Special Topics in Applications (AIL861)
Artificial Intelligence for Earth Observation
Lecture 23

Instructor: Sudipan Saha

Unsupervised CD for Hyperspectral Images

✓ Recent deep learning based unsupervised CD methods use transfer learning.

✓ They are dependent on the availability of suitable pre-trained feature extractor.

✓ Challenges w.r.t. hyperspectral input:

❑ Networks pre-trained on multi-spectral images cannot be reused for hyperspectral images.

❑ Large variation among different hyperspectral sensors.

Sensor	Bands	GSD (m)
DESIS	180	30
EnMAP	228	30
PRISMA	185	30
HISUI	185	30
HySIS	256	30
Shalom	241	10
CCRSS	328	30

Brute Force Solutions

How to use a model trained for 3 channel input for hyperspectral image

- ✓ Use only 3 channels of the hyperspectral input.
- ✓ Replicate the weight of 1st layer 3*64*3*3 to N_c*64*3*3.

Requirement

An unsupervised CD method for hyperspectral images such that:

- ✓ The method can be used for different hyperspectral sensors without any adaptation.
- ✓ The method can use all bands of the hyperspectral input.
- ✓ The method captures the semantics of different change classes.

Untrained Model

- ✓ Untrained Model:
- ❑ Can be initialized to as many input band as we desire.
- ❑ Can be initialized to as many layers as we desire.

- ✓ Deep image prior:
- ❑ Relevant prior works in computer vision [1].
- ❑ They show that even an untrained model can capture image semantics to some extent.
- ❑ Main idea: structure of the network is more important than weights.

1. Ulyanov, D., Vedaldi, A. and Lempitsky, V., 2018. Deep image prior. In *Proceedings of the IEEE conference on computer vision and pattern recognition* (pp. 9446-9454).

Strategy

✓ Untrained however initialized:

❑ With appropriate weight initialization strategy, e.g., He weight initialization.

❑ Weight initialization does not involve any training.

✓ Tackles large number of bands:

❑ Can tackle inputs having very different structure from multispectral images.

✓ Tackles varying band composition:

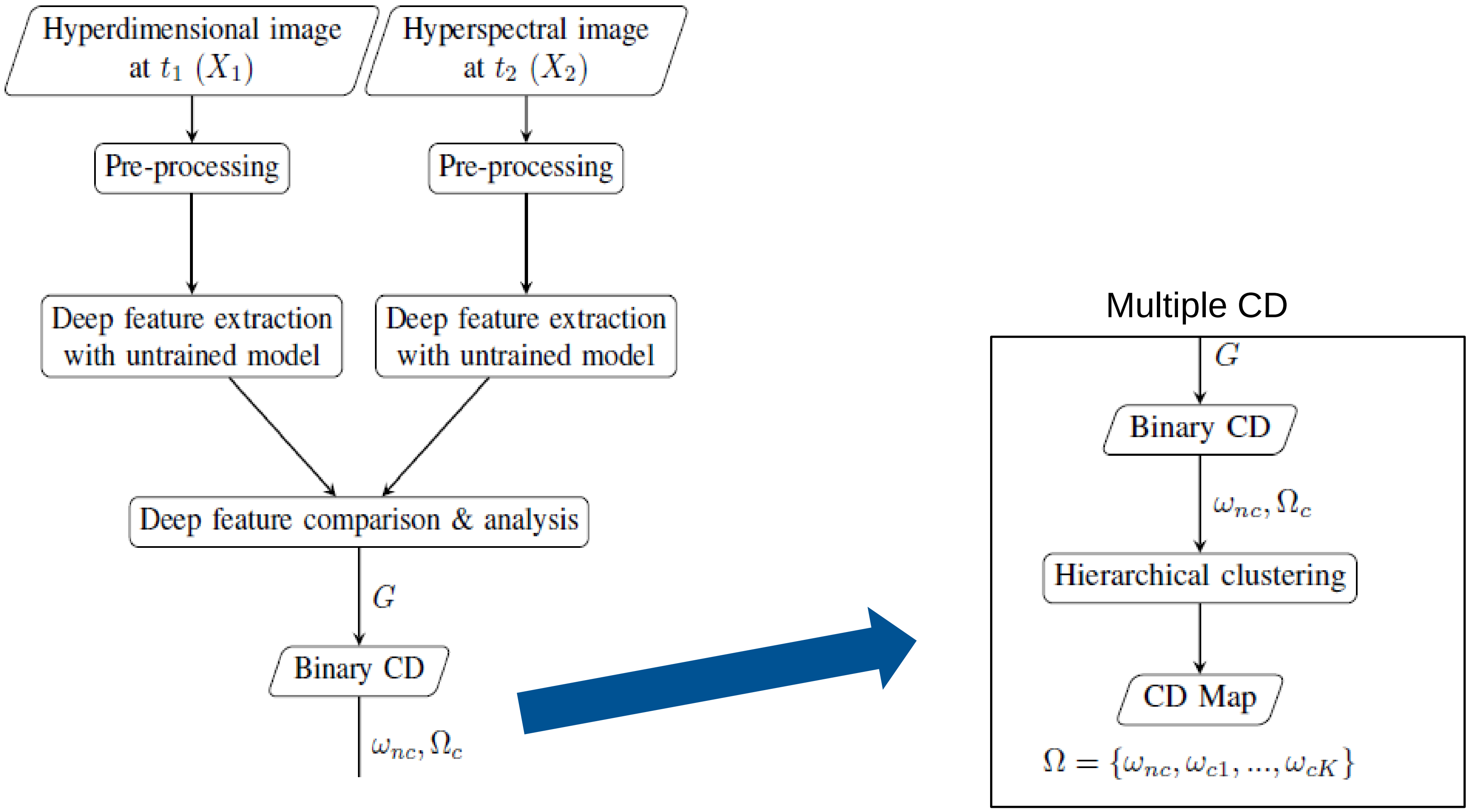
❑ Different band composition for different hyperspectral sensors.

Network

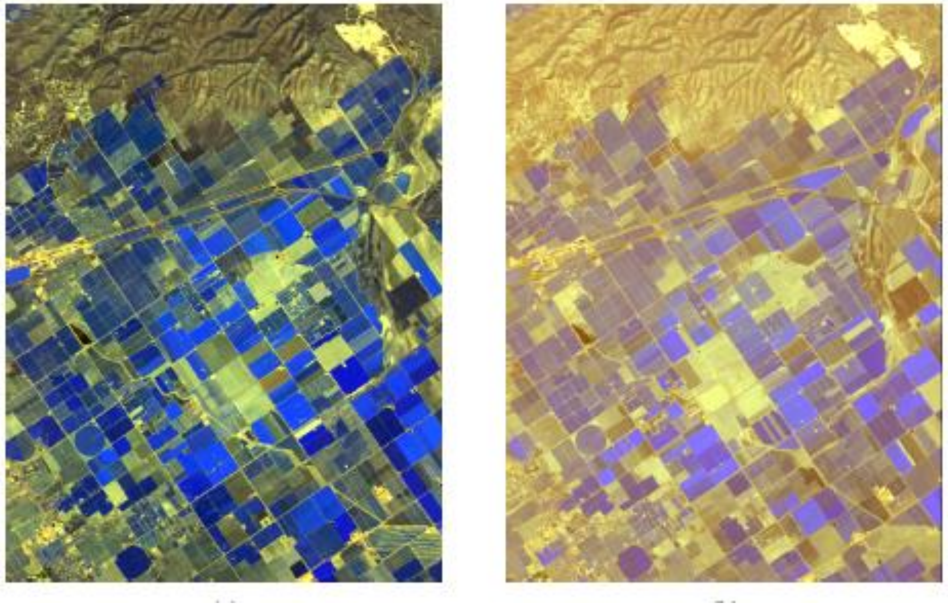
Network structure assuming 224-channel input

Layer number	Layer type	Input Kernel	Output Kernel	Kernel size
1	convolutional	224	896	(3,3)
2	convolutional	896	896	(3,3)
3	convolutional	896	896	(3,3)
4	convolutional	896	896	(3,3)
5	convolutional	896	896	(3,3)

Hyperspectral CD with Untrained Network

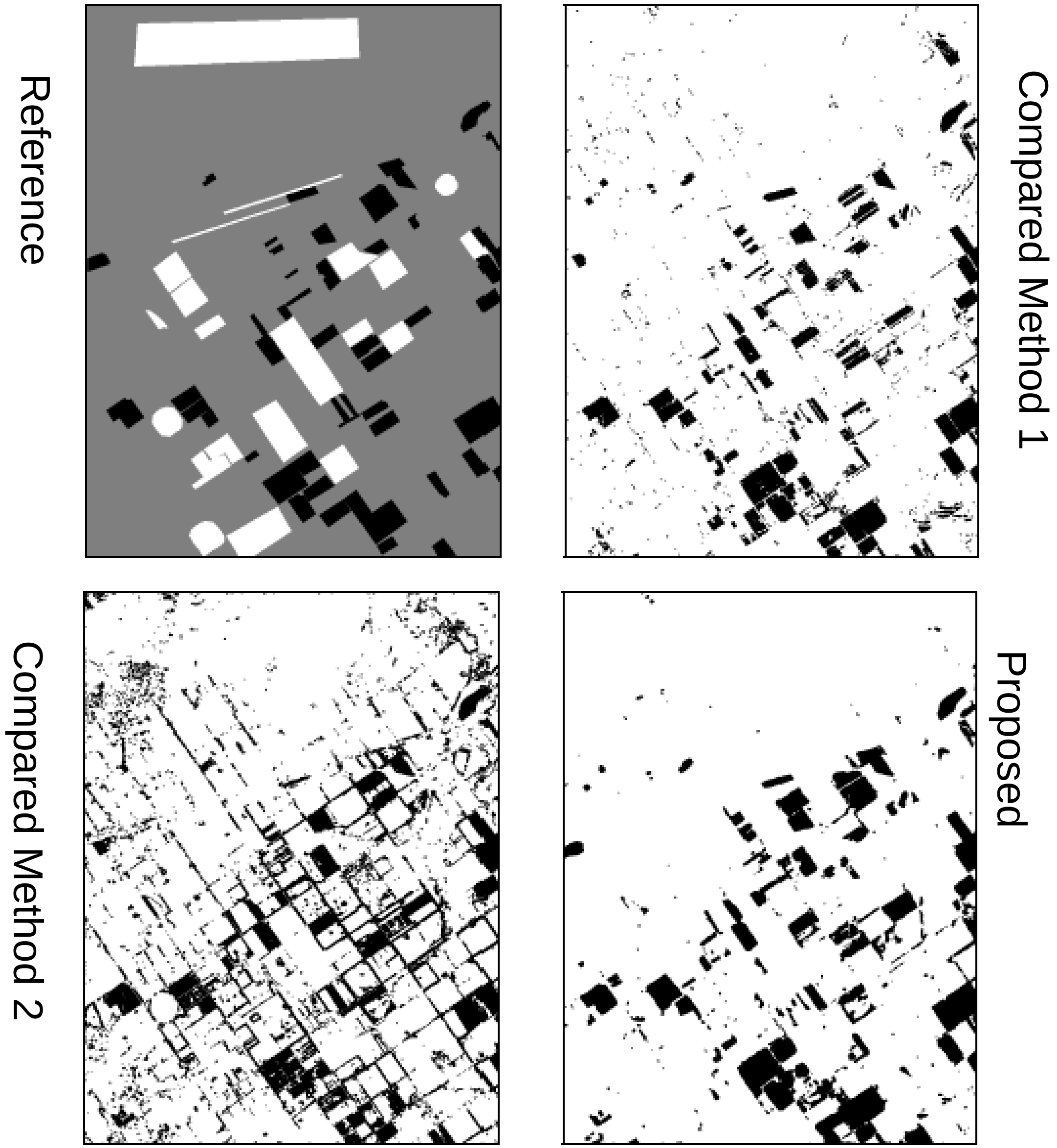


Santa Barbara Scene



https://citius.usc.es/investigacion/datasets/hyperspectral-change-detectiondataset

Santa Barbara Scene



Santa Barbara Scene

- ✓ Sensitivity: accuracy computed over reference changed pixels
- ✓ Specificity: accuracy computed over reference unchanged pixels

- ✓ Result improves initially as more layers are added.

Method	Sensitivity	Specificity
Proposed (2 layers)	83.86	98.71
Proposed (3 layers)	83.90	98.96
Proposed (4 layers)	85.86	98.97
Proposed (5 layers)	87.98	98.57
Proposed (6 layers)	84.74	98.48

Untrained Model in Other Tasks?

How good are untrained features for other tasks, e.g., classification?

3D Convolution

- ✓ Input and output dimension of convolution
- ✓ Source of name: how the kernel moves.

3D Convolution

```
import torch
import torch.nn as nn

m = nn.Conv3d(13, 26, 3, stride=1)
#Basic syntax: torch.nn.Conv3d(in_channels, out_channels, kernel_size, stride=1)
print(m)

input = torch.randn(100, 13, 100, 200, 200)
#Input dimension is (N, C_in, D, H, W)
#N number of inputs in batch
#C_in: number of channels (like 3 for RGB image)
#D depth (like number of frames in the video)
# H and W - spatial dimension
output = m(input)

print(input.shape)
print(output.shape)
```

m: Conv3d(13, 26, kernel_size=(3, 3, 3), stride=(1, 1, 1))