Convolutional Neural Network Classification

of

Hyperspectral Imagery in the San Francisco Bay Area, California

By

Daniel Giudici

Department of Engineering

Submitted in partial fulfillment of the requirements

for the degree

of Masters of Science in Computer and Engineering Science
The undersigned have examined the thesis entitled “Convolutional Neural Network Classification of Hyperspectral Imagery in the San Francisco Bay Area, California” presented by Daniel Giudici, a candidate for the degree of Master of Science in Computer and Engineering and hereby certify that it is worthy of acceptance.

9/16/16
Date

Dr. Matthew Clark

9/16/16
Date

Dr. Farid Farahmand

9/16/16
Date

Dr. Balasubraman Ravikumar
Abstract

Purpose:

Hyperspectral image (HSI) data has the potential to have a high degree of intra-class variability and low inter-class variability; this makes engineering features and creating robust classifiers challenging. Convolutional Neural Networks (CNNs) have found success in classifying classical 2-d imagery and recently HSI data. CNNs, unlike other neural network topologies learn features from the data in the form of convolutional kernels. These trained convolutional kernels bring out or filter the class-specific discriminating features from the data. These features are then used to classify the data. To date there has been little discussion on visualization and interpretation of these learned convolutional kernels and their respective impact on classification as applied to hyperspectral data. This work will introduce an architecture as well as provide methods to visualize and determine how important these learned convolutional kernels are on the classification task.

Procedure:

A convolutional neural network architecture was developed, trained and utilized to classify hyperspectral imagery. As a comparison, Random Forests (RF) and Support Vector Machine (SVM) classifiers were also trained on the same dataset. These data are from the San Francisco Bay area in year 2015, across three temporal seasons. Comparisons between these classifiers and the application of temporal data within the data are provided. The analysis of the convolutional
network's inner data product visualizations are provided to show insight into what significant features exist in the data and how important these features are for classification accuracy.

**Findings:**

The CNN developed provided classification accuracies comparable to SVM. Both CNN and SVM performed better than RF. Analysis of the inner products of the CNN provided insight to the distinctive features within the spectral and temporal domain. All classification methods perform well when generating land cover maps. The Random Forest classifier mis-classified some obvious land cover areas. The Support Vector Machine (SVM) and CNN generated maps disagree on harder to classify areas.

**Conclusions:**

Described within is a classifier implementation that can be applied to hyperspectral imagery data that produces classification accuracies comparable to existing classification methods. Unbalanced/Balanced accuracies of 91.0%/74.0% for SVM, 89.8%/75.5% for CNN and 84.2%/65.8% for RF were developed. Visualizations methods were developed to show the distinguishing characteristics between the classes across the dimension of the convolution defined within this topology. This type of classifier is a good candidate for HSI applications because of the interpretability of these visualizations and relatively high classification accuracy. Kernel importance is provided as a method and metric to determine how important a learned feature is with respect to the classification task.
Keywords: Hyperspectral Imagery, Convolutional Neural Networks, Support Vector Machine (SVM), Random Forests (RF), Machine Learning.
Table of Contents

Abstract ......................................................................................................................... iv
Table of Contents .......................................................................................................... vii
1 Introduction .................................................................................................................. 1
2 Methods ......................................................................................................................... 4
  2.1 Classifier Tuning and Architectures ........................................................................ 4
  2.1.1 Random Forest ...................................................................................................... 5
  2.1.2 Support Vector Machine ......................................................................................... 6
  2.1.3 Convolutional Neural Network .............................................................................. 7
  2.2 Data ......................................................................................................................... 20
  2.2.1 Study area ............................................................................................................ 20
  2.2.2 Land cover reference data ................................................................................... 22
  2.2.3 HyspIRI Preparatory Science Campaign Data Products ..................................... 24
  2.2.4 Data Pre-Processing ............................................................................................ 25
  2.2.5 Classifier Comparison and Accuracy Evaluations .............................................. 26
3 Results ......................................................................................................................... 28
  3.1 Accuracy Assessment .............................................................................................. 28
  3.2 Feature Evaluation .................................................................................................. 34
  3.3 Performance ........................................................................................................... 40
4 Discussions .................................................................................................................. 41
  4.1 Feature Map and Land Cover Classification Analysis ............................................. 41
  4.2 Performance ............................................................................................................ 50
  4.3 Future work ............................................................................................................. 52
5 Conclusion .................................................................................................................. 53
6 References ................................................................................................................... 55
Appendix ......................................................................................................................... 58
7 Supplemental Material ................................................................................................. 59
  7.1 Code ........................................................................................................................ 59
  7.2 Saved Classifiers ...................................................................................................... 61
  7.3 Data Files ................................................................................................................ 61
List of Figures

Figure 1 CNN Flow Diagram................................................................. 8
Figure 2 Training data, showing 25 spectral signatures from each class .......... 10
Figure 3 Detailed CNN Architecture ..................................................... 11
Figure 4 Reference and training data collected in the San Francisco Bay Area study area.
........................................................................................................... 21
Figure 5 Data standardization example ..................................................... 26
Figure 6 Training and testing population distributions ................................. 27
Figure 7 Confusion Matrix HSICNet (CNN) - balanced data. Call-out boxes show
Producer’s Accuracy. Class index definition found in Table 1........................ 29
Figure 8 Confusion Matrix HSICNet (CNN) – unbalanced data ................. 30
Figure 9 Confusion Matrix SVM - balanced data .................................... 31
Figure 10 Confusion Matrix SVM - unbalanced data ................................. 32
Figure 11 Confusion Matrix RF - balanced data ....................................... 33
Figure 12 Confusion Matrix RF - unbalanced data ................................... 34
Figure 13 Feature importance for Random Forest classification ................ 35
Figure 14 Convolutional feature map for class: Annual Crops .................... 38
Figure 15 Convolutional feature maps for each class. These feature maps are the result of
convolving the kernels and the input data ............................................... 39
Figure 16 RF, CNN & SVM classified land cover areas of San Francisco Bay Area.
White areas indicate pixels that were not classified (e.g., water, no data) ........ 44
Figure 17 RF, CNN & SVM classified land cover areas, San Francisco Peninsula ... 45
Figure 18 Satellite view of the city of San Francisco ................................. 46
Figure 19 RF, CNN & SVM classified land cover areas, San Francisco South Bay .... 47
Figure 20 Satellite view of San Francisco South Bay ................................. 47
Figure 21 RF, CNN & SVM classified land cover areas, San Francisco East Bay .... 48
Figure 22 Satellite View of San Francisco East Bay ................................. 48
Figure 23 RF, CNN & SVM classified land cover areas, San Francisco North Bay .... 49
Figure 24 Satellite View of San Francisco North Bay ................................ 50
Figure 25 CNN training - validation accuracy ........................................ 51
List of Tables

Table 1 One-Hot classification scheme ................................................................. 15
Table 2 Land Cover Classification System (LCCS) ..................................................... 23
Table 3 Accuracy performance per classifier and data .................................................. 28
Table 4 Top 10 Features for Random Forest classification ......................................... 35
Table 5 Kernel Importance Matrix. Class index definitions found in Table 1 ............... 36
Table 6 Classifier training and classification times ..................................................... 41
1 Introduction

Land-cover maps provide information for natural resource and ecosystem service management, conservation planning, and the assessment of human-induced and natural drivers of land change. The societal impacts of accurately classifying land areas extend from urban monitoring, to observing functional physiological and seasonal changes within a landscape [1]. The automated classification of land-cover with remotely sensed imagery is a challenging task and can be aided by the analysis of hyperspectral imagery, as there is the potential for distinguishing information to be captured within this type of data that may not be captured with standard imagery or multi-spectral sensors. These data consist of hundreds of spectral bands as compared to standard imagery or multi-spectral sensors that span relatively few and select spectral bands. The analysis of hyperspectral data presents issues due to the amount of data and the inter- and intra-spectral variability among classes [2]. The classification task becomes more difficult at larger spatial extents, such as in images acquired by spaceborne hyperspectral sensors. This is due to potentially non-linear spectral mixing at low spatial resolutions (larger ground cover per pixel).

Traditional classification methods for hyperspectral imagery predominantly consist of mixing pure spectral profiles (endmembers) to create a combined spectral profile that is similar to the data being classified. This type of method is called spectral mixture analysis (SMA). This family of methods require large libraries of pure spectral profiles that are specific to the region being classified. This type of classification is the most closed form solution available currently, analytically processing an exhaustive combination of pure endmembers that most closely match the data to be classified [3].
Due to the non-linearity within the data due to the mixing of nearby pixels and reflectance captured by each individual pixel, machine learning has been recommended for the classification of these data. There are various machine learning approaches as described in [4], a subset of recommended supervised learning classifiers will be discussed within this work.

Machine learning has proven to be successful in the classification of hyperspectral imagery for distinguishing land-cover [5]. There are many different varieties of classification techniques implemented on different platforms. Picking the classifier to analyze a dataset at times falls to the familiarity of computational platform and or algorithm for a specific field. Random Forests (RF) and Support Vector Machines (SVM) are relatively new, yet widely adopted machine learning classifiers for the remote sensing community. They have provided robust results across many platforms and datasets, surpassing many other families and implementations of classifiers [6]. Convolutional Neural Networks (CNNs) are the leading machine learning classifier for image recognition tasks using 2-dimensional image data [7,8,9]. Applications of CNNs have extended into the classification for other contiguous data types, like speech recognition utilizing 1-dimensional data [10]. Predominantly, CNNs have been utilized to classify remotely-sensed images with respect to the spatial dimension, making the classification of data based on groups of pixels. This is similar to what is done in standard image classification [7,8]. Only recently CNN architectures have been introduced and implemented to work with the hyperspectral imaging modality [14]. Single dimensional spectral data as well as the classification of land cover area based on joint spatial and
spectral domains, have started to be explored [9]. Although not always discussed, a potentially useful feature of CNN architectures is that the inner resulting data products can provide some insight into what the classifier has learned to make its classification, making this classifier type less of a “black box”. This interpretative aspect of CNN will be explored within this work, enabling the network to show where within the spectral profile lies the distinctive information for the classification. CNN architectures require the data to be in a contiguous format as convolutional layer of the network distinguish, or filter, local “structural” features or patterns from contiguous regions throughout the data. The network then performs the subsequent classification based on these learned features.

In this work the leading machine learning classification methods, RF and SVM, are assessed with respect to a CNN. Furthermore, to support the eventual applications based on spaceborne hyperspectral imagery these analyses were performed with simulated Hyperspectral Infared Imager (HyspIRI) imagery. The HyspIRI satellite is a mission currently being considered by NASA to improve our ability to map and monitor the Earth's ecosystems and provide timely information on natural disasters [1]. The HyspIRI sensor will create images using energy from 214 spectral measurements from visible to shortwave wavelengths (400-2500 nm) with a 30-m spatial resolution, with an additional thermal sensor and a 16-day repeat cycle. In this study, Airborne Visible/Infrared Imaging Spectrometer (AVIRIS) data acquired over the San Francisco Bay Area in spring, summer and fall of 2015 were used to simulate multi-temporal HyspIRI satellite imagery. Classes defined by the international Land Cover Classification System
(LCCS) were used for mapping land cover [11]. This scheme consisted of 12 land cover classes.

The inner data products of CNN architectures as applied to HSI data have not been thoroughly discussed in the literature to date. Another goal of this study is to show how the inner data products of CNN can provide insight into the classification task. With the analysis of the feature maps that the CNN creates, local regions within the spectral dimension of the data that are excited can be shown to have an impact on the classification accuracy. The importance of these feature maps can be then traced back to how important they are to the classification task. Additional processing of the feature maps extracted from the CNN enables a illustrative visual that shows the features that separate the classes. By standardizing, scaling and capturing only the magnitude of the convolutional kernel feature maps, the spectral band and temporal importance can be visually explored with respect to the resulting classification accuracy.

2 Methods

2.1 Classifier Tuning and Architectures

Three classifiers were assessed in this work: Random Forest (RF), Support Vector Machine (SVM) and Convolutional Neural Network (CNN). RF and SVM are known and robust algorithms recommended by the remote sensing community. Additionally, literature states that they are robust classifiers for various data sets [6]. There are various implementations across various programming platforms and languages for classification algorithms. Scikit-learn is a machine learning library, developed for the Python programming language. This library provides access to established data mining and data
analysis tools and algorithms. This library was utilized for the training and classification utilizing RF and SVM classifiers, and was used as a baseline for comparison of the CNN developed within this work.

The application of CNN to data from different disciplines is relatively new to the machine learning field. The adaptations of the respective CNN architectures require a more flexible platform for their development. As a CNN is a deep network topology, it can readily be formed as a computational graph. Google's recently released open source computational platform, Tensorflow™, is directly suited for this application. Tensorflow™ was used for the CNN definition, training and classification portion of this work. A brief background and explanation the classifiers and the hyper-parameters of each classifier are described below.

2.1.1 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 is applied to the classifier and the mean prediction across the individual trees is captured [12]. This mean prediction 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 (min_samples_leaf). As an empirical rule, the number of features utilized in RF classifiers is \( \sqrt{N_f} \), where \( N_f \) is the number of features. The number of trees included in the ensemble typically improves classification accuracy until a critical point is reached, where accuracy does not increase
For the minimum leaf size, the smaller this value is the more prone the RF is to capturing noise in the training data. These parameters should each be considered when training a classifier on a dataset [13].

For this work the standard $MaxNumberofFeatures = \sqrt{N_f}$ will be utilized. The number of trees was increased until there was not a significant change in prediction accuracy, one thousand trees was utilized for this work. A search across the minimum leaf sizes was performed and the default value of one for the leaf size performed the best.

2.1.2 Support Vector Machine

A SVM is a classifier that separates the data by defining a hyperplane(s) through the data to segment the classes. In a visual sense, it is kin to drawing the line between the two classes that defines the maximum separation or margin between the classes. 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 Support Vector Classifier that utilized one-vs-rest classification scheme was utilized for this work. This implementation simplifies the tuning of the classifier down to one parameter, C. This parameter trades off misclassification of training examples against 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 [13]. A linear search for this tuning parameter was performed from $1e-2$
to $1 \times 10^10$ and the highest accuracy classifier was chosen; a $c$ value of 0.1 was utilized for this work.

### 2.1.3 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. Work done by Wei Hu and colleagues [14] mentions these structural features as the interclass appearance present within the data. As an extension from this previous literature [14], it will be shown within this work that these 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 onto a subsequent classification process.

The basis for the network developed for this work has been modified from the work done in [14] to suit this data set. A flow diagram of the CNN process utilized here is shown below Figure 1. In larger networks, additional convolutional and pooling operations/layers are combined in different ways to capture distinct features from the data. Due to the HSI data used here being single dimensional, the architecture is able to be simplified as follows:
The training of CNN architectures can be performed by the standard back-propagation algorithm used in traditional neural networks. Regularizing methods such as Dropout, Regularization of the Feature Classification NN and the utilization of an adaptive learning rate are applicable to CNNs and were implemented within this network.

2.1.3.1 Architecture

The feature generation and feature classification nature of the 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 [7,8]. These complex networks required the learning of a large number of features and a high level of complexity for these. The HSI data classification is inherently different from standard imagery classification. This is because the purpose is to classify individual pixels of the data and not the full image. Standard imagery is 2-
dimensional, containing a width and a height. The convolutional operation for these networks occurs with multi-dimensional kernels and in various configurations. For simplicity, only the spectral domain of HSI data was considered in this study; and thus, the convolutional operation of the network only needs to operate on this single dimension (i.e., 1-dimensional). This reduces the convolutional kernels shape to only span this dimension. This greatly simplifies the architecture of the HSI CNN from what has been developed for standard imagery as shown in [7,8].

As will be noted later, the data utilized consists of three single-date imagery acquisitions: one for spring, summer and fall. Each season's spectral profile data will be horizontally appended to each other so that all three seasons will be shown in each classified spectral signature. Figure 2 shows twenty-five randomly selected three season spectral profiles from each of the 12 LCCS classes used in this study. A description of these data is described in more detail in the Data section of this work. Note that the X-axis is the band number (1-558) and bad bands have been removed.
A similar CNN architecture was recently introduced for HSI data classification [14]. The network in [14] consists of a single convolution layer paired with a pooling layer which feeds a fully connected neural network and was the basis for the CNN developed within this work. The network developed here was further refined from what was previously presented with the addition of Dropout and Regularization of the fully connected layer. Additionally, tuning of the convolutional layers was based on a technique developed within this work that reduces the number of kernels trained within the network based on impact the kernels have on the classifier accuracy.

A more detailed diagram of the network is shown in Figure 3.
The single convolutional layer accepts the 1-dimensional spectral profile input data and performs the convolutional operation on the input data with each kernel in the architecture. This filtering of the input data with each kernel creates the features for classification. Hyperspectral imagery classification using the dataset used in this study has had good results by utilizing 86 engineered features, or spectral metrics, spanning the
spectral dimension [5]. The spectral metrics presented in [5] perform a similar function as the kernels for this network. For this network, the number of engineered features from previous literature was used as the rough starting point for the number of kernels to be trained for this network. Because the learned features within the CNN are not tied to any chemical or physical phenomenon, rather only what is required to distinguish the classes, the number of kernels was able to be drastically reduced.

The convolutional operation within this network is similar to the formulation of a Finite Impulse Response (FIR) digital filter. For $K_n$ kernels within the CNN, the network performs a convolution on the data based on the following formula.

$$y(n)_i = \sum_{a=0}^{K_s-1} w_{i,a} x(n - a) \text{ for } i = 1, 2, 3 ... N_k$$

Where $K_s$ is the size of the kernel, $w$ is the kernel weights, $x$ is the input data, $N_k$ is the number of kernels in the network and $n$ is the index into the input data.

The convolution that occurs within a digital FIR filter is based on the following formulation.

$$y(n) = \sum_{m=0}^{M} b_m x(n - m)$$

Where $M$ is the size of the filter, $b$ is the coefficients of the digital filter, $x$ is the input time series data and $n$ is the index into the input data. Comparing these two formulations, it is apparent that the length of the convolutional kernel is effectively the size of the digital filter, or the number of coefficients or poles within the FIR filter. In this way, the convolutional network can be thought to be learning $N_k$ digital filter coefficients for the
The convolution of the input data (length, $N_d$), with each of the kernels within the network (number of kernels, $N_k$) creates a feature map with dimensions $N_d \times N_k$.

The pooling layer can be thought of as a down-sampling of the convolutional feature map. 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 condenses where the features are observed within the convolutional feature map providing a level of spatial invariance to the feature generation. This also reduces the overall number of connections to the fully connected network and in effect the total number of trainable parameters for the network, reducing overall training time.

This Pooled Feature map is then provided to the feature classification network. In this architecture it consists of a fully connected Neural Network (NN). This fully connected network layer consist 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 encourages “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 others. This could in theory change how and what the convolutional kernels learn.

The output of the hidden layer is connected to a final softmax output layer that produces a probabilistic output per class; a vector of length of the number of classes, with each real valued scalar value representing the probability that the input data belongs to a specific class. This probability or confidence that the Softmax layer calculates is shown in the formula below.

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

Where $Z_j$ are the weights applied to each softmax layer node, $K$ 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. For example a soft max probability vector of [0.191, 0.191, 0.425, 0.191] would result in a one hot vector of [0,0,1,0]. The subsequent class aligned with this one-hot encoding would be the classification result. For the data presented within this work the one-hot representation of the twelve classes within the data are shown in Table 1.
<table>
<thead>
<tr>
<th>Class Name</th>
<th>Class Index</th>
<th>One Hot Class Label</th>
</tr>
</thead>
<tbody>
<tr>
<td>Annual Crops</td>
<td>0</td>
<td>000000000001</td>
</tr>
<tr>
<td>Bare</td>
<td>1</td>
<td>000000000010</td>
</tr>
<tr>
<td>Built-Up</td>
<td>2</td>
<td>000000000100</td>
</tr>
<tr>
<td>Closed-canopy Evergreen Needleleaf Trees</td>
<td>3</td>
<td>000000001000</td>
</tr>
<tr>
<td>Closed-canopy Deciduous Broadleaf Trees</td>
<td>4</td>
<td>000000100000</td>
</tr>
<tr>
<td>Closed-canopy Evergreen Broadleaf Trees</td>
<td>5</td>
<td>000001000000</td>
</tr>
<tr>
<td>Closed-canopy Shrubs</td>
<td>6</td>
<td>000010000000</td>
</tr>
<tr>
<td>Dune Vegetation</td>
<td>7</td>
<td>001000000000</td>
</tr>
<tr>
<td>Perennial Crops</td>
<td>8</td>
<td>010000000000</td>
</tr>
<tr>
<td>Tidal Marsh</td>
<td>9</td>
<td>100000000000</td>
</tr>
<tr>
<td>Herbaceous</td>
<td>10</td>
<td>100000000000</td>
</tr>
<tr>
<td>Urban Vegetated</td>
<td>11</td>
<td>100000000000</td>
</tr>
</tbody>
</table>

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 based on the labeled data. This loss is then propagated back through the network to modify the network’s interconnections, based on the how much impact the respective weights had on the classification. This is the gradient of the network, and is essentially the degree to which the weights of the network have impacted that classification and resulted in a respective loss. Modifying the network based on the loss calculated by the classification or mis-classification of the labeled data is effectively the backpropagation algorithm. As training progresses the back-propagation modifies the
network to reduce the loss function. This decreasing loss function indicates the classified data more closely matches the labeled data.

To help generalize the fully connected NN dropout was utilized on the hidden layer. This operation removes half of the nodes at various times during training, forcing the network to learn the data set across the weights of the network. The loss function utilized the softmax cross entropy of the classification and L2 regularization based on the fully connected NN weights. L2 regularization on the fully connected NN was utilized to encourage the network to utilize smaller weights for the classification, so one feature would not dominate the classification.

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 or an exponential decaying learning rate. The Adagrad optimizer adjusts the learning rate, the extent the network can be modified, based on the gradient recently seen by the network and is suited for relatively sparse data. As the dataset analyzed here is relatively unbalanced, 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.
2.1.3.2 Hyper-parameter Tuning

Dominant tunable hyper-parameters of this architecture that should be tuned on a per data basis are $K_s$ the kernel size (also known as number of taps within a digital filtering context), $N_k$ the number of kernels, and $N_h$ the number of hidden nodes within the classification network. These parameters are show on the block diagram Figure 3. While other characteristics are able to be modified within the network, these were determined to be the most influential. Other parameters such as dropout level and regularization level on the hidden weights could be adjusted, but it seemed to be that as long as these parameters were defined their presence increased the accuracy of the network. Values of 50% dropout and 1e-4 for regularization was utilized for the network.

The kernel size dictates the size of the feature to capture. It is the size of the local receptive field considered for the convolution with the data. A rule of thumb originating from the currently engineered features is to use local receptive fields that are roughly 10 spectral points long for a single season. These engineered features have been implemented with good results in [5]. A single-date data capture (single season) of data spans 186 valid data points, so roughly about 5 percent (10/186) of the data range has been determined to be acceptable when creating features from other methods. This $N_s$ of 10 was utilized here. It is also desirable to keep the kernel size down to reduce the fringing effects of the convolutional operation. As this network is currently implemented with zero padded convolution, fringe effects at the beginning and end of the spectral dimension of the feature maps is expected. The zero padded convolution is utilized to preserve the input to output size of the convolution operation. The fringing artifacts due
to this should be learned by the classification network to not be a distinguishing feature as it will be present throughout all of the data. The larger the kernel size the larger these fringing effects will show up within the feature maps, and will make these inner data products less interpretable. To minimize the potential for this to have a significant effect the size of the kernel should be minimized. This will enable only minimal areas of the feature maps to contain convolutional fringe effects.

In a digital FIR filters, the number of taps within a kernel are the coefficients for that filter. FIR filter design tunes these coefficients in such a way to control step and frequency responses. With more coefficients the designer is better able to achieve the desired step and frequency responses. In the CNN context the coefficients of these filters are modified to optimize a loss function and not a visual performance indicator; such as a step or frequency response. Because of this, as the number of coefficients increase the interpretation of these filter(s), via the usual methods (frequency and step responses) becomes less clear. This nuance limits the usefulness of digital FIR filter analysis techniques as applied to convolutional networks. This makes what has been learned by the kernels hard to extract by these methods. Some work was done to analyze the digital domain frequency response of the kernels as if they were digital filters, but this additional analysis relating convolutional kernels to FIR filters provided mixed results. When treating a kernel as a FIR filter, as the kernel size increased, effectively the number of coefficients increased in the analogous FIR filter. This makes the FIR filter more descriptive, but in this CNN context the kernel is learning features based on a loss function and not a physical phenomenon, like an explicit high pass or low pass filter. The
resulting frequency and impulse responses are hard to analyze, and do not contain the
typical structure one would expect when designing a FIR filter. The salient result of
analyzing the kernels as FIR filters is the idea that adding more tunable coefficients
enables the convolutional network to be able to learn more descriptive features. However,
as the kernels increase in size the less interpretable and generalized the features may
become.

The capacity of the network to learn features is a combination of the number of
kernels and the kernel size. The number of kernels contained within the network
represents the number of features able to be learned. If each of those features are very
descriptive, and has a large kernel size, then fewer of them would be required to achieve
similar results. It was empirically determined that the number of kernels are able to be
reduced until removing a kernel from the classifier always has negative effect on the
accuracy. This ensures that the feature maps with respect to each kernel are excited. With
this kernel reduction performed it is believed that the network does not have too much
excess capacity and the likelihood of over training is minimized. On the other hand, the
number of kernels should be increased to reduce the kernel size and to increase the
individual kernels interpretability. This becomes a balancing act between number of
kernels for interpretation and the overall capacity of the network. An acceptable balance
between these parameters were: \( N_k = 7 \) and \( K_s = 10 \) for these data.

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. The number of hidden layer nodes, $N_h$, was chosen to be 500 for this network.

### 2.2 Data

The data utilized for this work is an extension of the work recently analyzed for land cover classification [5]. The major difference between the data being the dataset utilized here was taken from 2015, whereas the data utilized in the previous literature was captured in 2013. These data will be described in the following section.

#### 2.2.1 Study area

Imagery in this study covered a box centered (37° 52’ 45.73” N, 122° 13’ 8.54” W) on the San Francisco Bay Area in northern California, USA (Figure 4). This area includes the Temperate Coniferous forests biome in the north (ecoregion California North Coast) and the Mediterranean Forests, Woodlands, and Scrub forests biome to the south (ecoregions California Central Coast and Great Central Valley). The region has a mostly Mediterranean climate with average annual precipitation of 689 mm, and a seasonal average of 162 mm in spring [March – May], 6 mm in summer [June – August], 123 mm in fall [September – November], and 397 mm in winter [December through February] (NOAA, 1981-2010 U.S. Climate Normals: Ground stations within box). Average minimum and maximum temperatures are 9°C and 21°C, respectively. There is a range of fine-scale variation in climate across the study area largely determined by distance from coast and topography as a marine air mass and advection fog moderates annual climate variability by providing summer moisture and cooler temperatures along the
coast and adjacent low-elevation areas that it penetrates; and consequently, annual precipitation tends to be higher toward the coast and in the north and summer temperatures tend to be highest and winter temperatures lowest in the east.

![Figure 4 Reference and training data collected in the San Francisco Bay Area study area.](image)

Natural vegetation in the study area is largely determined by fine-scale variability in climate, interlinked with topographic controls (elevation, position, aspect) and coastal influence, as well as variation in soils and disturbance. In general, evergreen needle leaf forests (conifer) and evergreen broadleaf forests are mainly in the west; in the interior mountains, deciduous broadleaf forests intergrade with needle leaf and evergreen broadleaf trees, shrublands and annual grasslands; the eastern slopes of mountains to the agricultural central valley are dominated by annual grasslands, which senesce in late
spring. Dense urban areas rim the San Francisco Bay and agriculture is important beyond
the bay, particularly perennial crops (vineyards) in the northwest, annual crops in the
southwest and a diverse mix of annual and perennial crops in the Central Valley to the
east.

2.2.2 Land cover reference data

Land cover reference data is described in more detail in [5]. Data consists of
percent land cover of the following twelve “land-cover components” within a sample
polygon: evergreen needleleaf trees; evergreen broadleaf trees; deciduous broadleaf trees;
shrubs; upland grasses and forbs; dune vegetation; tidal salt marsh; annual crops;
perennial crops; impervious surfaces; urban landscape; and, non-vegetated (beaches,
dunes, rocks, bare soil). Percent cover data were visually estimated to the nearest 10%
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 with simple
random and manual methods over areas of well-mixed land cover components, and most
polygon centers were further than 1000 m to a neighboring sample (Figure 1). Each
polygon was classified into one of 20 discrete Land Cover Classification System (LCCS)
[11] classes using a decision-tree of rules applied to a polygon’s percent cover data
(Table 2). Open-canopy trees (woodlands) and shrubs (shrublands) have >10-65% tree or
shrub cover, respectively; closed-canopy trees (forests) and shrubs (thickets) have >65%
This study focused on only closed-canopy trees and shrubs, resulting in 12 LCCS classes (Table 1).

Table 2 Land Cover Classification System (LCCS)
Discrete classes derived from percent cover types. 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.

<table>
<thead>
<tr>
<th>Land Cover/Use Class</th>
<th>Cover types and abundance</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>If &gt; 10% Vegetated</td>
</tr>
<tr>
<td></td>
<td>If &gt; 50% Natural/Semi-natural Vegetation</td>
</tr>
<tr>
<td>Tidal Marsh</td>
<td>&gt; 50% Tidal salt marsh</td>
</tr>
<tr>
<td></td>
<td>If &gt; 10% of cover is woody vegetation (trees + shrubs) and &gt; 10% of woody vegetation is trees &gt; 75% of relative tree cover is needleleaf tree</td>
</tr>
<tr>
<td>Evergreen Needleleaved Trees (ENT)</td>
<td>&gt; 75% of tree cover is needleleaf tree</td>
</tr>
<tr>
<td></td>
<td>&gt; 75% of relative tree cover is broadleaf (evergreen + deciduous)</td>
</tr>
<tr>
<td>Evergreen Broadleaved Trees (EBT)</td>
<td>&gt; 75% of tree cover is evergreen</td>
</tr>
<tr>
<td>Deciduous Broadleaved Trees (DBT)</td>
<td>&gt; 75% of broadleaf tree cover is deciduous</td>
</tr>
<tr>
<td>Mixed Broadleaved Trees (MBT)</td>
<td>Evergreen &amp; Deciduous relative cover each &lt; 75%</td>
</tr>
<tr>
<td>Mixed Broadleaved/Needleleaved Trees (MBT/NT)</td>
<td>Needleleaf and broadleaf relative cover each &lt; 75%</td>
</tr>
<tr>
<td>Shrubs</td>
<td>and &gt; 10% of woody vegetation is shrubs</td>
</tr>
<tr>
<td>Herbaceous</td>
<td>&gt; 75% of herbaceous cover is upland grasses and forbs</td>
</tr>
<tr>
<td>Dune Vegetation</td>
<td>&gt; 75% of herbaceous cover is dune vegetation</td>
</tr>
<tr>
<td></td>
<td>Else Cultivated/Managed Vegetation</td>
</tr>
<tr>
<td>Perennial Crops</td>
<td>&gt; 50% perennial crops</td>
</tr>
<tr>
<td>Annual Crops</td>
<td>&gt; 50% annual crops</td>
</tr>
<tr>
<td>Urban Vegetated</td>
<td>&gt; 50% urban landscape</td>
</tr>
<tr>
<td></td>
<td>Else Not Vegetated</td>
</tr>
<tr>
<td>Built-up</td>
<td>&gt; 50% impervious surface</td>
</tr>
<tr>
<td>Bare</td>
<td>&gt; 50% non-vegetated</td>
</tr>
</tbody>
</table>
2.2.3 HyspIRI Preparatory Science Campaign Data Products

Hyperspectral images used in this study were collected as part of the NASA HyspIRI Preparatory Science Campaign and include year 2015 spring, summer and fall scenes from the AVIRIS Classic sensor [1]. The AVIRIS-C sensor collects spectral radiance data over 224 bands from 370 to 2500 nm with 10 nm sampling. The sensor was flown on a NASA ER2 jet at 65,000 ft (~20 km) over a ~30,000 km$^2$, an area roughly the size of a Landsat satellite image. There were 11 nominal flight runs over the San Francisco Bay Area per season. Each run had a ~12 km swath width and flight paths provided 20% image overlap among runs. Elevation in the study area ranged from 0 to 1186 m (mean = 188 m). Images were acquired on April 30 (spring), June 11 (summer), and November 2 (fall), in year 2015.

Georeferenced and orthorectified surface reflectance data were downloaded from the NASA Jet Propulsion Laboratory (JPL) website (http://aviris.jpl.nasa.gov). JPL resamples AVIRIS images to a fixed 18-m pixel size. For 30-m products, JPL spatially resamples at-sensor radiance data and adds a noise function to approximate the potential noise equivalent delta radiance function of HyspIRI [5]. Per-pixel apparent surface reflectance was retrieved by JPL from 18-m and 30-m radiance products using an improved version of ATREM [15]. Bands in SWIR with strong atmospheric water vapor absorption and poor signal-to-noise were removed, leaving 186 bands for analysis per season. The 18-m reflectance product, which is not intended to simulate HyspIRI data, was used for training the cloud mask. Clouds and water in images were masked prior to image classification following methods in [5].
2.2.4 Data Pre-Processing

Since CNN architectures work with contiguous data, each season's spectral profile (single-date imagery) was horizontally stacked to create a single three-season input vector $N_d = 186(spectral\ datapoints) \times 3(seasons)$. This is important to note for the subsequent feature map analysis.

Standardizing data for Neural Networks is a common practice and enables the network to operate on data that has a similar dynamic range [16]. 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.

$$X_{i, new\ spectra} = \frac{x_{i, raw\ spectra} - \mu_{i, raw\ spectra}}{\sigma_{i, raw\ spectra}} \text{ for } i = 1, \ldots, N_d$$

Where $N_d$ is the number of spectra or data, $\mu_{i, raw\ spectra}$ and $\sigma_{i, raw\ spectra}$ are the mean and standard deviation of all the training at the spectral index $i$. The figure below shows an example spectral profile before and after standardization.
This standardization of the data boosted performance of the CNN roughly 1% in classification accuracy and made the training more stable. This same standardized data was applied to RF and SVM and it did not make any noticeable impact on accuracy of these classifiers. As per standard practice, the mean and standard deviations for this standardization were developed from the training dataset and then extended to condition the validation data and classification data. This is to prevent any knowledge of the validation data to influence the training of the classifier.

2.2.5 Classifier Comparison and Accuracy Evaluations

The quantity of training and testing data for each class was not equal (Figure 6).
To not over represent classification accuracy, confusion matrices were created subjecting the respective classifier to the full test dataset and a balanced test dataset. The balanced validation data is a subset of the unbalanced validation data that consists of an equal number of data points per class. To maximize the number of samples assessed while still being equal per class, the minimum bin count of the population distribution (83 samples) was utilized for the number of data per class. By doing this the total number of points used for evaluation has been reduced, but it equally represents under-represented classes within the data. Both balanced and un-balanced confusion matrices are presented to provide a more complete comparison of the classifiers.
3 Results

3.1 Accuracy Assessment

All the classifiers were assessed with independent validation data. Each classifier was considered with respect to the overall accuracy when balanced population distributed data and un-balanced population distributed data were applied to the classifiers (Table 3).

*Table 3 Accuracy performance per classifier and data*

<table>
<thead>
<tr>
<th>Balanced - Data</th>
<th>CNN</th>
<th>SVM</th>
<th>RF</th>
</tr>
</thead>
<tbody>
<tr>
<td>Single Season</td>
<td>73.90%</td>
<td>74.80%</td>
<td>58.90%</td>
</tr>
<tr>
<td>Three Season</td>
<td>75.50%</td>
<td>74.00%</td>
<td>65.80%</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Un-Balanced - Data</th>
<th>CNN</th>
<th>SVM</th>
<th>RF</th>
</tr>
</thead>
<tbody>
<tr>
<td>Single Season</td>
<td>88.50%</td>
<td>89.50%</td>
<td>82.20%</td>
</tr>
<tr>
<td>Three Season</td>
<td>89.80%</td>
<td>91.00%</td>
<td>84.20%</td>
</tr>
</tbody>
</table>

The main results from this show that CNN and SVM are the predominant top performers, with Random Forests performing from 6.8 to 15% worse. The differences between the SVM and the CNN accuracies are interesting as the CNN shows to be on the order of 1% better than the SVM to generalize across all the classes when equally represented (i.e., balanced), whereas the SVM seems to better classify the data when it classifies data from a population distributed similar to the original training data. From this it would be expected that SVM would perform better if the classification data population distribution accurately reflected the training data population distribution. From a different perspective, the CNN seems to be a slightly more generalized classifier. If the training data did not have a similar population distribution as the data to be classified, the CNN would perform better. It is also important to note that across the classifiers three season data seemed to have a positive effect on the classification accuracies for all classifiers.
This should be expected, as this provides more data that the classifiers are able to utilize that may be class distinguishing.

This can be expanded for each classifier by showing the confusion matrices for each. As the three season data performed better it will be referred to throughout the rest of the analysis. Shown below are the 6 confusion matrices for each classifier with respect to balanced and un-balanced three-season data.

![Confusion Matrix: HSICNet - balanced](image)

*Figure 7 Confusion Matrix HSICNet (CNN) - balanced data. Call-out boxes show Producer’s Accuracy. Class index definition found in Table 1.*
Figure 8 Confusion Matrix HSICNet (CNN) – unbalanced data.
Figure 9 Confusion Matrix SVM - balanced data
Figure 10 Confusion Matrix SVM - unbalanced data
Figure 11 Confusion Matrix RF - balanced data
An interesting feature is the under-represented class 7 “Dune Vegetation” did not perform well with any classifier tested. This decreases the overall balanced classification accuracies but has relatively little effect on the unbalanced accuracies.

### 3.2 Feature Evaluation

The RF classifier is able to provide a ranking of the feature importance via the method of Mean Decrease Accuracy by effectively capturing the resulting impact of not utilizing a feature (spectra) for classification [17]. For the Random Forest classifier, the feature importance can be visualized as a plot (Figure 13). The top 10 features are in spring blue-green wavelengths and fall short wave IR wavelengths (Table 4).
Table 4 Top 10 Features for Random Forest classification

<table>
<thead>
<tr>
<th>Rank</th>
<th>Ranked Features for RF</th>
<th>Spring Spectra (nm)</th>
<th>Summer Spectra (nm)</th>
<th>Fall Spectra (nm)</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>4</td>
<td>394.9196593</td>
<td></td>
<td></td>
</tr>
<tr>
<td>2</td>
<td>6</td>
<td>414.2786593</td>
<td></td>
<td></td>
</tr>
<tr>
<td>3</td>
<td>5</td>
<td>404.5966593</td>
<td></td>
<td></td>
</tr>
<tr>
<td>4</td>
<td>3</td>
<td>385.2466593</td>
<td></td>
<td></td>
</tr>
<tr>
<td>5</td>
<td>7</td>
<td>423.9646593</td>
<td></td>
<td></td>
</tr>
<tr>
<td>6</td>
<td>497</td>
<td></td>
<td></td>
<td>1741.733659</td>
</tr>
<tr>
<td>7</td>
<td>501</td>
<td></td>
<td></td>
<td>1781.583659</td>
</tr>
<tr>
<td>8</td>
<td>498</td>
<td></td>
<td></td>
<td>1751.693659</td>
</tr>
<tr>
<td>9</td>
<td>8</td>
<td>433.6546593</td>
<td></td>
<td></td>
</tr>
<tr>
<td>10</td>
<td>9</td>
<td>443.3496593</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

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 showing to have
significant impact can then be analyzed further and regarded as a distinguishing characteristic of the data. The kernel importance matrix for this network is shown in Table 5.

Table 5 Kernel Importance Matrix. Class index definitions found in Table 1.

<table>
<thead>
<tr>
<th>Kernel</th>
<th>% Decrease from Baseline Classifier</th>
</tr>
</thead>
<tbody>
<tr>
<td>6</td>
<td>-1.2</td>
</tr>
<tr>
<td>5</td>
<td>-6.0</td>
</tr>
<tr>
<td>4</td>
<td>-2.4</td>
</tr>
<tr>
<td>3</td>
<td>-3.6</td>
</tr>
<tr>
<td>2</td>
<td>-13.3</td>
</tr>
<tr>
<td>1</td>
<td>-9.6</td>
</tr>
<tr>
<td>0</td>
<td>-3.6</td>
</tr>
</tbody>
</table>

The number of kernels trained were reduced to ensure that all the kernels were important to some degree. It was empirically determined that the number of kernels trained within the network could be reduced until the Kernel Importance Matrix always showed a negative total average % impact on the classification accuracy for all kernels. Reducing the number of kernels beyond this started to have a negative impact on the classification accuracy. While this was determined experimentally, it proved to be useful in reducing the total size of the network. Additionally, in this way the learned information is more condensed in fewer trainable kernels and the features that each kernel is extracting up should be more expressive.

A useful visualization to see what is being excited by the convolutional filters is by reviewing 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. By visualizing this convolutional feature map, it enables the identification of where within the 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.

It was observed that directly reviewing the raw feature map is slightly naive in this context and it should be formatted to be in the context and scale of the input data. As feature maps are connected to the same locations within the hidden layer, if the feature maps from each of the classes all are excited in the same area then they do not provide any distinctive information to the classification network and would have little effect in determining the class of the data. Additionally, as any deviation from zero within the feature map indicates that the kernel learned something in that region, this can be visualized as any deviation from zero within the feature map. Therefore, for visualization purposes standardizing the feature maps with respect to spectra and taking the magnitude of the feature map provide a more useful view of the feature maps. The scale of the feature map was stretched to the scale of the input data as well. The respective 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 each class illustrating typical structure of the spectral profile across all three seasons is
provided as a silhouette. Formatted feature map for class 0 ("Annual Crops") data and feature maps for all the classes other classes are shown in Figure 14 and Figure 15.

Figure 14 Convolutional feature map for class: Annual Crops
Figure 15 Convolutional feature maps for each class. These feature maps are the result of convolving the kernels and the input data.

An interesting aspect from this analysis is the excitation around areas of relatively low signal-to-noise, such as those in low blue, far shortwave infrared and wavelengths bordering the atmospheric absorption bands. Current engineered metrics look at various aspects of the data, of which are based on vegetation chemistry, physical phenomenon and structure. These engineered metrics utilize narrowband ratio (indices), spectral derivatives, and absorption-feature fitting techniques that span the different spectral ranges. All of these metrics occur in areas that are well behaved continuous areas and the areas around these low-fidelity areas are not thoroughly explored. Because some formatted feature maps are excited in these regions, this result indicates that that these areas are in fact important to the classification accuracy of the network and warrant
further investigation. This is traceable back to the kernel importance matrix (Table 5) showing that removing these kernels would negatively affect classification accuracy.

Other notable features are the different seasonal variations that are important and the distinguishing spectral regions for the individual class data. The data are represented within this network as three horizontally stacked datasets comprising a three-season long input vector. By observing under which third of the feature map excitations occur, distinct phenological information can be extracted about the classes. This effectively enables the feature maps to show under which season features are more prominent for respective classes than others.

With more temporal data, this type of analysis could help guide when to extract features or indicate subsets of data for additional analysis.

### 3.3 Performance

The computers utilized for this work were an 8 Core Xenon 3.7 Ghz processor with 32 GB ram and a NVIDIA K2200 GPU, as well as a second i7-2640M (2.8GHz) dual core processor with 8GB of RAM. As Tensorflow has been designed to scale from multiple-core, GPU(s), and multiple computer clustered configurations the CNN easily scaled to utilize a GPU. The scikit-learn library does not have this capability so RF and SVM were at a disadvantage when comparing the metrics. If different RF and SVM implementations were considered, the performance for these classifiers may be increased.

The CNN has been tested on both CPU and GPU based platforms on a single computer. The GPU based implementation performed all graph computations within the GPU. The performance increase moving from a CPU to GPU platform was on the order
of 5 to 8 times depending on the learning batch size (larger batch size increased performance within the GPU space).

The highest performing implementations of these classifiers performed within this work are provided in Error! Reference source not found. Error! Reference source not found. Error! Reference source not found. Error! Reference source not found. Error! Reference source not found. Table 6

Table 6 Classifier training and classification times

<table>
<thead>
<tr>
<th></th>
<th>CNN</th>
<th>SVM (s)</th>
<th>RF (s)</th>
</tr>
</thead>
<tbody>
<tr>
<td>to initial validation accuracy steady state (s)</td>
<td>1,500</td>
<td>358</td>
<td>3,452</td>
</tr>
<tr>
<td>4mil epoch standard training session (s)</td>
<td>21,146</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Training Time</td>
<td>3,452</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Classification Time (per Gb of data)</td>
<td>65</td>
<td>590</td>
<td>768</td>
</tr>
</tbody>
</table>

4 Discussions

4.1 Feature Map and Land Cover Classification Analysis

*Phenological Changes:* The usage of CNN is promising with respect to classification accuracy as well as visualization of features within the data. From Figure 15 the seasonal changes and contribution with respect to each class is visible. This visualization, paired with the confusion matrices, provide a point of exploration into the distinctive nature of the classes. Herbaceous land cover is a good example of this, as the location where the excited features are distinct with respect to the other classes are mainly occurring within the spring and summer regions. Within this timeframe, Herbaceous land cover grasses and forbs are likely growing and in the fall are senesced (non-photosynthetic) and apparently not providing any distinguishing reflectance features.
from other classes. Annual crops also show interesting excitations as there is some high excitations within the fall data collection. This could be due to the anthropogenic factors to manage the harvest of specific crops during seasons that other classes, such as Herbaceous; this class in particular, is not expressive. Other classes are naturally more expressive over the summer time frame. This is also seen in Closed-Canopy Deciduous Broad-Leaf Forests with a majority of the excitations occurring within the Summer seasons when canopies have full photosynthetic capacity (i.e., maximum leaf area index). When looking at Urban Vegetated Area, it makes sense that there is more uniform excitation throughout the seasons as they are maintained by municipalities and are in most contexts attempted to stay consistently photosynthetic with regular watering; i.e., green lawns, business park vegetation, golf courses, etc. This can also be said for the Bare class as well, where the spectra are almost uniformly excited across all three seasons of the feature map. This makes perfect sense as Bare, by definition should not contain any plant or land cover that would be influenced by the season, but rather the intra-season spectra content should dominate. This is visible by the additional excitation around the bad band areas within each season of the class data. In a sense, Bare can be considered to show the background excitation without natural or anthropogenic vegetation signal excitations. The Perennial Crops in contrast to the Annual Crops are much more skewed with features to be excited within the spring and summer seasons. This is likely because during the fall these crops are not very expressive due to leaf senescence timed with natural cycles, as compared to Annual Crops which are forced, due to harvesting and farming factors.
**Kernel Uniqueness:** An interesting aspect of the architecture and trained kernels is that all kernels are excited in similar regions to roughly the same amounts, at least on the same order of magnitude. This could likely be improved in future work to provide a regularizing learning feature on the kernels that enforces kernel uniqueness. Encouraging the kernels to be orthogonal may be able to further segment the feature space captured by the CNN, and provide more insight into what feature each kernel is filtering out from the data.

**Distinctive Features:** A key aspect of Figure 15 is that excitations are subtracted from the average excitation at that spectra. This figure thus shows the excitations that are distinct from the average. It is visible that areas where there is a change in reflectance are likely areas for distinction between classes. As the bad spectral bands are removed and seasonal data is horizontally stacked, these abrupt transitions are considered as distinctive features by the network. Table 5 and Figure 15 shows that these features do in fact contribute to classification accuracy and are learned to be distinctive. What is the cause for these transitions to be expressive of class uniqueness goes into the context of what land cover exists within each class and across which spectral region or transition it is excited. As mentioned, the Bare class is a good example as the intra-seasonal spectral content is the main area of excitation. This makes sense as this feature of the data should be consistent throughout the seasons and the seasonal changes from spring to summer or summer to fall should be minimal. Conversely, Perennial Crops are almost dominated by the Spring-Summer transition, where the seasonal change for this class is the salient feature. While this network only captures this features across the spectral domain it
could be extended in future work and give justification to learn patterns on a seasonal basis. This could be done by vertically stacking the data rather than horizontally stacking the data and training per-pixel 2-d kernels.

**Classified Land Cover Maps:** The maps created when classifying the data with the CNN and SVM are shown in Figure 16. It is visible that all classifiers segment the data relatively well, and are relatively consistent throughout their respective classification.

![Classified Land Cover Maps](image)

*Figure 16 RF, CNN & SVM classified land cover areas of San Francisco Bay Area. White areas indicate pixels that were not classified (e.g., water, no data)*

A close up review of several areas around this region provides additional insight.

A close up of the city of San Francisco shows all classifiers provide a similar
classification. Only upon even further close up inspection can some differences be seen. Notable features like Golden Gate Park and the Presidio are readily visible from all classifications. A discrepancy is the area near Hunters Point (east side of peninsula) where high-resolution satellite imagery shows this area should be classified as Bare and Herbaceous whereas the SVM does not make this determination. It also seems to make sense to trust the CNN classifier more for this class as the balanced accuracy classification was much higher for the CNN than the SVM, ~95% versus ~77% for the classification accuracy of the bare class (see confusion matrices).

Figure 17 RF, CNN & SVM classified land cover areas, San Francisco Peninsula
When reviewing classification of the San Francisco South Bay there is a significant portion of Salt Tidal and Marsh land that RF does not appropriately classify whereas the CNN and the SVM perform better. The differences in other locations are less significant between the SVM and CNN classifier, but there is a slight difference for the SVM classifier to not classify areas as Bare where the CNN would, similar to as what was previously mentioned. Upon close inspection of the Moffett Field air strip the areas in between the runways are classified as Bare and Herbaceous for the CNN and the SVM classified maps, respectively. This area has likely similar spectral components and is a
more challenging area for classification as due to the maintenance of the air field to keep local grasses low and similar to what would be expressed as Bare.

Figure 19 RF, CNN & SVM classified land cover areas, San Francisco South Bay

Figure 20 Satellite view of San Francisco South Bay
Reviewing the east bay of San Francisco the classification consists of more Annual and Perennial crops, and these classifications are roughly consistent throughout each of the classifiers.

Figure 21 RF, CNN & SVM classified land cover areas, San Francisco East Bay

Figure 22 Satellite View of San Francisco East Bay
The classification results from the North Bay shows the built-up regions inter-mixed with the upland grasses that span a majority of the region. This area seems to have some heavy inter-mixing of closely related spectral profiles, as the classification within much of the north bay has a significant amount of salt-pepper noise and mixing of classifications. While it is apparent that the classifications as viewed from the broad scale show very similar patterns, the noise apparent at a finer scale does not seem to be consistent or biased in any direction for each classifier, pointing to a higher degree of spectral mixing in these areas and in effect less inter-class spectral variability of this data.

*Figure 23 RF, CNN & SVM classified land cover areas, San Francisco North Bay*
4.2 Performance

For the methods utilized within this work, the classification time for the CNN is significantly better than both SVM and RF. However, performance should be considered cautiously. As SVM and RF are not utilized with GPU compatible implementations, it should be expected that their performance processing large amounts of data would be reduced. This could be potentially improved by augmenting the implementation or utilizing a different library.

Additionally, the training time for the CNN effectively can vary drastically depending on the training stopping criteria. A set epoch number of 4 M epochs was used.
as the stopping criteria for this work. This was done to attempt to reach the point of overtraining of the network where the network’s validation accuracy begins to decrease as well as ensuring that the training was not stopped at a local minimum on the training gradient. As overtraining was never reached within the tuning of this network, and some versions of the network did not converge as fast as the final implementation, the full duration of the 4M training epochs was used throughout development of testing hyper-parameters. This took a little over 6 times that of RF to train. When reviewing the CNN learning curve (Figure 25), it is apparent that the training asymptotes at roughly epoch 75,000. If the training was stopped there the accuracy of the classifier would have been the same, and the training time would have been reduced by around 14 times. As the network’s architecture is now less variable a more sophisticated stopping criteria could be applied to reduce the training time.

*Figure 25 CNN training - validation accuracy*
4.3 Future work

The application of Convolutional Neural Networks to hyperspectral data has only recently been explored and there are thus many avenues for future work. The consideration of the spatial domain within classification could enable significant improvements. Some recent work has been done in this regard and could be expanded [17,18]. In a general sense this could consist of considering multiple pixels at one time for classification of a specific pixel. For this consideration of the spatial dimensions as well as the spectral dimension, the input to the CNN would be a 3-dimensional tensor of $N_x \times N_y \times N_d$. With $N_x$ and $N_y$ being the number of pixels over a local region of HSI data and $N_d$ being the spectral dimension utilized within this work. The convolution would take place across the spatial domain, similar as what is done in standard image classification but also add information from the spectral domain. By incorporating the spatial context of pixels, this approach would be expected to alleviate some of the high frequency noise present within some of the existing land cover classifications.

An additional extension would be to vertically stack the temporal season data creating a $N_d \times N_{ns}$ sized input data array, where $N_d$ is the number of data points within a single seasons data capture and $N_{ns}$ is the number of seasons.

These two data formats would dictate architectures of networks that could convolve higher dimension kernels across the input data, or perform a multiple layered convolutional operations. The additional benefit would be that the kernel(s) would enable the network to capture patterns within the spatial, 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, by a similar review of the convolutional feature maps as previously presented.

It was observed when reviewing the feature maps that the kernels were exciting the convolutional feature maps in similar regions, across all kernels. This points to the idea that the kernels are not capturing unique information. If these kernels were “encouraged” to learn information that is “unique” it could enable the feature maps to show different structural features. One potential way to do this would be to provide a regularization on the kernels that would penalize kernels that were similar in some regard.

As datasets from future satellites are more readily available, the scaleability of implementations should be considered. While the CNN classifier in this study was not specifically developed to be extended to multiple computers, the Tensorflow™ platform supports this extension to be scalable beyond CPU and GPU based platforms to clustered and multiple computer computing platforms. Future development to scale this network to multiple computers to handle larger datasets and perform in-depth performance studies could be performed.

5 Conclusion
A broad goal of this work was to implement and explain how a Convolutional Neural Network could be applied to multi-temporal hyperspectral images. The results show that a CNN architectures can be applied to simulated spaceborne hyperspectral reflectance data (HyspIRI) to achieve high classification accuracy rates comparable to Support Vector Machines, and surpassing Random Forest classifiers.
Unbalanced/Balanced accuracies of 91.0%/74.0% for SVM, 89.8%/75.5% for CNN and 84.2%/65.8% for RF were demonstrated. Spectral and temporal information is readily visible through feature map visualizations and their respective importance is traceable back to kernel importance. Maps generated through the classification of these data provide useful information about the land cover and from a qualitative perspective seem realistic. In summary, the CNN is a promising classifier for future HSI classification tasks and this study identifies future work to increase CNN performance, scalability and incorporation of spatial and temporal information.
6 References


7 Supplemental Material

7.1 Code
The deliverable work products from this work consist of the python code shown below. To run this code Tensorflow 0.7.1 was utilized with CUDA 7.5 for GPU support. The numpy, scipy, and scikit-learn python libraries are required to run the RF and SVM classifiers. The code files shown below were provided to the graduate committee to accompany this report.

**HSIcNet_Main.py**
The code contained within this file consists of the main thread that:

- imports data
- instantiates the HSIcNet classifier
- trains the classifier
- produces visuals
- creates classification maps.

**RF_SVM_Main.py**
The code contained within this file consists of the main thread that:

- imports data
- instantiates the the RF and SVM classifiers: creates or recalls
- trains the classifiers
- produces visuals
- creates classification maps

**DataIO_main.py**
The code contained within this file provides the means to test and implement the HIS_data class.

**HSI_data.py**
The code contained within this file consists of the data structure to captured data as provided from the Center for Interdisciplinary Geospatial Analysis. Randomization and standardization helper functions are provided with this class. The functions to provide the generation of the GEOTiff classified maps are provided within this class.

**HSIcNet.py**
This code defines the implementation interface to:

- define
- train
- classify
- save
- recall
- kernel extraction
- kernel impact: calculations and visualizations

for the CNN architecture as described within this network. The impact matrix requires the modification of the network parameters and is so captured with in this code, helper functions to support this are provided within.

**Kernel_eval.py**
This code consists of kernel analysis functions. Providing feature map generation. Ancillary FIR filter analysis functions have been implemented here for future development.
7.2 **Saved Classifiers**
The classifiers utilized to develop the results described within this report are described below.

- CNN Classifier Filename: CNN_stdzd_Classifier-3Season
- CNN Graph Parameters: GraphParams_CNN_stdzd_Classifier_3Season.PKL
- SVM Classifier Filename: SVM_stdzd_Classifier_3Season.PKL
- RF Classifier Filename: RF-stdzd_Classifier_3Season.PKL

7.3 **Data Files**
The training data file utilized for the training of this report is:

- Bay_Area_2015_hyspiri_reflectance_30m_3season_training_filtered.csv

The data utilized to create the classification maps for each of the classifiers is on the order of hundreds of gigabytes and would be cumbersome to provide with this report and will not be provided. Dr. Matthew Clark should be contacted to gain access to this data.