Hyperclass: A Framework for Hyperspectral Data Visualization and Analysis

NASA Goddard Innovation Lab

Abstract

Hyperclass is an interactive workbench supporting visual data analysis of sensor data. It provides an extendable interactive interface and toolsuite which can be used to jumpstart the development of novel methods for addressing a wide range of data analysis challenges in both the earth and space sciences. We have chosen, as a science driver for the initial stage of development, the development of innovative semi-supervised machine learning methods for landscape classification using hyperspectral imagery.

1. Introduction

The NASA Land-Cover/Land-Use Change (LCLUC) program is aimed at using satellite observations to improve our understanding of LCLUC as an important component of Earth System Science. The LCLUC program includes studies that detect and quantify changes in land cover and land use; examine their impact on the environment, climate, and society; or model future scenarios of LCLUC and its various impacts and feedbacks. The LCLUC program is aimed at developing the capability for periodic satellite-based inventories of land cover to characterize and monitor changes in land cover and land use.

Hyperspectral remote sense imaging technology, HSI, is widely used for monitoring Earth's surface. HSI, which provides better discrimination among land cover classes then traditional multispectral sensors with low spectral resolution, is becoming increasingly important with NASA's upcoming deployment of new hyperspectral sensors. The sensors provide a vast amount of spectral and spatial information, which is exploited in HSI classification applications for agriculture, environmental management, urban planning, mineral detection and urban mapping.

Hyperspectral image classification deals with the problem of pixel-wise labeling of the hyperspectral spectrum. With the recent development of new hyperspectral remote capturing sensors, HSIs normally contain millions of pixels with hundreds of spectral wavelengths (channels), making HIS classification intrinsically challenging. While huge volumes of HSI data are increasingly becoming available, ground truth labels remain scarce, due to the immense manual efforts required to collect them. Semi-supervised machine learning is an effective method for addressing the shortage-of-labels problem.

2. Semi-Supervised Machine Learning

Semi-supervised machine learning (SSL) can be viewed as an intermediate approach between supervised (CNN, SVM, Random Forest) and unsupervised (K-Means, DBScan) leaning methods. Given a relatively small number of labeled points, the SSL algorithm attempts to infer the labels of other similar points, thus greatly increasing the number of training samples for the classifier.

We formulate our SSL approach by representing each spatial location in the HS image as a point in an NB dimensional spectral space, where NB is the number of bands. The collection of points representing the current image tile can be viewed as a manifold in the spectral space, with an NB-dimensional Euclidean metric providing a distance measure. Hyperclass captures the structure of this manifold by constructing a NN nearest-neighbor graph, where NN (the number of neighbors) ranges from 5 to 20 (typically 8). The SSL algorithm is then based on the following two assumptions: 1) nearby points in the spectral space are likely to have the same labels, and 2) points on the same spectral manifold are likely to have the same labels. The second assumption allows us to use the structure of the unlabeled points to help guide the inference of new labels.

With this background we can delineate the SSL workflow as the following sequence of operations:

1) The Point Labeling Console (Section 4) is used to assign labels, representing land cover classes such as water, forest, bare earth, etc., to spatial locations in the hyperspectral image.
2) An extended set of labels is inferred using the Hyperclass inference algorithm (Section 5).
3) The labels, defined in the 400-dimensional image space, are embedded in a reduced (e.g. 10) dimensional space using the UMAP algorithm.
4) A linear Support Vector Classifier (SVC) is trained using the labels in the reduced dimensional space in order to compute the optimal classification boundaries.
5) This classification can then be applied to a different block/tile by applying the UMAP transformation computed in step 3 to reduce the dimensionality of the new data and then applying the classification learned in step 4 to assign labels to the spatial locations in the image.

3. Data Management

A single hyperspectral can quite large, so that it may be computationally impractical to process the entire image at one time. Hyperclass addresses this issue be splitting an image into square tiles and loads only a single tile into memory at any one time. Each tile is then split into square blocks, and only a single block is displayed in the Hyperclass Console at any one time. The hyperclass configuration parameters can be used to set the tile and block sizes. The 'Tiles' menu can be used to easily load a different tile and/or block.

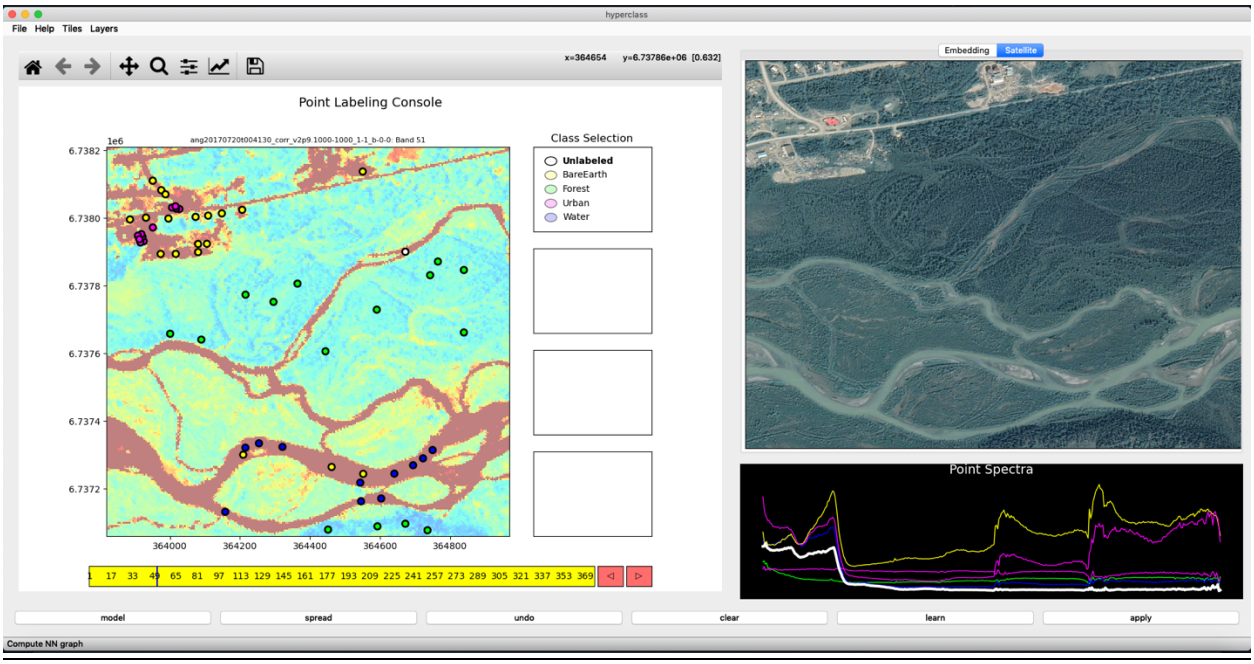

Figure 1. The Hyperclass console

4. Console GUI

The hyperclass console (shown above) provides the GUI for all operations performed using hyperclass.

The hyperclass console is composed of the following panels plus a toolbar:

- **Point Labeling Console**

  The Point Labeling Console (Figure 1) is used to display spatial maps of the spectral bands and allocate classification labels to points on the map. It allows users to browse or step through the band images using the selection tool at the bottom of the panel. It has the following features:

  - *Navigation controls*, located at the top left of the panel, enable zoom and pan operations on each band image.
  - *Information box*, located at the top right of the panel, displays the coordinates and data value for any selected point on the image.
  - *Class Selection widget*, located on the right side of the panel, is used to specify the current class for labeling purposes.
  - *Label assignment* (representing the current class) occurs by left-clicking on a point on the image. Any label can be deleted by right-clicking on it.
  - *Probe Label* (represented by the 'Unlabeled class') is a special class that can be used for displaying information about specific locations on the image without assigning a label. There is never more than one 'Unlabeled' label on the image at any given time.
  - *Transparency Configuration,* available through the 'Layers' menu, enables separate control of the transparency of the band and the inherited label layers, for customized visibility.

- **Embedding Panel**

  The Embedding Panel (Figure 2) is used to display a UMAP embedding of the current hyperspectral block in three dimensions. The collection of points representing the current block can be viewed as a manifold in the (NB-dimensional) spectral space. Hyperclass captures the structure of this manifold by constructing a NN nearest-neighbor graph, where NN (the number of neighbors) ranges from 5 to 20 (typically 8). The UMAP mapping operation creates an embedding of this graph in three-dimensional space which endeavors to preserve the structure of the manifold. This embedding is visualized as a point cloud in the Embedding Panel. Each point in the cloud corresponds to a specific location in the raster image. Visualizing the manifold structure of the hyperspectral dataset gives important insights to guide further analysis and training. The embedding panel has the following features:

  - *Label Display*: The UMAP embedding of every label that is assigned in the Labeling Console is displayed as a marker in the Embedding Panel. As the labels are spread across the graph, the points in the point cloud are assigned the color of their inferred label.
  - *Manifold Mapping*: Executing an option-<right click> on a point in the point cloud places markers on both the selected point and the corresponding raster location in the Labeling Console.
  - *Point size configuration*, available through the 'Layers' menu, allows the user to adjust the size of the point in Embedding Panel for customized visibility.
  - *Navigation* in the Embedding Panel is enabled using click-and-drag operations: Left click -> pan/rotate, Right click -> zoom.

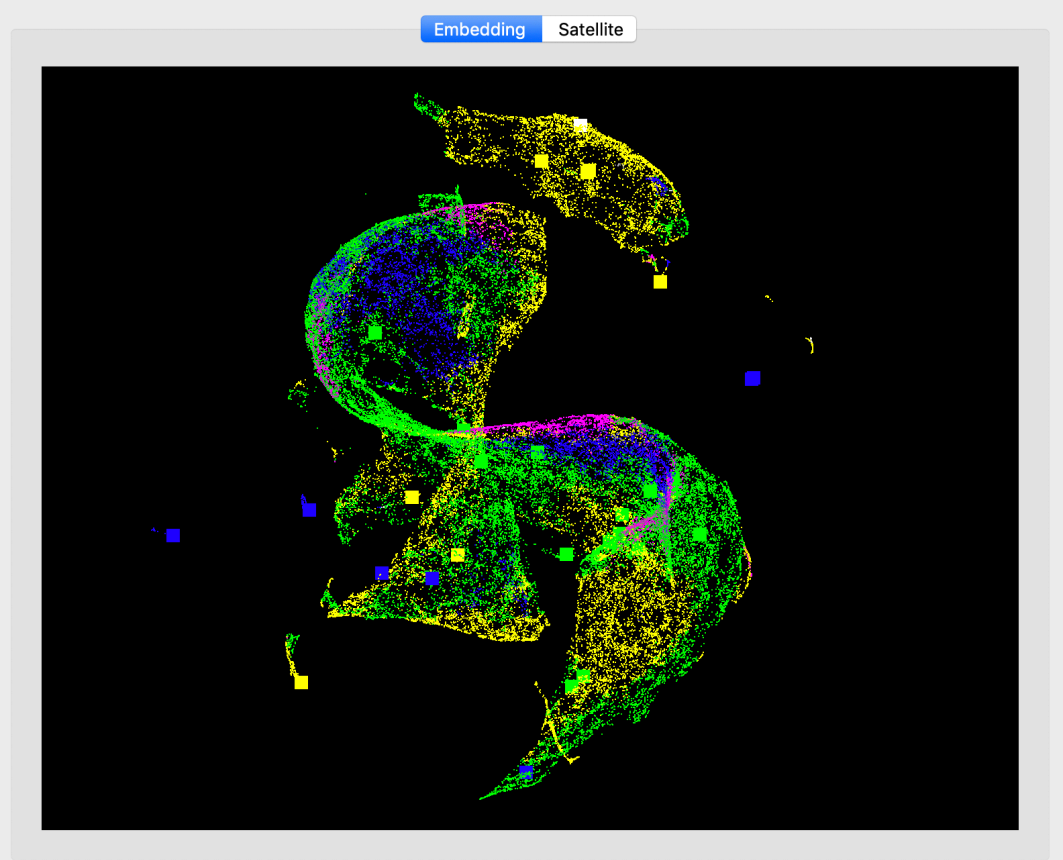

Figure 2. The Hyperclass Embedding Panel

- **Satellite Panel**

  The satellite panel displays a google maps satellite image of the region displayed in the Labeling Console. This view assists in choosing appropriate labels for spatial locations. The satellite panel has the following features:

  - *Synchronized navigation:* Pan and zoom operations performed in the Labeling Console are reflected in the Satellite Panel, so that the visual region in the Satellite Panel always matches that of the Labeling Console.
  - *Point Labeling:* Left-clicking on a point in the Satellite Panel creates a label at the corresponding spatial location in the Labeling Console. The unlabeled class can be used for exploration purposes.

- **Point Spectral Panel**

  The Point Spectral Panel is used to display a line plot of the band values at a selected spatial location. Visualizing the full spectrum at various locations gives important information regarding the spectral separability of the corresponding classes. A spectral plot is displayed in this panel whenever a point is selected in the Labeling Console, regardless of whether that selection originated in the Labeling Console, the Satellite Panel, or the Embedding Panel.

- **Actions Panel**

The Actions Panel, a buttonbox across the bottom of the console, is used to execute the steps in the semi-supervised machine learning workflow. It provides the following action buttons:

- *Model:* Computes a UMAP embedding of the NN-graph representing the block of hyperspectral data displayed in the Labeling Console. If labels have been assigned to the data then a (semi-)supervised embedding will be computed. The results of the 3D embedding are displayed as a point cloud in the Embedding Panel (as explained in the section above).
- *Spread:* Given a set of user-specified labels, this action propagates the labels across the NN-graph, so that points that are similar to the current labeled points inherit the label of the most similar point. The spread action initiates 5 iterations of the spreading label algorithm (the algorithm is explained in more detail in Appendix 1). The inherited labels are displayed on the UMAP embedding of the NN-graph and, projected back into geographic space, on the band image in the Labeling Console.
- *Learn:* This action uses the current set of labeled point to learn a classification mapping that can be applied to other tiles/blocks. First, first a supervised UMAP embedding is computed to reduce the dimensionality of the data (the number of reduced dimensions, typically 8, is one of the settings parameters). Next, a linear SVM classifier is trained on the labeled points in the reduced dimensional space.
- *Apply:* This step applied the classification that is learned in the previous step to a new block of data. First, the UMAP transform that is computed in the previous step is used to reduce the dimensionality of the data. Then the SVM classifier that was trained in the previous step is used to classify the data. The result of the classification operation is displayed in the Labeling Panel.
- *Undo:* The action will undo the last operation. It is typically used to undo a label assignment in the Labeling Panel.
- *Clear:* This action clears all the assigned and inherited labels, as well as their displays in the various panels, placing the data block in its original state.

5. Appendix: Manifold-Aware Label Inference

Given a relatively small number of labeled points, this algorithm attempts to infer the labels of other similar points, thus greatly increasing the number of training samples for the classifier. It is based on the assumption that the collection of points representing the current image tile can be viewed as a manifold in the (NB-dimensional, NB=number of bands) spectral space, with an NB-dimensional Euclidean metric providing a distance measure. Hyperclass captures the structure of this manifold by constructing a NN nearest-neighbor graph, where NN (the number of neighbors) ranges from 5 to 20 (typically 8). Each edge in this graph is weighted by the distance between the connected vertices in spectral space. The graph distance between any pair of connected vertices is defined as the weight of the connecting edge, and the distance along any path through the graph is defined as the sum of the weights of all edges traversed in that path. We can then define the graph distance between any two vertices of the graph as the distance along the shortest path between them.

Using this framework we can define a simple method for inferring the label of any unlabeled vertex V. Assume we have a set of unlabeled vertices C. We can then compute the graph distance from V to all of the vertices in C and assign to V the class of the vertex in C which is closest (minimum graph distance). In order to effectively perform this operation we must construct an algorithm which efficiently computes the distance along the shortest path from any given point to each of the labeled points. Computational tractability imposes the following constraints on this algorithm: 1) a brute-force search over paths to find the shortest is intractable, 2) since the algorithm is being implemented in python we must avoid iterating over vertices (since large iterations in python are extremely inefficient and the number of vertices could easily be ~500,000). Given these constraints, the hyperclass incremental spreading activation algorithm, implemented using numpy arrays, provides a very efficient method for implementing the label inference method described above. If this algorithm is run to convergence, then every point in the graph will be assigned the label of the closest

(minimum graph distance) labeled point. Alternately, one can iterate the algorithm a limited number of times to provide a partial labeling which covers those vertices that are closest to the preexisting labeled vertices.