CSC591/791-Spatial and Temporal Data Mining

Final Project Report

Team 1

Preya Shabrina - pshabri

Pushpendra Pathak - ppathak

Avi Agrawal - aagraw24

Cody Morazan - camoraza

Unsupervised Change Detection in Satellite Images Using Convolutional Neural Networks

Kevin Louis de Jong
Department of Computer Science
University of Pretoria
Pretoria, South Africa
kevinkatdj@gmail.com

Anna Sergeevna Bosman
Department of Computer Science
University of Pretoria
Pretoria, South Africa
annar@cs.up.ac.za

Abstract**—**This project deals with an efficient unsupervised method for detecting changes in high resolution satellite imagery. A convolutional neural network (CNN) based on the U-Net architecture is used for semantic segmentation, or feature extraction of the image and difference images are calculated to effectively determine the occurrence of any image changes. Since the difference images are generated by using feature maps of CNN, it means that the CNN does not need to learn non linear mapping between images and thus is unsupervised. The key feature of the approach is that it being unsupervised it has the capability of being utilized with other CNN's that are trained to perform feature detection.

Keywords-Semantic segmentation; Unet; Convolutional Neural Network; Change Detection

I. Introduction

Change detection is an important process that has many uses in different fields of interest such as military, business, agriculture, etc.. The main objective of this project is to implement an efficient method that includes unsupervised change detection of high resolution satellite images which are temporally different for the same scene. Furthermore, we wish to classify them into one of the three labels or classes- buildings, roads/parking lots and any other change which the other two classes does not encompass. Previous methods involved classifying temporal images and committing post-classification analysis to detect change. However it required very high accuracy of classification, greater resources and time which acted as a bottleneck. In this project we implement a new method of change detection by constructing a different image of the previous image to do the same task in the high resolution satellite imagery with lesser resources and time.

A. Convolutional Neural Networks

CNNs have been used effectively in a wide range of applications associated with computer vision. The name is derived from the operation that sets CNNs apart from other neural networks (NNs): the convolution operation. During training, CNN learns a set of weight matrices, or kernels, that convolve over an image to extract image features. Given an $n \times n$ input and a $k \times k$ kernel, the convolution operation slides the kernel over the input, and calculates the Hadamard product for each overlap between the kernel and the input. Convolving a single kernel with an input image produces a feature map, i.e. an $m \times m$ matrix of activations, where $m = (n - k + 2p)/s + 1$, where p is an optional padding parameter, and s is the stride, or step size used to slide the kernel. A feature map captures the presence of a specific feature across the image by exposing the correlations between neighbouring pixels. Convolutional layers are collections of feature maps, which result from applying multiple kernels to the original image, or previous convolutional layers. Early convolutional layers extract simple features, such as lines and edges, whereas later layers extract more complex features such as shapes, patterns, and concepts. Thus, feature maps capture a compressed hierarchical representation of the objects present in the input image. Further compression is achieved by applying a max pooling operation between convolutional layers, which reduces the dimensionality of feature maps, favouring stronger activation signals. The ability to construct compressed hierarchical image representations makes CNNs appealing for the purpose of change detection, and semantic segmentation

The task of constructing a difference image first requires the successful completion of the task of semantic segmentation, or feature extraction. This is accomplished by the implementation of a CNN. The CNN is of great importance as its simplified training process allows for effective manipulation of the difference images to correctly produce change detection results.

B. Contribution of the Paper

The paper gives an efficient algorithm to detect changes in satellite imagery. It also demonstrates how the output of a trained U-net model, i.e, semantically segmented input image can be used to detect changes efficiently in satellite imagery.

The structure of the rest of the report is as follows: (II) Related Work, (III) Methodology, (IV) Implementation, (V) Results, (VI) Conclusion, (VII) Github Repository Link, and (VIII) References.

II. Related Work

Change detection in optical and satellite imagery is of high interest and an extensively explored problem in the field of computer vision. Employing neural networks to detect changes has a long history in computer vision research. Researchers have exploited simple neural networks to special variants of neural networks such as convolutional neural networks (CNN), recurrent neural networks (RNN), and SegNets to detect changes in images. Efficiency becomes a big concern in image change detection when it comes to the large-scale spatial temporal data collected from satellites or earth observation systems. Many researchers have focused on improving efficiency of existing algorithms for change detection when applied to satellite imagery. But, more research is required. Below we present a literature review of explored methods to address the problem of change detection in simple optical images and also in satellite images.

The study by Ghosh et al. [15] uses a modified version of the Hopfield NN in conjunction with traditional DI techniques to help consider the spatio-contextual information of each pixel.

A recent study by Gong et al [16] has employed DNNs for change detection. DNNs have the ability to extract a compressed hierarchical feature representation of an image, which enables a meaningful semantic comparison of temporally different images.

The study by Chu et al.[6] uses a deep belief network (DBN) to increase the changed areas and decrease the unchanged areas in a DI. Their framework consists of two DBNs, where each DBN learns features from one of two temporally different images.

Gong et al.[12] make use of unsupervised feature learning performed by a CNN to learn the representation of the relationship between two images. The CNN is then fine-tuned with supervised learning to learn the concepts of the changed and the unchanged pixels.

O. Yousif et al.[13] investigated a nonlocal means (NLM) denoising algorithm that combines local structures with a global averaging scheme in the context of change detection using multitemporal SAR images. Both visual and quantitative analyses have proven the efficiency of the PCA-NLM algorithm in improving urban change detection using multitemporal SAR images.

III. Methodology

A. *Dataset*

For this project, the dataset used is the Vaihingen dataset, provided by the International Society for Photogrammetry and Remote Sensing (ISPRS). The dataset is made up of 33 satellite image segments of varying size, each consisting of a true orthophoto (TOP) extracted from a larger TOP mosaic. Of these segments, 16 are provided with the ground truth semantically segmented images. All images in the dataset are that of an urban area, with a high density of man-made objects such as buildings and roads. We used the exact same dataset. Download link: [http://www2.isprs.org/commissions/comm3/wg4/2d-sem-label-vaihingen.html].

B. *U-net architecture*

U-net architecture is a convolutional network architecture that is used for fast and precise segmentation of images with very less training data. A U-net uses convolutional layers stacked together to make an encoder which encodes an input image into feature representation at different levels with the help of max pooling. U-net also uses deconvolutional layers to up-sample the output of the encoder. In order to recover object detail better during up-sampling, U-net copies and concatenates high resolution features from layers in the encoder path to features in the corresponding layer of the decoder path. This operation also serves to recover information on the position of pixels before the image was compressed by the convolution and the max pooling operations. The decoder performs up-sampling and concatenation, followed by regular convolution operations. Figure 1 shows the implemented U-net architecture.

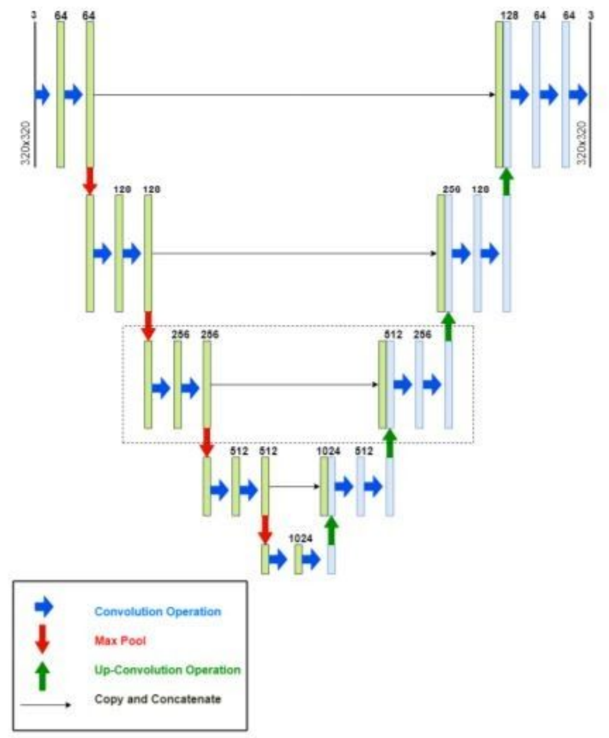

Figure 1 : U-net architecture

The encoding is implemented in five levels of U-net. The input image size was 320*320*3. For the first four levels, two convolutions of inputs are implemented with a kernel size of 3*3. A stride of 1 and padding of 1 was used. After every convolution, batch normalization is performed. In these four levels, after both the convolutions on each level, max pooling operation is performed to prevent over fitting. A stride of 2 and padding 0 was used for max pooling. For the final layer, the process was similar apart from the absence of max pooling operation. The deconvolution for upscaling (decoding) has 5 levels as well. The first four levels had one deconvolution operation followed by two convolution operations in each layer. The last layer had a third convolution layer to make sure that the generated output had the correct number of channels.

The first level of encoding had 64 kernels and doubled on every subsequent layer. Thus at the fifth layer, we had 1024 kernels. In decoding, the kernels halved with every layer resulting in 64 again on the final layer. However we applied one additional convolution as mentioned before which is why the final output had 3 kernels only.

C. *Training*

For training of the model, adaptive moment estimation of gradient descent was implemented because it requires very less tuning of parameters. The model was trained to perform semantic image segmentation with a mini-batch size of 4. The log loss function was taken as the loss function The training was done for 20 epochs. The learning rate was 0.0002.

D. *DI algorithm*

Let f1 be a feature map of image 1 and f2 be a feature map of image 2. Let threshold value be t. Assuming the dimensions of feature maps are m x m, initially the DI(difference image) matrix is a m x m matrix of zeros.

```
for i=1 to m do:
        for j=1 to m do:
                If $f2_{ij}$ - $f1_{ij}$ >=  t do:
                        $DI_{ij} = f2_{ij}$
```

Algorithm 1 : Difference Image Generation Algorithm

E. *Change detection*

The U-net is tweaked a little to accept two images as input. It is a convention that the first image comes first temporally and the second image comes after.. On every level of the U-net, a difference image is constructed for the images using their feature maps which were extracted beforehand using the CNN. To construct the difference image, the corresponding values of feature maps of input image 1 and 2 are compared. If the difference of values between them is less than the threshold set, the value in the difference image matrix is set to zero. If not, it is set to be the same as that of the feature map from the second image. This gives us different difference images in different levels of the U-net with the same dimensions as of feature maps. These difference images are then feeded into the upscaling of the segmented image.The threshold values for different levels taken are 0.4, 0.6, 0.8, 1 and 1.2 for levels 1, 2, 3, 4 and 5 respectively.

F. *Evaluation*

In our project, we are identifying the changes in two satellite images and then classifying the pixels in a specific label out of the three available ones. Hence the evaluation is performed on both these operations separately.

For the identification of changes in the images, we make a contingency table stating the true positives (TP), true negatives (TN), false positives (FP) and false negatives(FN). The accuracy of the change identification is thus given by the sum of true positives and true negatives divided by the sum of all the pixels in the image. Mathematically it can be stated as-

PCC1 = (TP+TN)/(TP+TN+FP+FN)

True positive refers to the number of pixels that were changed in image 2 and were correctly identified as changed. True negative refers to the number of pixels that were not changed in image 2 and were correctly identified as not-changed. False positive refers to the number of pixels that were not changed in image 2 and yet identified as changed. False negatives are the number of pixels that were changed in image 2 and yet were identified as not-changed.
For the evaluation of classification of pixels in the right class or labels, we use another evaluating formula which is a simple ratio of total correctly labelled pixels and total pixels in the image. The mathematical representation of the formula can be written as-

PCC2 = Correctly classified pixels / Total pixels

IV. Implementation

A. *Language and Libraries*

We used Python (3.6.10) with Jupyter notebook for our implementation. To read images, we used *Pillow*, for visualization we used *Matplotlib*, for array manipulation we used *Numpy* and *math,* to design and train U-net we used *Tensorflow* and *Keras,* and finally for evaluation we used *Scikit-learn.metrics.*

B. *Dataset*

We used the same dataset used in the paper. The details are already provided in the previous section.

C. *Data Preprocessing*

Within our selected dataset, there were 16 TOP images with ground truth. We used 15 images for training and kept 1 for testing purposes. All images were cut into 320x320 segments. Partial images near the border were eliminated. After augmentation we had 601 images for training. Each image was normalized to have a mean of 0.5 and std of 0.5.

D. *U-net design*

We designed the exact same U-net described in the previous section with all parameters adopted from the paper. The only difference is we didn't 0 padding in any of the layers so that no pixel information is lost, whereas in the paper, 0 padding was used in all deconvolution and max pooling layers.

E. *Training*

We trained our U-net model with 601 images [all parameters adopted from paper]. But, while training we used 80%-20% train-validation split, whereas nothing was mentioned regarding validation in the paper. We saved our best model to disk as best_validated_model.hdf5 which we used in later phases.

F. *Change Detection*

We implemented the Difference Image Generation Algorithm and implemented the change detection system as described in the paper with all parameters intact.

G. *Test Data Preparation*

After augmentation we had 36 total images from our single test image. We selected 12 images from those 36 images, such that each selected image had part of Immutable Terrain [roads, driveways, parking lots - class 0 (red)], Background [vegetation, cars, others - class 1(green)], and buildings[class 2 (blue)]. We used these 12 images to evaluate the trained U-net for semantic segmentation. So, the evaluation was based on 1,228,800 (12x320x320) pixels. Since, our dataset didn't contain any image from the same location but from different time, to evaluate the change detection method we created 6 simulated images from 2 of our test images. In the first 3 simulated images we changed the original images by 5%, 10%, and 15%. In the other 3 simulated images, we applied Gaussian noise of variance 10, 20, and 40 to the 5% changed image. We used these simulated images along with the original images for change detection.

H. *Evaluation*

To evaluate the entire system, we implemented PCC1 and PCC2 as described in the previous section. To evaluate semantic segmentation using our trained U-net, we used PCC2 to identify how many of the pixels were classified correctly. To evaluate the change detection method, we used PCC1 to identify if the changed pixels were identified correctly and we used PCC2 to identify if the changed pixels were labeled correctly.

V. Results

A. *Evaluation of Semantic Segmentation*

Our trained model achieved a validation accuracy of ~87% at 20th epoch. The log loss at each epoch is plotted in figure 2. However, the accuracy reported in the paper was 89% which is slightly better than us.

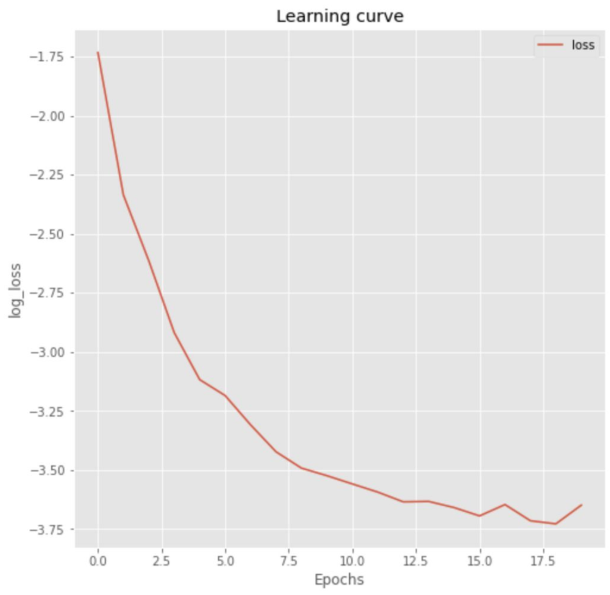

Figure 2 : Log Loss at Each Epoch of Training.

The model was trained on a personal laptop and it took around 6-7 hours. The U-net marked Immutable Terrain [class 0] with color red, Background [class 1] with green, and buildings [class 2] with blue. The overall accuracies on the test images are reported in the table 1.

Table 1 : Semantic Segmentation Results

	Class Label	Color Label	PCC1
Immutable Terrain	0	Red	84.0%
Background	1	Green	71.8%
Buildings	2	Blue	91.6%
Overall	-	-	82.9%

We achieved fairly high accuracies comparable to the paper for buildings (the paper reported 92.2%) and immutable terrains(the paper reported 88.1%), but low accuracy for background (The paper reported 86.1%). This can be due to the fact that the dataset was small and less training pixels belonged to this class. Also, this class has less differentiable features to learn for the model. Sample labelling performed by the trained U-net is depicted in figure 3.

B. *Evaluation of Change Detection*

We performed change detection on the 6 simulated images. The results are summarized in table 2.

Change detection for the simulated images are depicted from figure 4-9. Notice the unchanged pixels are pushed towards color blue. The reason behind this can be the normalization technique adopted or the pixel value managing mechanism of the libraries used.

Also, Notice that for test case 2 and 3 (figure 5 and 6), we obtained the lowest accuracy. The changed segment of these two test cases had a mixture of multiple class objects and the change detection algorithm struggled to find proper labels near the boundary.

Additionally, for test case 4, 5, and 6 (figure 7, 8, and 9), although we introduced gaussian noise of varying variances to the images, the noise had no impact on the segmentation process. This indicates towards the robustness of the change detection algorithm in the presence of noise.

For all the 6 test cases, our implementation got higher values for PCC1 than the paper (The paper reported 91.2, 88.7, 87.5, 91.0, 86.2, and 81.5% respectively). However, we got slightly lower accuracies for PCC2 for the test cases (The paper reported 93.0, 91.2, 90.7, 92.0, 89.2, and 85.4% respectively).

Table 2 : Change Detection Results

Pixels Changed	**PCC1**	**PCC2**
5%	99.2%	87.9%
10%	97.8%	61.5%
15%	85.6%	78.7%
Gaussian Noise Variance	**PCC1**	**PCC2**
10	99.3%	88.8%
20	99.2%	88.6%
40	99.2%	89.2%

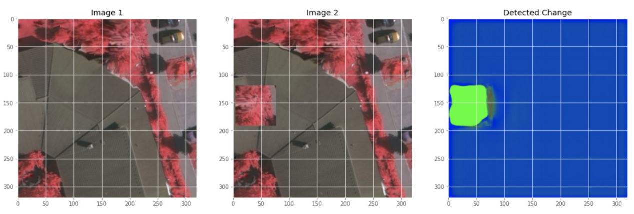

Figure 4 : a) Original image; b) 5% changed image; c) Detected change.

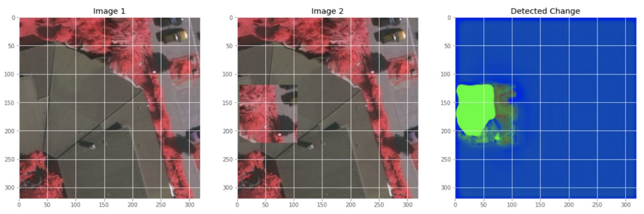

Figure 5 : a) Original image; b) 10% changed image; c) Detected change.

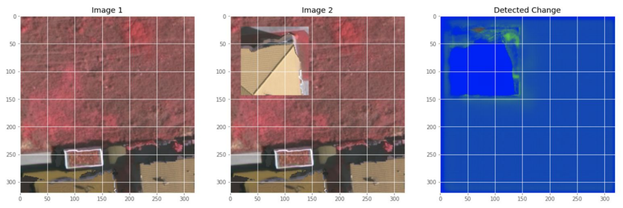

Figure 6: a) Original image; b) 15% changed image; c) Detected change.

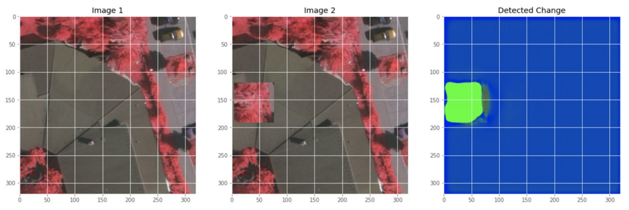

Figure 7 : a) Original image; b) 5% change + Gaussian noise of variance 10; c) Detected change.

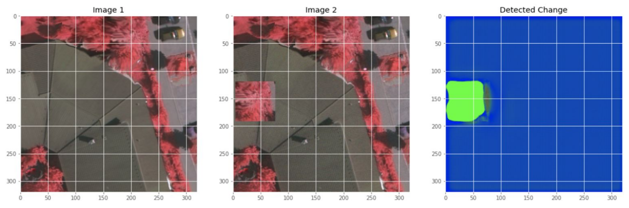

Figure 8 : a) Original image; b) 5% change + Gaussian noise of variance 20; c) Detected change.

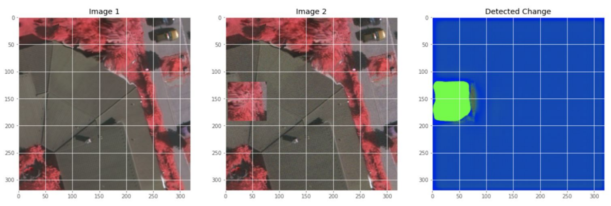

Figure 9 : a) Original image; b) 5% change + Gaussian noise of variance 40; c) Detected change

VI. Conclusion

The paper presents a unique method to detect changes in satellite imagery and the project was highly educational for us. We learnt image preprocessing, designing and training neural nets using *Tensorflow*, change detection, and several evaluation techniques. We faced least difficulties while designing and training neural nets and implementing change detection algorithms. We realized that image preprocessing takes a lot of time. Finding a library for correct preprocessing is a very crucial task. Because different libraries support the same task, but they handle images differently. In some cases, if the library does not support the particular format(for example .tif images are not supported by *OpenCV* and *Matplotlib*) of the input images, they do not show any error, rather they misinterpret pixel values killing the accuracy of the entire system silently. We were getting low accuracies for a very long time due to the use of an inappropriate library. Now, our results are close to the results reported in the paper. But we haven't been able to reach the exact accuracy. As a future work, if we are to implement this project again, we would like to focus on small details and research more about the libraries to find a more appropriate one to achieve higher accuracies. Also, from the results we got and results reported in the original paper, we found out that the trained U-net struggles near the boundary. The authors attributed this phenomena to the fact that they didn't really focus on parameter tuning (ex. different thresholds applied on the different encoder layers, training epochs etc.) and U-net training. So, we would also like to observe the impact of better training and parameter tuning on the change detection mechanism. Moreover, we chose a very small dataset and the future work requires scaling it to a larger dataset. This dataset also had simulated changes in images and we need to see if real changes would give us high accuracy too.. Also, the threshold values for constructing the difference images were taken by us arbitrarily which can be automated for best results. The model uses CNN hence is resistant to noise but we need to verify what the effects of camera angles, elevation, reflection, sunlight has on our model's accuracy. To scale the model to make difference images accurately while taking in consideration the atmospheric changes has to be worked upon.

VII. Github Repository

https://github.ncsu.edu/pshabri/CSC591_791_ChangeDetectionInSatelliteImagery_Team_1

Acknowledgment

We would like to thank the authors of the paper and Dr. Ranga Raju Vatsavai for giving us the opportunity to implement a research paper as a course project which tremendously enriched our knowledge. We also thank the authors of the paper for their wonderful contribution in the field of Spatial and Temporal data mining.

VIII. References

[1] R. J. Radke, S. Andra, O. Al-Kofahi, and B. Roysam, "Image change detection algorithms: A systematic survey," IEEE Transactions on Image Processing, vol. 14, no. 3, pp. 294–307, 2005.

[2] X. Chen, L. Vierling, and D. Deering, "A simple and effective radiometric correction method to improve landscape change detection across sensors and across time," Remote Sensing of Environment, vol. 98, no. 1, pp. 63–79, 2005.

[3] T. L. Sohl, "Change analysis in the United Arab Emirates: an investigation of techniques," Photogrammetric engineering and remote sensing, vol. 65, no. 4, pp. 475–484, 1999.

[4] S. Yang, Y. Li, G. Feng, and L. Zhang, "A method aimed at automatic landslide extraction based on background values of satellite imagery," International Journal of Remote Sensing, vol. 35, no. 6, pp. 2247–2266, 2014.

[5] Y. LeCun, K. Kavukcuoglu, C. Farabet et al., "Convolutional networks and applications in vision," in Proceedings of the IEEE International Symposium on Circuits and Systems, 2010, pp. 253–256.

[6] Y. Chu, G. Cao, and H. Hayat, "Change detection of remote sensing image based on deep neural networks," in Proceedings of the International Conference on Artificial Intelligence and Industrial Engineering, 2016, pp. 262–267.

[7] H. Noh, S. Hong, and B. Han, "Learning deconvolution network for semantic segmentation," in Proceedings of the IEEE international conference on computer vision, 2015, pp. 1520–1528.

[8] O. Ronneberger, P. Fischer, and T. Brox, "U-net: Convolutional networks for biomedical image segmentation," in Proceedings of the International Conference on Medical Image Computing and Computer-Assisted Intervention, 2015, pp. 234–241.

[9] A. Singh, "Digital change detection techniques using remotely-sensed data," International Journal of Remote Sensing, vol. 10, no. 6, pp. 989– 1003, 1989.

[10] E. F. T. D. Vermote, D. Tanre, J. L. Deuz ´ e, M. Herman, J. J. Morcrette, ´ and S. Y. Kotchenova, "Second simulation of a satellite signal in the solar spectrum-vector (6SV)," in 6S User Guide Version 3, 2006, pp. 1–55.

[11] D. Hoja, M. Schneider, R. Muller, M. Lehner, and P. Reinartz, "Comparison of orthorectification methods suitable for rapid mapping using direct georeferencing and RPC for optical satellite data," in The International Archives of the Photogrammetry, Remote Sensing and Spatial Information Sciences, vol. 37, January 2008, pp. 1617–1624.

[12] M. Gong, L. Su, M. Jia, and W. V. Chen, "Fuzzy clustering with a modified MRF energy function for change detection in synthetic aperture radar images," IEEE Transactions on Fuzzy Systems, vol. 22, pp. 98– 109, 2014.

[13] [13] O. Yousif and Y. Ban, "Improving urban change detection from multitemporal SAR images using PCA-NLM," IEEE Transactions on Geoscience and Remote Sensing, vol. 51, pp. 2032–2041, 2013

[14] X. Long Dal and S. Khorram, "Remotely sensed change detection based on artificial neural networks," Photogrammetric Engineering and Remote Sensing, vol. 65, pp. 1187–1194, 1999.

[15] S. Ghosh, L. Bruzzone, S. Patra, F. Bovolo, and A. Ghosh, "A context sensitive technique for unsupervised change detection based on hopfield type neural networks," IEEE Transactions on Geoscience and Remote Sensing, vol. 45, pp. 778–789, 2007.

[16] M. Gong, J. Zhao, J. Liu, Q. Miao, and L. Jiao, "Change detection in synthetic aperture radar images based on deep neural networks," IEEE transactions on neural networks and learning systems, vol. 27, no. 1, pp. 125–138, 2015.

[17] D.-A. Clevert, T. Unterthiner, and S. Hochreiter, "Fast and accurate deep network learning by exponential linear units (ELUs)," in Proceedings of the International Conference on Learning Representations, 2016, pp. 1–14.

[18] G. Lin, C. Shen, A. Van Den Hengel, and I. Reid, "Efficient piecewise training of deep structured models for semantic segmentation," in Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, 2016, pp. 3194–3203.

[19] D. P. Kingma and J. Ba, "Adam: A method for stochastic optimization," in Proceedings of the International Conference on Learning Representations, 2015, pp. 1–15.

[20] International Society for Photogrammetry and Remote Sensing (ISPRS). (2018) 2D Semantic Labeling - Vaihingen data. [Online]. Available: http://www2.isprs.org/commissions/comm3/wg4/2d-sem-label-vaihingen.html

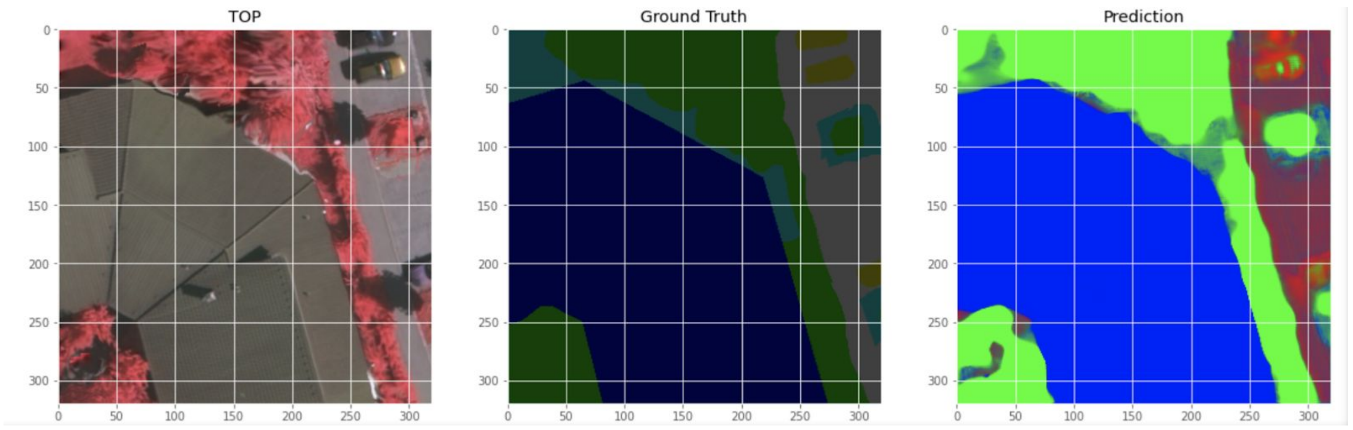

Figure 3 : a) Original image; b) Ground Truth; C) Semantically Segmented image (notice the color label for each class - Terrains are pushed towards color red)