HyperHound: a Framework for Hyperspectral Image Analysis and Target Detection using Deep Learning Models

Rosario Di Carlo[1,*], Roberto Morelli[1] and Alessandro Nicolosi[2]

[1]Lab of Artificial Intelligence, Leonardo Labs, Via Pieragostini 80, Genova, 16149, Italy
[2]Lab of Artificial Intelligence, Leonardo Labs, Via Tiburtina Km. 12.400, Roma, 00156, Italy

Abstract

Hyperspectral images have shown great potential for the target detection task. These images collect the reflectance physical value over a large electromagnetic spectrum providing a fingerprint that characterizes uniquely distinct materials. In this work, a framework is developed to recognize different materials using several approaches ranging from classical methods to deep learning ones. Different learning paradigms are investigated considering both supervised and self-supervised methods. The main difference between these approaches concerns the labeling process. Indeed, while the former method requires labeling the data, the latter approach is based on pseudo-labels generation described in this contribution.

Keywords

Hyperspectral, deep learning, target detection, HSI

1. Introduction

Hyperspectral imaging (HSI) [1] is an advanced technology that allows for the collection of a wide range of spectral data acquired by remote sensors. It has been shown to be useful for various applications, including object detection, classification, and material recognition. In particular, hyperspectral images provide unique material fingerprints that can be used to identify different materials.

In recent years, there has been an increasing interest in developing machine learning models that can accurately recognize materials from hyperspectral images. Deep learning has emerged as a promising approach to solving complex problems in various fields. Among the different deep learning models, convolutional neural networks (CNNs) [2] have become dominant for processing visual-related tasks. The concept of CNNs was first introduced in a paper by LeCun et al. [3] and has since been improved upon by subsequent research [4] and refined and simplified by other studies [5] [6] .

This paper proposes a framework that leverages both classical and deep learning approaches for material recognition in hyperspectral images. Different learning paradigms, including supervised and self-supervised methods, are investigated and evaluated for their performance on a benchmark dataset. The approach demonstrates promising results and can have practical applications in fields such as remote sensing, geology, environmental monitoring, and target detection.

Ital-IA 2023: 3rd National Conference on Artificial Intelligence, organized by CINI, May 29–31, 2023, Pisa, Italy
*Corresponding author.
rosario.dicarlo.ext@leonardo.com (R. D. Carlo); roberto.morelli.ext@leonardo.com (R. Morelli); alessandro.nicolosi@leonardo.com (A. Nicolosi)
0000-0002-3616-5507 (R. D. Carlo); 0000-0001-5090-9026 (R. Morelli); 0009-0007-5071-5687 (A. Nicolosi)

2. HyperHound Framework

HyperHound is a framework developed specifically for analyzing hyperspectral images. It has been designed to allow for easy implementation and testing of various models for target detection. This framework comes with a broad range of capabilities, with its main features described below and provided with a simple user interface (UI) shown in Fig. 1:

- **Compatibility with the PIX format**: support loading files in PCI (Geomatics Database File) format, splitting them into smaller patches, and visualizing them for the analysis process.
- **Datasets**: Integration of both publicly available and privately collected datasets to enable model evaluation and comprehensive data analysis.
- **Implementation of classic target detection models**: target detection is a critical task in computer vision, which involves identifying specific objects of interest within an image, HyperHound implements several algorithms that provide a solid foundation for measuring the performance of newer and more advanced models. Implementing these classic models allows us to compare the results of different models and evaluate their relative strengths and weaknesses. Some of the classical models implemented are Euclidean distance, CEM, MF, and ACE.
- **Data labeling**: the interface of HyperHound provides two options for labeling data, individual

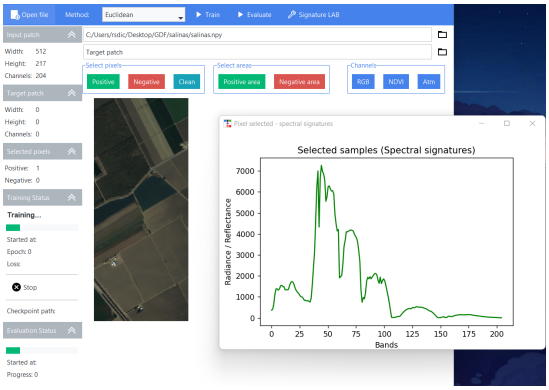

Figure 1: UI of Hyperhound framework loading Salinas data and analyzing a spectral signature of the selected pixel.

pixel labeling and bounding box selection. The labeled data can be used to train a classification model.

- **Functionalities of Inference and Training**: HyperHound implements functionalities to both perform inference with pre- trained deep learning models and training models on the fly from the interface. The inference process is optimized by splitting the input image into smaller slices and processing them in parallel on a GPU.
- **Database of spectral signatures**: consisting of laboratory- sampled materials collected from online sources. This resource enables comparisons between the reflectance of individual pixels and available materials, enabling the computation of similarity scores.
- **Atmospheric correction**: Integration with Py6S [7], a Python implementation of the 6S model [8], to compute atmospheric correction of the spectral image according to the atmospheric conditions during the data acquisition.

3. Methods

3.1. Standard Methods

Classical hyperspectral image target detection algorithms, such as Spectral Angle Mapper (SAM) [9] and Spectral Information Divergence (SID) [10] are two straightforward detection algorithms that measure the "distance" between the spectrum of the test pixel and the prior spectral signature of the target. Also Constrained Energy Minimization (CEM) [11] [12] matched filter (MF) [13], and adaptive coherence/cosine estimator (ACE) [14] [15] are typically developed using constrained least square regression methods or hypothesis testing methods that assume a Gaussian distribution. However, real-world hyperspectral data obtained through remote sensing often exhibits strong non-linearity and non-Gaussianity, which can result in a decline in the performance of these classical detection algorithms.

3.2. Self-supervised

Self-supervised learning is a type of machine-learning technique in which a model is trained to learn patterns and relationships within a dataset without the need for explicit labeling or supervision. For the scope of this work, this method is used to learn a space topology to cluster similar hyperspectral signatures. In this sense, starting from a reference signature, this algorithm can detect similar targets from the images analyzed. To overcome the labeling burden, an unsupervised method is used to generate pseudo-labels. The strategy used in this work is described in the evaluation and results section and leverages a clustering pre-text task. Once pseudo-labels are generated, contrastive learning is used to train the model to cluster properly signatures belonging to distinct classes. It is worth remembering that these are the classes defined in a self-supervised manner, that is, using an unsupervised pre-text task. A fully connected neural network was chosen to learn the distance metric for class discrimination.

3.3. Fully-connected neural network (FCNN)

A fully-connected neural network consists of a series of fully connected layers that connect every neuron in one layer to every neuron in the other layer.

Each neuron represents a computational unit that processes its input and passes its results to each neuron of the next layer. Layer by layer a hierarchical representation of the input is learned to improve the classification task that consists of producing a probability for each pixel to belong to the target object. For the hyperspectral images, the input of the FCNN is represented by all the channels of a single image pixel that are processed consequently by all the fully-connected layers. Indeed, the first layer of the network has an input dimension equal to the hyperspectral channels while the other layers have a number of neurons that gradually decreases. The last layer has a number of neurons equal to the dimension of the code used to encode the pixel given in the input.

Indeed, the network is trained to encode the input into a sequence of numbers in a latent space. In this way, pixels belonging to the same class are clustered together to reduce their distance into the latent space. To promote this behavior, the training proceeds by means of a metric learning approach as explained at the beginning of this paragraph.

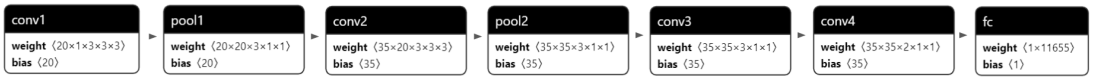

Figure 2: CNN3D Architecture

3.4. Supervised

The supervised learning method involves training a model using labeled training data, which consists of a set of inputs and their corresponding outputs or class labels. The model's parameters are updated iteratively during the training phase to accurately predict the desired outputs. In the testing phase, the model is evaluated against new input or test data to assess its ability to predict the correct labels. With sufficient training, the model can predict the labels of new input data. However, this approach requires a large amount of labeled training data to fine-tune the model parameters. Therefore, it is most appropriate for situations where much-labeled data is available. The HyperHound framework facilitates this labeling process and the following training. The model adopted to test the framework is a convolutional neural network with 3D convolutional filters.

3.5. 3D Convolutional neural network (3D-CNN)

Identifying ground objects in hyperspectral imaging requires both spectral and spatial information. To effectively classify these objects, a 3D convolutional neural network (CNN) was implemented. The network processes each pixel of the images by considering the relation between adjacent channels, in addition to spatial patterns across neighboring pixels. The input of the 3D-CNN is a patch of 7x7 pixels, where **N** is the number of channels in the hyperspectral image. The architecture consists of a series of 3D convolutional layers, with decreasing filter numbers leading to the last fully- connected layer. This final layer takes the flattened concatenation of a set of feature maps from the last convolutional layers as input and outputs the probability of the center pixel of the input patch belonging to a target object. A scheme of this neural network is reported in Fig. 2.

3.6. Hyperparameters optimization

The model included in the HyperHound framework is the result of extensive hyperparameter optimization. To scale the search for optimizing hyperparameters, Ray Tune, a Python library designed to execute experiments and tune hyperparameters at any scale, was utilized. This was done using the Leonardo HPC system, specifically the Davinci-1 infrastructure, which comprises a total of 80 nodes, each equipped with four Nvidia A100 GPUs.

4. Evaluation and Results

The following paragraphs begin by presenting one of the datasets used to validate the self-supervised approach. Subsequently, the training and validation processes were expanded to other hyperspectral datasets [16]. Finally, the supervised method, including the labeling process and performance evaluation, is reported.

4.1. Salinas

Salinas is a hyperspectral dataset collected by the 224-band AVIRIS sensor over Salinas Valley, California, and is characterized by high spatial resolution (3.7-meter pixels). The area covered comprises 512 lines by 217 samples. 20 water absorption bands were discarded: [108-112], [154-167], for a total number of bands equal to 224. This image was available only as at-sensor radiance data. It includes vegetables, bare soils, and vineyard fields. Salinas ground-truth contains 16 classes.

4.2. Data Labelling

The self-supervised approach for labeling involves generating samples that are labeled without full supervision. One method for accomplishing this is through the use of endmembers, which are defined as pure spectral signatures that can be linearly combined to represent the hyperspectral image pixels. Endmembers can be thought of as the basis vectors of a geometrical subspace. During image acquisition, due to the relatively low spatial resolution of hyperspectral sensors, some pixels may collect a mix of signatures from different materials. This means that each pixel can be seen as a superimposition of each endmember. By identifying the endmembers in the hyperspectral image, it is possible to obtain a set of pure spectral signatures that can be used to label the image data. Once the endmembers have been identified, they can be used in a variety of ways to label the image data. For example, one approach is to use spectral unmixing to estimate the abundance fractions of each endmember in each pixel. Nevertheless, some methods exist to unmix the pixel to find the basic constituents of each material, but their application doesn't guarantee the optimality

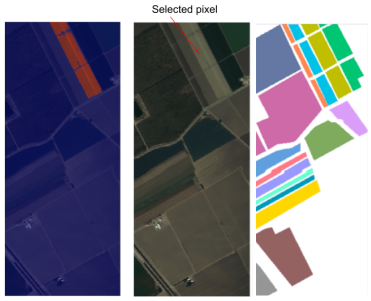

Figure 3: Evaluation on Salinas dataset using only one selected target pixel belonging to the class "Stubble". Heatmap of the classification on the left, Selected target pixel on the center, GT on the right.

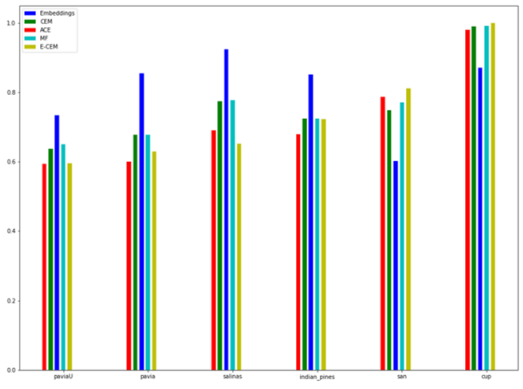

Figure 4: Comparison of the mean AUC on different datasets. The blue bar refers to the self-supervised model performance.

of the solution. Indeed, the method used should define both the exact number of pure signatures inside a picture and the relative abundances of each end member that represents the coefficient of the linear mixing. Both these parameters are unknown and so the solutions are ill-defined. However, a guess about the number of endmembers is made to perform the unmixing. Once the endmembers are defined, the dataset generation can be provided by sampling the coefficients that define their linear mixing following the equation:

$$x_i = \sum_{i}^{n} e_i \times c_i; . \tag{1}$$

where c_i are the randomly generated coefficients and e_i represents the n endmembers extracted from the source image. This sampling is repeated to generate all the dataset samples. The key step in this process is the labeling step, where the endmember corresponding to the highest coefficient is used to label the sample, in other words:

$$y_i = Argmax(c_i); . \tag{2}$$

So, in the end, a dataset with a custom number of samples is generated with several classes equal to the number of endmembers.

4.3. Performance

All the models were evaluated on different hyperspectral datasets, each containing one or multiple classes to detect. For each dataset, one of the classes was designated as the target class, while the others were considered background classes.

To identify the target class, a representative pixel was selected and the distance between that pixel and all other pixels in the dataset was computed. If the similarity score between the target pixel and another pixel was above a certain threshold, that pixel was classified as a target pixel; otherwise, it was classified as a background pixel.

This process was repeated for each class, with the threshold value varied to calculate the relative area under the curve (AUC) performances. The final AUC value was calculated by taking the mean value over all the classes. Using this approach, the ability of different models to detect target objects in hyperspectral datasets with multiple classes can be accurately measured. In Fig. 3 is shown the inference results of the trained model, which was able to successfully detect the target class. Fig. 4 shows the results on 6 datasets: Pavia University, Pavia Centre, Salinas, Indian Pines, San Diego, Cuprite. It can be observed that the self- supervised model outperforms other models on most of the tested datasets.

It is important to highlight that the datasets used in the study consist of a single image with pixel-based labeling, resulting in a scarcity of data variability. This may lead to a high correlation between the training and validation sets, which could negatively impact the model's robustness. This limitation can be observed in the paragraph below evaluating the same models on data acquired with real-world conditions variability.

4.4. Proprietary dataset

The dataset used for the supervised task is a proprietary dataset. It consists of 4 images collected with a hyperspectral sensor during an aerial acquisition. The images were pre- processed by performing the L1 pre-processing chain, which consists of the following operations:

- Spectral and Radiometric Calibration
- Geo-Referencing
- Geo-Rectification

Table 1
Dataset train validation and test splitting proportion

Data	Train	Validation	Test
Tiles containing the target (613×613)	11		7
Patches (7×7)	1050	450	Full tiles

These operations are missing the L2 pre-processing chain, which involves atmospheric correction and conversion of values to reflectance. In addition, the images have artifacts probably due to the vibrations the sensor was subjected to during the flight. With these limitations, a single pixel may contain a mixture of multiple hyperspectral signatures making the detection task harder.

4.5. Data Labelling

Each of the 4 images was cropped into 200 smaller tiles, each measuring 613 × 613 in size, for a total of 800 tiles. Through ground surveys, it was determined that 18 of these tiles contained the targets to identify. Of these 18 tiles, 11 were included in the training-validation sets, while the remaining 7 tiles were used to test the models. The labeling process is provided using the HyperHound interface. Through this interface, it is possible to display an image and collect a set of pixels to represent both target and background samples. This collection can be performed by using both bounding boxes or dot annotations, for a finer pixel selection. This procedure was repeated on all 18 tiles used for the training, validation, and test. A patch of dimension 7x7 was cropped around each pixel collected to provide the input in the form of images to the 3D CNN used for the training. The total number of patches collected for training and validation was nearly 1500 with a proportion of 1:14 between target and background samples. The partition of data into training, validation, and testing is summarized in Table 1.

4.6. Performance

The objective was to identify target areas, and a detection metric was used to evaluate the model's performance. The F_1 score was chosen as the evaluation metric since it handles class imbalances better than accuracy and other metrics. An algorithm was developed to associate the model's predictions with the ground-truth labels, and assess the model's performance. The output of the model is a heat-map representing the probability of a pixel belonging to a target area. Therefore, the first step was to apply a threshold to obtain a binary mask, where each cluster of fully-connected pixels represents a predicted object. Subsequently, if a predicted object partially or fully overlaps with an object in the ground-truth mask,

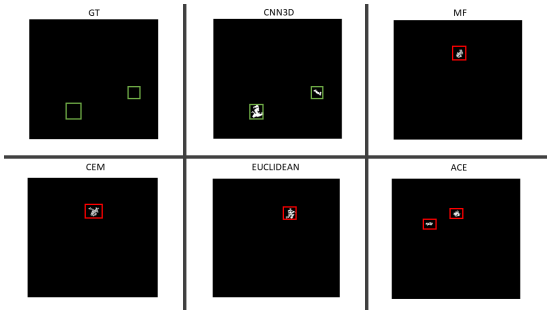

Figure 5: Comparison between the proposed model (CNN3D, center top row) and the other standard methods on the tile n°1 of the test set.

Table 2
Performance metrics.

Tile Num.	TP	FP	FN
1	2	0	0
2	2	2	0
3	1	0	0
4	0	3	2
5	1	1	0
6	1	2	0
7	1	1	0

it is considered a true positive (TP). On the other hand, if there is no overlap between a predicted object and a ground-truth object, it is considered a false positive (FP). Finally, if a ground-truth label is not associated with any predicted object, the false negative (FN) count is increased by one unit. The results are provided in the table (2).

The proposed model was evaluated on seven images that were not included in the training or validation set and achieved an F_1 score of 0.6 in identifying the targets. An example of detection comparison on a test image is reported in Fig. 5. The first patch (top left corner) represents the ground truth, that is, a completely black image with green boxes corresponding to the targets to detect. The image is black in order to preserve sensitive information and the original image pertaining to this test is not shown. The remaining patches represent the predictions of all the competing methods. Notably, only the CNN3D was able to detect correctly all the targets

on this test tile.

5. Conclusions

This article presents the HyperHound framework, which has been developed for hyperspectral image analysis. The framework provides an effective solution for analyzing hyperspectral data by applying deep learning techniques. Two types of deep learning models were analyzed using the framework: self-supervised and supervised. The self-supervised approach is particularly useful in addressing the challenges of a lack of labeled data and the difficulty of pixel-level ground truth annotation. The model learns to predict features from the input data itself, without any explicit supervision. This approach is particularly effective when the ground truth data is not available, and it has shown good results in many literature datasets. However, the self- supervised models are less robust and their detection metrics are generally lower compared with supervised models. The supervised model, on the other hand, utilizes ground truth data to train the model. This type of model yields good results, providing accurate results even under different real-world conditions where classical and unsupervised models often fail.

In this study, it is highlighted that many hyperspectral datasets used as benchmarks lack sufficient data, and the training and validation data are often highly correlated, resulting in models that are not robust to different real-world conditions. However, the supervised models have shown significant improvement and are particularly useful for man-in-the-loop applications. They provide an excellent tool for guiding and facilitating the task of an expert analyst in identifying targets, which is a challenging task in hyperspectral data analysis. Therefore, the HyperHound framework and supervised models provide a promising direction for hyperspectral data analysis, and they hold great potential for addressing the challenges of real-world applications.

6. Citations and Bibliographies

References

[1] D. Landgrebe, Hyperspectral image data analysis, IEEE Signal processing magazine 19 (2002) 17–28.

[2] G. E. Hinton, R. R. Salakhutdinov, Reducing the dimensionality of data with neural networks, science 313 (2006) 504–507.

[3] K. Fukushima, S. Miyake, T. Ito, Neocognitron: A neural network model for a mechanism of visual pattern recognition, IEEE transactions on systems, man, and cybernetics (1983) 826–834.

[4] Y. LeCun, L. Bottou, Y. Bengio, P. Haffner, Gradient-based learning applied to document recognition, Proceedings of the IEEE 86 (1998) 2278–2324.

[5] D. C. Ciresan, U. Meier, J. Masci, L. M. Gambardella, J. Schmidhuber, Flexible, high performance convolutional neural networks for image classification, in: Twenty-second international joint conference on artificial intelligence, Citeseer, 2011.

[6] P. Y. Simard, D. Steinkraus, J. C. Platt, et al., Best practices for convolutional neural networks applied to visual document analysis., in: Icdar, volume 3, Edinburgh, 2003.

[7] R. T. Wilson, Py6s: A python interface to the 6s radiative transfer model., Comput. Geosci. 51 (2013) 166–171.

[8] E. F. Vermote, D. Tanré, J. L. Deuze, M. Herman, J.-J. Morcette, Second simulation of the satellite signal in the solar spectrum, 6s: An overview, IEEE transactions on geoscience and remote sensing 35 (1997) 675–686.

[9] F. A. Kruse, A. Lefkoff, J. Boardman, K. Heidebrecht, A. Shapiro, P. Barloon, A. Goetz, The spectral image processing system (sips)—interactive visualization and analysis of imaging spectrometer data, Remote sensing of environment 44 (1993) 145–163.

[10] Y. Du, C.-I. Chang, H. Ren, C.-C. Chang, J. O. Jensen, F. M. D'Amico, New hyperspectral discrimination measure for spectral characterization, Optical engineering 43 (2004) 1777–1786.

[11] L. Gao, B. Yang, Q. Du, B. Zhang, Adjusted spectral matched filter for target detection in hyperspectral imagery, Remote sensing 7 (2015) 6611–6634.

[12] Y. Cohen, D. G. Blumberg, S. R. Rotman, Subpixel hyperspectral target detection using local spectral and spatial information, Journal of Applied Remote Sensing 6 (2012) 063508–063508.

[13] D. Manolakis, E. Truslow, M. Pieper, T. Cooley, M. Brueggeman, Detection algorithms in hyperspectral imaging systems: An overview of practical algorithms, IEEE Signal Processing Magazine 31 (2013) 24–33.

[14] E. J. Kelly, K. M. Forsythe, Adaptive detection and parameter estimation for multidimensional signal models, Technical Report, Massachusetts Inst of Tech Lexington Lincoln Lab, 1989.

[15] X. Jin, S. Paswaters, H. Cline, A comparative study of target detection algorithms for hyperspectral imagery, in: Algorithms and Technologies for Multispectral, Hyperspectral, and Ultraspectral Imagery XV, volume 7334, SPIE, 2009, pp. 682–693.

[16] B. A. M Graña, MA Veganzons, Hyperspectral remote sensing scenes, 2011. URL: https://www.ehu.eus/ccwintco/index.php/Hyperspectral_Remote_Sensing_Scenes.