remote sensing

MDPI

Article

One-Dimensional Convolutional Neural Network Land-Cover Classification of Multi-Seasonal Hyperspectral Imagery in the San Francisco Bay Area, California

Daniel Guidici [1] and Matthew L. Clark [2,*]

[1] Department of Engineering Science, Sonoma State University, 1801 E Cotati Ave, Rohnert Park, CA 94928, USA; daniel.giudici@gmail.com

[2] Center for Interdisciplinary Geospatial Analysis (CIGA), Department of Geography, Environment and Planning, Sonoma State University, 1801 E Cotati Ave, Rohnert Park, CA 94928, USA

* Correspondence: mateolclark@gmail.com; Tel.: +1-707-664-2558

Academic Editors: Qi Wang, Nicolas H. Younan, Carlos López-Martínez and Prasad S. Thenkabail
Received: 10 May 2017; Accepted: 14 June 2017; Published: 20 June 2017

Abstract: In this study, a 1-D Convolutional Neural Network (CNN) architecture was developed, trained and utilized to classify single (summer) and three seasons (spring, summer, fall) of hyperspectral imagery over the San Francisco Bay Area, California for the year 2015. For comparison, the Random Forests (RF) and Support Vector Machine (SVM) classifiers were trained and tested with the same data. In order to support space-based hyperspectral applications, all analyses were performed with simulated Hyperspectral Infrared Imager (HyspIRI) imagery. Three-season data improved classifier overall accuracy by 2.0% (SVM), 1.9% (CNN) to 3.5% (RF) over single-season data. The three-season CNN provided an overall classification accuracy of 89.9%, which was comparable to overall accuracy of 89.5% for SVM. Both three-season CNN and SVM outperformed RF by over 7% overall accuracy. Analysis and visualization of the inner products for the CNN provided insight to distinctive features within the spectral-temporal domain. A method for CNN kernel tuning was presented to assess the importance of learned features. We concluded that CNN is a promising candidate for hyperspectral remote sensing applications because of the high classification accuracy and interpretability of its inner products.

Keywords: hyperspectral imagery; 1-dimensional (1-D); Convolutional Neural Network (CNN); Support Vector Machine (SVM); Random Forests (RF); machine learning; deep learning; TensorFlow; multi-seasonal; regional land cover

1. Introduction

Land-cover maps provide information for natural resource and ecosystem service management, conservation planning, urban planning, agricultural monitoring, and the assessment of long-term land change. The automated classification of land cover from satellite imagery is a challenging task due to spectral mixing, intra-class spectral variability, and low spectral contrast among classes. Hyperspectral, or imaging spectroscopy, data consist of hundreds of spectral bands, and capture more spectral detail and variability relative to conventional multispectral sensors used for mapping land cover. Terrestrial hyperspectral applications have shown success in mapping composition, physiology, and biochemistry of vegetation, ecosystem disturbance, and built-up environments [1]. The analysis of hyperspectral data presents issues in classification, due to large data volumes, and increased spectral variability as recorded by hundreds of correlated bands [2]. Additionally, the classification task becomes more

difficult when presented with the larger spatial extents and temporally detailed data collected by spaceborne hyperspectral sensors with repeat measurements.

A traditional classification method for hyperspectral imagery involves Multiple Endmember Spectral Mixture Analysis (MESMA), which consists of unmixing image spectra with pure spectral profiles (endmembers), and assigning the class through endmembers selected in the unmixing solution [3]. This family of methods typically requires regionally specific libraries of pure spectral profiles that are from field spectra, synthetically generated, or selected from large libraries of image spectra [4]. This type of classification is the most closed form solution available currently, analytically processing an exhaustive combination of endmembers that most closely match the data to be classified [3]. Further, MESMA assumes linear mixing, which is often violated by the interaction of photons with components within an individual pixel and from nearby pixels.

Machine learning is an alternative domain of classification techniques that can accurately distinguish land cover in hyperspectral and multi-seasonal imagery [5]. In contrast to MESMA, these classifiers can learn non-linear decision spaces and do not require training data optimization steps to select spectrally pure endmembers. There are many different varieties of machine learning algorithms implemented on different platforms. Picking the classifier to analyze a dataset at times falls to the user's familiarity with the computational platform and/or algorithm for a specific field. Random Forests (RF) and Support Vector Machines (SVM) are widely adopted machine learning classifiers in the remote sensing community [6,7]. These algorithms have provided robust results across many platforms and datasets, surpassing many other families and implementations of classifiers [8].

Convolutional Neural Network (CNN) is a leading machine learning classifier for image recognition tasks that use 2-dimensional (2-D) image data [9,10], such as identifying faces in photographs of people. Applications of CNN have extended into the classification of other contiguous data types, like speech recognition utilizing 1-dimensional (1-D) data [11]. In remote sensing, there have been several recent applications of CNNs and other similar "deep" network topologies [12–18]. The form in which CNNs are applied to remote sensing data can vary significantly depending on the data available, as there is no universal deep network classification architecture. The application of a CNN to classify land cover from 2-D visual images has been performed with good results in their respective applications. For example, Kussel et al. [15] achieved a 95% overall accuracy with land-cover classification and Li et al. [19] had a 96% correct detection of plants within a scene. In some applications, segmentation and previously learned features can be transferred and leveraged as part of the CNN classification task [12,20–22]. The method within [18,23,24] applies down-selected spectra to a 2-D CNN architecture, thereby mainly exploiting the spatial extent of the data. The spectral dimension is reduced because large number of features present with hyperspectral imagery typically poses a problem for some classifiers, and many studies use dimensionality reduction (DR) to aid in the classification process [13,18,23–26]. In [23–25] the spectral dimension was reduced, for example by applying principal component analysis [23,24], comparing salient band vectors in a manifold ranking space to consider hyperspectral data structure [25], or clustering similar bands and extracting features [26]. All these methods reduce the number of spectral bands before subjecting the data to the classifier. The method provided in [6] pre-calculates features based on physical and chemical composition. All these DR methods have shown an increase in classification accuracy as compared to using the full hyperspectral data. In contrast, some studies have shown increased classification accuracy when utilizing the full spectra, invalidating the need for complex DR preprocessing. For example, the work done in [13] shows the comparison between a 3-dimensional (3-D) CNN, which utilizes the full spectral dimension, and other methods that do not utilize the full spectral dimension. The 3-D CNN spanning the full spectral dimension increased accuracy by up to 3%. In this case, the CNN classifier determines what are the distinctive features from the initial data, without a DR pre-processing step. As another example, Hu et al. [17] utilized single-season hyperspectral data and 1-D CNN across the full spectral dimension to classify land cover with 90 to 93% overall accuracy, and CNN outperformed SVM by 1 to 3%. A distinction between the extraction of key spectra by

using DR and spectral feature generation from the CNN is that the CNN approach extracts features based on spectral characteristics directly driven from reducing classification error. This makes CNN a promising method for the exploitation of the distinctive aspects of the spectral dimension of the data and warrants further investigation with hyperspectral land-cover mapping applications at various spatial and temporal scales.

A broad goal of this study is to assess the accuracy of a 1-D CNN for classifying land cover from multi-seasonal hyperspectral imagery. Accuracy from this CNN is compared to those from the two leading machine learning classification methods in remote sensing, RF and SVM. These methods were utilized as a control group due to their high accuracy rates, high prevalence within the field and the robust libraries available for their implementation. In order to support applications based on spaceborne hyperspectral imagery, our analyses were performed with simulated Hyperspectral Infrared Imager (HyspIRI) imagery, a satellite mission currently being considered by NASA. Our analyses are regional in scale, covering the San Francisco Bay Area, California, and land-cover classes followed the global Land-Cover Classification System (LCCS) [27].

A CNN architecture requires the data to be in a contiguous format as convolutional layers of the network distinguish, or filter, local features or patterns from neighboring regions throughout the data. The neural network then performs the subsequent classification based on these learned features. A potentially useful feature of CNN architectures is that the inner resulting data products, such as properties of convolutional filters, can provide some insight into what the classifier has learned to make its classification. The inner data products of CNN architectures as applied to 1-D hyperspectral data have not been discussed in the literature. Thus, another goal of this study is to show how the inner data products of CNN can provide insight into the classification task and the features extracted from the spectra through the training of the network. With the analysis of feature maps that the convolutional layer of this network creates, local regions within the spectral dimension of the data that are excited by the convolutional layer of the network can be shown to have an impact on the classification accuracy. The importance of these learned features can be then traced back to how important they are to the classification task, by zeroing them from the network and re-computing the accuracy. Additional processing of the feature maps extracted from the CNN enables an illustrative visual that reveals which spectral areas assist in separating the classes. By standardizing, scaling and capturing only the magnitude of the convolutional kernel feature maps, the spectral-temporal band importance can be visually explored.

2. Materials and Methods

2.1. Study Area

The study area was the San Francisco Bay Area in northern California, USA (Figure 1) and is described in Clark and Kilhman [6]. Natural vegetation in the study area includes evergreen needleleaf forests (conifer), evergreen broadleaf forests, deciduous broadleaf forests, mixed forests, shrublands and grasslands. Anthropomorphic land cover includes dense urban areas around the San Francisco Bay and perennial crops (e.g., vineyards, fruit orchards) and annual crops (e.g., strawberries, cotton, rice).

2.2. Simulated HyspIRI Imagery

Hyperspectral imagery for the ~30,000 km^2 study area was from NASA's Airborne Visible Infrared Imaging Spectrometer (AVIRIS) "Classic" sensor, flown in spring, summer and fall of year 2015 [1]. The AVIRIS-C sensor images spectral radiance in 224 bands from 370 nm (visible) to 2500 nm (shortwave infrared, SWIR) with 10 nm sampling and SNR of >1000:1 at 600 nm and >400:1 at 2200 nm [28,29]. There were twelve, ~12 km swath flight runs per season that provided 20% image overlap among runs. Complete spring and fall runs were used while different dates were used to make a complete, cloud-free summer dataset (Table 1, Figure 1).

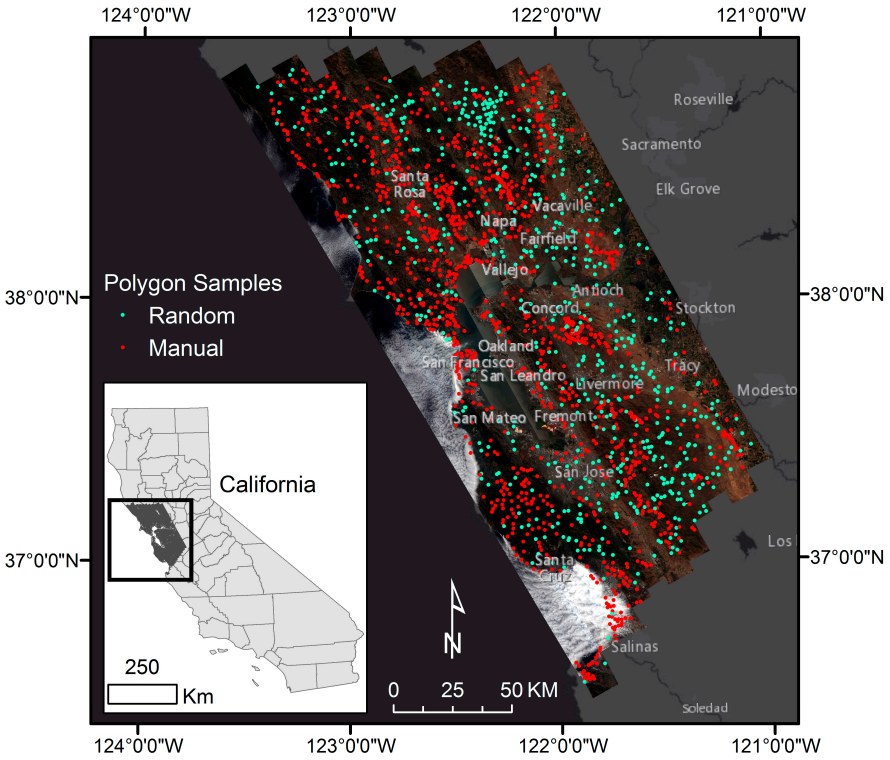

Figure 1. Study area overview with an AVIRIS-C, 11 June 2015 RGB mosaic of 12 individual flight runs. Reference data are red and cyan points. Inset shows the multi-seasonal image extent with water and cloud mask applied.

Table 1. Summary of San Francisco Bay Area AVIRIS-C images for the year 2015.

Season	Image Collection Dates (UTC)	Runs Used	Average Solar Zenith	Average Solar Azimuth
Spring	30 April (16:48–21:25)	Runs 06–17	26.7°	147.0°
Summer	11 June (18:30–23:42) 15 June (21:32–22:51)	Runs 09–20 Runs 12, 15	20.1° 33.3°	227.2° 254.5°
Fall	2 October (17:19–22:32)	Runs 06–13; Runs 16–20	42.2°	191.9°

Hyperspectral images from AVIRIC-C were used to simulate HyspIRI as part of a preparatory science campaign [1,6]. The current configuration of HyspIRI is a satellite sensor with 30-m spatial resolution, 185 km swath width, and 16-day repeat global coverage [1]. The measurements would cover 380 to 2510 nm in ≤10-nm contiguous bands (~214 bands). The AVIRIS-C radiance data were processed by NASA's Jet Propulsion Laboratory into HyspIRI Precursor Data Products (Figure 1) and can be downloaded at http://aviris.jpl.nasa.gov. The 30-m simulated HyspIRI products include, in order: orthrorectification and 90 × 90-m Gaussian-weighted resampling of at-sensor radiance to 30 m pixels, addition of noise approximating a HyspIRI visible-SWIR Noise Equivalent Delta Radiance (NEdL) function, and ATREM-based per-pixel atmospheric correction and apparent surface reflectance retrieval [6,30]. Bands in the shortwave infrared with strong atmospheric water vapor absorption and poor signal-to-noise were removed, leaving 186 bands for analysis per season. Clouds and water in simulated images were masked prior to image classification using a Random Forests classifier developed in Clark and Kilham [6]. The analysis of multi-season data was performed on an image cube of seasonal spectral data stacked in temporal sequence (spring, summer, fall), resulting in an input vector of 558 bands; $N_d = 186\ (bands)\ \times\ 3\ (seasons)$.

2.3. Land-Cover Reference Data

Reference data for training and testing were collected using visual interpretation of high-resolution imagery in Google Earth. Detailed methods are found in Clark and Kilham [6] and summarized below. Data were percent cover of the following twelve "land-cover components" within a polygon: evergreen needleleaf trees (ENT); evergreen broadleaf trees (EBT); deciduous broadleaf trees (DBT); shrubs; herbaceous; dune vegetation; tidal marsh; annual crops; perennial crops; impervious surfaces; urban landscape; and, bare non-vegetated (beaches, dunes, rocks, bare soil). Percent cover data were visually estimated in 10% intervals using high-resolution Google Earth imagery in 100-, 250- or 500-m square polygons. Different polygon sizes were chosen depending on patch size in order to maximize pixels collected in large patches (e.g., 500-m square chosen), while minimizing mixed pixels in relatively small patches (e.g., 100-m square chosen). Polygons were located over areas of well-mixed land-cover components, with most centers further than 1000 m to a neighboring sample. Initially, a simple random method was used to locate samples, which led to under-represented samples of some classes. This necessitated manual placement to ensure an adequate sample of class types (Figure 1). Each polygon was classified into one of twelve discrete Land-Cover Classification System [27] classes using a decision-tree of rules applied to a polygon's percent cover data (Table A1; [6]). Open-canopy trees (woodlands) and shrubs (shrublands) have >10–65% tree or shrub cover, respectively; closed-canopy trees (forests) and shrubs (thickets) have >65% tree or shrub cover, respectively. In this study, we focus on mapping closed-canopy tree and shrub classes.

There were 1495 total reference polygons for the twelve LCCS classes. Reference reflectance spectra were extracted from polygons overlaid on the three-season image cube for each run (Figure 2). To exclude potentially mixed pixels at edges and minimize geolocation error, polygons were buffered inward using a half-pixel buffer, and only pixels that were 100% contained by the buffered polygon were selected. Due to scene overlap, there were some polygons that were located in two images. In these cases, all available pixels were extracted. Reference data were split at the polygon level into training and testing sets using the same polygon designations found in Clark & Kilham [6], except polygons in areas of fires between years 2013 and 2015 were removed from the analysis. Training spectra were then filtered at the polygon level to remove outliers following methods in Clark & Kilham [6]. Single- and three-season data were processed for outliers separately for each group of variables. There were a total of 71,362 training pixels and 52,510 testing pixels (Figure 3).

2.4. Classifier Architectures and Tuning

Three machine classifiers were assessed in this work: Random Forest (RF), Support Vector Machine (SVM) and Convolutional Neural Network (CNN). We used Python's Scikit-learn machine learning library for training and classification. This library provides access to established data mining and data analysis tools and algorithms for implementing RF and SVM. The adaptations of the respective CNN architectures require a relatively flexible platform for development. A CNN is a deep network topology that can be formed as a computational graph and Google's open-source computational platform, TensorFlow™, is directly suited for this application. In this study, we used TensorFlow™ for the CNN definition, training and classification portion of this work. This platform had the additional advantage of using the GPU to speed data processing, whereas Scikit-learn relied solely on the CPU.

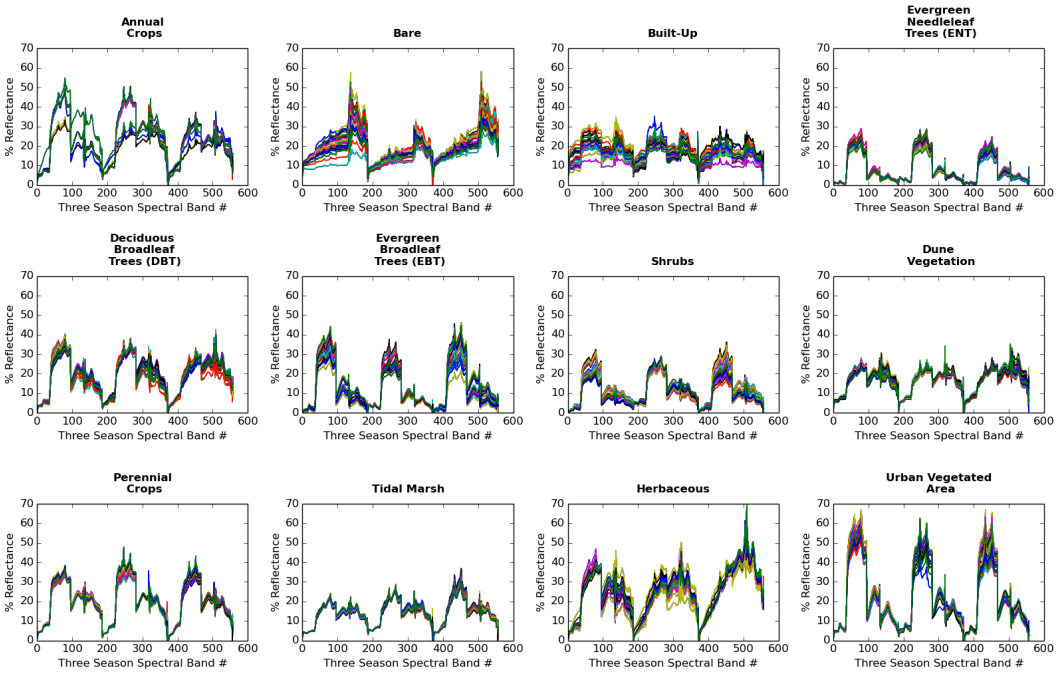

Figure 2. Twenty-five randomly selected three-season reflectance spectra from each of the twelve LCCS classes. Note that the x-axis is the band number (1–558) with seasons in spring- summer-fall sequence. Bad bands have been removed within each season (186 bands per season).

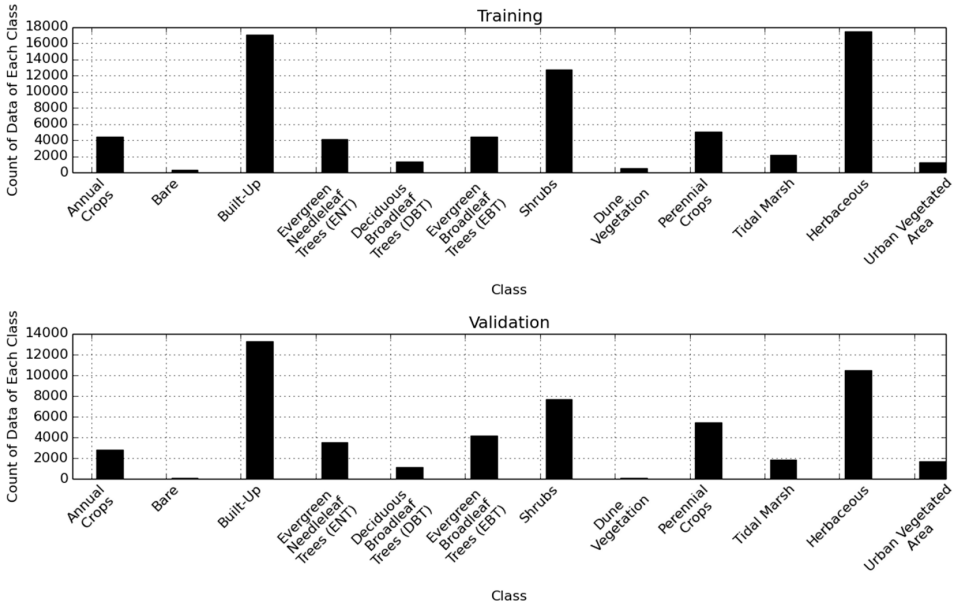

Figure 3. Training and testing reference data distributions.

Below we provide a brief background and explanation of the RF, SVM and CNN classifiers and their respective hyper-parameters. An overview of the method is shown in Figure 4. Training and testing reference data included duplicate pixels from areas of scene overlap, thus providing spectral variation for differences in sensor view and sun angle in the classification process. We did not have a validation dataset due to the limited number of samples of some classes within the data. Training data were thus used to train the classifier, while hyper-parameter tuning was performed by observing training and test data accuracies. The final accuracy assessment was performed by extracting the

most trusted classification from each classifier in the areas of scene overlap. This consists of utilizing data classifications that result from classifications that have the highest confidence for RF or highest probability for SVM and CNN (Section 2.6; Figure 4, LCCS-labeled Land-Cover Samples).

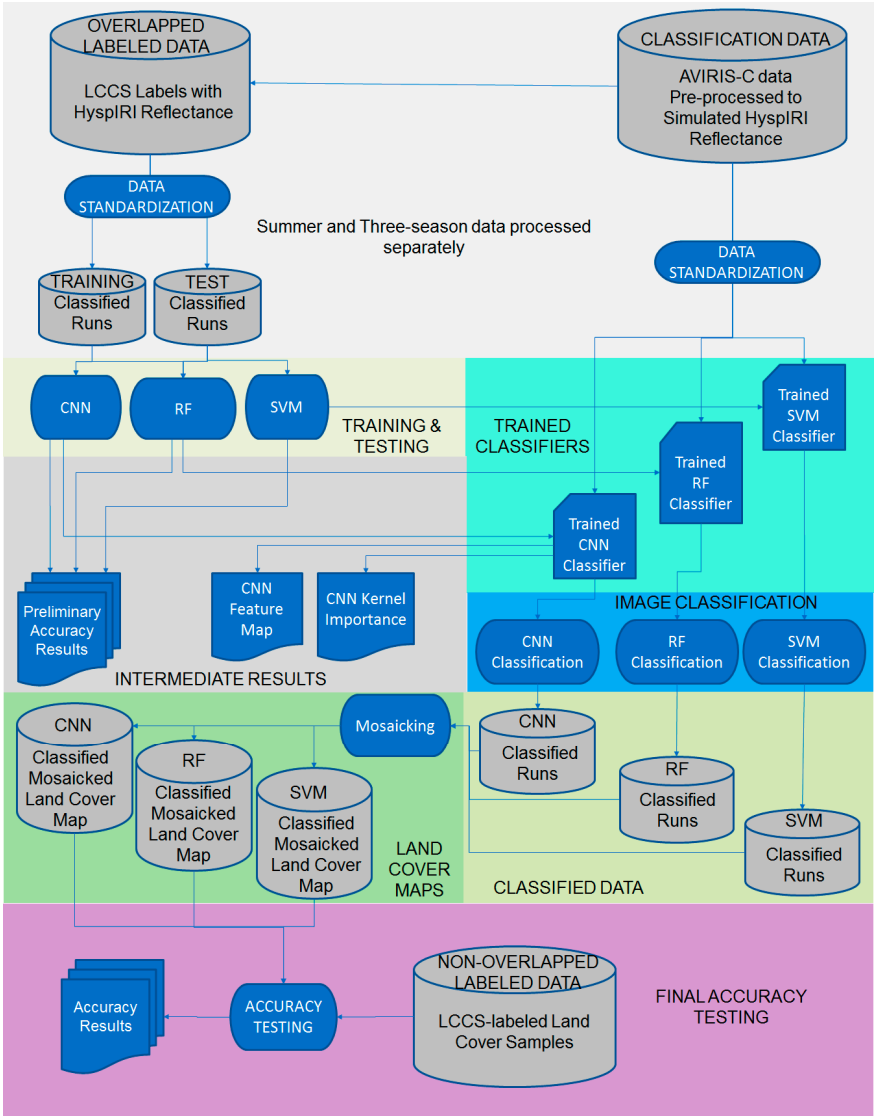

Figure 4. Overview of machine learning classification and accuracy assessment methodology.

2.4.1. Random Forest

The RF classifier is a technique of training an ensemble of decision trees through randomized draws of training data. Once the ensemble of trees is created, data are applied to the classifier and the prediction across the individual trees is captured [31]. The plurality of predictions is the resulting classification of this algorithm.

There are three main parameters when tuning a Random Forest classifier: the number of estimators (or trees), the maximum number of features utilized at each decision point within a tree and the minimum samples at each leaf. As a general rule, the number of features utilized in RF classifiers is $\sqrt{N_f}$, where N_f is the number of features. In hyperspectral imagery, each feature is an individual reflectance band. The number of trees included in the ensemble typically improves classification accuracy until a critical point is reached, beyond which accuracy does not increase further. For the minimum leaf size, smaller values result in RF classifiers that are more prone to capturing noise in the

training data. Optimization of these parameters should be considered when training a RF classifier on a particular dataset [32].

For this work the standard $Max\ Number\ of\ Features = \sqrt{N_f}$ was utilized. The number of trees was increased until there was not a significant change in prediction accuracy, resulting in a parameter of 1000 trees. A search across the minimum leaf sizes was performed, resulting in a parameter of 1 sample in leaves.

2.4.2. Support Vector Machine

The SVM classifier separates training data by defining a hyperplane(s) through the data to segment the classes. In two dimensions, this can be visualized as a line drawn between two classes that defines the maximum separation in spectral space. For non-linear and high dimensional classification tasks, the data are mapped to an even higher dimensional space and the "line" that is drawn between the classes is, in effect, a plane which creates the most separation between the classes. How this mapping to the higher-dimensional space is performed is dictated by hyper-parameters. A standard linear Support Vector Classifier that utilizes a one-vs.-rest classification scheme was utilized for this work. This implementation simplifies the tuning of the classifier down to one parameter, cost (C). This parameter accounts for the trade-off between misclassification of training examples and simplicity of the decision surface. A low C makes this decision surface smooth, while a high C aims at classifying all training examples by giving the model freedom to select more samples as support vectors [33]. A linear search for this tuning parameter was performed from 1.0×10^{-2} to 1.0×10^{10} and the highest accuracy classifier was chosen; a C value of 0.1 was utilized for this work.

2.4.3. Convolutional Neural Network

At a broad level, a CNN is a deep-network topology that typically combines convolutional filter layers in conjunction with a classification network, which for this work is a fully connected Neural Network (NN). Through the standard back-propagation training process, convolutional filters are trained to capture salient structural feature information from the sequential input data. Hu and colleagues [17] mention these structural features as the "intraclass appearance and shape variation" within spectra. As an extension from this previous literature, here we demonstrate that these structural features actually represent those features present within the data that distinguish the classes from each other. The architecture feeds these features, or filtered input data, into a subsequent classification process.

A flow diagram of our CNN process is shown in Figure 5. For some datasets, like those consisting of 2-D imagery, larger networks are required.

CNN Architecture

The feature generation and feature classification nature of CNN can become complex with multiple layers that extract different levels of features from the data. A few examples of more complex CNN architectures as implemented for image classification tasks can be reviewed in [9,10]. These complex networks required the learning of a large number of features and a high level of neural network complexity. These traditional CNN architectures are implemented in two dimensions (i.e., 2-D, width and a height). The convolutional operation for these networks occurs with multi-dimensional kernels and in various configurations, with the goal to classify the full or subsets of the image. In contrast, hyperspectral data classification can be a pixel-based operation spanning only the spectral dimension (i.e., 1-D), or in the spatial and spectral dimensions (i.e., 3-D). To simplify implementation and compare results to RF and SVM, we chose to apply the CNN across the spectral domain of hyperspectral pixels; and thus, the convolutional operation of the network only needs to operate on this single dimension. This in turn greatly simplifies the architecture of the hyperspectral CNN relative to 2-D or 3-D implementations [9,10,13,15].

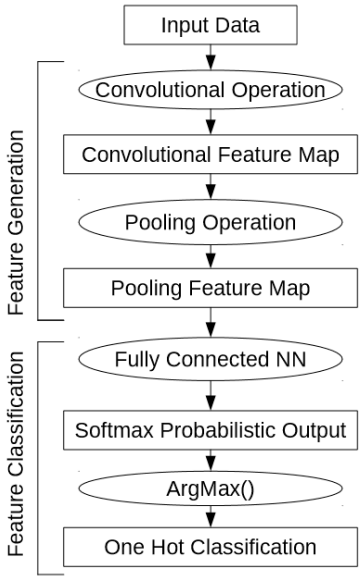

Figure 5. Convolutional Neural Network (CNN) flow diagram.

Our CNN architecture resembles a network that has been recently introduced for hyperspectral data classification in [17]. The network consists of a single convolution layer paired with a pooling layer that feeds a fully connected neural network. Our network was further refined by adding Dropout and Regularization of the fully connected layer [34,35]. Additionally, here we tune the convolutional layers based on a technique that reduces the number of kernels trained within the network by accounting for their impact on the classifier accuracy.

The single convolutional layer accepts the 1-D spectral profile input data and performs the convolutional operation on the input data with each kernel in the architecture (Figure 6). This filtering of input data with each kernel creates the features for classification. In our implementation of CNN, the number of initial kernels to be trained was set at 86, the number of single-season engineered metrics used in our previous research with these hyperspectral data and the RF classifier [6]. In that study, engineered "spectral metrics" targeted vegetation biochemical and structural properties found in reflectance spectra, and perform a similar function as the kernels for this network. Because the learned features within the CNN are not tied to any pre-selected spectral features, rather only spectral properties that best distinguish the classes, the number of kernels in the CNN was drastically reduced by evaluating kernel importance (Section 3.2).

The pooling layer can be thought of as a spectral down-sampling of the convolutional feature map (Figure 6). A Max Pooling operation was utilized here. This layer accepts the convolutional feature map, evaluates pairs of data elements across the spectral dimension of the feature maps and passes the maximum value onto the next layer. This down-samples the data by a factor of two while preserving the maximum excitations from the convolutional feature map. This operation reduces the size of the feature map while preserving the features observed within the convolutional feature map. This provides a level of invariance to the spectral location of feature excitation within the feature generation network and reduces the overall number of connections to the fully connected network; and subsequently, this process will decrease the total number of trainable parameters for the network, thereby reducing overall training time.

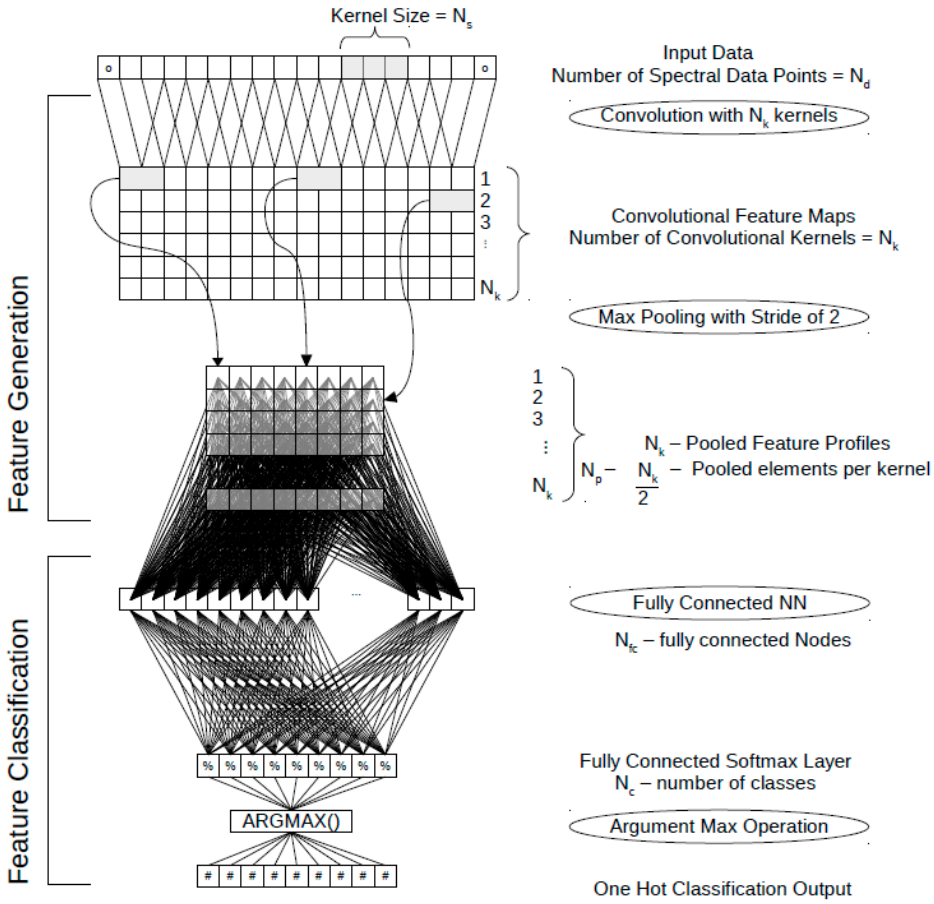

Figure 6. Detailed Convolutional Neural Network (CNN) architecture.

This Pooled Feature map is then provided to the feature classification network (Figure 6). In this architecture the feature classification network consists of a fully connected neural network. This fully connected network layer consists of a hidden layer with 500 nodes. This network's learning is regularized by including standard neural network regularization techniques; specifically, dropout with a 50% dropout level and L2 regularization on the weights connecting the pooled feature map and the fully-connected NN's hidden nodes. These two techniques enable the classification network to learn information throughout the fully connected network and encourage "smaller" weights to be utilized within the classification network. This ensures that individual weights between the two networks (feature generation and feature classification) are not exorbitantly larger than the other weights within the same layer, helping to prevent overfitting of the training data. By applying a penalty for large weights at this layer, the network is encouraged to learn information through all of the kernels and not just a few of them.

The output of the hidden layer is connected to a final Softmax output layer that produces a probabilistic output per class, or a vector of length of the number of classes, with each value representing the probability that the input data belongs to a specific class (Figure 6). This probability or confidence that the Softmax layer calculates is shown in Equation (1).

$$\sigma(z)_j = \frac{e^{z_j}}{\Sigma_{k=1}^{K} e^{z_k}} \; for \; j = 1, \ldots K \tag{1}$$

where Z_j are the inputs from the previous fully-connected layer applied to each Softmax layer node and K is the number of Softmax layer nodes (i.e., the number of classes). Finding the location

of the argument with the largest probability value via the Argmax() function provides a one-hot representation of the class (Figure 6).

Stochastic gradient decent was utilized for training of the CNN. During training, this learning method subjects randomized labeled data to the network and calculates a loss function. This loss is then propagated back through the network to modify the network's interconnections, based on the impact the respective weights had on the previous epoch's classification. This is the gradient of the network, and is essentially the degree to which the weights of the network have impacted a given classification and resulted in a respective loss. Modifying the network based on the loss calculated by the classification or misclassification of the labeled data is effectively the back-propagation algorithm. As training progresses, the back-propagation modifies the interconnection of the network to reduce the loss function. This decreasing of the loss function indicates the classified data more closely matches the labeled data. Learning curves are provided in Appendix Figures A2 and A3 for the training and testing data, respectively, and show overfitting is not occurring within the testing data.

The TensorFlow™ platform has built-in implementations for calculating the gradients of the network and the minimization of the loss function to be back-propagated through the network. The Adagrad adaptive learning rate algorithm was utilized from the TensorFlow™ platform. The usage of an adaptive learning rate provided a significant boost to the performance of this architecture on the order of 4% overall classification accuracy. Additionally, the Adagrad adaptive learning rate greatly sped up the training of the network as compared to fixed and the exponential decaying learning rate optimizers. The Adagrad optimizer adjusts the learning rate, the extent the network can be modified, based on the gradient recently seen by the network. As this optimizer is suited for relatively sparse data and the dataset analyzed here is relatively unbalanced (Figure 3), this optimizer seemed to be an appropriate fit for this architecture. Empirical testing proved that this algorithm reached a maximum accuracy with a seed-learning rate of 0.1.

Hyper-Parameter Tuning

Dominant tunable hyper-parameters of this architecture are: K_s the kernel size, N_k the number of kernels, and N_h the number of hidden nodes within the classification network (Figure 6). While other characteristics can be modified within the network, these three hyper-parameters were determined to be the most influential. Other parameters such as dropout level and regularization level on the hidden weights could be adjusted, but our experience was that just having these parameters defined would increase the accuracy of the network. We used values of 50% dropout and 1×10^{-4} for regularization.

The kernel size dictates the size of the feature to capture. It is the size of the local receptive field considered within the spectral dimension for the convolution with the data. A rule of thumb originating from the currently engineered features utilized in [6] is to use local receptive fields that are roughly 10 bands long (10 bands $\times$ 10 nm spacing = 100 nm) for a single season. Informal experiments indicated that K_s could change in size as long as the Number of Kernels, N_k, varied inversely with this value. We believe that this is because the modification of these two parameters effectively varies the capacity of the network to learn the data, as long as the network is sufficiently able to learn the parameters equivalent accuracy should be able to be achieved. Due to familiarity with local receptive fields of 10 in prior work this was used as the kernel size in our CNN implementation.

The number of kernels contained within the network represents the number of features able to be learned. As the capacity of the network to learn features is a combination of the number of kernels and the kernel size. If each of those features are very descriptive, and has a large kernel size, then it would be expected that fewer of them would be required to achieve similar results. With the kernel size set, it was determined that the number of kernels can be reduced until zeroing a kernel from the classifier always has negative effect on overall accuracy. This ensures that the feature maps with respect to each kernel are excited and that each kernel is important to increasing overall accuracy.

Evaluating Kernel Importance

A main goal of training deep network topologies is to appropriately load the weights of the network. If a network has too many nodes or has too many layers, it may not effectively learn the appropriate weights for its interconnections, in effect not learning the decision space. As applied to CNNs, too many kernels can produce excess capacity that does not contribute to an increase in overall accuracy of the classification. With that in mind, we developed a way to verify that each kernel contributes to an increase in accuracy. CNNs can be evaluated with a similar method to the Mean Decrease Accuracy utilized in RF to determine feature importance. By zeroing out each convolutional filter or kernel and assessing the effect on the overall accuracy, information on how important that kernel is to the classification task can be determined. Kernels with significant impact on accuracy can be regarded as containing information of distinguishing characteristic(s) of the data.

In our study, the number of kernels trained was reduced to ensure that all the kernels were important to some degree. Each time the number of kernels was reduced the network was retrained with the training data to appropriately learn the new kernels. It was determined that the number of kernels trained within the network could be reduced until the Kernel Importance Table always showed a negative average percent impact on the classification accuracy for all kernels (Section 3.2). Reducing the number of kernels beyond this started to have a negative impact on the overall classification accuracy. While this was determined experimentally, it proved to be useful in reducing the total size of the network. Additionally, as the learned information is more condensed in fewer trainable kernels, the features that each kernel is extracting should be more expressive. With this kernel reduction performed, we expect that the network does not have too much excess capacity and the likelihood of over training is minimized. In this study, we started with 86 kernels and were able to reduce this drastically. Acceptable parameters for a three-season CNN were: $N_k = 7$ and $K_s = 10$. These parameters were also applied to the single-season CNN.

Hidden Layer Nodes

The number of hidden layer nodes within the classification portion of the network determines the capacity of the classification network to make an accurate "combination" of the features learned. It is important that this network has enough capacity with the upper limit being mainly to keep the size of the network only large enough to not limit the classification accuracy of the network. Here we chose 500 hidden layer nodes, N_h.

Convolutional Filter Visualization

Useful visualizations for CNN are the feature maps generated by the network. Feature maps are created by applying individual pieces of data to the network and extracting the resulting convolutional excitations for each class of data. When a single piece of data is applied to the network, every kernel is convolved with this data point. This creates the $N_d \times N_k$ convolutional feature map. Visualizing this convolutional feature map enables the identification of where within the temporal-spectral signature the data is being excited or filtered by a respective kernel. This is the information that is provided to the feature classification network.

As feature maps are connected to the same spectral locations within the hidden layer, if the feature maps from each of the classes all are excited in the same area then those excitations do not provide any distinctive information to the classification network and would have little effect in determining the class of the data. Thus, for visualization purposes the mean across all class feature maps was removed from individual class feature maps. As any deviation from zero within the modified feature map indicates that the kernel learned something in that region, viewing the magnitude of the mean-removed feature map provides a more useful view. Additionally, averaging feature maps from 75 spectra of the same class provides the average excitations from convolution with those data, showing general trends for each class. The resulting visualization feature maps can then shed some light on the distinguishing

characteristics of the class, on a class-by-class basis. To provide context for the feature map, a random piece of data from the feature map's class is provided as a silhouette. This was done to illustrate the typical structure of that class's spectral profile across all three seasons. An example is provided in Figure 7 for Annual Crops.

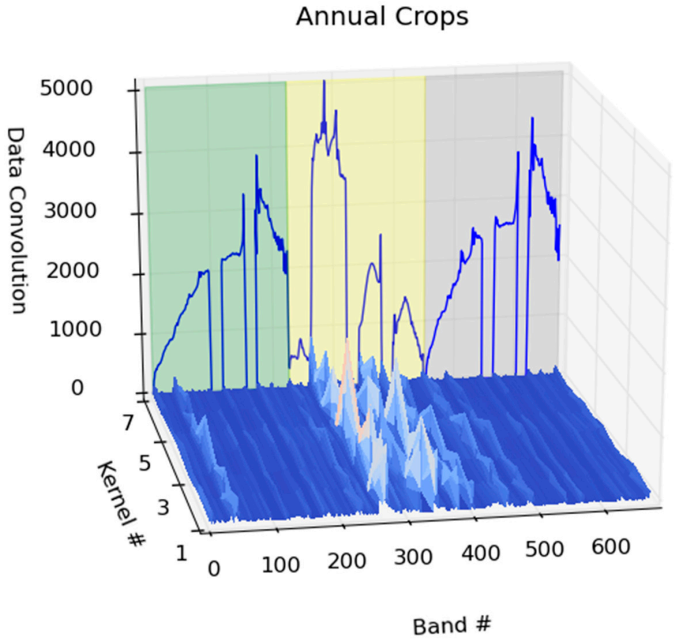

Figure 7. An example convolutional feature map for an Annual Crop three-season spectrum. The example class spectrum is shown in background. Green, yellow and gray background areas represent bands in the spring, summer and fall, respectively. Each season has 186 bands, ordered 370 to 2500 nm. However, in this figure the original 224 bands per season are shown in order to display the bad data gaps (e.g., atmospheric absorption windows).

2.5. Data Pre-Processing

Standardizing data for neural networks is a common practice and enables the network to operate on data that has a similar dynamic range [36]. This frees the network from having to "learn" this dimension or characteristic of the data. The zero mean and standardization of the data with respect to spectra effectively removes common structural content from the data and then scales the data to have an appropriate dynamic range on a per spectra basis. The formula below is the operation performed for this standardization for each spectral band.

$$X_{i\_newspectra} = \frac{x_{i\_rawspectra} - \mu_{i\_rawspectra}}{\sigma_{i\_rawspectra}} \; for \; i = 1, \ldots N_d \tag{2}$$

where N_d is the number of spectra or data, $\mu_{i\_rawspectra}$ and $\sigma_{i\_rawspectra}$ are the mean and standard deviation of all the training at the spectral index i.

This standardization of the data boosted performance of the CNN roughly 1% in classification accuracy and made the training more stable. These same standardized data were applied to RF and SVM but did not make any noticeable impact on accuracy of these classifiers (Figure 4). As per standard practice, the mean and standard deviations for this standardization were developed from the training dataset and then extended to condition the testing and classification image data. This is to prevent standardization knowledge of the testing data to influence the training of the classifier.

2.6. Classification Post-Processing and Accuracy Assessment

The final classified LCCS map products were created from mosaics of classified runs from respective classifiers (Figure 4). In areas of scene overlap, the map class was determined by selecting the pixels with the maximum number of votes (RF) or highest probability (CNN and SVM). An accuracy assessment was conducted using confusion matrices, overall percent accuracy and kappa statistics based on respective classifiers applied to independent test data. In this case, test data did not include duplicate pixels in areas of scene overlap; that is, the class from the respective mosaicked map was chosen as the reference class in areas of overlap.

3. Results

3.1. Accuracy Assessment

With single-season (summer) data, CNN had 0.5% and 9.3% significantly higher overall accuracy (OA) than SVM and RF, respectively ($Z > 3.2$, $p < 0.01$, Table 2). With three-season (spring, summer, fall) data, CNN had 0.3% and 7.3% significantly higher OA than SVM and RF, respectively ($Z > 2.1$, $p < 0.05$, Table 2). Overall accuracy for three-season data was 1.9 to 3.5% significantly higher than single-season data for all three classifiers ($Z > 2.1$, $p < 0.05$). Given their superior performance, remaining results will focus on three-season CNN and SVM classifications.

Table 2. Overall percent accuracy per classifier using single (summer) or three seasons (spring, summer, fall) of data. Kappa statistics are provided in parentheses.

	CNN	SVM	RF
Single Season	88.0 (0.86)	87.5 (0.86)	78.7 (0.75)
Three Season	89.9 (0.88)	89.5 (0.88)	82.2 (0.80)

The three-season CNN map had average producer accuracies (PA) of 76.7%, and that ranged from 0.0 (Dune Vegetation) to 98.3% (Built-up, Table 3). The CNN map user accuracies (UA) averaged 80.0% and ranged from 0.0 (Dune Veg.) to 97.9% (Tidal Marsh). The SVM map had average PA of 73.5%, and that ranged from 0.0 (Dune Vegetation) to 98.8% (Built-up, Table 4). The SVM map UA averaged 80.0% and ranged from 0.0 (Dune Veg.) to 97.9% (Tidal Marsh). The class accuracy for Dune Vegetation was 0.0% due to low class sample size (n = 132 pixels) and spectral confusion with Herbaceous, Annual Crops, Bare, and DBT (Tables 3 and 4). Being located along the coast, Dune Vegetation was not prevalent in the study area and was thus underrepresented in reference data. Annual Crops had 12.3% greater PA with CNN over SVM, with confusion in both classifiers spanning Perennial Crops, Herbaceous and Urban Vegetation. Bare had a 35.3% greater PA in CNN over SVM, with greater confusion with Annual Crops and Built-up in SVM. In both classifiers, deciduous broadleaf forests (DBT) tended to be confused with evergreen broadleaf forests (EBT). Evergreen broadleaf forests were confused with conifers (ENT), deciduous forests (DBT), and shrubs.

The mosaicked classified maps created with RF, CNN and SVM are shown in Figure 8. At this scale, it is visible that all classifiers map land cover relatively well, and are relatively consistent throughout their respective classification. In general, the RF map tended to be more speckled than SVM or CNN maps. The RF map had more obvious problems with classifying shrublands in the southeast corner of the map (Figure 8B), with over-mapping of Perennial Crops; RF also had over-mapping of Built-up in the Tidal Marsh areas along the northeast bay. Some class differences toward the coast, such as near San Francisco, are explained by clouds in the imagery that were not fully masked from the analysis.

Table 3. CNN confusion matrix for the three-season classification.

	Annual Crops	Bare	Built	ENT	DBT	EBT	Shrubs	Dune Veg.	Perennial Crops	Tidal Marsh	Herbaceous	Urban Veg.	Total	User
1. Ann. Crops	3513	62	35	0	0	0	0	45	394	110	41	52	4252	83%
2. Bare	3	479	30	0	0	0	0	22	16	5	1	10	566	85%
3. Built-Up	33	106	13,210	4	2	0	5	0	70	77	6	617	14,130	93%
4. ENT	1	0	2	11,101	0	302	9	0	17	3	0	2	11,437	97%
5. DBT	0	0	0	2	1032	431	16	0	2	4	20	3	1510	68%
6. EBT	2	0	2	394	369	5297	100	0	1	2	3	4	6174	86%
7. Shrubs	11	0	5	262	24	350	7066	0	4	29	156	16	7923	89%
8. Dune Veg.	3	0	0	1	0	0	8	0	0	6	0	0	18	0%
9. Per. Crops	631	0	9	0	10	6	8	0	5359	29	3	37	6092	88%
10. Tidal Marsh	2	0	0	0	0	0	3	1	0	4930	98	2	5036	98%
11. Herbaceous	876	73	52	1	3	21	52	64	148	467	9357	66	11,180	84%
12. Urban Veg.	57	0	92	8	1	17	0	0	0	32	0	1772	1979	90%
Total	5132	720	13,437	11,773	1441	6424	7267	132	6011	5694	9685	2581		
Producer	68%	67%	98%	94%	72%	82%	97%	0%	89%	87%	97%	69%		

Table 4. SVM confusion matrix for the three-season classification.

	Annual Crops	Bare	Built	ENT	DBT	EBT	Shrubs	Dune Veg.	Perennial Crops	Tidal Marsh	Herbaceous	Urban Veg.	Total	User
1. Ann. Crops	2881	256	15	0	0	0	0	21	226	115	38	6	3558	81%
2. Bare	0	225	7	0	0	0	0	25	0	0	0	0	257	88%
3. Built-Up	83	223	13,272	151	1	2	2	7	61	78	65	575	14,520	91%
4. ENT	1	0	1	11,438	1	411	3	0	0	5	0	4	11,864	96%
5. DBT	18	0	4	0	1047	230	11	67	0	2	8	0	1387	75%
6. EBT	3	0	0	131	351	5351	59	0	11	0	0	2	5908	91%
7. Shrubs	0	0	1	34	17	389	7145	0	1	22	129	3	7741	92%
8. Dune Veg.	25	0	0	6	0	1	1	0	0	0	2	0	35	0%
9. Per. Crops	678	5	19	0	20	9	1	0	5560	61	19	65	6437	86%
10. Tidal Marsh	10	0	10	0	0	0	2	1	0	4856	74	5	4958	98%
11. Herbaceous	1124	11	16	1	4	28	43	11	152	524	9307	109	11,330	82%
12. Urban Veg.	309	0	92	12	0	3	0	0	0	31	43	1812	2302	79%
Total	5132	720	13,437	11,773	1441	6424	7267	132	6011	5694	9685	2581		
Producer	56%	31%	99%	97%	73%	83%	98%	0%	92%	85%	96%	70%		

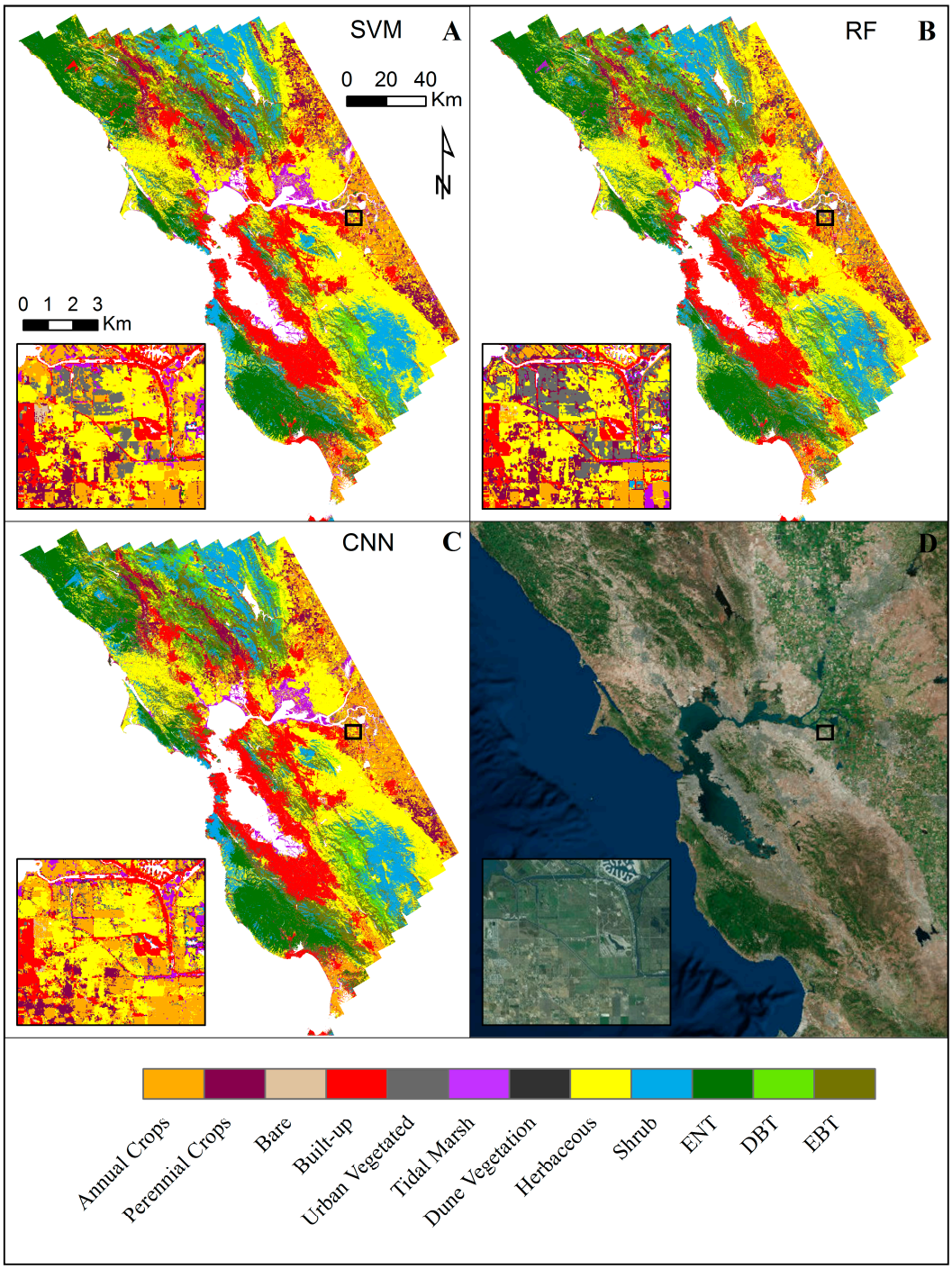

Figure 8. Classified land-cover maps for (**A**) Support Vector Machine; (**B**) Random Forests; and (**C**) Convolutional Neural Networks. White areas indicate pixels that were not classified (e.g., water, clouds, no data); (**D**) Natural color mosaic of imagery from June 2015.

The insets show an example of the classifications in the Bay Area delta agricultural zone. This area is predominantly agricultural land with some urban areas. In this anthropogenic landscape, homogenous geometric shapes are dictated by property lines (Figure 8D). The RF classifier performs the worst with many areas with more speckle (i.e., more heterogeneity) of classes across the landscape. The RF and SVM maps both misclassified several parcels with Urban Vegetated that were mapped as Annual Crops by CNN. This is an area of irrigated sod production, which has temporal and spectral

similarities to irrigated golf courses found in the Urban Vegetated class. Despite this spectral similarity, CNN tended to correctly map these areas as agriculture.

3.2. CNN Kernel Importance and Visualization

The three-season CNN kernel importance matrix (Figure 9) shows the impact of zeroing a kernel from the CNN on class producer accuracy. For example, conifer forests (ENT, class index 4) producer accuracy decreases 95.2% if Kernel 7 is removed from the CNN. With Kernel 4 removed, six classes have a 15.9 to 65.31% decrease in accuracy. In contrast, some kernels have a positive impact if removed from the network. For example DBT (class index 5) and Urban Vegetation (class index 12) producer accuracy increases 6% and 10.8% if Kernel 5 is removed from the CNN. On Average, the most important kernel was Kernel 7 while the least important was Kernel 5.

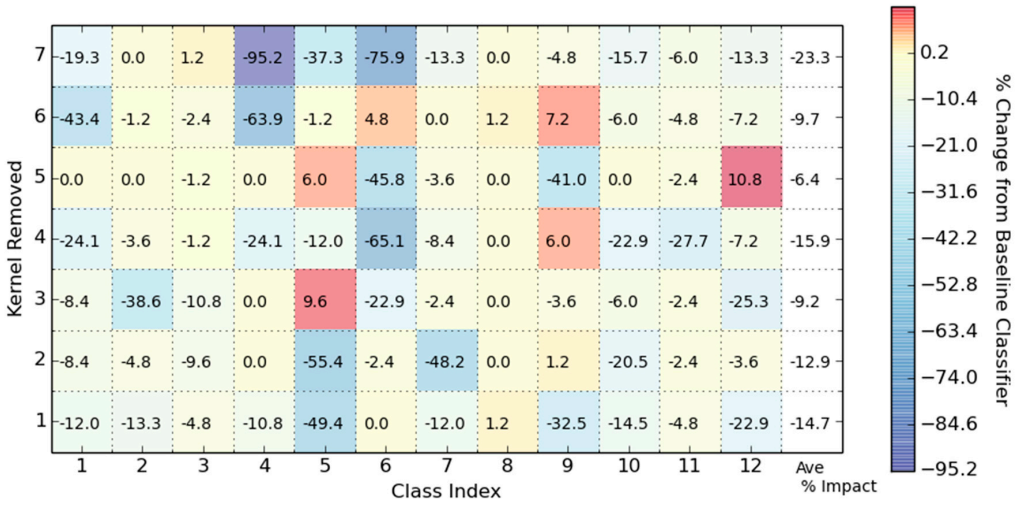

Figure 9. Kernel Importance Matrix. This table shows the percent change in producer accuracy when zeroing a kernel from the CNN. Class index definitions found in Table 3.

3.3. CNN Feature Map Visualization

Convolutional feature maps show, for an example spectrum from each class, kernel excitations subtracted from the average excitation across all classes at a given wavelength (Figure 10). Figure 10 shows the average feature map excitations from 75 randomly selected spectra per class. In general, feature maps show that kernels tend to excite in specific spectral-temporal regions for a given class, forming strips in the feature map. At these regions of activity, the kernels tend to be excited at different magnitudes (e.g., peaks and valleys along a strip).

By observing under which season of the feature map the excitations occur, distinctiveness of phenological spectral variation can be observed within classes. This effectively enables the feature maps to show under which season features are more prominent for respective classes. For example, for the Annual Crop spectrum there is heavy kernel excitation in the summer (Figure 10). The forest spectra (EBT, DBT, ENT) have excitations throughout the seasons. In contrast, Tidal Marsh and Shrubs spectra have relatively low excitation throughout the year.

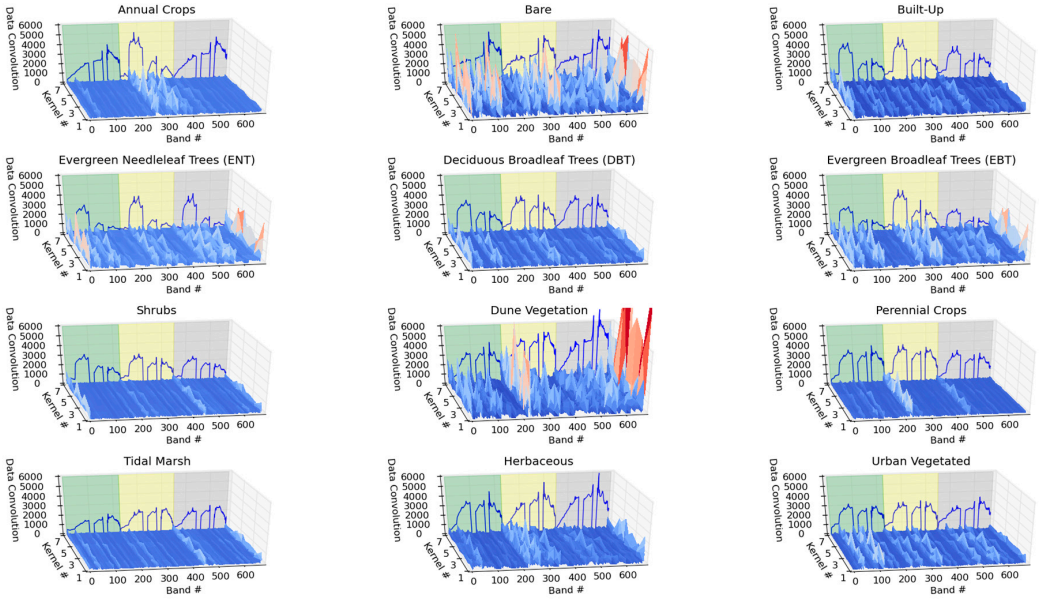

Figure 10. Convolutional feature maps for each class for the three-season CNN. These feature maps are the result of averaging the convolution of the kernels with 75 spectra per class (Section 2.4.3). An example class spectrum is shown in background. Green, yellow and gray background areas represent bands in the spring, summer and fall, respectively. Each season has 186 bands, ordered 370 to 2500 nm. However, in this figure the original 224 bands per season are shown in order to display the bad data gaps (e.g., atmospheric absorption windows).

4. Discussion

4.1. Classified Land-Cover Maps

The classification scheme in this study included twelve classes defined by LCCS rules designed for global-scale applications. Forest classes had closed-canopies of one leaf type (e.g., deciduous or evergreen), and thus excluded more mixed forest or open-canopy classes that have more spectral variability, and subsequent class confusion and decrease in overall accuracy [6]. However, the broad classes include spectral-temporal variability over a large region. This aspect of this study is unique for testing machine-learning classifiers with hyperspectral and multi-seasonal datasets.

We found that the 1-D (per-pixel) SVM and CNN classifiers had over 7% overall accuracy improvement over the RF classifier for these data and a classification scheme. This is similar to a 5% increase in overall accuracy observed for 1-D CNN over RF in a study focused on crop classification in Ukraine with multi-temporal multispectral Landsat and Sentinel-2 images [15]. We found that CNN only had 0.3 (three-season) to 0.5% (summer) greater overall accuracy than the popular SVM classifier. This result is similar to a previous study with 1-D hyperspectral imagery (not multi-temporal) that found CNN to improve overall accuracy by 0.9 to 2.6% relative to SVM [17]. In our study, we found the addition of multi-seasonal data modestly (but significantly) improved classification accuracy by 1.9 to 3.5% over using summer-only data. The Kussal et al. study [15] did not specifically isolate individual seasons for testing multi-temporal data, and to our knowledge there are no studies that compare the relative advantage of temporal data in CNN. However, multi-temporal data have been shown to increase overall accuracy from ~2 to 8% with a variety of classifiers [37]

Our previous research in this study area with 2013 AVIRIS data and the same training and testing reference data was based on the RF classifier [6]. We found reflectance data yielded a 62.9% and 78.8% overall accuracy with single- and three-season data, respectively. The RF classifier in the current study produced 15.8% (single-season) and 3.4% (three-season) higher overall accuracy, respectively. The current study was not designed to replicate our previous research design. Some of the discrepancy

in results is explained by a difference in image years, improvements in the atmospheric correction process, and differences in processing code (Python vs. R). Further, our previous work used a RF node size of 10 to avoid overfitting and to train a more generalized classifier, while the current study used a node size of 1. Despite these differences, our results indicate that the CNN and SVM classifiers offer improved performance relative to RF, and both classifiers should be considered in hyperspectral classification applications.

All classifications in our study tended to have class confusion among spectrally similar forest classes, a result in accordance with our previous RF study [6]. Annual Crops were broadly confused with Herbaceous cover, and to lesser extent Urban Vegetated. Not all agriculture in this area is irrigated or plowed, and tends to have similar physiognomy and phenology as grasslands, which are dominated by exotic annual grasses; and thus, spectral-temporal confusion between these two classes is expected.

4.2. CNN Feature Maps and Land Cover

The usage of CNN is promising with respect to classification accuracy as well as visualization of features within the data. From convolutional feature maps, the seasonal and spectral contribution to the CNN for each class is visible as excitations by the kernels (Figure 10). This visualization, paired with the confusion matrices, provide a point of exploration into the distinctive nature of the classes. For example, the example Annual Crop spectrum has heavy kernel excitation within the summer reflectance bands (Figure 7). This indicates that the Annual Crop class contains distinctive features within the summer season as compared to the other classes. This could be due to the lack of chlorophyll absorption features in the spring and fall visible spectrum and presence of these features in summer; the example spectrum appears to be green in the summer. The kernels may thus be sensitive to chemical properties of crops. In contrast, herbaceous cover follows the cycle of precipitation (wet spring, dry summer and fall) and senesces in the early summer and is fully senesced by the fall (Figure 2—note lack of chlorophyll absorption in summer and fall). The Herbaceous example spectrum has convolutional excitations spread more evenly across seasons relative to Annual Crops, with some excitations occurring in fall where annual crops had none (Figure 10). The Bare class by definition has relatively low plant cover and minimal chlorophyll absorption signal, although there is high within-class spectral diversity (Figure 2). With the example spectra, kernels are more uniformly excited across all three seasons of the feature map, indicating that this class is less influenced by seasonal variation than within-season spectral properties. This is visible by the additional excitation around the low-signal area (e.g., low blue, high SWIR) within each season of the class data.

An interesting aspect from feature maps is the kernel activations around areas of low blue, far shortwave infrared and around atmospheric absorption bands. These are spectral regions that tend to have more noise due to lower solar irradiance, atmospheric effects (e.g., scattering, absorption), and spectrometer sensitivity. Engineered metrics used in remote sensing target vegetation chemistry and structure with narrowband ratios (e.g., indices such as NDVI), spectral derivatives, and absorption-feature fitting techniques that span the different spectral ranges with well-known continuous areas [6]. In general, engineered metrics avoid these noise-prone spectral regions. However, because some CNN feature maps are excited in these regions, these results indicate that these regions can be important to classification accuracy and warrant further investigation.

One criticism of neural networks and SVM is that they are relatively "black box"; classification performance may be high, but it is difficult to understand the relative importance of spectral and temporal information in the input vector. The CNN feature maps and kernel importance matrix introduced in this study have potential to provide more insight into the classifier and should be explored further with other datasets. In comparison to RF variable importance, we found these CNN diagnostic tools to be less easy to interpret [6]. With RF variable importance, there is a direct linkage between a predictor variable (e.g., reflectance band) and the impact on class and overall accuracy with its removal. Despite the very different architectures between CNN and RF, we found that RF importance (Figure A1) tended to cluster around some of the same regions as identified by the CNN

feature maps, such as blue (spring, summer, fall), high SWIR (spring, summer) and the border of SWIR atmospheric absorption (band index 500, which is 1761 nm from fall imagery).

4.3. Future Work with CNN Hyperspectral Image Classification

The application of CNN to hyperspectral data has only recently been explored and there are thus many avenues for future work. The consideration of the spatial domain is an area of current investigation. The extension of this architecture to include the spatial dimension as in [13] could continue to add more distinctive information that could aid in the classification process. A 3-D convolutional network paired with a respective importance figure (Figure 9) could shed light on to the spatial as well as the spectral features that impact the classification on a per class basis. In a similar manner, the temporal dimension of image data should be investigated. As in this study, this could be done by vertically stacking the temporal season data creating a $N_d \times N_{ns}$ sized input data array, where N_d is the number of bands within a single season of image data and N_{ns} is the number of seasons (rather than horizontally stacked data in temporal sequence, as in our study). This data format would dictate the architecture of networks that could convolve higher dimension kernels across the input data, or perform multiple layered convolutional operations. The additional benefit would be that the kernel's local receptive field could enable the network to capture patterns within the temporal and spectral dimensions in an integrated machine-learning framework. If there is distinctive information within these dimensions, this could increase the accuracy as well as provide more ancillary information about the distinctive aspects of the data through analysis of convolutional feature maps.

It was observed when reviewing the feature maps that the kernels were exciting the convolutional feature maps in similar spectral regions, across all kernels. This appeared as strips of excited kernels. This indicates that the kernels are not capturing unique information. If kernels were "encouraged" to learn information that is "unique" this would enable the feature maps to show different structural features in the data on a per kernel basis. One potential way to do this would be to provide regularization on the kernels during training that would penalize kernels that are not orthogonal. Although this may enhance CNN as a tool for exploring features that discriminate classes, this approach may not improve classification accuracy.

As data from future hyperspectral satellites are more readily available, such as from the Environmental Mapping and Analysis Program (EnMAP) satellite due to launch in 2019, the scalability of classification techniques should be considered. As TensorFlow has been designed to scale from multiple-core, GPU(s), and multiple-computer clustered configurations, the CNN can readily scale for large global and temporal datasets. The Python Scikit-learn library used for RF and SVM in this study does not have this capability, and thus does not offer a direct comparison of GPU processing with CNN and TensorFlow. The computers utilized for this work were an 8 Core Xenon 3.7 Ghz CPU processor with 32 GB ram and a NVIDIA K2200 GPU, as well as a second i7-2640M (2.8 GHz) dual-core CPU with 8 GB of RAM. We tested the CNN architecture with both CPU- and GPU-based configurations. The GPU-based implementation performed all graph computations within the GPU, and was 5 to 8 times faster than CPU-based processing, depending on the learning batch size (larger batch size increased performance within the GPU space). With our data, the GPU-based CNN was slower to train than SVM; however, image classification was 9 times faster with CNN (Table A2). For applications that use large datasets, particularly with the inclusion of spatial and temporal information in the classification design, the CNN with TensorFlow or similar machine learning platform may have a considerable advantage in processing time while offering relatively high classification accuracy.

5. Conclusions

A broad goal of this work was to implement and explain how a Convolutional Neural Network (CNN) with a one-dimensional architecture could be applied to multi-seasonal hyperspectral images. The results show that CNN can be applied to simulated spaceborne hyperspectral reflectance data (HyspIRI) to achieve high classification accuracy rates comparable to that from Support Vector

Machine (SVM), and surpassing the Random Forest (RF) classifier. Highest overall accuracies were with three-season data (spring, summer, fall), with 89.9% for CNN, 89.5% for SVM and 82.2% for RF. Single-season (summer) classifications had overall accuracies that were 1.9 to 3.5% lower than with three-season data. Spectral and temporal information is readily visible through CNN feature map visualizations and their respective importance is traceable back to kernel importance. In summary, the CNN is a promising classifier for future hyperspectral classification tasks and this study identifies future work to increase CNN performance, scalability and incorporation of spatial and temporal information.

Supplementary Materials: The following are available online at www.mdpi.com/2072-4292/9/6/629/s1. The code utilized within this work can be cloned from the following repository: https://ciga_ssu@bitbucket.org/ciga_ssu/hsi-cnn-repo.git.

Acknowledgments: Funding for this project was provided by NASA HyspIRI Preparatory Airborne Activities and Associated Science Research, grant NNX12AP09G. Simulated HyspIRI-like reflectance products were provided by NASA Jet Propulsion Lab (JPL), with key involvement by Bo Cai Gao, David R Thompson, Sarah Lundeen and Robert Green, as well as Philip Dennison of Univ. of Utah for AVIRIS-to-HyspIRI resampling (NASA grant NNX12AP08G). Kelsey Dunn helped prepare maps for this paper. Bala Ravikumar and Farid Farahmand of Sonoma State University provided initial review and input into this study.

Author Contributions: D.G. and M.C. conceived and designed the experiments; D.G. performed the experiments; D.G. and M.C. analyzed the data; M.C. contributed materials and analysis tools; D.G. and M.C. wrote the paper.

Conflicts of Interest: The authors declare no conflict of interest.

Appendix A

Table A1. The Land-Cover Classification System (LCCS) rules applied to percent cover estimates of land cover in reference samples in order to create discrete class labels. Open-canopy (open) trees and shrubs have >10–65% trees or shrubs, respectively. Closed-canopy (closed) trees and shrubs have >65% trees or shrubs, respectively. This study focused on closed-canopy tree and shrub classes only.

Rules Based on Land-Cover Abundance	LCCS Class
1. If >10% Vegetated	
2. If >50% Natural/Semi-natural Vegetation	
3. If >50% tidal salt marsh	**Tidal Marsh**
3. If >10% of cover is woody vegetation (trees + shrubs) and >10% of woody vegetation is trees	
4. If ≥75% of relative tree cover is needleleaf trees	**Evergreen Needleleaved Trees (ENT)**
4. If >75% of relative tree cover is broadleaf trees	
5. ≥75% of tree cover is evergreen	**Evergreen Broadleaved Trees (EBT)**
5. ≥75% of broadleaf tree cover is deciduous	**Deciduous Broadleaved Trees (DBT)**
3. If >10% of cover is woody vegetation (trees + shrubs) and >10% of woody vegetation is shrubs	**Shrubs**
3. Else herbaceous cover	
6. ≥75% of herbaceous cover is upland grasses and forbs	**Herbaceous**
6. ≥75% of herbaceous cover is dune vegetation	**Dune Vegetation**
2. Else Cultivated/Managed Vegetation	
7. >50% perennial crops	**Perennial Crops**
7. >50% annual crops	**Annual Crops**
7. >50% urban landscape	**Urban Vegetated**
1. Else Not Vegetated	
8. >50% impervious surface	**Built-up**
8. >50% non-vegetated	**Bare**

Table A2. Classifier training and classification times in seconds for three season data. For CNN, (1) initial steady-state validation accuracy; and (2) a 4-mil epoch standard training session. The CNN classifier utilized GPU-based processing, while SVM and RF were restricted to CPU-based processing.

	CNN-1 (s)	CNN-2 (s)	SVM (s)	RF (s)
Training Time	1500	21,146	358	3452
Classification Time per GB	65	65	590	768
Classification Time Per Pixel	0.0008	0.0008	0.00773	0.00959

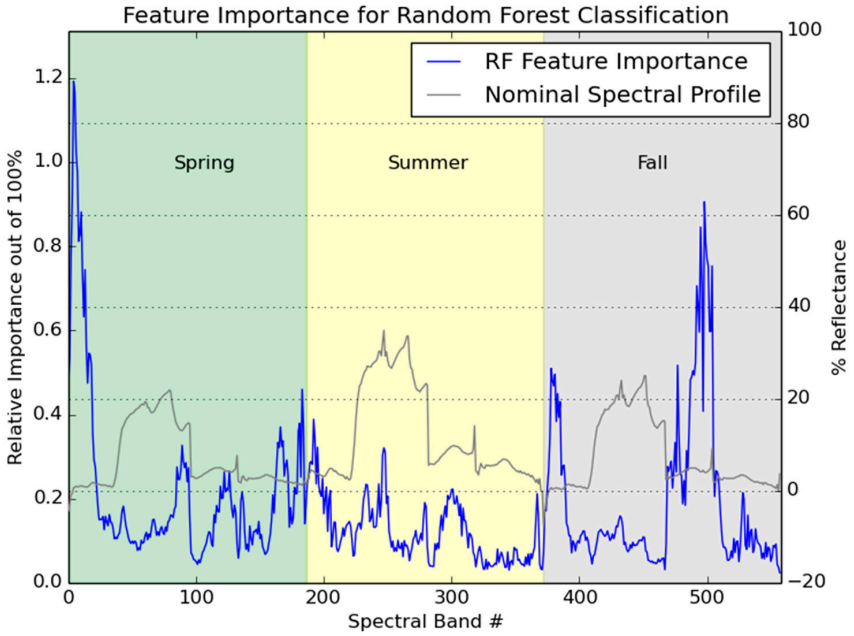

Figure A1. Feature importance for the three-season Random Forests classifier. Nominal spectral profile for ENT shown for context.

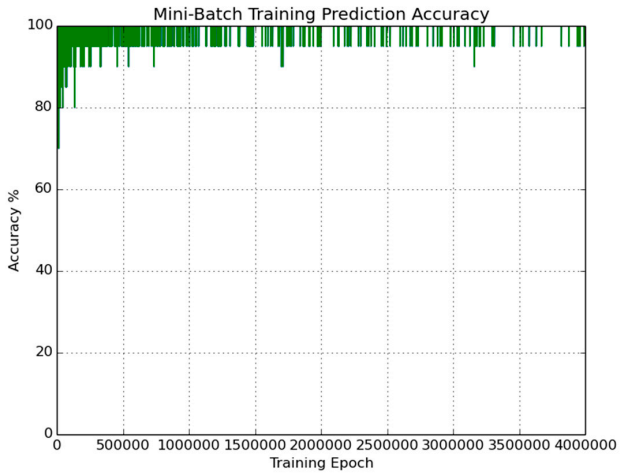

Figure A2. Mini-batch training accuracy curve.

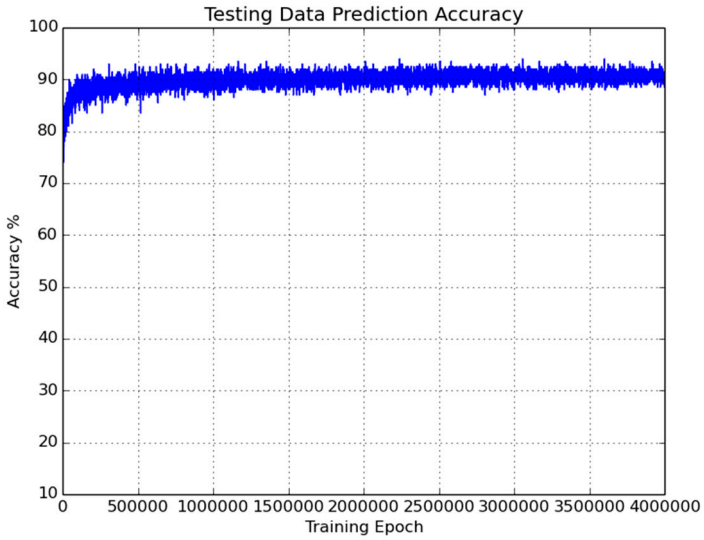

Figure A3. Testing accuracy curve.

References

1. Lee, C.M.; Cable, M.L.; Hook, S.J.; Green, R.O.; Ustin, S.L.; Mandl, D.J.; Middleton, E.M. An Introduction to the NASA Hyperspectral InfraRed Imager (HyspIRI) Mission and Preparatory activities. *Remote Sens. Environ.* **2015**, *167*, 6–19. [CrossRef]
2. Somers, B.; Asner, G.P.; Tits, L.; Coppin, P. Endmember Variability in Spectral Mixture Analysis: A Review. *Remote Sens. Environ.* **2011**, *117*, 1603–1616. [CrossRef]
3. Roberts, D.A.; Gardner, M.; Church, R.; Ustin, S.; Scheer, G.; Green, R.O. Mapping Chaparral in the Santa Monica Mountains Using Multiple Endmember Spectral Mixture Models. *Remote Sens. Environ.* **1998**, *65*, 267–279. [CrossRef]
4. Franke, J.; Roberts, D.A.; Halligan, K.; Menz, G. Hierarchical Multiple Endmember Spectral Mixture Analysis (MESMA) of Hyperspectral Imagery for Urban Environment. *Remote Sens. Environ.* **2008**, *113*, 1712–1723. [CrossRef]
5. Camps-Valls, G. Machine Learning in Remote Sensing Data Processing. Proceedings of IEEE International Workshop on Machine Learning for Signal Processing (MLSP), Grenoble, France, 2–4 September 2009; pp. 1–6.
6. Clark, M.L.; Kilham, N.E. Mapping of Land Cover in Northern California with Simulated HyspIRI imagery. *ISPRS J. Photogramm. Remote Sens.* **2016**, *119*, 228–245. [CrossRef]
7. Mountrakis, G.; Im, J.; Ogole, C. Support Vector Machines in Remote Sensing: A Review. *ISPRS J. Photogramm. Remote Sens.* **2011**, *66*, 247–259. [CrossRef]
8. Fernandex-Delgado, M.; Cernadas, E.; Barro, S.; Amorim, D. Do We Need Hundreds of Classifiers to Solve Real World Classification Problems? *J. Mach. Learn. Res.* **2014**, *15*, 3133–3181.
9. Krizhevsky, A.; Sutskever, I.; Hinton, G. Imagenet Classification with Deep Convolutional Neural Networks. Proceedings of 25th International Conference on Neural Information Processing Systems, Lake Tahoe, NV, USA, 3–6 December 2012; pp. 1097–1105.
10. Szegedy, C.; Liu, W.; Jia, Y.; Sermanet, P.; Reed, S.; Anguelov, D.; Erhan, D.; Vanhoucke, V.; Rabinovich, A. Going Deeper with Convolutions. In Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, Boston, MA, USA, 7–12 June 2015.
11. Abdel-Hamid, O.; Mohamed, A.; Jiang, H. Convolutional Neural Networks for Speech Recognition. *IEEE/ACM Trans. Audio Speech Lang. Process.* **2014**, *22*, 1533–1545. [CrossRef]
12. Castelluccio, M.; Poggi, G.; Sansone, C.; Verdoliva, L. Land Use Classification in Remote Sensing Images by Convolutional Networks. Available online: http://arxiv.org/abs/1508.00092 (accessed on 8 May 2017).

13. Li, Y.; Zhang, H.; Shen, Q. Spectral-Spatial Classification of Hyperspectral Imagery with 3D Convolutional Neural Network. *Remote Sens.* **2017**, *9*, 67. [CrossRef]
14. Langkvist, M.; Kiselev, A.; Alirezaie, M.; Loutfi, A. Classification and Segmentation of Satellite Orthoimagery Using Convolutional Neural Networks. *Remote Sens.* **2016**, *8*, 329. [CrossRef]
15. Kussul, N.; Lavreniuk, M.; Skakun, S.; Shelestov, A. Deep Learning Classification of Land Cover and Crop Types Using Remote Sensing Data. *IEEE Geosci. Remote Sens. Lett.* **2017**, *14*, 5. [CrossRef]
16. Chen, C.; Li, W.; Su, H.; Liu, K. Spectral-Spatial Classification of Hyperspectral Image Based on Kernel Extreme Learning Machine. *Remote Sens.* **2014**, *6*, 5795–5814. [CrossRef]
17. Hu, W.; Huang, Y.; Wei, L.; Zhang, F.; Li, H. Deep Convolutional Neural Networks for Hyperspectral Image Classification. *J. Sens.* **2015**, *2015*, 258619. [CrossRef]
18. Makantasis, K.; Karantzalos, K.; Doulamis, A.; Doulamis, N. Deep Supervised Learning for Hyperspectral Data Classification through Convolutional Neural Networks. In Proceedings of the 2015 IEEE International Symposium Geoscience and Remote Sensing (IGARSS 2015), Milan, Italy, 26–31 July 2015; pp. 4959–4962.
19. Li, W.; Fu, H.; Yu, L.; Cracknell, A. Deep Learning Based Oil Palm Tree Detection and Counting for High-Resolution Remote Sensing Images. *Remote Sens.* **2017**, *9*, 22. [CrossRef]
20. Wang, J.; Luo, C.; Huang, H.; Zhao, H.; Wang, S. Transferring Pre-Trained Deep CNNs for Remote Scene Classification with General Features Learned from Linear PCA Network. *Remote Sens.* **2017**, *9*, 225. [CrossRef]
21. Hu, F.; Xia, G.-S.; Hu, J.; Zhang, L. Transferring Deep Convolutional Neural Networks for Scene Classification of High-Resolution Remote Sensing Imagery. *Remote Sens.* **2015**, *7*, 14680–14707. [CrossRef]
22. Mesay, B.B.; Zeggada, A.; Nouffidj, A.; Melgani, F.; A Convolutional Neural Network Approach for Assisting Avalanche Search and Rescue Operations with UAV Imagery. *Remote Sens.* **2017**, *9*, 100.
23. Liang, H.; Li, Q. Hyperspectral Imagery Classification using Sparse Representations of Convolutional Neural Network Features. *Remote Sens.* **2016**, *8*, 99. [CrossRef]
24. Zhao, W.; Du, S. Spectral-Spatial Feature Extraction for Hyperspectral Image Classification: A Dimension Reduction and Deep Learning Approach. *IEEE Trans. Geosci. Remote Sens.* **2016**, *54*, 8. [CrossRef]
25. Wang, Q.; Lin, J.; Yuan, Y. Salient Band Selection for Hyperspectral Image Classification via Manifold Ranking. *IEEE Trans. Neural Net. Learn. Syst.* **2016**, *27*, 6. [CrossRef] [PubMed]
26. Yuan, Y.; Lin, J.; Wang, Q. Hyperspectral Image Classification via Multitask Joint Sparse Representation and Stepwise MRF Optimization. *IEEE Trans. Neural Netw. Learn. Syst.* **2016**, *46*, 12. [CrossRef] [PubMed]
27. Di Gregorio, A. *Land Cover Classification System: Classification Concepts and User Manual*; LCCS (No.8); Food and Agriculture Organization: Rome, Italy, 2005.
28. Green, R.O.; Eastwood, M.L.; Sarture, C.M.; Chrien, T.G.; Aronsson, M.; Chippendale, B.J.; Faust, J.A.; Pavri, B.E.; Chovit, C.J.; Solis, M.S.; et al. Imaging Spectroscopy and the Airborne Visible/Infrared Imaging Spectrometer (AVIRIS). *Remote Sens. Environ.* **1998**, *65*, 227–248. [CrossRef]
29. Thorpe, A.K.; Frankenberg, C.; Aubrey, A.D.; McFadden, J.P. Mapping Methane Concentrations from a Controlled Release Experiment using the Next Generation Airborne Visible/Infrared Imaging Spectrometer (AVIRIS-NG). *Remote Sens. Environ.* **2016**, *179*, 104–115. [CrossRef]
30. Thompson, D.R.; Gao, B.C.; Green, R.O.; Roberts, D.A.; Dennison, P.E.; Lundeen, S.R. Atmospheric Correction for Global Mapping Spectroscopy: ATREM advances for the HyspIRI preparatory campaign. *Remote Sens. Environ.* **2015**, *167*, 64–77. [CrossRef]
31. Breiman, L. Random forests. *Mach. Learn.* **2001**, *45*, 5–32. [CrossRef]
32. Ensemble Methods. Available online: http://scikit-learn.org/stable/modules/ensemble.html#forest (accessed on 8 May 2017).
33. RBF SVM Parameters. Available online: www.scikit-learn.org/stable/auto_examples/svm/plot_rbf_parameters.html (accessed on 8 May 2017).
34. Srivastava, N.; Hinton, G.; Krizhevsky, A.; Sutskever, I.; Salakhutdinov, R. Dropout: A Simple Way to Prevent Neural Networks from Overfitting. *J. Mach. Learn. Res.* **2014**, *15*, 1929–1958.
35. Nielsen, M. Chapter 3, Improving the Way Neural Networks Learn, Neural Networks and Deep Learning. Available online: http://neuralnetworksanddeeplearning.com/chap3.html (accessed on 8 May 2017).

36. CS231n Convolutional Neural Networks for Visual Recognition. Available online: http://cs231n.github.io/convolutional-networks/ (accessed on 8 May 2017).
37. Khatami, R.; Mountrakis, G.; Sehman, S. A Meta-analysis of Remote Sensing Research on Supervised Pixel-based Land-cover Image Classification Processes: General guidelines for Practitioners and Future Research. *Remote Sens. Environ.* **2016**, *177*, 89–100. [CrossRef]