Cairo University
Faculty of Engineering

Satellite Imagery

CMP40XX

Team #6

Project Document

Provided By:

Name	Sec	BN	ID
Nour Ziad Almulhem	2	30	9204005
Ahmed Sayed Sayed Madbouly	1	3	9202111
Ahmed Hany Farouk	1	10	9202213
Eslam Ashraf Ibrahim	1	12	9202256

Provided to: Dr/ Sandra Wahid

Eng/ Mohamed Sayed

Contribution and workload:

Eslam Ashraf Nour Ziad	Traditional Methods SNUnet trial
Ahmed Hany Ahmed Madbouly	Experimental tasks UnetPlusPlus and deep learning approaches

Utilities:

For the sake of achieving the objective of this project we used some utility functions, it is to be explained in the next few lines.

a. Jaccard index

For Jaccard score calculation we used 3 approaches

- Calculating using the Jaccard equation using the true positive, false negative and false positive.
  Jaccard_Score = tp / (tp + fp + fn)
- We used JaccardIndex from torchmetrics with number of classes 2 and binary task
- Calculate the Jaccard score also using metrics from sklearn for every batch and got the average value across all batches which is in our case the MIoU

```
print('----- Test Floor Info')

print("test floor Jaccard score: ", np.array(test_floor_j_score_arr).mean())
print(f'for each epoch jaccard score 2: test floor : {test_floor_jaccard.compute()}')
print(f'TP: {tp_T2}, FP: {fp_T2}, FN: {fn_T2}')
print(f'jaccard 3 test floor : {tp_T2 / (tp_T2 + fp_T2 + fn_T2)}')
```

b. Scoring Functions

For measuring the performance of our obtained solution, we calculated the accuracy, F1 score, recall and precision running on the test set, which is a subset of 486 image of the dataset.

Traditional Approach

We used 2 different traditional approaches to achieve our goal of change detection. For our solution we used otsu's thresholding algorithm to obtain our threshold which calculates a threshold value based on intensity distribution among the pixels in the image

The thresholding algorithm tries to get an equal number of pixels above and below its intensity.

a. Image differencing + otsu as a thresholding algorithm

Image differencing change detection technique is performed by subtracting the digital number (DN) value of a pixel in one date for a given band from the DN value of the same pixel for the same band of another date.

For each difference image, a threshold value based on standard deviation (SD) is required to delineate the changed pixels from the unchanged pixels. To determine the most appropriate threshold value

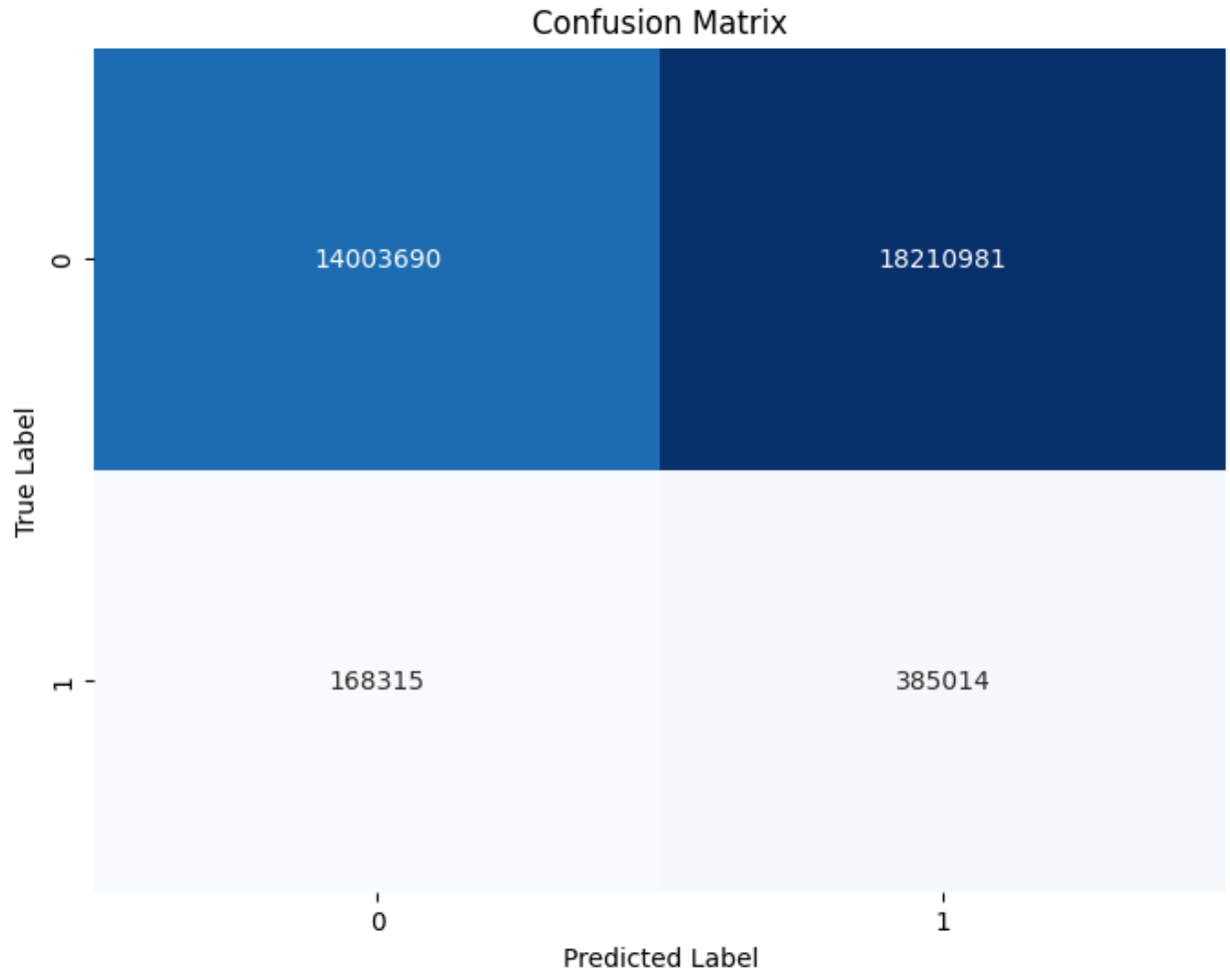
the Jaccard score = 31.04336820803229%

Accuracy: 43.91%

Precision: 2.07%

F1 Score: 4.02%

Recall: 69.58%



b. CVA + otsu as a thresholding algorithm

Change Vector Analysis (CVA) as change detection technique has useful capabilities of extracting and identifying land cover changes in terms of change-magnitude and change-direction from two different temporal satellite imageries.

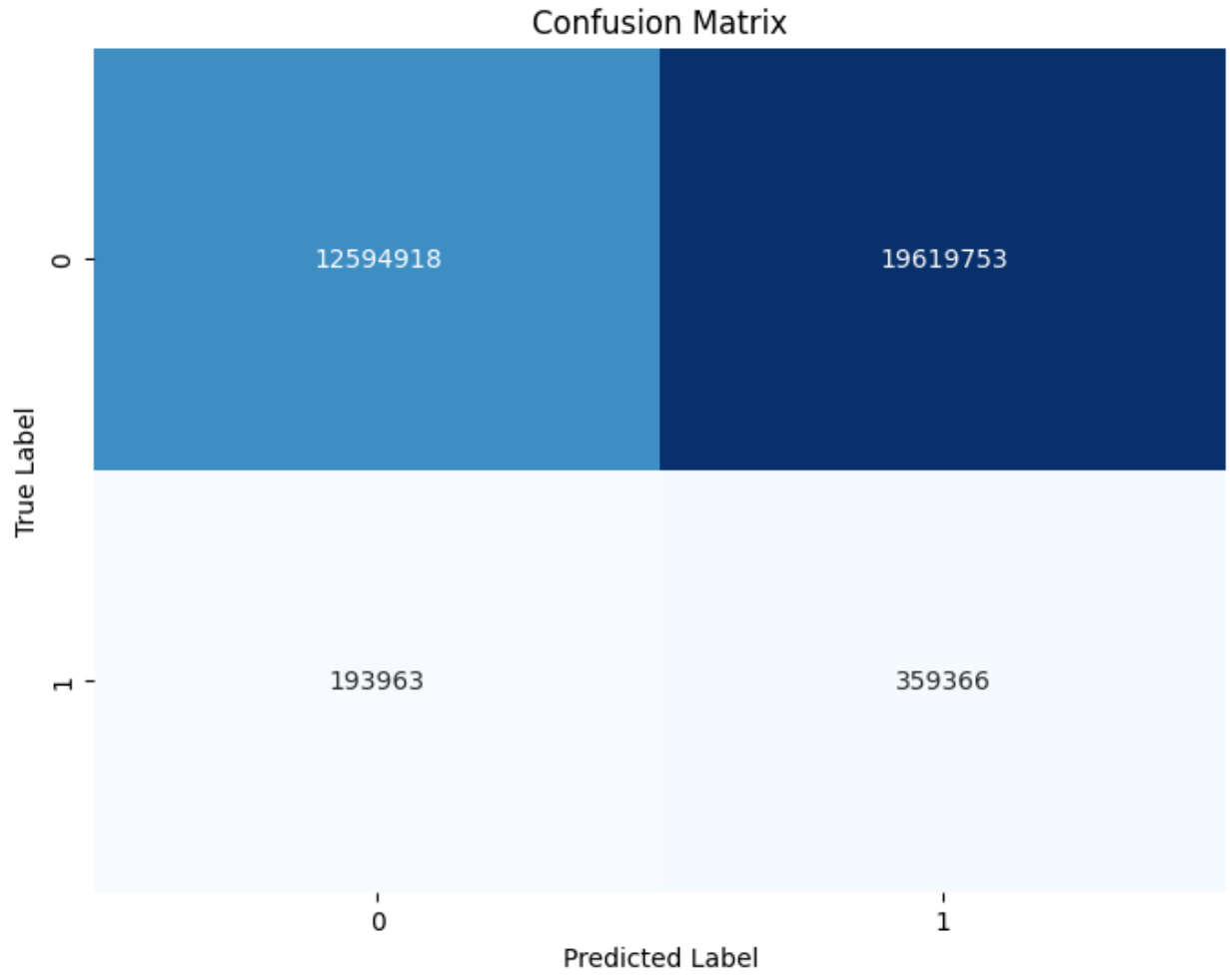
the Jaccard accuracy = 24.684784855737227

Accuracy: 39.53%

Precision: 1.80%

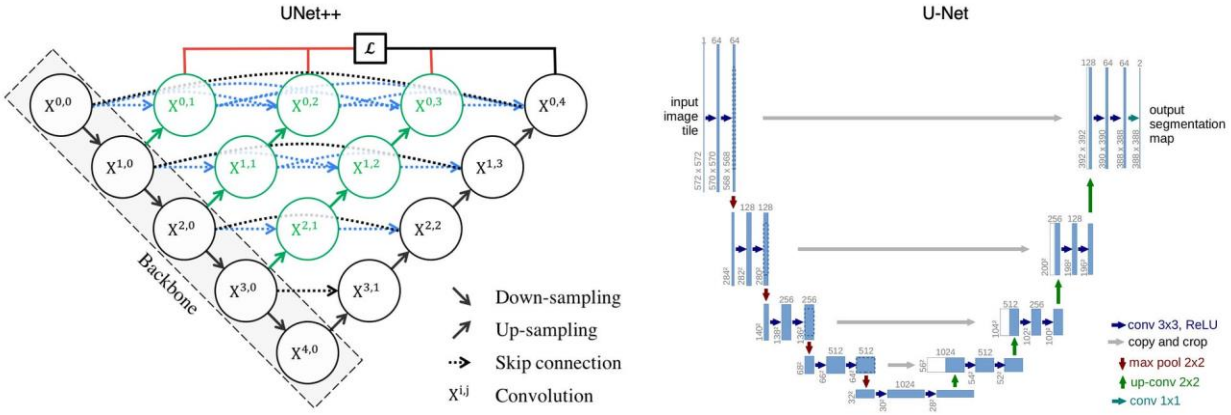
F1 Score: 3.50%

Recall: 64.95%

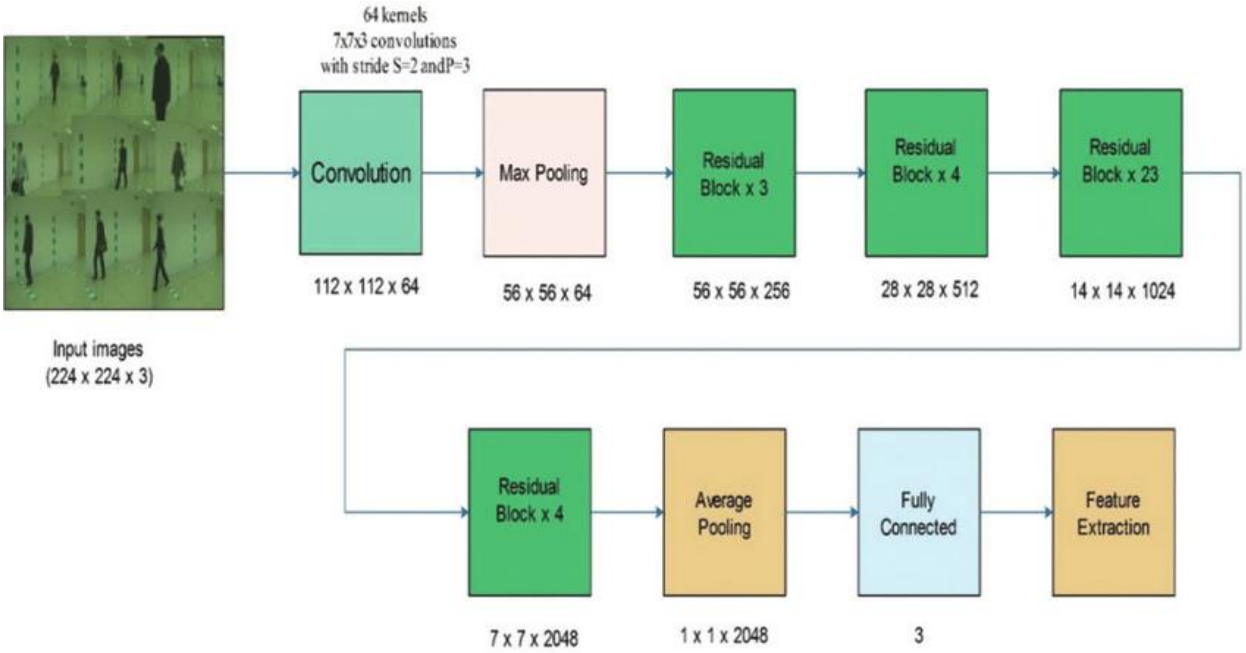


Deep learning Approach:

Our approach used basically Unet plus plus model along with se_resnet101 as an encoder.



Our approach utilized the Unet++ model architecture coupled with se_resnet101 as an encoder. The Unet++ architecture, known for its ability to capture complex features effectively, was chosen over the standard Unet model due to its superior performance in capturing intricate features. Additionally, the use of se_resnet101 as our encoder was based on recommendations from prior literature surveys. This selection was made to leverage the proven effectiveness of se_resnet101 in feature extraction tasks, aligning with the goals of our study.



Our approached steps were:

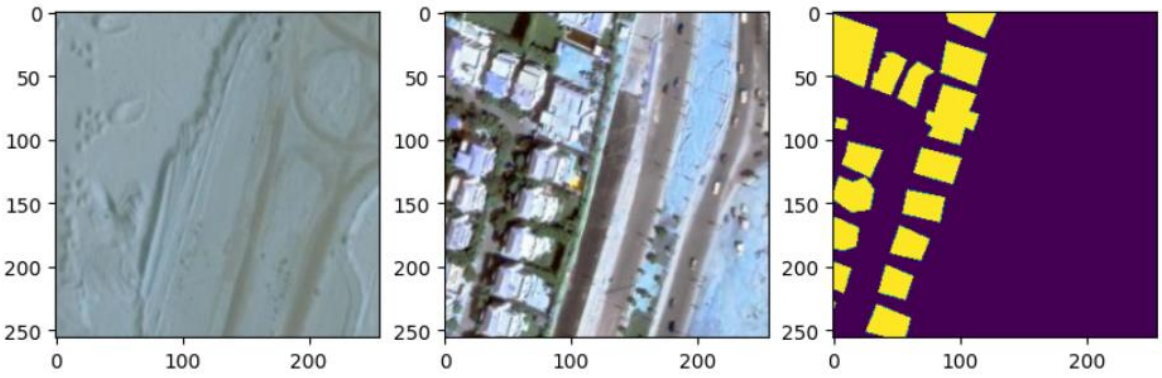
Downloading the data set into our environment, splitting data into train and validation with size 0.2 – this step was so sensitive as it determines the performance of the model – getting more test data with no change images won't be indicative of the performance of our model.

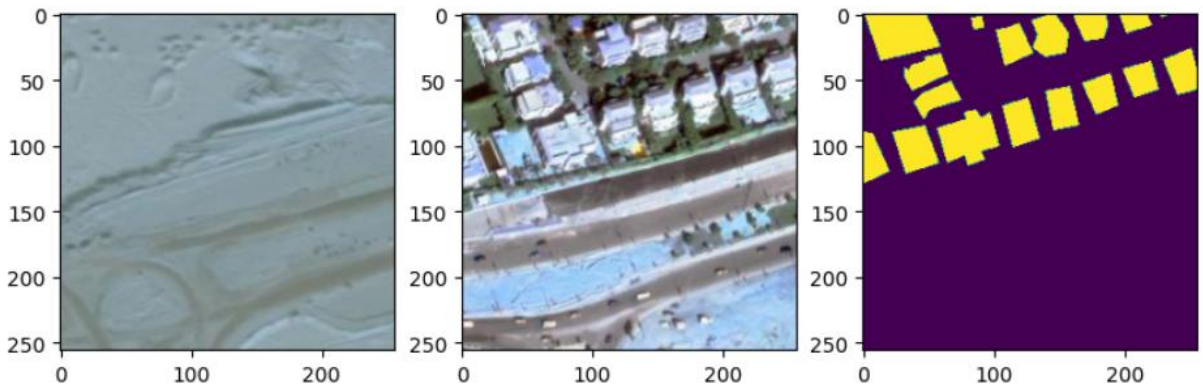
Also, on the other hands having all change data will achieve a non-indecative Jaccard score as well.

Augmenting dataset was an important step to do, this was we provide more features to be captured when training the model which gave us a higher performance.

we tried two ways of augmentation of the rest images where there is a change:

a. First way:
    i. The image itself
    ii. The image rotated vertical or horizontal random choice.
b. Second way:
    i. The image itself
    ii. The image rotated vertical.
    iii. The image rotated horizontal.
    iv. The image rotated 90 degrees.
    v. The image rotated 180 degrees.
    vi. The image rotated 270 degrees.





One of the trials was reducing the no-change images from training, as their high number affected our training the model was giving a high weight to the no-change over the change.

We defined the data set where we read the image and make the preprocessing whether adjust brightness or any rotation mentioned above.

Then we initiate the UnetPlusPlus and use the se_resnet101 for decoder and encoder

The model:

2. It consists of three main components:
    a. Encoder: Responsible for extracting high-level features from input images.
    b. Decoder: Reconstructs images from the encoded features.
    c. Segmentation Head: Used for generating a change mask indicating areas of difference between the input images.
3. Forward Pass:
    a. The forward method defines the forward pass of the model, where input images (x1 and x2) are processed to detect changes.
4. Encoding:
    a. The input images are fed through the encoder to obtain latent features.
    b. Each input image is passed through the encoder independently, resulting in two sets of latent features.
5. Feature Combination:
    a. Latent features from the two input images are combined to emphasize differences between them.
    b. This is achieved by taking the absolute difference between corresponding features.
6. Decoding:
    a. The combined features are then decoded using the decoder to reconstruct the change representation.
7. Change Mask Generation:

    a. The output of the decoder is passed through the segmentation head.
    b. The segmentation head generates a change mask indicating areas of significant difference between the input images.
8. Output Processing:
    a. The output is passed through a sigmoid activation function to obtain a binary change mask.
9. Conclusion:
    a. The ChangeNet model enables the identification of changes between input images, providing a valuable tool for various applications, including image differencing and change detection tasks.

We also tried using the encoders and decoders for se-resnet1-1 in different way.

- Connecting the 2 input images and feeding the model.
- Adding the result of the encoding stage and feed the decoder.
- Subtracting the results of the encoding stage and feeding the model with the subtracted result image.

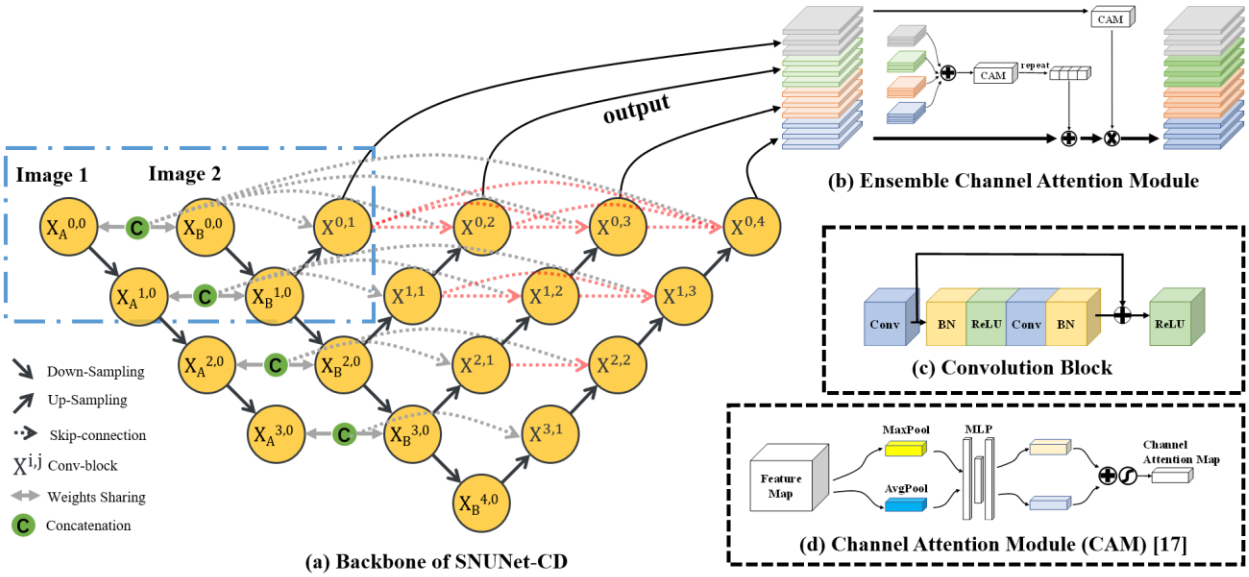
For our loss function we used 2 losses as a combination, which was Binary Cross Entropy loss, and dice loss to be more indicative and it performed well on our training.

Using Jaccard loss was also indicative, but this combination gave a better result on the validation and test subsets.

We then calculate Accuracy, precision, recall, F1 and Jaccard Score and confusion matrix.

And finally, we ended up creating an interface for detecting single images.

SNUnet, was also a trial for us, we used the SNUnset architecture as it provides way better results.



SNUNet-CD alleviates the loss of localization information in the deep layers of neural network through compact information transmission between encoder and decoder, and between decoder and decoder.

For better results, Ensemble Channel Attention Module (ECAM) is proposed for deep supervision. Through ECAM, the most representative features of different semantic levels can be refined and used for the final classification.

This trial gave us a maximum score of 40 on the subset we tested on, so we ignored using a complex architecture and provided the pretrained se-resnet one.

Metrics and results:

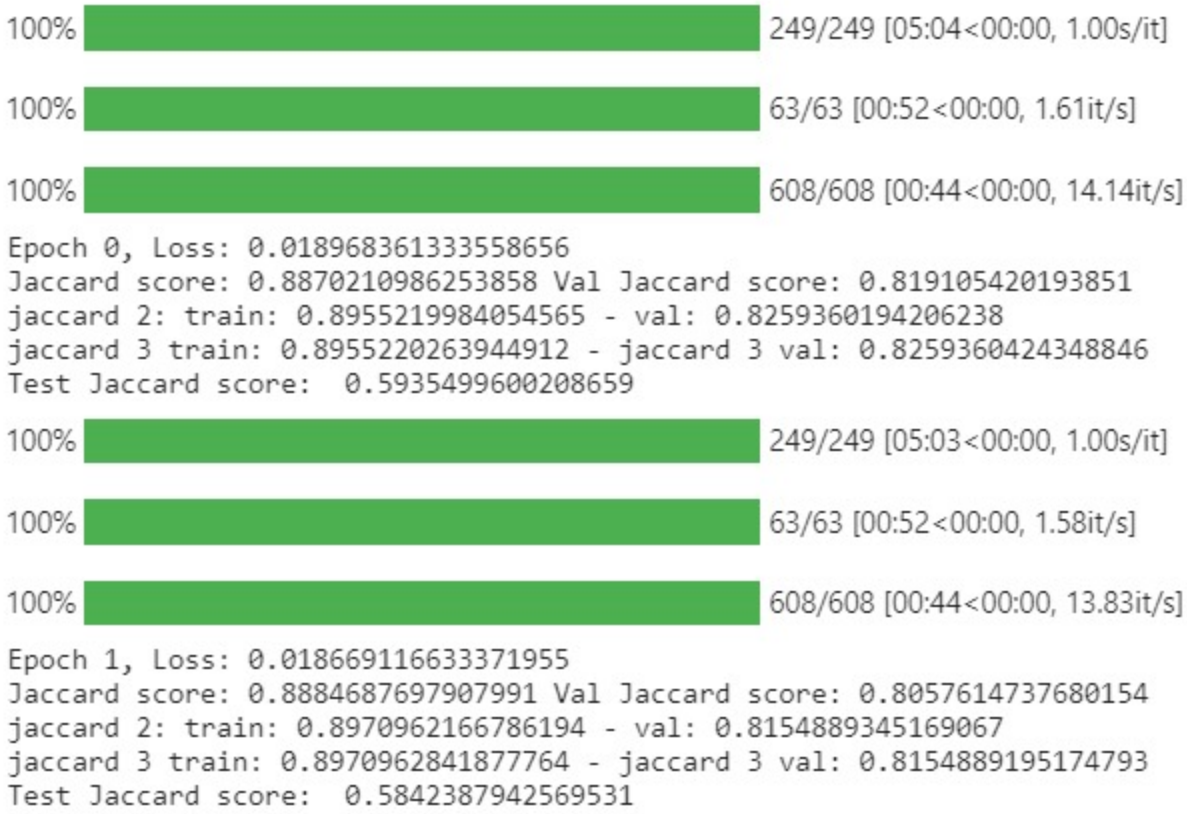
We calculated with more than one way:

For Jaccard score we calculated:

1. We use built in function of ski-learn then take the average of each batch.
2. We use built in function of Torch metrics.
3. We use the Tp/ Tp + Fp + Fn.

```
found a best dev:)
----- Train Info
Epoch 0, Loss: 0.31434181185643 Jaccard score:  0.6600009067559223
for each epoch jaccard score 2: train: 0.637493371963501
TP: 87657586, FP: 34468608, FN: 15377336
jaccard 3 train: 0.6374933501707193
----- Val Info
Val Jaccard score:  0.8251810129456357
for each epoch jaccard score 2: val: 0.6951078772544861
TP: 3654408, FP: 530126, FN: 1072791
jaccard 3 val: 0.6951078733005853
----- Test Floor Info
test floor Jaccard score:  0.9276644863409004
for each epoch jaccard score 2: test floor : 0.0
TP: 0, FP: 114162, FN: 0
jaccard 3 test floor : 0.0
100% |████████████████████| 649/649 [15:27<00:00, 1.09s/it]
100% |████████████████████| 974/974 [01:17<00:00, 12.93it/s]
100% |████████████████████| 2599/2599 [03:08<00:00, 14.06it/s]
----- Train Info
Epoch 1, Loss: 0.15170450879401529 Jaccard score:  0.7698371273265596
for each epoch jaccard score 2: train: 0.7727810740470886
TP: 90034527, FP: 13472237, FN: 13000395
jaccard 3 train: 0.7727810700456613
----- Val Info
Val Jaccard score:  0.761165124184187
for each epoch jaccard score 2: val: 0.6977570652961731
TP: 3991038, FP: 992611, FN: 736161
jaccard 3 val: 0.6977570933300232
----- Test Floor Info
test floor Jaccard score:  0.8514813389765294
for each epoch jaccard score 2: test floor : 0.0
TP: 0, FP: 507351, FN: 0
jaccard 3 test floor : 0.0
```

Here is also an example where for training loop where we print the Jaccard with the three ways.



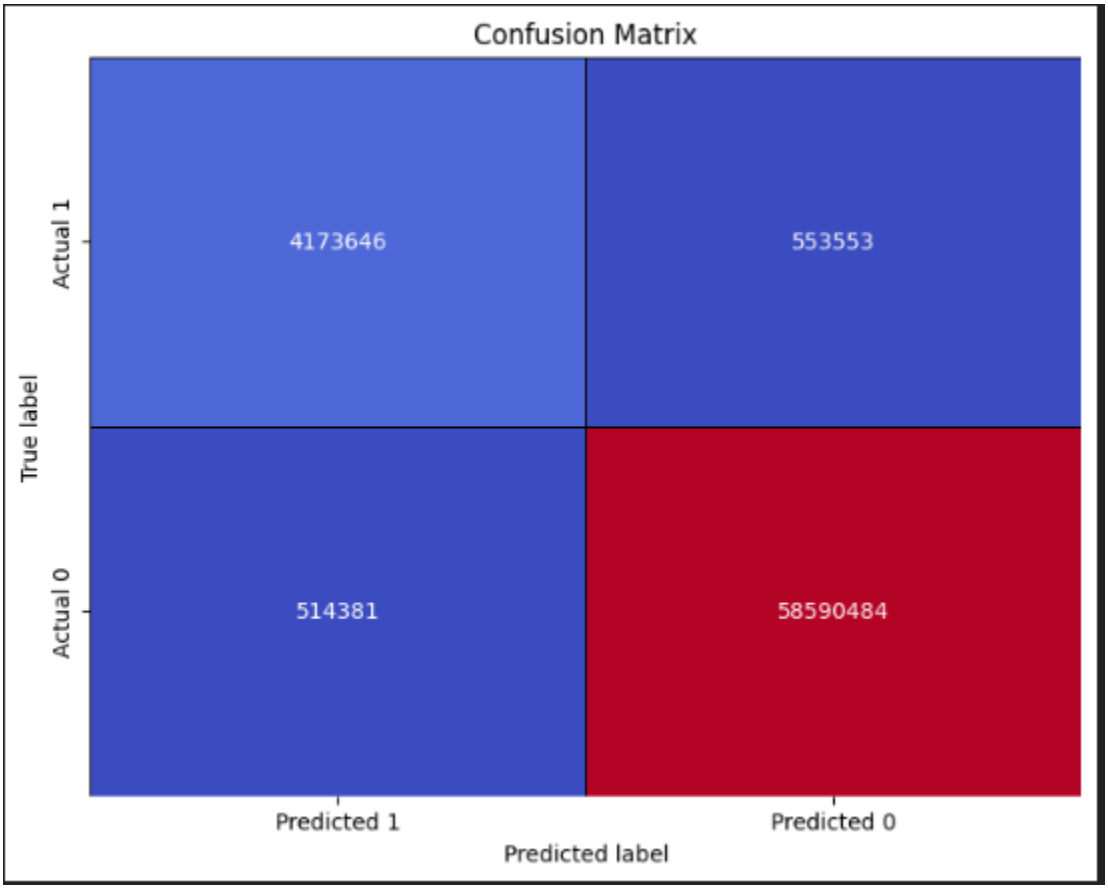
For Accuracy, precision, recall and F1-score we calculated it:

1. We use built in function of ski-learn then take the average of each batch.
2. We use the Tp, Fp, Fn and Tn

```
Val Jaccard score:  0.88313346405328 49
for Val data jaccard score 2: Val: 0.7962572574615479
Jaccard Score for class 3: 0.7962572354137493
---
TP: 4173646, FP: 514381, FN: 553553, TN58590484
---
Precision: 0.9372311824650706
Recall: 0.9317094210770301
F1 Score: 0.9135132886553159
Accuracy: 0.98326963076111 72
---
Precision: 0.8902777223765 99
Recall: 0.8829004236969926
F1 Score: 0.8865737264299338
Accuracy: 0.98326963076111 72
---
```

Here is an example for the Jaccard score, accuracy, F1-score, recall, precision for only 2 epochs.

For Confusion matrix we used Tp, Fp, Fn and Tn and plot the matrix

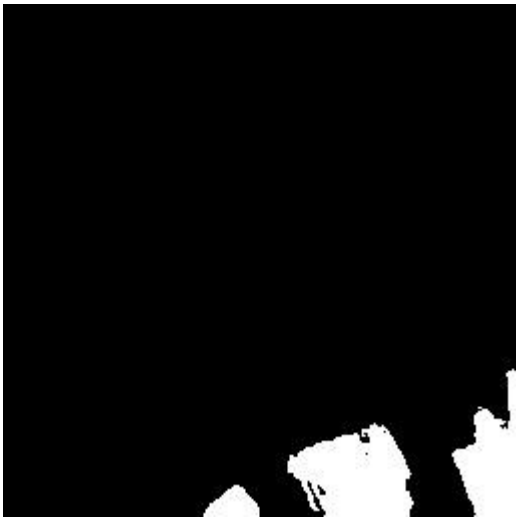


Our Detection:

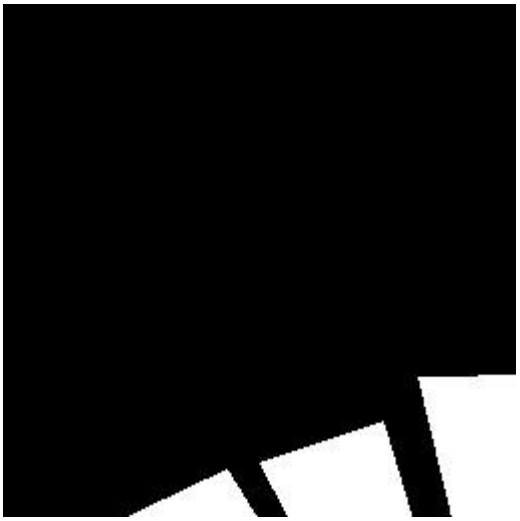
We eventually achieved a Jaccard score of 61.4

Our prediction

ground truth

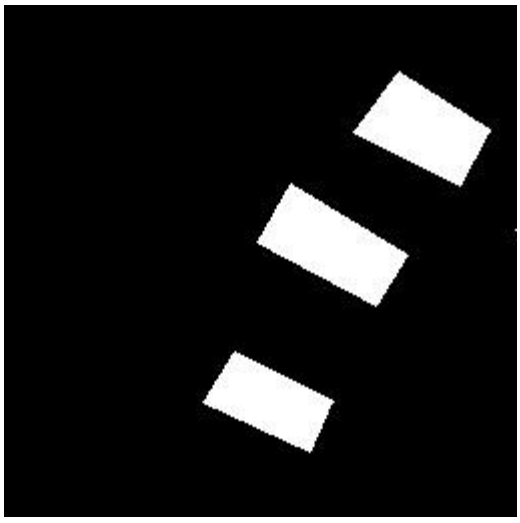


Our prediction



ground truth

Our prediction



ground truth

Unsuccessful trials:

1. We tried to blur the images, but we found that it removed some info we need
2. We tried to detect no change images by histogram where we tried to get patterns in histogram to get whether this is a no change image or not. I mean by no change that there is no budlings in the image
3. SVM, we tried to use SVM with the histograms, but we found that it may affect the accuracy as we needed it to be 100% precision.
4. Adjust brightness, we tried to some adjustments on the brightness, but we found that it not always works as:
    a. Incorrect Target Brightness Value: Ensure that the target brightness value is appropriate for your images. If it's too high or too low, the adjustment might not produce the desired results.
    b. Color Space Conversion: The conversion between HSV and BGR color spaces might introduce artifacts or inaccuracies, especially if the input images have very saturated or unusual colors.

Appendix:

Evaluation of change detection techniques for monitoring land-cover changes: A case study in new Burg El-Arab area

SNUNet-CD: A Densely Connected Siamese Network for Change Detection of VHR Images