fold cross-validation using K-Means++ and GMM for cluster numbers 2–6. Due to the dataset sizes, often the number of clusters $k > 4$ was impossible as too few samples would be sent to a single U-Net, which is dataset dependent. In addition, the performance of the U-Nets would drop significantly as shown in Fig. 3. The minimum number that k can be set to for CEU-Net is 2. If $k = 1$, then there would be no clustering, just the full dataset. This would not require an ensemble CEU-Net approach. Therefore, the $k = 1$ case is equivalent to our single U-Net using the entire dataset. Due to the class disparity of these datasets, increasing clusters and more complex clustering methods decrease CEU-Net performance. When the number of

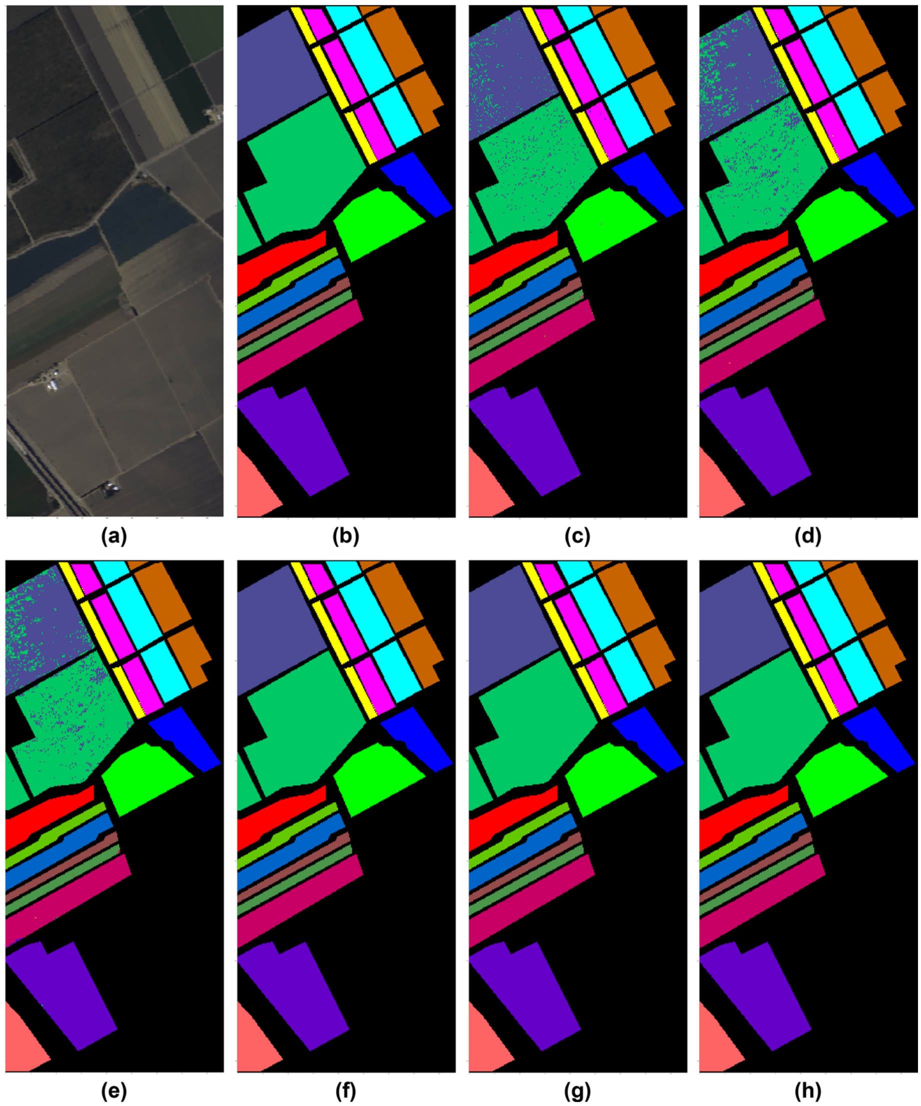

Fig. 8 The classification map for the Salinas dataset. **a** RBG Image **b** Ground Truth **c**–**e** Predicted classification maps for HybridSN, single U-Net, and CEU-Net with no patching respectively. **f**–**h** Predicted classification maps for HybridSN, single U-Net, and CEU-Net with patching with a patch size of 10x10 pixels

clusters are increased, we have less data for the associated network to learn with, therefore decreasing accuracy. We can see this steady decrease of performance for the K-Means graph (left) in Fig. 3. We can also see the increased complexity of GMM lead to decreased performance in 4 out of 6 datasets. For a similar reason, the higher performing clustering method GMM results in clusters that are too small due to the class disparity present in these, and most HSI, datasets.

Weight study

Our CEU-Net is an ensemble method that uses a linear combination of the prediction of each sub-U-Net to give better overall accuracy on average than the single model. In ensemble networks, there are often sub-classifiers that contribute more to an ensemble prediction than others. Therefore, we tested different weighted loss average ensembles to analyze if giving attention/weight to certain sub-classifiers is useful for CEU-Net in the context of HSI semantic segmentation. Equivalently, this experiment investigates the multipliers in the linear combinations of each sub-U-Net prediction presented in (3).

For this study we execute three different ensemble manipulations:

1 Constant weights: In this method, we give equal attention to each sub-model in the CEU-Net by multiplying the loss of each by a constant.
2 Abundance weights: In this method, the weight we use for the sub-model loss is equal to the percent pixel abundance that goes to each sub-model. Therefore, for example, if 60% of the data goes to one model, the weight will be 0.6.
3 Random weights: In this method, the weights will be assigned at random as long as the sum of the weights is equal to 1.

Fig. 4 shows that manipulating the attention of the ensemble network via loss weight modification has little bearing on overall test accuracy. We observe a constant loss weight modifier slightly increases overall test accuracy for CEU-Net in each dataset. Therefore, all of our CEU-Net semantic segmentation results will be using constant weights.

Semantic segmentation

All models for each dataset trained for semantic segmentation without patching are shown in Table 5. The empirically optimal feature reduction technique was used before each semantic segmentation: PCA. Using PCA, bands were reduced to 30. For each test, 5-fold cross-validation was performed to ensure more stability in the test results within a certain standard deviation versus a single value. This standard deviation is given in the form of a $\pm$ error for each test metric [43]. We observe that our CEU-Net outperforms the baseline architectures and our single U-Net for four out of six datasets. For the Salinas and Botswana datasets, OA and Kappa were higher in our single U-Net while AA was higher in CEU-Net. Statistical significance testing between each of the semantic segmentation methods from the data shown in Table 5 are presented in Appendix Table 10. Observe that all p-values for each test are well below $\alpha = 0.05$, and therefore all semantic segmentation results are statistically significant.

For our second semantic segmentation study, we investigate how CEU-Net performs with patching relative to the baselines. Results are shown in Tables 6, 7, and 8. The empirically optimal feature reduction technique was used before each semantic segmentation: PCA. Bands were reduced to 30. CPC was used with a patch sizes of 5 x 5, 10 x 10, and 15 x 15. For our ensemble network baselines, the results from [38–40] vary in feature reduction method and patch sizes compared to the single network baselines. In addition, we pick the best metrics from the ensemble works with the best ensemble size T. Therefore, we place $T = x$ next to each method to represent the ensemble size used in the corresponding experiments.

Our single U-Net and CEU-Net networks outperform the baseline HybridSN in almost all datasets used except Botswana. For ensemble methods, our CEU-Net outperforms the ensemble baselines, EECNN [38], Deep CNN Ensemble [39], and TCNN-E-ILS [40], in all datasets except for TCNN-E-ILS for KSC, despite having CEU-Net running with smaller patch sizes.

Seeing the results of Tables 5, 6, 7, and 8 patching appears to increase overall test accuracy each time. However, the cost of patching is dramatic in runtime, while in some datasets the gain in test accuracy is small. Figure 5 shows the exponential increase in runtime that a relatively small patch size of 10 x 10 creates.

The output of our CEU-Net model is a classification map that defines where pre-defined classes are within an HSI. Figure 6 shows the output of our CEU-Net for the Indian Pines dataset, Fig. 7 for the Pavia University dataset, and 8 for the Salinas dataset with classification maps from the baselines for comparison purposes.

Discussions

Patching

In general, we see patching increase our accuracy for each dataset except Houston. However, our networks, primarily CEU-Net, outperformed all other models without patching. All other datasets show overall accuracies closer to their patched counterparts with accuracies above 90%. Once the clustering method is an unsupervised non-deterministic method that does not use neighborhood information, we expected smaller accuracy versus patching for these datasets, however, our method can be used in datasets where patching is not as useful or difficult to implement. As we observe, patching dramatically increases runtime even with a smaller patch size of 10 x 10. In comparison to our baseline models, HybridSN uses a 25 x 25 patch size [6] and AeroRIT uses a 64 x 64 patch size [10] by default, which increases runtime significantly.

Feature reduction

Out of the feature reduction methods, it appears that PCA is the better feature reduction method when compared to neural network autoencoder approaches for spectral-only information for these datasets. Most papers that used autoencoders used neighborhood information in the feature reduction process via patching, making the information used for feature reduction not purely spectral information [14]. Our experimental results show that PCA outperforms autoencoders in all testing metrics when only spectral information is considered.

PCA however relies on finding a single principal axis and is solely linear in nature. This can lead to worse performance in data with very different classes like mixed man-made and natural targets. However, these datasets have ground truths that are made up of mostly man-made structures or natural targets with very little overlap. Therefore, linear separation is easier with PCA. In addition, autoencoders are prone to over-fitting due to the high number of parameters. This can be exacerbated due to a small number of labeled samples in the more common HSI datasets.

Semantic segmentation

U-Net has seen great results for semantic segmentation on medical imagery and it is shown to work well in the hyperspectral domain as well. Single U-Net outperformed each baseline in all datasets when not patching them.

In addition to our single U-Net's success, we extend it with CEU-Net which outperforms single U-Net in most datasets. This proves that it is not only possible to cluster pixels by their spectral signatures without knowing their individual class, but that it outperforms single network models as well on average. In addition, CEU-Net has better runtime compared to single U-Net in all datasets with patching and without patching, with the exception of Botswana and KSC without patching. Single U-Net performs better in runtime for Botswana and KSC without patching due to the limited number of labeled samples allowing the overhead of the clustering method in CEU-Net to dominate the overall runtime. This issue is remedied with larger datasets like Salinas and Pavia University.

The CEU-Net model outperforms the baselines in all datasets and Single U-Net in most with no patching, as shown in Table 5. Due to the easy segmentable nature and the more spectrally unique classes of Salinas, the benefits of the ensemble are not as useful versus datasets like Indian Pines. Botswana by far has the lowest number of samples, so separating them into different clusters can degrade performance due to having too few samples to train on or each sub-U-Net.

With Patching, CEU-Net and Single U-Net both outperform the baseline in five out of six datasets on average, as shown in Tables 6, 7, and 8. For the ensemble networks, CEU-Net outperformed all ensemble baselines in all datasets except TCNN-E-ILS in the KSC dataset. CEU-Net outperformed all of these networks with smaller patch sizes and a drastically smaller ensemble size due to the intelligent clustering sub-sampling.

The number of clusters in CEU-Net is a hyperparameter that can take any number as long as there are sufficient data points in each cluster to train each sub-U-Net separately. Clusters can be hand-picked or determined using any clustering algorithm. For the scope of this paper, two unsupervised clustering techniques, K-Mean++ and GMM, were explored with a varying number of clusters to experimentally show that our CEU-Net works with any clustering method and/or the number of clusters. These results are shown in Fig. 3.

Our proposed CEU-Net increases overall accuracy by partitioning the dataset into similar pixels via unsupervised clustering. This way, each sub-model can become an expert in similar pixels, allowing the network to detect minuscule differences between them that more generalized networks might miss; thereby increasing test accuracy. Then, combining the results of each specialized sub-model results in an overall accuracy larger than an individual model can achieve. The strength of CEU-Net is further increased with the addition of patching if the dataset benefits.

Conclusion

In prior works, there has not been a proper investigation, to our knowledge, on the role neighborhood information should have in HSI semantic segmentation. Through our discussion, we showed the weaknesses of patching for complex datasets, but also its strengths under particular conditions. By exploring feature reduction and semantic segmentation techniques without using neighborhood information, our single U-Net achieves competitive accuracies against baselines without it. We further debuted a novel network called CEU-Net that outperforms all baselines with a preprocessing step that is unsupervised and does not use neighborhood information. We believe that Clustering Ensemble U-Net can be used in future works on many datasets, especially ones that are extra challenging with complex and overlapping class labels where neighborhood information is weak. In addition, we showed CEU-Net and Single U-Net outperform the baseline networks like HybridSN with patching as a preprocessing step. When compared to other recent hyperspectral ensemble methods, CEU-Net outperformed all methods in all datasets except for KSC with the TCNN-E-ILS method. However, CEU-Net was able to outperform these methods with smaller patch sizes and a drastically smaller ensemble size. This shows that CEU-Net and Single U-Net are strong-performing general-purpose HSI semantic segmentation techniques that can be used in many different and diverse datasets.

Appendix

Statistical analysis of feature extraction and semantic segmentation methods

Each table in this section shows the statistical significance of our results by calculating p-values via Single Factor ANOVA between all feature reduction methods (Table 9), and between all semantic segmentation methods (Table 10). All values are well under the accepted $\alpha = 0.05$, and therefore, we can reject the null hypothesis that there is no difference between the results of each method and accept the alternative hypothesis that there is a statistically significant difference between the results of each method.

Table 9 Detailed ANOVA Single Factor p-value results between all feature reduction methods as shown in Table 3. With the accepted $\alpha = 0.05$, all p-values are well under the alpha, and therefore the tests are statistically significant

ANOVA	OA	AA	Kappa	OA	AA	Kappa
	IP (k = 2)			Salinas (k = 3)		
p-value ($\alpha = 0.05$)	4.75E−22	5.88E−23	3.31E−23	6.28E−18	6.67E−16	1.48E−18
	KSC (k = 2)			Botswana (k = 3)		
p-value ($\alpha = 0.05$)	3.96E−14	8.50E−17	3.95E−15	3.74E−13	2.87E−13	8.93E−14
	PU (k = 2)			Houston (k = 2)		
p-value ($\alpha = 0.05$)	5.73E−20	3.28E−22	1.11E−22	1.56E−22	1.32E−21	7.11E−23

Table 10 Detailed ANOVA Single Factor p-value results between all semantic segmentation methods as shown in Table 5. With the accepted $\alpha = 0.05$, all p-values are well under the alpha, and therefore the tests are statistically significant

ANOVA	OA	AA	Kappa	OA	AA	Kappa
	IP (k = 2)			Salinas (k = 3)		
p-value ($\alpha = 0.05$)	8.58E−27	2.07E−26	9.12E−28	7.51E−14	8.78E−09	1.39E−16
	KSC (k = 2)			Botswana (k = 3)		
p-value ($\alpha = 0.05$)	1.41E−11	2.19E−12	7.42E−14	5.31E−18	6.01E−18	5.73E−20
	PU (k = 2)			Houston (k = 2)		
p-value ($\alpha = 0.05$)	5.71E−16	6.17E−17	5.4E−16	1.03E−20	1.65E−18	2.66E−21

Layer summary of 2D and 3D convolutional autoencoders

Table 11 denotes the layer-wise summary of both the 2D and 3D convolutional autoencoders used in the feature reduction experiments.

Class-wise classification results for no-patching CEU-Net

Class-wise classification results for Indian Pines, Salinas, Pavia University, Kennedy Space Center, Botswana, and Houston datasets are summarised in Tables 12, 13, 14,

Table 11 The layer-wise summary of the 2D (left) and 3D (right) convolutional autoencoders used in experiments. *w* is the input spectral dimension, and *r* is the desired reduced spectral dimension size. Note that layer 1F in each network is the end of the encoder and 2A is the start of the decoder

Layer #	Layer Name	Output Shape	Layer #	Layer Name	Output Shape
0	Input Layer	(1,1,w)	0	Input Layer	(1,1,1,w)
1A	Conv2D_1	(1,1,w)	1A	Conv3D_1	(1,1,1,w)
1B	MaxPooling2D_1	(1,1,w)	1B	MaxPooling3D_1	(1,1,1,w)
1C	Conv2D_2	(1,1,60)	1C	Conv3D_2	(1,1,1,60)
1D	MaxPooling2D_2	(1,1,60)	1D	MaxPooling3D_2	(1,1,1,60)
1E	Conv2D_3	(1,1,r)	1E	Conv3D_3	(1,1,1,r)
1F	MaxPooling2D_3	(1,1,r)	1F	MaxPooling3D_3	(1,1,1,r)
2A	Conv2D_4	(1,1,r)	2A	Conv3D_4	(1,1,1,r)
2B	UpSampling2D_1	(1,1,r)	2B	UpSampling3D_1	(1,1,1,r)
2C	Conv2D_5	(1,1,60)	2C	Conv3D_5	(1,1,1,60)
2D	UpSampling2D_2	(1,1,60)	2D	UpSampling3D_2	(1,1,1,60)
2E	Conv2D_6	(1,1,w)	2E	Conv3D_6	(1,1,1,w)
2F	UpSampling2D_3	(1,1,w)	2F	UpSampling3D_3	(1,1,1,w)
2G	Conv2D_7	(1,1,w)	2G	Conv3D_7	(1,1,1,w)

Table 12 Detailed classification test results for the Indian Pines Dataset in terms of Precision, Recall, and F1-Score. Testing was done with PCA 30 CEU-Net no-patching with a 75%/25% Training/Testing split

Class Labels	Precision	Recall	f1-score	Support
Alfalfa	0.70	0.70	0.70	10
Corn Notill	0.93	0.80	0.86	378
Corn Mintill	0.90	0.87	0.89	223
Corn	0.65	0.86	0.74	51
Grass Pasture	0.94	0.93	0.94	120
Grass Trees	0.91	0.99	0.95	174
Grass Pasture M	0.92	1.00	0.96	11
Hay Windrowed	0.97	0.97	0.97	110
Oats	1.00	1.00	1.00	3
Soybean Notill	0.87	0.92	0.90	246
Soybean Mintill	0.90	0.92	0.91	605
Soybean Clean	0.85	0.89	0.87	158
Wheat	0.96	1.00	0.98	43
Woods	0.94	0.97	0.95	301
Buildings etc.	0.83	0.65	0.73	103
Stone Steel Towers	1.00	1.00	1.00	27
Accuracy			0.90	2563
Macro Average	0.89	0.91	0.90	2563
Weighted Average	0.90	0.90	0.90	2563

Table 13 Detailed classification test results for the Salinas Dataset in terms of Precision, Recall, and F1-Score. Testing was done with PCA 30 CEU-Net no-patching with a 75%/25% Training/Testing split

Class Labels	Precision	Recall	f1-score	Support
Broccoli Green I	1.00	1.00	1.00	505
Broccoli Green II	1.00	1.00	1.00	931
Fallow	1.00	1.00	1.00	492
Fallow Rough Plow	0.99	0.99	0.99	345
Fallow Smooth	0.99	1.00	1.00	686
Stubble	1.00	1.00	1.00	957
Celery	1.00	1.00	1.00	925
Grapes Untrained	0.93	0.91	0.92	2842
Soil Vineyard Develop	1.00	1.00	1.00	1559
Corn Sensed Green Weeds	0.99	0.99	0.99	789
Lettuce Romaine 4wk	1.00	1.00	1.00	276
Lettuce Romaine 5wk	1.00	1.00	1.00	462
Lettuce Romaine 6wk	1.00	0.98	0.99	218
Lettuce Romaine 7wk	0.99	1.00	0.99	276
Vineyard Untrained	0.87	0.89	0.88	1818
Vineyard Vertical	1.00	0.99	1.00	452
Accuracy			0.96	13533
Macro Average	0.98	0.98	0.98	13533
Weighted Average	0.96	0.96	0.96	13533

Table 14 Detailed classification test results for the Pavia University Dataset in terms of Precision, Recall, and F1-Score. Testing was done with PCA 30 CEU-Net no-patching with a 75%/25% Training/Testing split

Class Labels	Precision	Recall	f1-score	Support
Asphalt	0.97	0.95	0.96	1693
Meadows	0.98	0.99	0.98	4629
Gravel	0.89	0.84	0.86	550
Trees	0.99	0.95	0.97	715
Painted Metal Sheets	1.00	1.00	1.00	331
Bare Soil	0.97	0.94	0.96	1313
Bitumen	0.88	0.92	0.90	341
Self-Blocking Bricks	0.88	0.92	0.90	882
Shadows	1.00	1.00	1.00	240
Accuracy			0.96	10694
Macro Average	0.95	0.95	0.95	10694
Weighted Average	0.96	0.96	0.96	10694

15, 16 and 17 respectively. Confusion matrices are available in Figs. 9, 10, 11, 12, 13 and 14 as heatmaps, and Tables 18, 19, 20, 21, 22 and 23 as numeric values. All results are from PCA 30 data and CEU-Net classification with no patching.

Table 15 Detailed classification test results for the Kennedy Space Center Dataset in terms of Precision, Recall, and F1-Score. Testing was done with PCA 30 CEU-Net no-patching with a 75%/25% Training/Testing split

Class Labels	Precision	Recall	f1-score	Support
Scrub	0.96	0.98	0.97	197
Willow Swamp	0.97	0.94	0.96	72
CP Hammock	0.84	0.90	0.87	63
Slash Pine	0.82	0.75	0.78	67
Oak/Broadleaf	0.85	0.79	0.82	43
Hardwood	0.89	0.85	0.87	47
Swamp	0.90	0.90	0.90	31
Graminoid Marsh	0.97	0.94	0.95	109
Spartina Marsh	0.95	1.00	0.97	135
Cattail Marsh	0.97	1.00	0.98	87
Salt Marsh	1.00	0.99	0.99	97
Mud Flats	0.99	0.97	0.98	121
Water	1.00	1.00	1.00	234
Accuracy			0.95	1303
Macro Average	0.93	0.92	0.93	1303
Weighted Average	0.95	0.95	0.95	1303

Table 16 Detailed classification test results for the Botswana Dataset in terms of Precision, Recall, and F1-Score. Testing was done with PCA 30 CEU-Net no-patching with a 75%/25% Training/Testing split

Class Labels	Precision	Recall	f1-score	Support
Water	1.00	1.00	1.00	69
Hippo Grass	1.00	1.00	1.00	19
Floodplain Grasses 1	1.00	0.98	0.99	62
Floodplain Grasses 2	0.93	1.00	0.96	51
Reeds 1	0.94	0.94	0.94	80
Riparian	0.93	0.90	0.91	70
Firescar 2	1.00	0.97	0.98	64
Island Interior	1.00	1.00	1.00	60
Acacia Woodlands	0.99	0.92	0.95	75
Acacia Shrublands	0.91	1.00	0.95	51
Acacia Grasslands	1.00	0.99	0.99	80
Short Mopane	0.84	1.00	0.91	36
Mixed Mopane	0.97	0.90	0.93	68
Exposed Soils	1.00	1.00	1.00	27
Accuracy			0.96	812
Macro Average	0.96	0.97	0.97	812
Weighted Average	0.97	0.96	0.96	812

Table 17 Detailed classification test results for the Houston Dataset in terms of Precision, Recall, and F1-Score. Testing was done with PCA 30 CEU-Net no-patching with a 75%/25% Training/Testing split

Class Labels	Precision	Recall	f1-score	Support
Healthy Grass	0.99	0.99	0.99	355
Stressed Grass	0.98	1.00	0.99	354
Artificial Turf	1.00	1.00	1.00	185
Trees	0.99	1.00	0.99	308
Soil	1.00	0.99	1.00	322
Water	1.00	1.00	1.00	69
Residential	0.99	0.97	0.98	316
Commercial	1.00	0.99	0.99	78
Roads	0.99	0.97	0.98	369
Highway	0.97	0.99	0.98	361
Railways	0.99	0.98	0.98	424
Parking Lot 1	0.95	0.98	0.97	354
Parking Lot 2	0.93	0.85	0.89	67
Tennis Court	1.00	0.99	1.00	126
Running Track	1.00	1.00	1.00	159
Accuracy			0.98	3847
Macro Average	0.99	0.98	0.98	3847
Weighted Average	0.98	0.98	0.98	3847

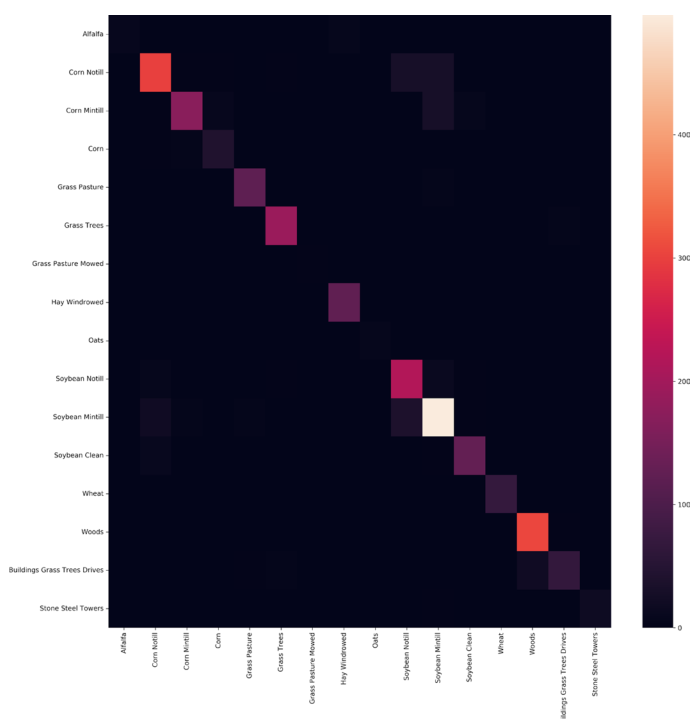

Fig. 9 Indian Pines Confusion Matrix for PCA 30 CEU-Net no patching with a 75%/25% Training/Testing split

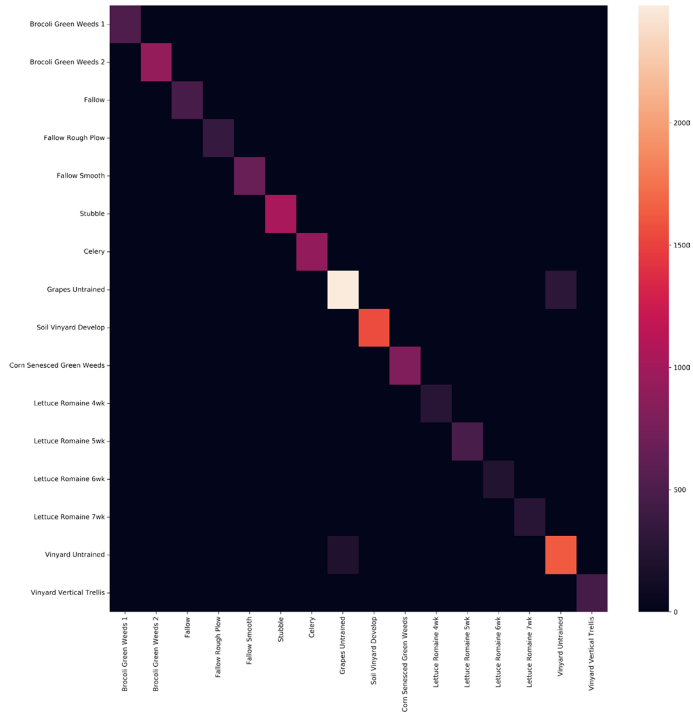

Fig. 10 Salinas Confusion Matrix for PCA 30 CEU-Net no patching with a 75%/25% Training/Testing split

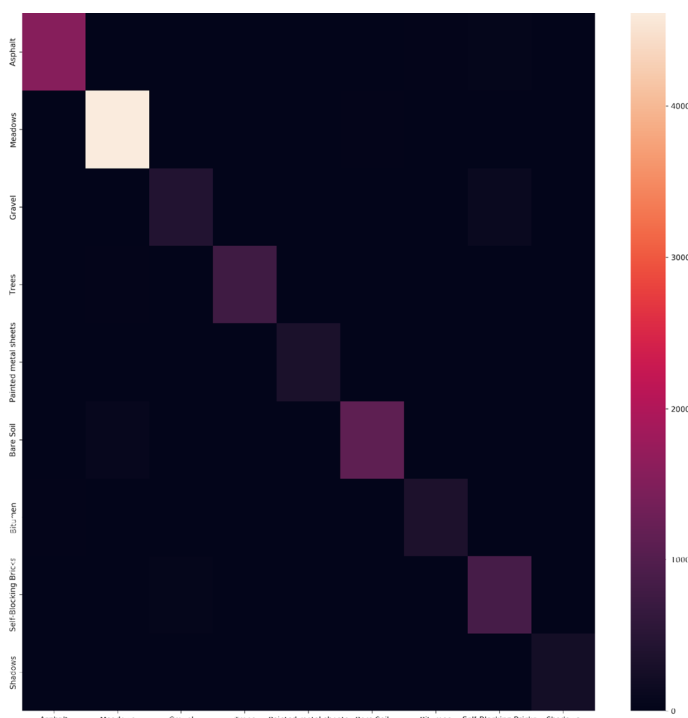

Fig. 11 Pavia University Confusion Matrix for PCA 30 CEU-Net no patching with a 75%/25% Training/Testing split

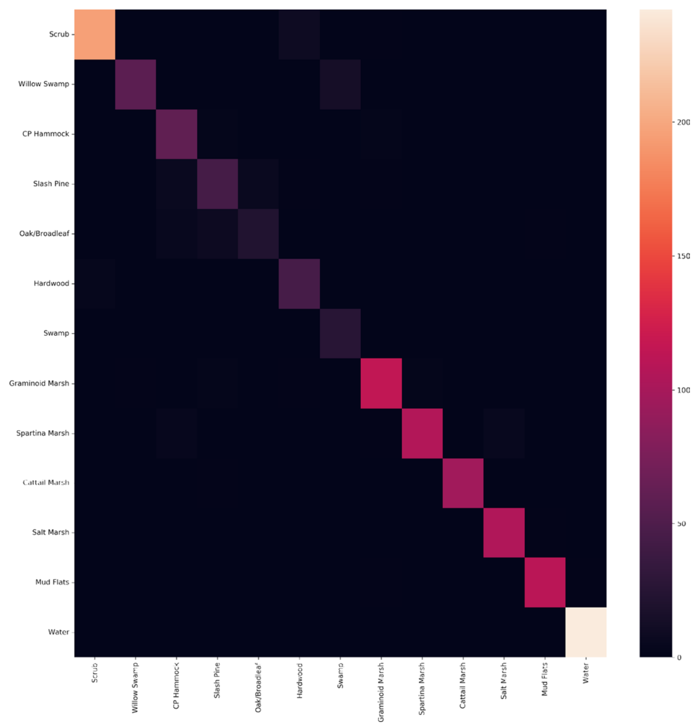

Fig. 12 Kennedy Space Center Confusion Matrix for PCA 30 CEU-Net no patching with a 75%/25% Training/Testing split

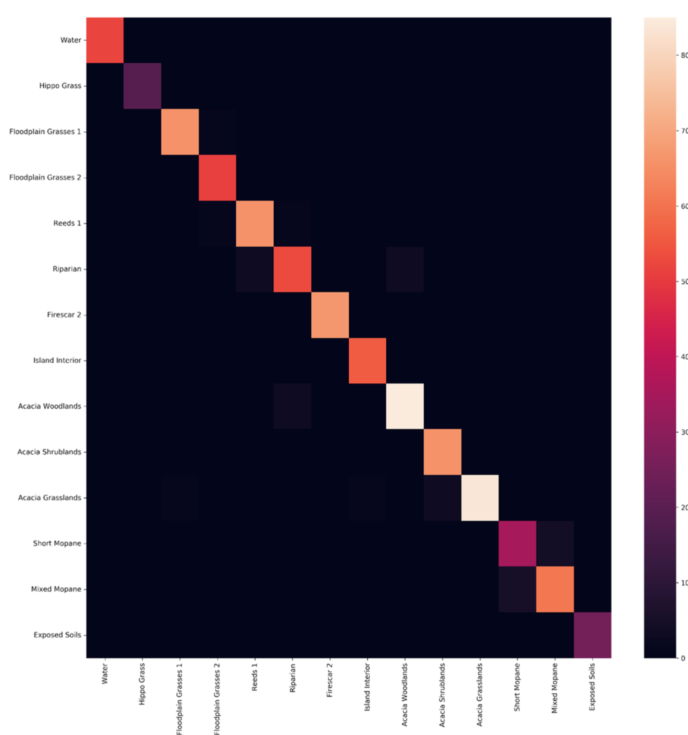

Fig. 13 Botswana Confusion Matrix for PCA 30 CEU-Net no patching with a 75%/25% Training/Testing split

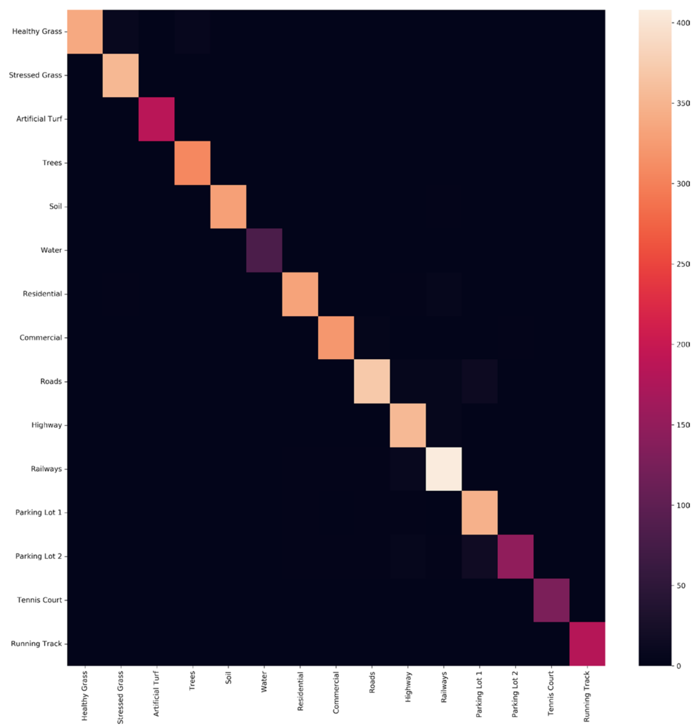

Fig. 14 Houston Confusion Matrix for PCA 30 CEU-Net no patching with a 75%/25% Training/Testing split

Table 18 Indian Pines Confusion Matrix numeric values for PCA 30 CEU-Net no patching with a 75%/25% Training/Testing split

	Alfalfa	Corn 1	Corn 2	Corn 3	Grass 1	Grass 2	Grass 3	Hay	Oats	Soybean 1	Soybean 2	Soybean 3	Wheat	Woods	Buildings etc.	Stone etc.
Alfalfa	**7**	0	0	0	0	0	0	2	0	1	0	0	0	0	0	0
Corn 1	0	**306**	5	6	0	1	0	0	0	20	35	5	0	0	0	0
Corn 2	0	1	**191**	7	0	1	0	0	0	1	15	7	0	0	0	0
Corn 3	0	0	2	**43**	2	2	0	1	0	0	0	1	0	0	0	0
Grass 1	0	1	0	0	**110**	2	0	0	0	0	1	2	0	1	3	0
Grass 2	0	0	0	0	1	**173**	0	0	0	0	0	0	0	0	0	0
Grass 3	0	0	0	0	0	0	**11**	0	0	0	0	0	0	0	0	0
Hay	3	0	0	0	0	0	0	**107**	0	0	0	0	0	0	0	0
Oats	0	0	0	0	0	0	0	0	**2**	0	0	0	0	0	1	0
Soybean 1	0	5	1	1	1	0	0	0	0	**222**	12	4	0	0	0	0
Soybean 2	0	10	9	1	0	1	1	0	0	12	**559**	10	0	0	2	0
Soybean 3	0	1	7	0	0	0	0	0	0	1	4	**144**	0	0	1	0
Wheat	0	0	0	0	0	0	0	0	0	0	0	0	**43**	0	0	0
Woods	0	0	0	0	0	0	0	0	0	0	0	0	0	**294**	7	0
Buildings etc.	0	0	0	0	2	11	0	0	0	0	0	0	0	21	**68**	1
Stone etc.	0	1	0	0	0	0	0	0	0	0	2	0	0	0	0	**24**

Highest performing values are highlighted in bold

Table 19 Salinas Confusion Matrix numeric values for PCA 30 CEU-Net no patching with a 75%/25% Training/Testing split

	Brocoli 1	Brocoli 2	Fallow 1	Fallow 2	Fallow 3	Stubble	Celery	Grapes	Soil Vinyard	Corn	Lettuce 4wk	Lettuce 5wk	Lettuce 6wk	Lettuce 7wk	Vinyard 1	Vinyard 2
Brocoli 1	**496**	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
Brocoli 2	0	**937**	0	0	0	0	0	0	0	0	0	0	0	0	0	0
Fallow 1	0	0	**519**	0	0	0	0	0	0	0	0	0	0	0	0	0
Fallow 2	0	0	0	**357**	1	0	0	0	0	0	0	0	0	0	0	0
Fallow 3	0	0	0	2	**678**	0	0	0	0	0	0	0	0	0	0	0
Stubble	0	0	0	0	0	**1019**	0	0	0	0	0	0	0	0	0	0
Celery	0	0	0	0	0	0	**869**	0	0	0	0	0	0	0	0	0
Grapes	0	0	1	0	0	0	0	**2465**	0	0	0	0	0	0	308	0
Soil Vinyard	0	0	0	0	0	0	0	3	**1596**	1	0	2	0	0	0	0
Corn	0	0	0	0	0	0	0	1	9	**802**	0	0	0	0	1	0
Lettuce 4wk	0	0	0	0	0	0	0	0	0	0	**238**	1	0	0	0	0
Lettuce 5wk	0	0	0	0	0	0	0	0	0	0	0	**484**	0	0	0	0
Lettuce 6wk	0	0	0	0	0	0	0	0	0	0	0	0	**229**	1	0	0
Lettuce 7wk	0	0	0	0	0	0	0	0	0	2	0	0	1	**288**	0	0
Vinyard 1	0	0	0	0	0	0	0	168	0	0	0	0	0	0	**1641**	0
Vinyard 2	0	0	0	0	0	0	0	0	0	2	0	0	0	0	0	**411**

Highest performing values are highlighted in bold

Table 20 Pavia University Confusion Matrix numeric values for PCA 30 CEU-Net no patching with a 75%/25% Training/Testing split

	Asphalt	Meadows	Gravel	Trees	Painted metal sheets	Bare Soil	Bitumen	Self-Blocking Bricks	Shadows
Asphalt	**1607**	0	10	0	0	0	43	33	0
Meadows	0	**4594**	0	4	0	31	0	0	0
Gravel	2	1	**448**	0	0	0	0	99	0
Trees	0	33	0	**681**	0	1	0	0	0
Painted metal sheets	0	0	0	0	**331**	0	0	0	0
Bare Soil	0	76	0	2	0	**1235**	0	0	0
Bitumen	26	0	0	0	0	0	**313**	2	0
Self-Blocking Bricks	11	2	32	0	0	5	2	**830**	0
Shadows	0	0	0	0	0	0	0	0	**240**

Highest performing values are highlighted in bold

Table 21 Kennedy Space Center Confusion Matrix numeric values for PCA 30 CEU-Net no patching with a 75%/25% Training/Testing split

	Scrub	Swamp 1	CP Hammock	Slash Pine	Oak	Hardwood	Swamp 2	Marsh 1	Marsh 2	Marsh 3	Marsh 4	Mud Flats	Water
Scrub	**194**	0	0	0	0	3	0	0	0	0	0	0	0
Swamp 1	0	**67**	0	0	0	0	4	0	1	0	0	0	0
CP Hammock	0	0	**59**	2	0	0	0	1	1	0	0	0	0
Slash Pine	1	0	12	**50**	3	0	0	1	0	0	0	0	0
Oak	0	0	1	7	**35**	0	0	0	0	0	0	0	0
Hardwood	5	0	0	2	0	**40**	0	0	0	0	0	0	0
Swamp	0	1	0	0	0	1	**29**	0	0	0	0	0	0
Marsh 1	2	0	0	0	0	0	0	**104**	3	0	0	0	0
Marsh 2	0	0	0	0	0	0	0	0	**135**	0	0	0	0
Marsh 3	0	0	0	0	0	0	0	1	0	**86**	0	0	0
Marsh 4	0	0	0	0	0	0	0	0	0	0	**96**	1	0
Mud Flats	0	0	0	0	0	0	0	0	0	3	1	**117**	0
Water	0	0	0	0	0	0	0	0	0	0	0	0	**234**

Highest performing values are highlighted in bold

Table 22 Botswana Confusion Matrix numeric values for PCA 30 CEU-Net no patching with a 75%/25% Training/Testing split

	Water	Grasses 1	Grasses 2	Grasses 3	Reeds 1	Riparian	Firescar 2	Island Interior	Acacia 1	Acacia 2	Acacia 3	Mopane 1	Mopane 2	Exposed Soils
Water	**52**	0	0	0	0	0	0	0	0	0	0	0	0	0
Grasses 1	0	**19**	0	0	0	0	0	0	0	0	0	0	0	0
Grasses 2	0	0	**66**	1	0	0	0	0	0	0	0	0	0	0
Grasses 3	0	0	0	**51**	0	0	0	0	0	0	0	0	0	0
Reeds 1	0	0	0	1	**65**	2	0	0	0	0	0	0	0	0
Riparian	0	0	0	0	2	**53**	0	0	3	0	0	0	1	0
Firescar 2	0	0	0	0	0	0	**67**	0	0	0	0	0	0	0
Island Interior	0	0	0	0	0	0	0	**56**	0	0	0	0	0	0
Acacia 1	0	0	0	0	0	0	0	0	**88**	0	0	0	0	0
Acacia 2	0	0	0	0	0	0	0	0	0	**66**	0	0	0	0
Acacia 3	0	0	1	0	0	0	0	1	0	2	**85**	0	0	0
Mopane 1	0	0	0	0	0	0	0	0	0	0	0	**38**	1	0
Mopane 2	0	0	0	0	0	0	0	0	0	0	0	5	**61**	0
Exposed Soils	0	0	0	0	0	0	0	0	0	0	0	0	0	**25**

Highest performing values are highlighted in bold

Table 23 Houston Confusion Matrix numeric values for PCA 30 CEU-Net no patching with a 75%/25% Training/Testing split

	Grass 1	Grass 2	Turf	Trees	Soil	Water	Residential	Commercial	Roads	Highway	Railways	Lot 1	Lot 2	Court	Track
Grass 1	**321**	2	0	0	0	0	0	0	0	0	0	0	0	0	0
Grass 2	2	**363**	0	0	0	0	0	0	0	0	0	0	0	0	0
Turf	0	0	**203**	0	0	0	0	0	0	0	0	0	0	0	0
Trees	1	0	0	**321**	0	0	0	0	0	0	0	0	0	0	0
Soil	0	0	0	0	**13**	0	0	0	0	0	0	0	0	0	0
Water	0	0	0	0	0	**80**	0	0	0	0	0	0	0	0	0
Residential	0	2	0	0	0	0	**43**	2	0	0	0	0	1	0	0
Commercial	0	0	0	0	0	0	2	**157**	0	0	0	0	0	0	0
Roads	0	0	0	0	0	0	2	2	**2**	0	0	0	1	1	0
Highway	0	0	0	0	0	0	0	0	0	**1**	0	0	1	0	0
Railways	0	0	0	0	0	0	0	1	0	0	**0**	0	0	0	0
Lot 1	0	0	0	0	0	0	0	2	0	0	0	**2**	1	0	0
Lot 2	0	0	0	0	0	0	0	0	0	0	0	0	**78**	0	0
Court	0	0	0	0	0	0	1	0	0	0	0	0	0	**124**	0
Track	0	0	0	0	0	0	0	0	0	0	0	0	0	1	**24**

Highest performing values are highlighted in bold

Abbreviations

HSI	Hyperspectral image
CPC	Center pixel classification
CNN	Convolutional neural network
KSC	Kennedy space center
DNN	Deep neural network
CEU-Net	Cluster ensemble U-Net
ML	Machine learning
SFS	Sequential feature selector
SVM	Support vector machine
PCA	Principal component analysis
SOM	Self-Organizing Maps
RNN	Recurrent neural network
LSTM	Long-short term memory
CAE	Convolutional autoencoder
LReLU	Leaky rectified linear unit
ANOVA	Analysis of variance
OA	Overall accuracy
AA	Average accuracy
GMM	Gaussian mixture models
SP	Spectral bands

Acknowledgements
This work has been supported by NSF 1920908 (EPSCoR RII Track-2); the findings are those of the authors only and do not represent any position of these funding bodies.

Author contributions
NS performed the review, developed the idea, wrote the code, executed experiments, and drafted the manuscript. SYS advised, developed mathematical notation, helped develop the concept and finalized this work. Both authors read and approved the final manuscript.

Funding
Not applicable.

Availability of data and materials
Complete CEU-Net code is available in GitHub: https://github.com/Sekeh-Lab/CEU-Net All datasets used, except Houston, are available on the GIC website, https://www.ehu.eus/ccwintco/index.php/Hyperspectral_Remote_Sensing_Scenes The Houston dataset is available from the authors on reasonable request per the instructions here: https://hyperspectral.ee.uh.edu/?page_id=1075.

Declarations

Ethics approval and consent to participate
Not Applicable.

Consent for publication
All Authors consent to the publication of this paper.

Competing interests
The authors declare that they have no competing interests.

Received: 18 October 2022 Accepted: 16 March 2023
Published online: 12 April 2023

References

1. Cook BD, Nelson RF, Middleton EM, Morton DC, McCorkel JT, Masek JG, Ranson KJ, Ly V, Montesano PM, et al. NASA Goddard's Lidar, hyperspectral and thermal (G-LIHT) airborne imager. Remote Sens. 2013;5(8):4045–66.
2. Garcia JL, Paoletti ME, Jiménez LI, Haut JM, Plaza A. Efficient semantic segmentation of hyperspectral images using adaptable rectangular convolution. IEEE Geosci Remote Sens Lett. 2022. https://doi.org/10.1109/LGRS.2022.3140950.
3. Yuan X, Shi J, Gu L. A review of deep learning methods for semantic segmentation of remote sensing imagery. Expert Syst Appl. 2021;169: 114417.
4. Kovacs KF. Cost of potential emerald ash borer damage in US communities, 2009–2019. Ecol Econ. 2009;69:569–78.
5. Grana M, Veganzons M, Ayerdi B. Hyperspectral remote sensing scenes. http://www.ehu.eus/ccwintco/index.php/Hyperspectral_Remote_Sensing_Scenes. Accessed: 2022-02-22.
6. Roy SK, Krishna G, Dubey SR, Chaudhuri BB. HybridSN: exploring 3-D-2-D CNN feature hierarchy for hyperspectral image classification. IEEE Geosci Remote Sens Lett. 2020;17(2):277–81. https://doi.org/10.1109/LGRS.2019.2918719.
7. Mei S, Ji J, Geng Y, Zhang Z, Li X, Du Q. Unsupervised spatial-spectral feature learning by 3D convolutional autoencoder for hyperspectral classification. IEEE Trans Geosci Remote Sens. 2019;57(9):6808–20.

8. Yu C, Han R, Song M, Liu C, Chang C-I. A simplified 2D–3D CNN architecture for hyperspectral image classification based on spatial-spectral fusion. IEEE J Sel Top Appl Earth Obs Remote Sens. 2020;13:2485–501. https://doi.org/10.1109/JSTARS.2020.2983224.
9. Shen H, Jiang M, Li J, Yuan Q, Wei Y, Zhang L. Spatial-spectral fusion by combining deep learning and variational model. IEEE Trans Geosci Remote Sens. 2019;57(8):6169–81. https://doi.org/10.1109/TGRS.2019.2904659.
10. Rangnekar A, Mokashi N, Ientilucci EJ, Kanan C, Hoffman MJ. Aerorit: a new scene for hyperspectral image analysis. IEEE Trans Geosci Remote Sens. 2020;58(11):8116–24. https://doi.org/10.1109/TGRS.2020.2987199.
11. Hu X, Zhong Y, Wang X, Luo C, Zhao J, Lei L, Zhang L. SPNet: Spectral patching end-to-end classification network for UAV-borne hyperspectral imagery with high spatial and spectral resolutions. IEEE Trans Geosci Remote Sens. 2022;60:1–17. https://doi.org/10.1109/TGRS.2021.3049292.
12. Sekeh SY, Hero AO. Feature selection for mutlti-labeled variables via dependency maximization. In: ICASSP 2019-2019 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP), IEEE, 2019; pp. 3127–3131
13. Romero A, Gatta C, Camps-Valls G. Unsupervised deep feature extraction for remote sensing image classification. IEEE Trans Geosci Remote Sens. 2015;54(3):1349–62.
14. Mei S, Ji J, Geng Y, Zhang Z, Li X, Du Q. Unsupervised spatial-spectral feature learning by 3D convolutional autoencoder for hyperspectral classification. IEEE Trans Geosci Remote Sens. 2019;57(9):6808–20. https://doi.org/10.1109/TGRS.2019.2908756.
15. Chen C, Zhang J-J, Zheng C-H, Yan Q, Xun L-N. Classification of hyperspectral data using a multi-channel convolutional neural network. In: International Conference on Intelligent Computing, Springer, 2018; pp. 81–92
16. He M, Li B, Chen H. Multi-scale 3d deep convolutional neural network for hyperspectral image classification. In: 2017 IEEE International Conference on Image Processing (ICIP), IEEE, 2017, pp. 3904–3908
17. Patel TR, Bodduluri S, Anthony T, Monroe WS, Kandhare PG, Robinson J-P, Nakhmani A, Zhang C, Bhatt SP, Bangalore PV. Performance characterization of single and multi GPU training of U-Net architecture for medical image segmentation tasks. In: Proceedings of the Practice and Experience in Advanced Research Computing on Rise of the Machines (learning), ACM, 2019; pp. 1–4. https://doi.org/10.1145/3332186.3333152.
18. Li X, Chen H, Qi X, Dou Q, Fu C-W, Heng P-A. H-DenseUNet: Hybrid densely connected UNet for liver and tumor segmentation from CT volumes. IEEE Trans Med Imaging. 2018;37(12):2663–74. https://doi.org/10.1109/TMI.2018.2845918.
19. Ronneberger O, Fischer P, Brox T. U-Net: Convolutional networks for biomedical image segmentation. In: International Conference on Medical Image Computing and Computer-assisted Intervention, Springer; 2015, pp. 234–241
20. Dietterich TG. An experimental comparison of three methods for constructing ensembles of decision trees: bagging, boosting, and randomization. Mach Learn. 2000;40(2):139–57.
21. Sewell M. Ensemble learning. RN. 2008;11(02):1–34.
22. Schapire, R.E.: Explaining adaboost. Empirical Inference: Festschrift in Honor of Vladimir N. Vapnik, 37–52(2013)
23. Zhan Y, Tian H, Liu W, Yang Z, Wu K, Wang G, Chen P, Yu X. A new hyperspectral band selection approach based on convolutional neural network. In: 2017 IEEE International Geoscience and Remote Sensing Symposium (IGARSS). IEEE, 2017
24. Labate D, Safaripoorfatide M, Karantzas N, Prasad S, Foroozandeh Shahraki F. Structured receptive field networks and applications to hyperspectral image classification, 2019; p. 23. https://doi.org/10.1117/12.2527712
25. Landgrebe D, Biehl L. A freeware multispectral image data analysis system. 2015. https://engineering.purdue.edu/~biehl/MultiSpec/. Accessed: 2022-01-12.
26. Ritchie, H., and M. Roser. 2021. Forests and deforestation. Our World in Data. Published Online at OurWorldInData.org. Retrieved from: https://ourworldindata.org/forests-and-deforestation. Accessed: 2022-03-01.
27. Moghimi A, Yang C, Marchetto PM. Ensemble feature selection for plant phenotyping: a journey from hyperspectral to multispectral imaging. IEEE Access. 2018;6:56870–84.
28. Tabrizi P, Rezatofighi S, Yazdanpanah M. Using PCA and LVQ neural network for automatic recognition of five types of white blood cells. In: 2010 Annual International Conference of the IEEE Engineering in Medicine and Biology, IEEE, 2010; pp. 5593–5596
29. Kohonen, T.: The self-organizing map. Proceedings of the IEEE 78(9), 1464–1480 (1990)
30. Hidalgo DR, Cortés BB, Bravo EC. Dimensionality reduction of hyperspectral images of vegetation and crops based on self-organized maps. Inf Process Agric. 2021;8(2):310–27.
31. Shamsolmoali P, Zareapoor M, Yang J. Convolutional neural network in network (CNNiN): hyperspectral image classification and dimensionality reduction. IET Image Proc. 2019;13(2):246–53.
32. Cao Z, Li X, Feng Y, Chen S, Xia C, Zhao L. ContrastNet: Unsupervised feature learning by autoencoder and prototypical contrastive learning for hyperspectral imagery classification. Neurocomputing. 2021;460:71–83.
33. Wang D, Du B, Zhang L, Xu Y. Adaptive spectral-spatial multiscale contextual feature extraction for hyperspectral image classification. IEEE Trans Geosci Remote Sens. 2020;59(3):2461–77.
34. Zhou F, Hang R, Liu Q, Yuan X. Hyperspectral image classification using spectral-spatial LSTMS. Neurocomputing. 2019;328:39–47.
35. Yu W, Huang H, Shen G. Multi-level dual-direction modifying variational autoencoders for hyperspectral feature extraction. IEEE Geosci Remote Sens Lett. 2022. https://doi.org/10.1109/LGRS.2022.3183408.
36. Yu W, Zhang M, Shen Y. Spatial revising variational autoencoder-based feature extraction method for hyperspectral images. IEEE Trans Geosci Remote Sens. 2020;59(2):1410–23.
37. Bao R, Xia J, Dalla Mura M, Du P, Chanussot J, Ren J. Combining morphological attribute profiles via an ensemble method for hyperspectral image classification. IEEE Geosci Remote Sens Lett. 2016;13(3):359–63.
38. Lv Q, Feng W, Quan Y, Dauphin G, Gao L, Xing M. Enhanced-random-feature-subspace-based ensemble CNN for the imbalanced hyperspectral image classification. IEEE J Sel Top Appl Earth Obs Remote Sens. 2021;14:3988–99.
39. Iyer P, Sriram A, Lal S. Deep learning ensemble method for classification of satellite hyperspectral images. Remote Sens Appl Soc Environ. 2021;23: 100580. https://doi.org/10.1016/j.rsase.2021.100580.
40. He X, Chen Y. Transferring CNN ensemble for hyperspectral image classification. IEEE Geosci Remote Sens Lett. 2020;18(5):876–80.

41. Yu L, Xie J, Chen S, Zhu L. Generating labeled samples for hyperspectral image classification using correlation of spectral bands. Front Comp Sci. 2016;10(2):292–301. https://doi.org/10.1007/s11704-015-4103-4.
42. Thatbrguy: Thatbrguy/hyperspectral-image-segmentation: Semantic Segmentation of hyperspectral imagesusing a U-net with depthwise separable convolutions. GitHub (2021).https://github.com/thatbrguy/Hyperspectral-Image-Segmentation. Accessed: 2021-12-03.
43. Berrar, D. Cross-validation. In Encyclopedia of Bioinformatics and Computational Biology; Elsevier: Amsterdam, The Netherlands, 2019; pp. 542–545.
44. of Medicine NL. Significance level. National Institutes of Health. 2019. https://www.nlm.nih.gov/nichsr/stats_tutorial/section2/mod11_significance.html. Accessed: 2022-01-10.
45. Carletta J. Assessing agreement on classification tasks: the kappa statistic. 1996. arXiv preprint cmp-lg/9602004
46. Arthur D, Vassilvitskii S. k-means++: the advantages of careful seeding. Stanford: Technical report; 2006.
47. McLachlan GJ, Basford KE. Mixture models: inference and applications to clustering, vol. 38. New York: M. Dekker; 1988.
48. Maugis C, Celeux G, Martin-Magniette M-L. Variable selection for clustering with gaussian mixture models. Biometrics. 2009;65(3):701–9.

Publisher's Note

Springer Nature remains neutral with regard to jurisdictional claims in published maps and institutional affiliations.