Cairo University

Faculty of Engineering
Cairo University

Remote Sensing & Satellite Imagery

Project Report

Team 16

Submitted to:

Eng. Muhammed Sayed

Submitted By:

NAME	SEC	BN	ID
Sarah Mohamed Hossam	1	29	9202618
Abdelrahman Fathy	2	3	9202846
Yasmine Ashraf Ghanem	2	37	9203707
Yasmin Abdullah Nasser	2	38	9203717

Classical Approach

Project pipeline

- Loading the dataset
- Data preprocessing
- Analysis and visualization
- Trying different methods
- Evaluation

Dataset Loading

The dataset was loaded as gray images, operating on different bands didn't make such a difference. We took into consideration the order of images and indices of A, B and the labels.
The dataset was split into training (70%) and testing (30%) sets.

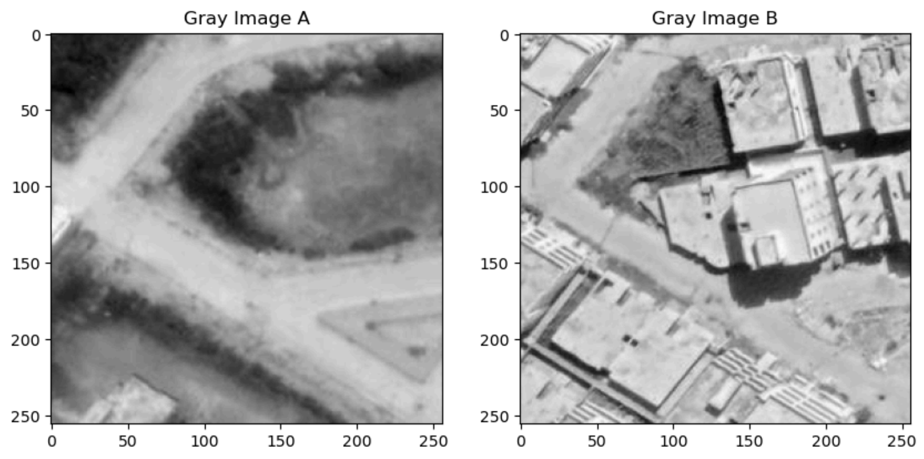
Data preprocessing

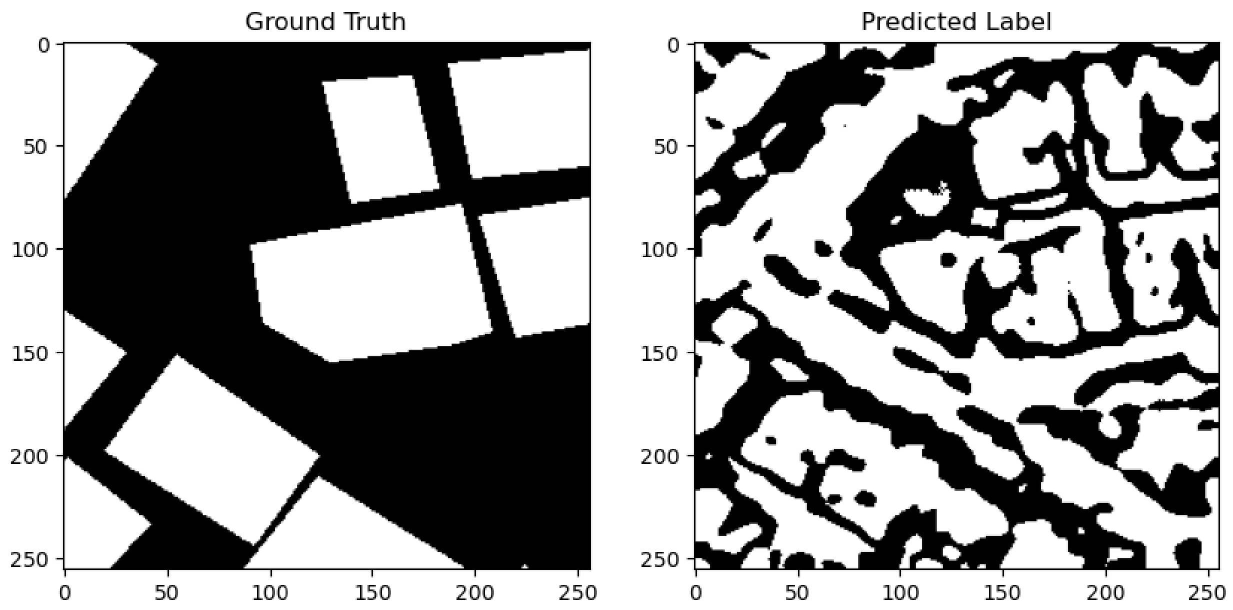
Basic preprocessing was done, radiometric correction, histogram equalization, gaussian blur, edge detection, opening and closing.

Analysis and visualization

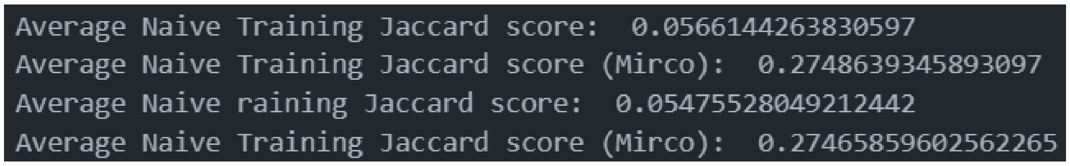
As we skimmed through the dataset, it was very noticeable that the detected changes were only concerned with the buildings, not the roads, vegetation… etc.
The main approach used was image differencing.

- Basic image differencing
  In this approach, we computed an absolute difference between the two images, and used some simple thresholding (tuned and tried different values)

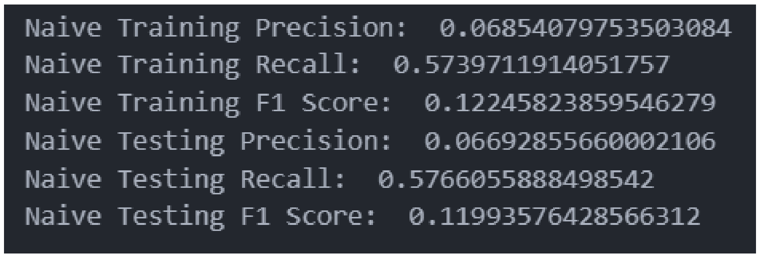




That approach obviously didn't have great results and resulted in very low evaluation metrics. We used the Jaccard score in both modes, binary and micro

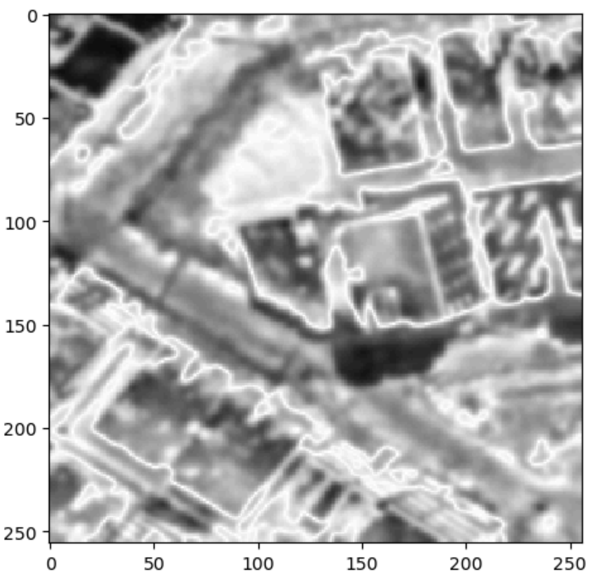
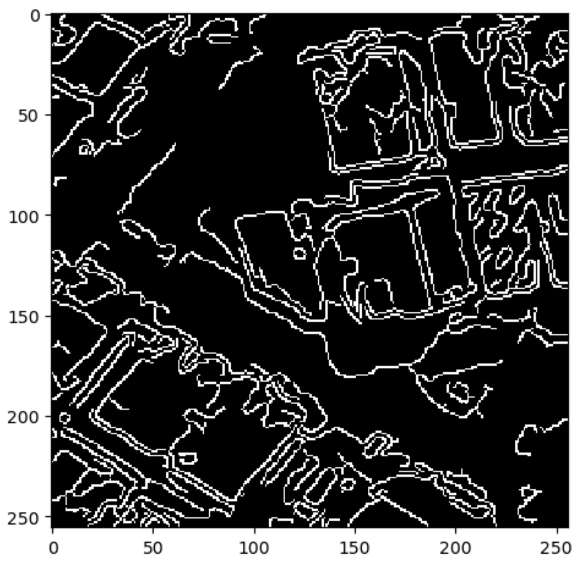


```
Average Naive Training Jaccard score:  0.0566144263830597
Average Naive Training Jaccard score (Mirco):  0.2748639345893097
Average Naive raining Jaccard score:  0.05475528049212442
Average Naive Training Jaccard score (Mirco):  0.27465859602562265
```

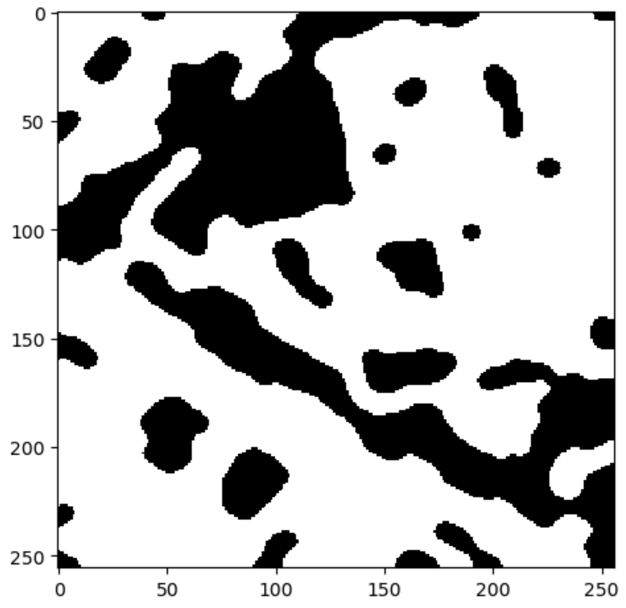


```
Naive Training Precision:  0.06854079753503084
Naive Training Recall:  0.5739711914051757
Naive Training F1 Score:  0.12245823859546279
Naive Testing Precision:  0.06692855660002106
Naive Testing Recall:  0.5766055888498542
Naive Testing F1 Score:  0.11993576428566312
```

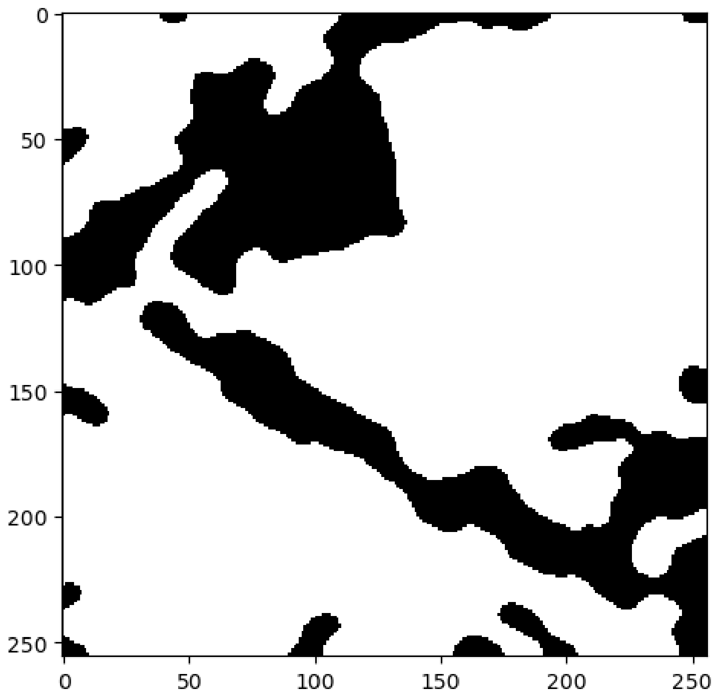
- Image differencing with some mask postprocessing
  Noticing the output mask using the above method had so many holes and inconsistent areas, so we tried a different method.
  First, we used histogram equalization for both images from A and B, then applied a gaussian filter to get rid of the harsh details and lines, then image differencing and inverting. When using image differencing, the buildings outlines are very much highlighted and visible.
  So we thought it might be useful to use edge detection to get those outlines.

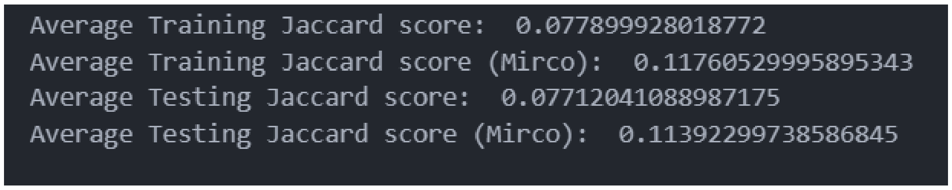
Next step was to emphasize those edges, and make them more visible, so we used opening and closing multiple times.



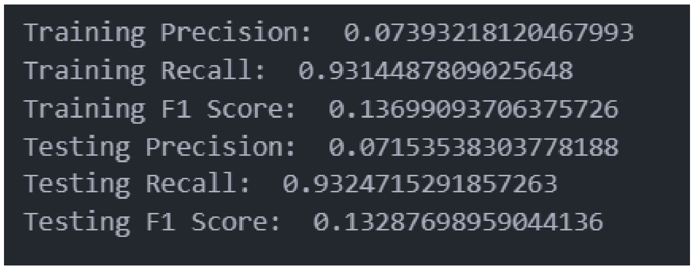
Lastly, we detected the closed contours and filled them with white. The output mask covered most of the areas that were covered by the original label



This method resulted in somewhat better results.



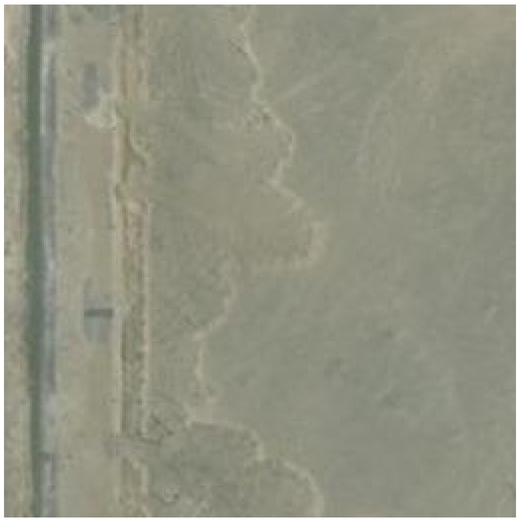
```
Average Training Jaccard score:  0.077899928018772
Average Training Jaccard score (Mirco):  0.11760529995895343
Average Testing Jaccard score:  0.07712041088987175
Average Testing Jaccard score (Mirco):  0.11392299738586845
```

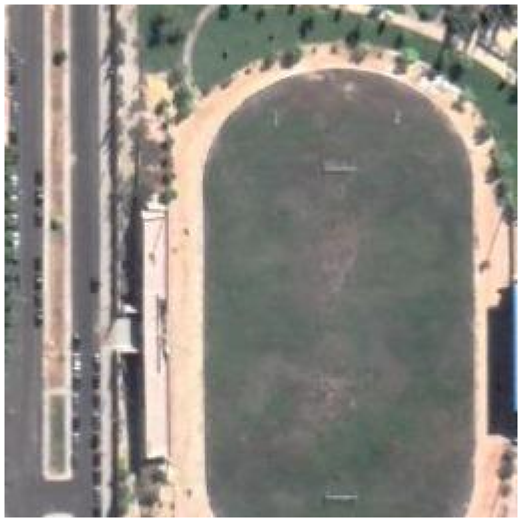


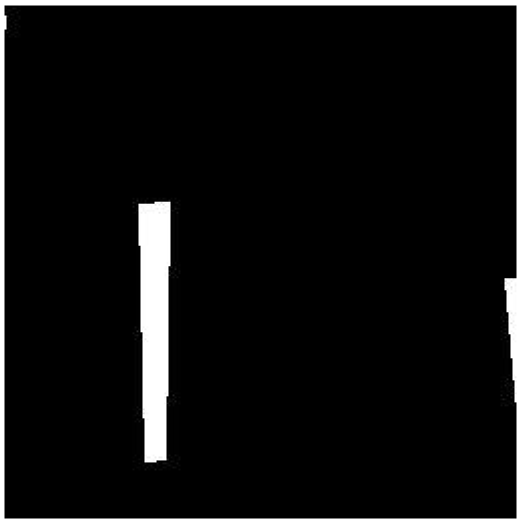
```
Training Precision:  0.07393218120467993
Training Recall:  0.9314487809025648
Training F1 Score:  0.13699093706375726
Testing Precision:  0.07153538303778188
Testing Recall:  0.9324715291857263
Testing F1 Score:  0.13287698959044136
```

The main issue was that the dataset had mostly pictures without change, and those had a mask of all zeros. To get such a mask, the images need to be almost identical, but again, it was noticed that the labels only highlighted the change in the buildings.

For example, given the 2 pictures and their label.



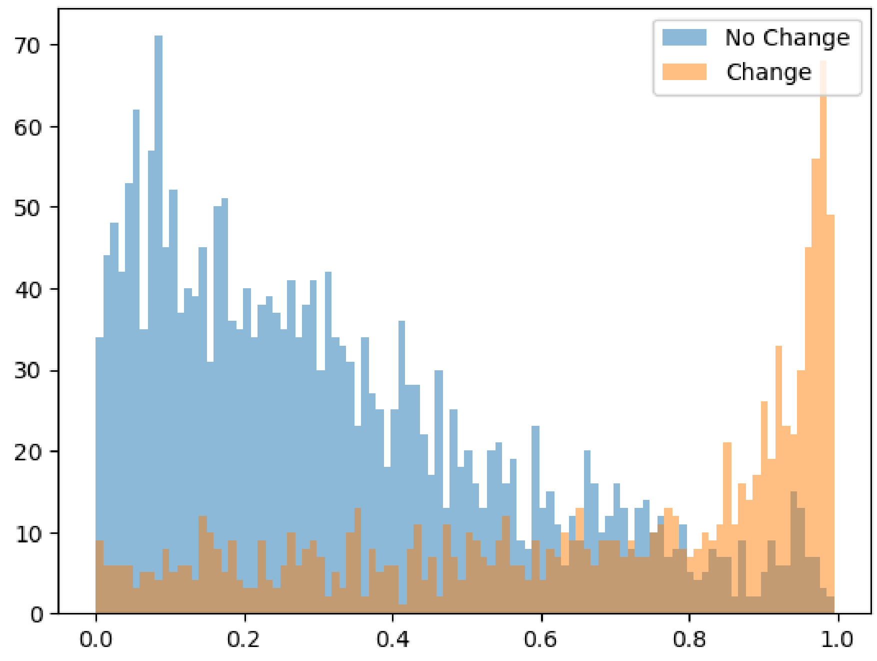


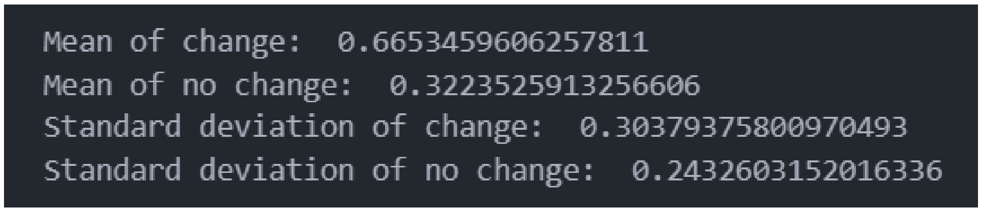


So, to get some higher scores, we thought we might try to first detect whether there's a change or not, if there isn't then output a black image, if there is, then output the calculated mask.

We tried a few methods for change detection. The one we picked relied on the variance of the resultant difference image.

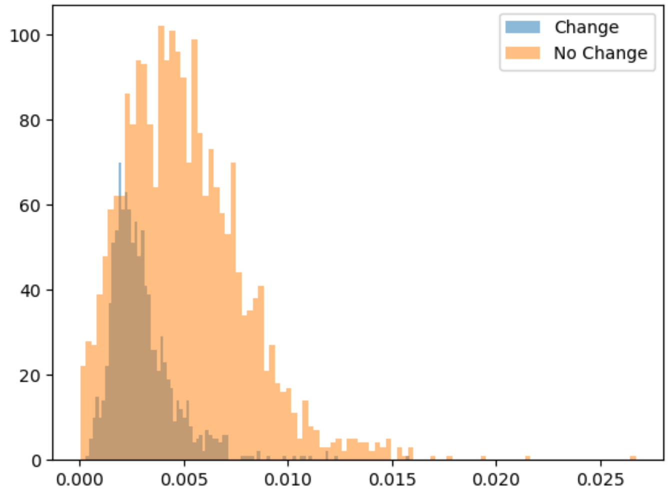
For all the training data, we plotted a histogram showing the distribution of the variance of the difference images resulting from the absolute difference between the image from A and the one from B, both tended to have normal distributions with shifted mean and standard deviation.



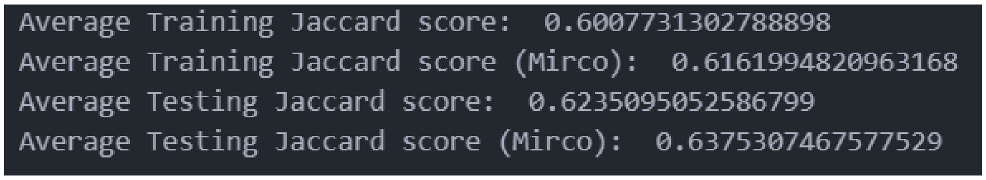


```
Mean of change:  0.6653459606257811
Mean of no change:  0.3223525913256606
Standard deviation of change:  0.30379375800970493
Standard deviation of no change:  0.2432603152016336
```

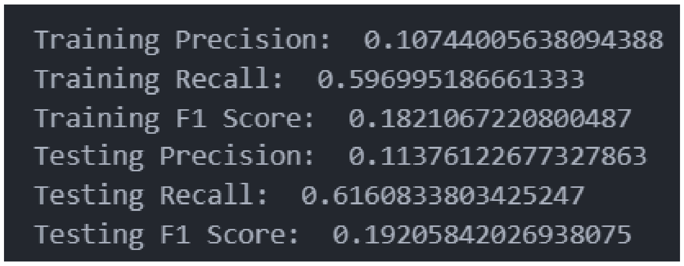
This helped a lot to identify whether there's a change or not.
Also, tried different approaches like the Jaccard Score, but it was difficult to differentiate between the resulting distributions



So in the final approach, we checked if the variance of the difference image is within the range of the *change* distribution, we then calculated the output mask. If no change was detected, then the output is a black image.



```
Average Training Jaccard score:  0.6007731302788898
Average Training Jaccard score (Mirco):  0.6161994820963168
Average Testing Jaccard score:  0.6235095052586799
Average Testing Jaccard score (Mirco):  0.6375307467577529
```



```
 Training Precision:  0.10744005638094388
 Training Recall:  0.596995186661333
 Training F1 Score:  0.1821067220800487
 Testing Precision:  0.11376122677327863
 Testing Recall:  0.6160833803425247
 Testing F1 Score:  0.19205842026938075
```

Even though this method isn't entirely accurate, it was our go to method.

Deep Learning Approach

- *UNet:*

  The first network we tried was a simple Unet given in the course for change detection which didn’t achieve the best results:

  - Mean Jaccard Score : 0.014
  - F1 Score: 0.023

  We found it challenging since the input of the UNet is a single image so we had to concatenate the images so that the input was a 6 channel image instead of 2-three-channel inputs.

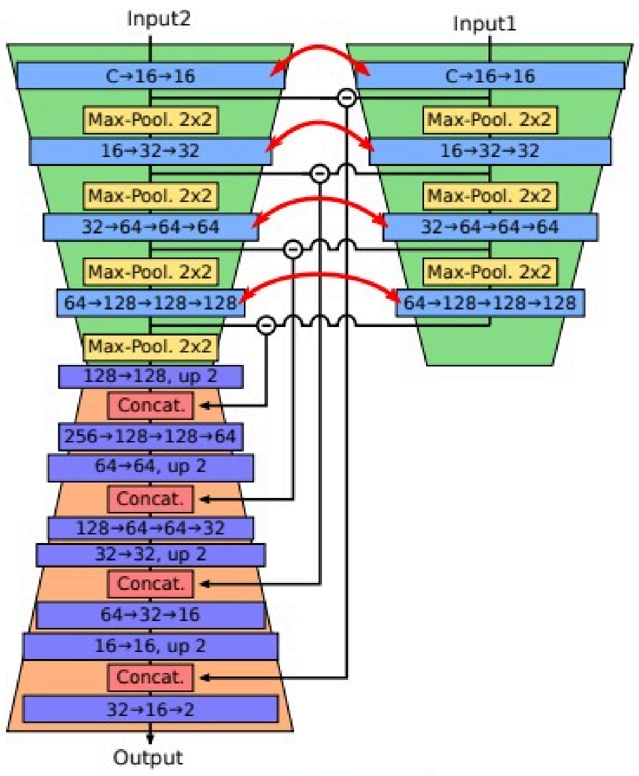
- *Siamese UNet*

  Then we tried the Siamese UNet suggested in the document as the base of our model with multiple variations and adjustments. The main advantage of the Siamese UNets over the traditional UNets is that the Siamese UNet takes as an input two images at different times and performs the same convolutional blocks on them to produce feature maps for the ‘What’ and ‘Where’ of the images to check for the differences between the outputs of the convolutional blocks.

  1. Siamese UNet Difference

     This is the simplest technique for the Siamese UNet where the feature maps of both images are subtracted from one another to highlight the differences:

     - Basic Architecture:



(c) FC-Siam-diff.

- Loss Function:
  - The paper proposed a NLLLoss function which is a built-in function in pytorch which was suggested to have the best results with the LogSoftmax function used in the algorithm. This got us:
    - Mean Jaccard Score: 0.689
  - We tried to implement a hybrid function that uses 2 loss function together; the idea was taken from the UNet lab where the Binary Cross Entropy Loss and the Dice Loss were combined:
    - The BCE combines a sigmoid activation function with the binary cross entropy loss.
    - BCE loss aims to minimize the difference between the predicted probability distribution and the ground truth binary labels
    - The Dice Loss aims to maximize the overlap between the predicted segmentation mask and the ground truth.

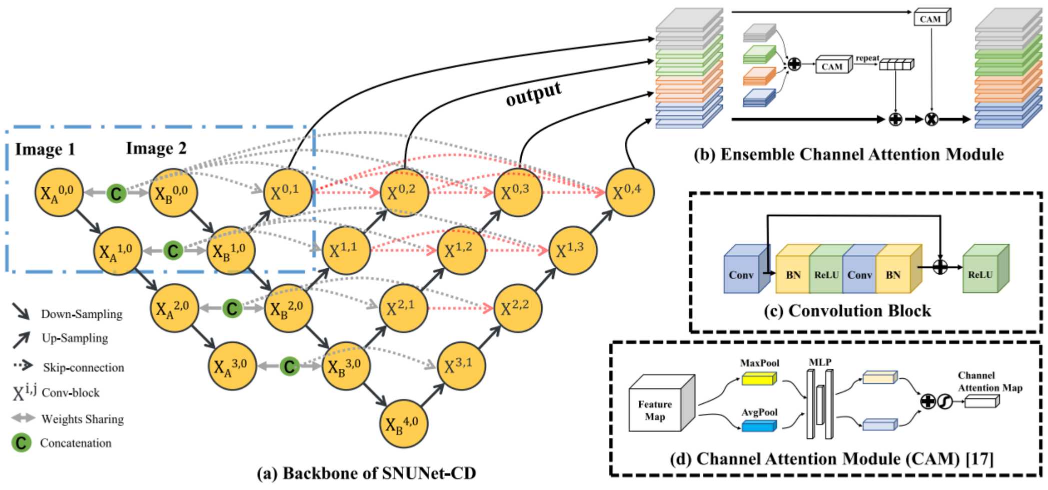
- Data Augmentation:

Lastly we augmented the data to apply some transformations like horizontal and vertical flips on some of the data so that the model can learn better on variations of the data however, the accuracy didn't increase significantly.

Best Results:
Epochs: 20.
Jaccard Mean: 0.6899
Micro Jaccard Mean: 0.9003

2. Siamese UNet ECAM
   The next approach had the same base but more layers were added. The paper suggested that the f1 score rose from 0.6 using the Siamese Difference to 0.9 when using the ECAM.

   - Basic Architecture



   - Loss Function
     - Same as the previous model where we tried a hybrid function but this time we combined the Dice Loss with the Focal Loss which is a variant of the BCE that focuses giving lower weights for easier examples
     - This technique is good with imbalance data.
     - This was a good approach since the previous model had a good base but the jaccard score came from classifying the images with no change and failed to classify accurately the class where there is actual change. However the results were not promising since we got:

- Jaccard Score: 0.03

The main disadvantage of this network is that it is much slower than the Siamese Difference as it has deeper layers and more computation within a single layer so for a single epoch the Siamese Difference takes a minute approximately, the ECAM takes 6 minutes to train a single epoch.

Conclusion

For the classical approach, the data was severely imbalanced, the change shown in masks only showed the change in the buildings and discarded so many other details.
The overall performance of the classical methods were not satisfying and don’t suit the problem.

As for the deep learning approach, the Siamese UNet has promising results and more layers could be added to make the model more complex; however, it would require higher processing power and have higher time and space complexity. However the deep learning approach achieved much better results than the classical approach for this particular application.