Change Detection in Aerial Images

Project instructor: Alex Golts, Rafael

Ayelet Alon, Inbal Tziperman Lotan and Tal Amir, Computer Science department, Technion.

Winter 2020

Table of contents

The Change Detection Task **3**

Generative Adversarial Networks **3**

Part A- Change Detection Using Semantic Segmentation 4

Methods **4**

Semantic Segmentation 4

Siamese Neural Network **4**

Database **5**

Performance Metrics **5**

Results **6**

Part B -Change Detection Using GAP Localization 8

Motivation **8**

Global Average Pooling and Localization **8**

Results **10**

Bbibliography 12

The Change Detection Task

Change detection in remote sensing images is an important part of many applications such as tracking urban changes for military purposes, tracking deforestation for climate change research, agricultural monitoring and more. Tracking changes manually is a tedious task and can be highly time consuming. In addition, since the task is extremely tedious, doing it manually often results in human errors. The automation of this task is therefore highly desirable. In this project, we aim to present an automation of the change detection task using both a fully and weakly supervised semantic segmentation based neural network.

We pose the change detection task as to identify differences between two photos, taken at the same geographical location at two different times. The detected changes should be a result of appearance of new or disappearance of existent objects in a scene. The images may contain other differences between them that are a result of many factors such as seasonal changes (snow, trees with/ without leaves), changes in brightness, shifted images due to slight changes in the capture angle and many more.

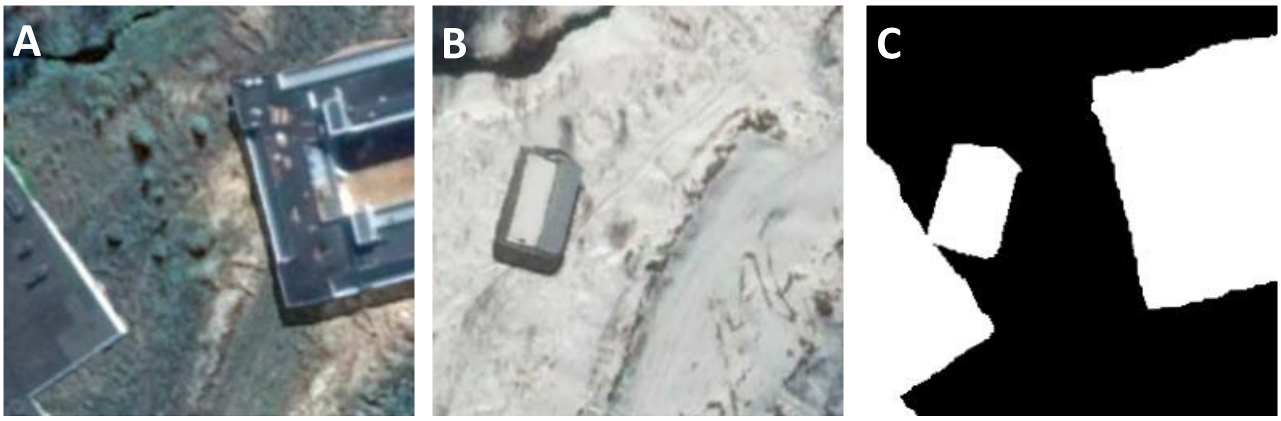


Figure 1: example of a pair of images (A, B) that has changed both due to seasonal changes (snow) and due to appearance and disappearance of objects. The binary map (C) shows only the changes of interest.

Generative Adversarial Networks

Generative Adversarial Networks (GAN) are a class of machine learning systems in which two neural networks interact with each other in order to mutually improve. Given a training set, this technique learns to generate new data with the same statistics as the training set. Traditionally, this technique was used for image generation tasks and proved to be useful for paired and unpaired image-to-image translation tasks. Previous work 0 has shown the automation of the change detection task as we described it here, using a GAN. The model described is composed of two major components: the Generator, and the Discriminator. The Generator is given two photos taken at the same geographical location at different times, and outputs a binary change map. The Discriminator takes three input images: two images for comparison and one image as a difference map, which can be the output from the Generator or a ground truth label. The Discriminator learns to distinguish between a difference map synthesized by the Generator and ground truth labels. The Discriminator parameters are then adjusted based on the classification error and the Generator parameters are updated using the Discriminator output and discrepancy between difference map and ground truth labels.

Using Semantic Segmentation Detection Change -Part A

Methods

In this project, we chose to use a discriminative model rather than a generative model to solve the change detection task. GAN models have a strong advantage in dealing with uncoupled data, learning the joint probability distribution of two random variables. They also excel at generating new photos with a set of attributes learned from the training set. However, they are hard to train requiring a lot of time, memory and are not as stable as discriminative models. Since in our case there is no obvious reason to use a generative model, as our data is coupled and requires learning of the conditional probability distribution $\mathrm{p}(\text{Change map}\ |\text{photos A}, \mathrm{B})$, we chose to implement the task using a discriminative model. If the performance of the model is like the GAN model, it would therefore be preferred due to the easier and less demanding training process. We chose to use a convolutional neural network originally proposed for semantic segmentation [2].

Semantic Segmentation refers to the process of linking each pixel in an image to a class label. It does not differentiate between different instances of the same class, but simply links a label class to each individual pixel. We used this technique here with two class labels: change and no change. The architecture of our network is based on the ERFnet (Efficient Residual Factorized ConvNet for Real-time Semantic Segmentation)1 [2]. The ERFnet introduces a new layer, which combines the use of residual connections and factorized convolutions in order to remain efficient while retaining remarkable accuracy. We chose this architecture to base our model on due to these attributes, and the fact that the code was easily customizable to fit the new task.

The input of our model is composed of two coupled images. Therefore, we decided to create a Siamese neural network. A Siamese neural network is a network in which one set of weights is used for the analysis of two different input vectors. Then, the information received from both network branches is concatenated into a single blob. In our project, we used the encoder of the ERFNet to analyze each input image separately. We then combined the information gathered by concatenating the two outputs of the encoder across the channel dimension, and fed the concatenated blob to the decoder. Figure 2 shows the network modified for change detection.

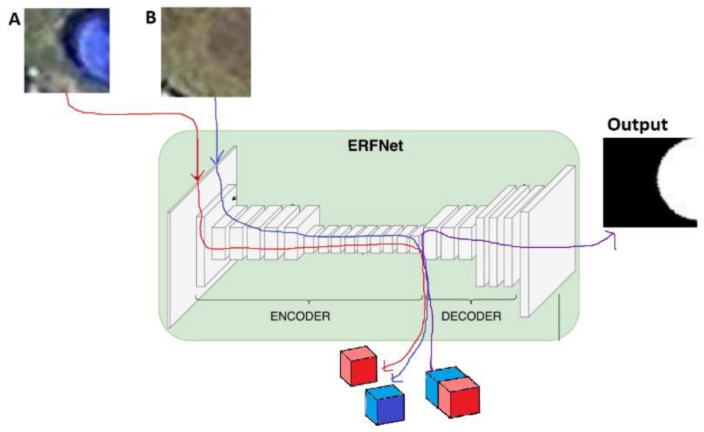


Figure 2: The architecture of our model receives two input images, runs them separately through the Encoder. The resulting feature vectors are concatenated across the channel dimension and runs the joint blob through the decoder. The result is a binary change map.

Database:

To train the network we used a database published in [1]. The database consists of 13,000 pairs of 256x256 pixel images, each taken at the same geographical location at different times. Each pair of photos also has a ground truth label- a binary change map of the same size as the photos, in which every pixel is labeled if it has changed due to the appearance or disappearance of an object between the photos, or not. We split the database into 10,000 training photos, and 3,000 photos left for testing the model.

Performance Metrics

In order to evaluate the performance of our model and compare it to the performance of [1], we calculated mean precision and recall across the test set. The way we calculated these metrics is to our understanding equivalent to what was done in [1], thus enabling us to compare the performance of the two models. To do so, first we used the connected components algorithm in order to extract connected regions from the ground truth labels and from the difference map synthesized by our network. We then defined a tight square area around each connected region in which we would then calculate the Intersection Over Union (IOU) (Figure 3A). We merged overlapping square regions in order to prevent the calculation of a certain area multiple times, thus increasing its weight in the total average (Figure 3B). We define a blob as detected if the IOU is greater than some threshold. Then, for the obtained classification values, we calculated the average values of Precision and Recall across the entire test dataset.

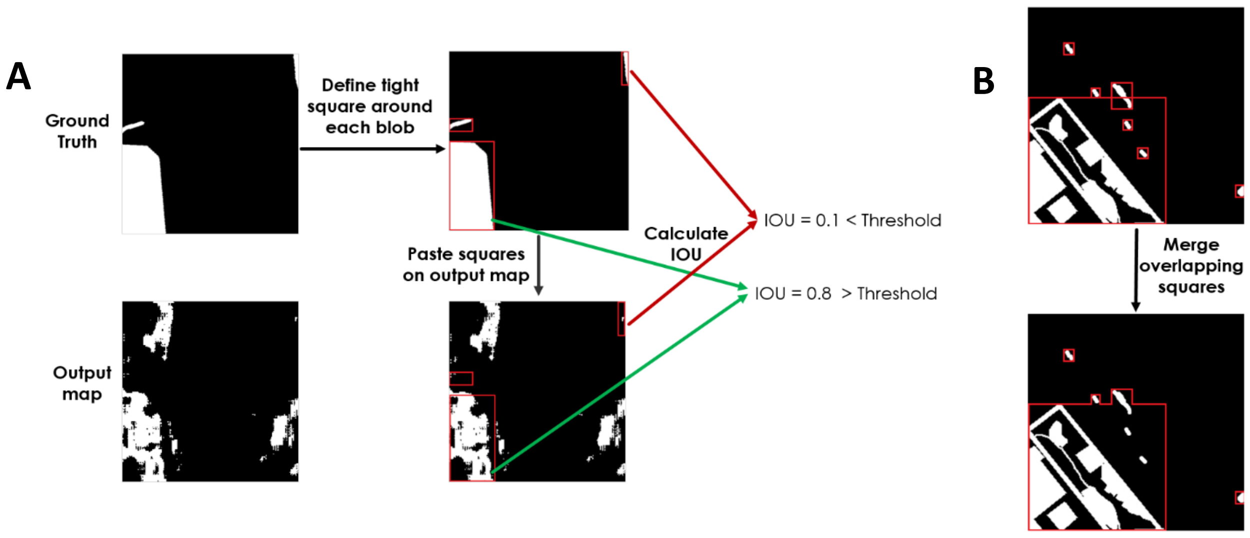

A
Ground Truth
Define tight square around each blob
Paste squares on output map
Calculate IOU
IOU = 0.1 < Threshold
IOU = 0.8 > Threshold
Output map
B
Merge overlapping squares

Figure 3: Performance metrics used to asses our model. (A) Each change in the ground truth label is classified as a detected change if the IOU, calculated in a tight square around the connected region in the map outputted by the model, is greater than some threshold. (B) overlapping square regions were merged in order to prevent the calculation of a certain area multiple times.

Results

We trained the neural network for 150 epochs on 10,000 training images. We then used the test set to evaluate the performance of our network. An example of the model's output is shown in figure 4. As expected, the lower the IOU threshold, the higher the Recall and Precision values were. This is because the trained network often does not find the exact shape of the change (Figure 5).

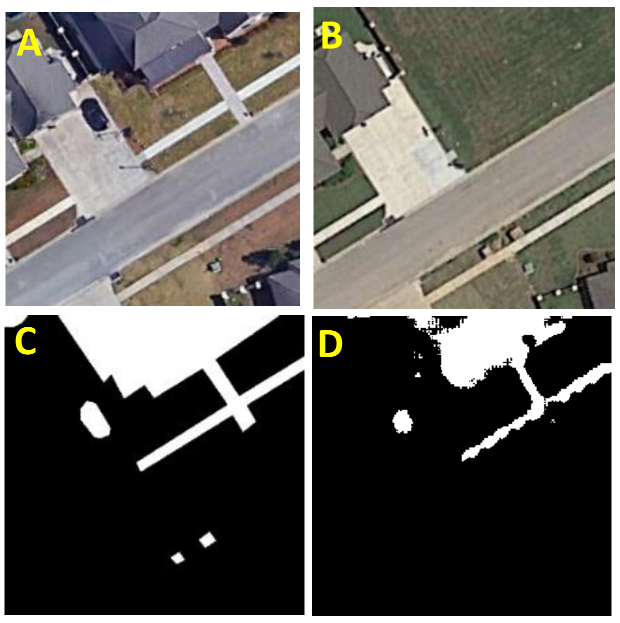
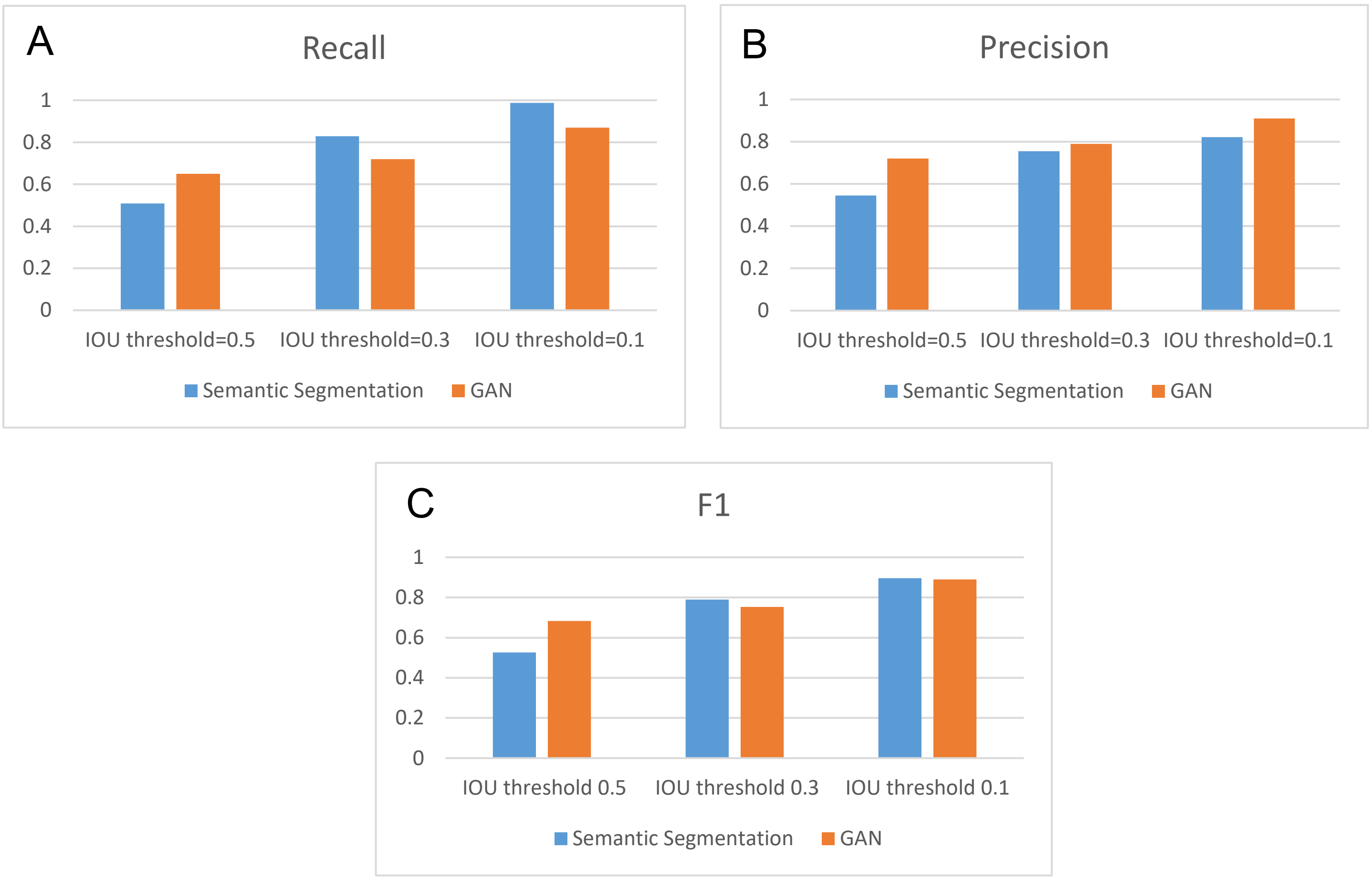


Figure 4: Example output of the model on an image from the test set. Coupled images (A,B), ground truth label (C), change map output (D).

As shown in Figure 5A, for IOU thresholds of 0.3, 0.1, the recall values calculated for the Semantic Segmentation model were higher than the values calculated for the GAN model. For the IOU threshold of 0.1, the system had a recall value of 0.987, meaning that almost all changes were found. For the IOU threshold of 0.5 our model produced a lower recall value than the one calculated for the GAN model. This could indicate that though our system is highly likely to find part of every change, and it is able to notify with high confidence that there was a change in a certain area of the image, it does not necessarily detect the exact shape of the change as well as the GAN model does.



<u>Figure 5:</u> Comparing the precision, recall and F1 metrics evaluating the performance of the two models.

The precision values (figure 5B) were slightly higher using the GAN model for all thresholds, indicating that the Semantic Segmentation model introduces more false positives than the GAN model. For many applications using the change detection task and in particular for military applications, it is far more important to have a high recall value for the model, indicating that more changes were in fact found, even if that results in more false positives.

We also calculated the F1 metric, which combines both the recall and precision metrics. We found that for thresholds 0.3, 0.1 the Semantic Segmentation is higher than the GAN model, but lower for the threshold of 0.5 (figure 5C).

Using GAP Localization Detection Change -Part B

Motivation

In order to train the model in the previous part of this project, we used a training set that included a ground truth label for each pair of photos. These ground truth labels are labor intensive. To alleviate the manual annotation procedure we trained the model using weak supervision. Weak supervision is a branch of machine learning where limited labels are used to provide supervision signal for producing more detailed predictions in test time. In this approach, we use inexpensive, easily obtained weak labels that are imperfect, but can nonetheless be used to create a strong predictive model.

We created a model in which the ground truth labels are simply a binary indication of whether the two input images contain a change or not. This type of label is much simpler to create but it is far less informative. Even so, we would like to output the same binary change map described in the previous part of this project.

Global Average Pooling and Localization

Global Average Pooling (GAP) is an operation that calculates the average value of each feature map. A tensor with dimensions HxWxD is reduced in size to have dimensions 1×1×D by simply taking the average of all values. In the last few years, GAP layers are used to minimize overfitting by reducing the total number of parameters in the model [4].

Researchers from MIT demonstrated that Convolutional Neural Networks with GAP layers that have been trained for a classification task can also be used for object localization[2]. That is, the network does not only output the classification of the image, it also tells us where the object is in the image. We used this technique to locate changes in our input data.

To implement this technique, a classifier is used to classify the input into N predefined classes. The output layer is thus a vector of length N where every value in the vector is the probability calculated by the model that the input belongs to the corresponding class. Just before the final output layer, we add a layer of GAP on the convolutional feature maps and use those as features for a fully connected layer that produces the desired output vector described. The architecture is shown in figure 6A. Each feature map in the final layer preceding the GAP layer acts as a detector for a different pattern in the image, localized in space. To get the location of the object detected by the classifier in the image, we need only to transform these detected patterns to detected objects.

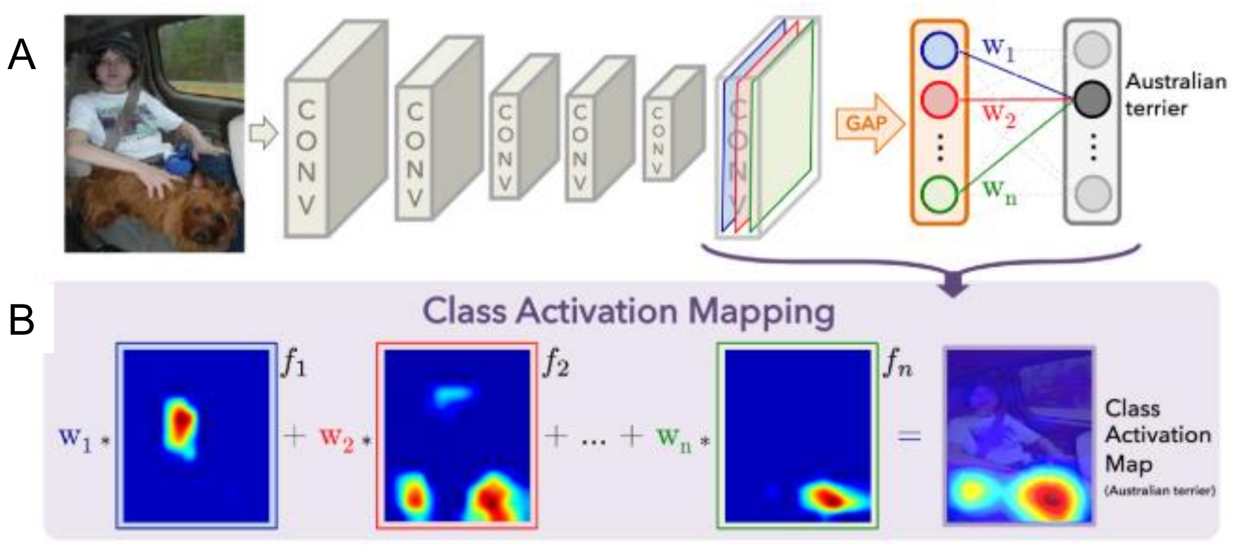


Figure 6: The architecture of the classifier used (A) and the assembly of the localization map (B) [3]2.

Each node in the GAP layer corresponds to a different activation map, and the weights connecting the GAP layer to the final dense layer encode each activation map’s contribution to the predicted object class. For example, if activation map *i* is important for the decision that the image is of class *j*, then the weight connecting the global average of map *i* to node *j* in the final vector, would be high. If however activation map *i* is not important for the decision that the image is of class *j*, then the weight connecting the global average of map *i* to node *j* in the final vector, would be very low, or even 0. Thus, to obtain the final localization map, we sum the contributions of each of the activation maps multiplied by its corresponding weight (figure 6B). By doing so, detected patterns that are more important to the predicted object class are given more weight.

We altered the model from the previous part of this project to classify image pairs into the change and no-change classes. We also added a GAP layer and a fully connected layer as described for the GAP Localization. The training set consists of 40,000 images, which were obtained by dividing each of the images from the GAN paper's database into four 128x128 pixels images. We divided the pictures since in the original database 99.29% of the training set image pairs and 99.17% of the test set image pairs contain change. In order to build a good classifier, we wanted to create a balanced database. After dividing each photo into 4 parts, 67.26% of the training set image pairs and 67.48% of the test set image pairs contain change.

Results

We trained the neural network for 150 epochs, and tested it using a test set which consists of 12,000-coupled pictures (these were again obtained by dividing the pictures from the test set into four). Though the classifier itself was not the main goal of this experiment, its performance is crucial to the success of the GAP localization done next. If the model cannot classify that there was change, it will certainly not be able to locate it. We used a Precision-Recall curve (Figure 7) which summarizes the trade-off between the true positive rate and the positive predictive value for a predictive model using different probability thresholds. By calculating the maximal harmonic average between the precision and the recall, we chose the probability threshold of 0.53. The Recall and precision values obtained for this threshold were 0.9 for both metrics.

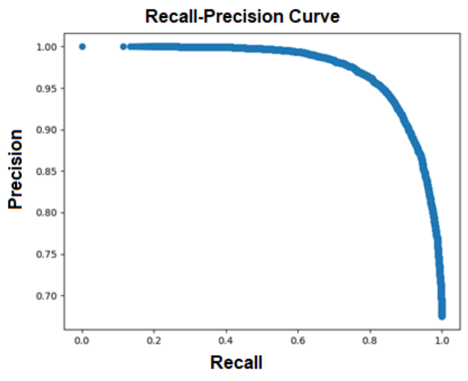


Figure 7: precision recall curve.

Using the GAP localization method described and the classifier we trained, we constructed a change map for the images in the test set. Figure 8 shows some examples of the change maps constructed by the model. The output is not a binary map, but a heat map indicating which areas in the photo are more or less important to the Classifier's decision.

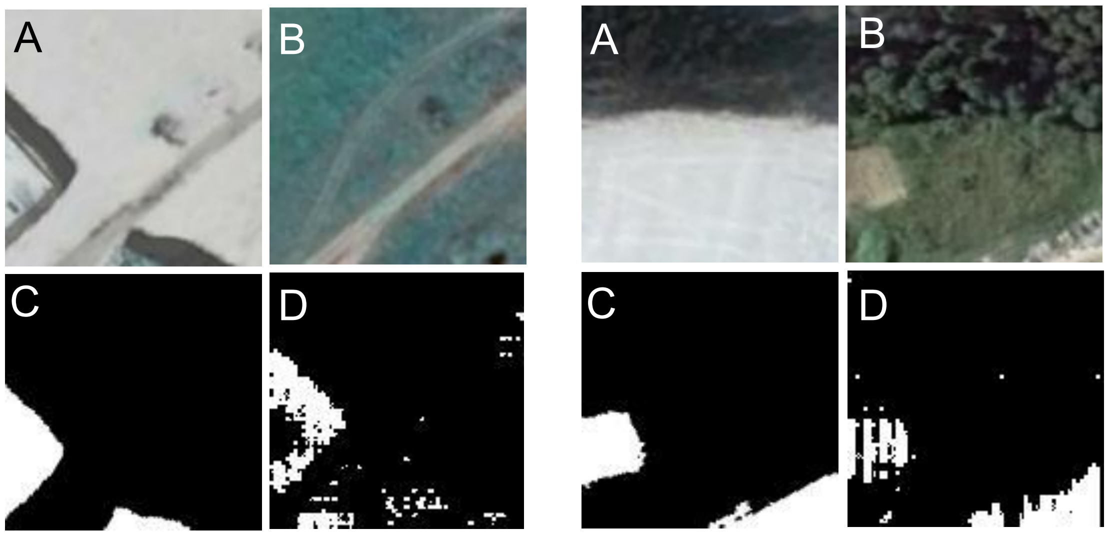


Figure 8: Example output from the GAP Localization method on test set images. Coupled images (A,B), ground truth label (C), binary change map output from the GAP localization method (D).

We transformed the heat maps received from the GAP localization model into binary heat maps using a threshold of 0.2. This threshold was chosen by looking at a histogram of pixel values and separating between maps with/ without change. Next, we wanted to test the performance of the binary change maps created, and compare them to the maps created by the Semantic Segmentation and GAN models. To do so, we used the same metrics described in the previous part of the project, classifying each change as detected or not via the IOU value, and calculating the precision and recall values across the entire test set (Figure 9).

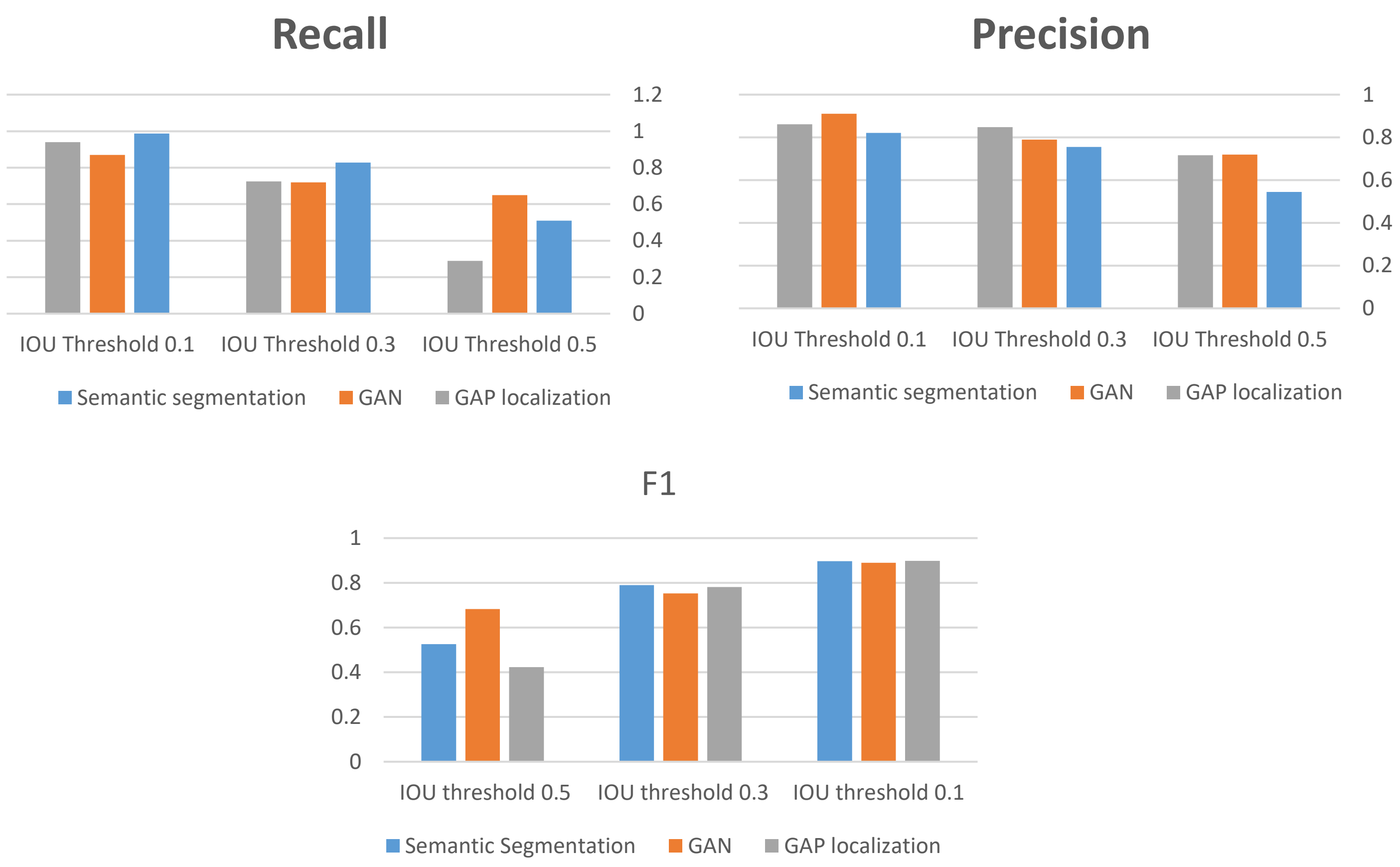


Figure 9: Comparing the precision, recall and F1 metrics evaluating the performance of the three models (calculated on each corresponding test set). Semantic Segmentation and GAN values are the same as shown in previous part of this project.

The precision values calculated for the maps created by the GAP Localization method exceeded the values calculated for the maps created by the Semantic Segmentation model for all tested IOU thresholds, and similar to the maps created by the GAN model.

The recall values calculated for the change maps created by the GAP Localization method were inferior to the ones corresponding to the GAN and Semantic Segmentation models, apart for the IOU threshold of 0.1 where the recall value of the GAN model was lower. We also calculated the F1 metric. We found that for thresholds 0.3, 0.1 the GAP is higher than the GAN model, but lower for the threshold of 0.5.

These results show that weak supervision can be used in order to train a network for change detection in aerial photos. This enables us to train the network on alternative databases if needed with minimal effort for creating the ground truth labels while still gaining a similar performance in detecting changes.

Bbibliography

1. Lebedev, M. A., et al. "Change Detection In Remote Sensing Images Using Conditional Adversarial Networks." International Archives of the Photogrammetry, Remote Sensing & Spatial Information Sciences 42.2 (2018).

2. Romera, Eduardo, et al. "Erfnet: Efficient residual factorized convnet for real-time semantic segmentation." IEEE Transactions on Intelligent Transportation Systems 19.1 (2017): 263-272.

3. Zhou, Bolei, et al. "Learning deep features for discriminative localization." Proceedings of the IEEE conference on computer vision and pattern recognition. 2016.

4. Lin, Min, Qiang Chen, and Shuicheng Yan. "Network in network." arXiv preprint arXiv:1312.4400 (2013).