Segmentation of Satellite Imagery

Jimut Bahan Pal

Under the Guidance Of

Tamal Maharaj

Department of Computer Science
Ramakrishna Mission Vivekananda Educational and Research Institute
Howrah - 711202

April 25, 2021

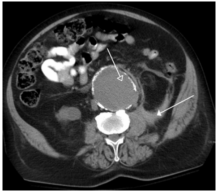
Table of Contents

What is Image Segmentation?

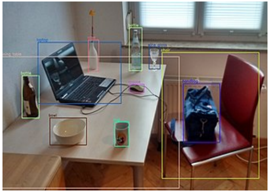
- Partitioning a digital image into set of image objects.
- Assigning labels to each pixel of image, which share some common characteristics.
- Makes an image easier to analyze.
- Object localization.

Where do we need Image Segmentation?

Surveillance (P.C.: Wikimedia)



Medical Segmentation (P.C.: Wikimedia)



Detection (P.C.: Wikimedia)

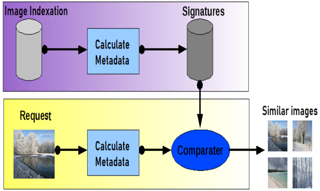
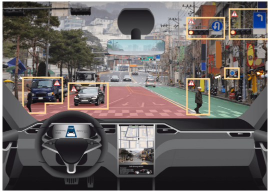


Image retrieval (P.C.: Wikimedia)

Machine Vision (P.C.: Cognex)

Finger Print Recognition (P.C.: Wikimedia)

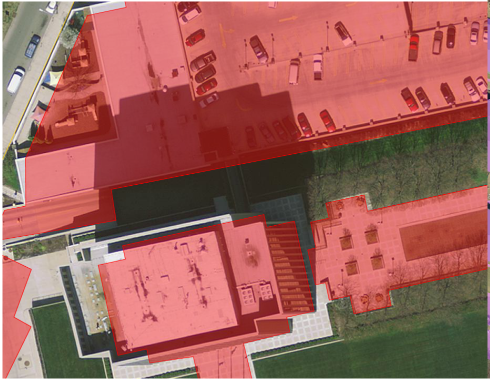
Image Tagging (P.C.: Facebook Research)



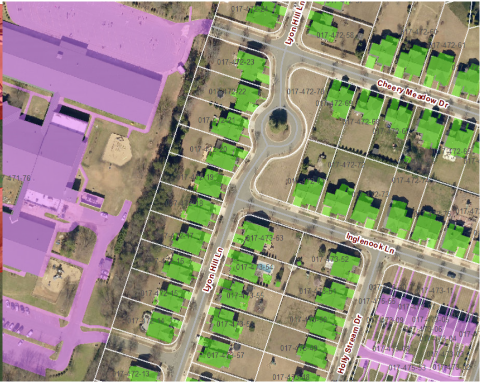
Self Driving Cars (P.C.: Cloudfront)

What is Segmentation of Satellite Imagery?

- Dividing aerial satellite images to different parts for better analysis.



(a) P.C.: eijournal



(b) P.C.: mecknc.gov

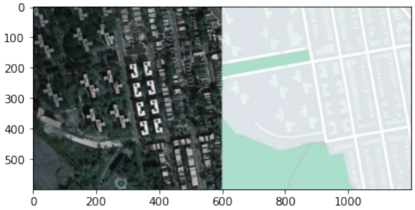
Why do we need Segmentation of Satellite imagery?

- Overview of settlement content from a plethora of data where manual labelling is not possible.
- Object tracking in Satellite video feeds.
- Automatic labelling of roads and buildings.
- Segmented maps can be used to calculate distances between two places via road.
- This method can be extended to robot vision and biomedical segmentation tasks.

Roads Dataset - Our Contribution to creation of datasets

Modified Pix2Pix dataset

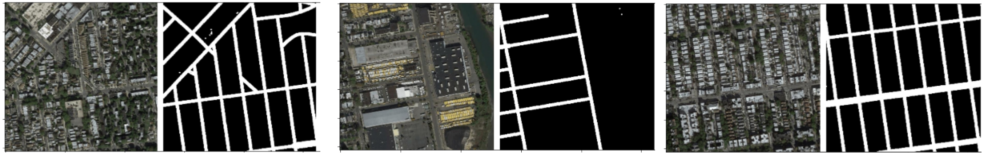
- 1096 images of dimension 600x1200 of image and mask pairs, i.e., each tile is 600x600x3 with single channel mask. a sample from the original dataset is shown below.



- Thresholded to get segmentation masks.

Datasets used (cont.)

- Image mask pair split to create the actual dataset of 1096 image mask pairs.
- The size of the dataset is 238.65 MB.
- Sample from the dataset is shown below.



Datasets used

Roads dataset

- Apple Maps Scraper a.k.a. jimutmap (2019)
  - We have created a scraper to get any amount of data from Apple Maps.
  - The tiles are related to the latitude and longitude by the following relation
  - $x\_tile = 2^{zoom} \times \frac{(longitude + 180)}{360}$
  - $lat\_radius = lat\_degree \times \frac{\pi}{180}$

- $$y\_tile = \frac{2^{zoom} \times \left(1 - \frac{\log\left(\tan lat\_radius + \frac{1}{\cos lat\_radius}\right)}{\pi}\right)}{2}$$

- Refer to Harvesine formula for more details, the derivations and calculations are out of scope for the presentation

- We have build a library and one could install the API (Application programming interface) via

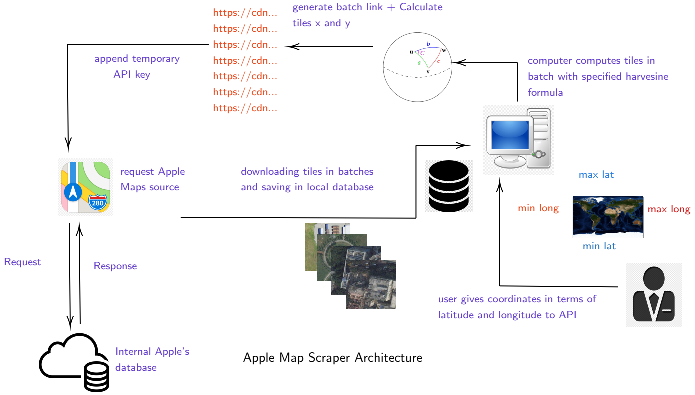
jimutmap 1.3.9 (https://pypi.org/project/jimutmap/)

```
pip install jimutmap
```

- Uses multi-threading to download huge quantity of maps within minutes

- The architecture overview of the jimutmap scraper is shown in the next page

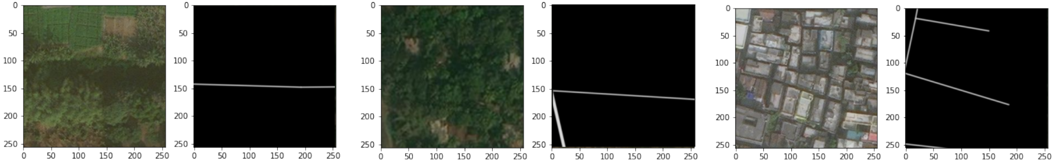
Roads Dataset - jimutmap



Architecture of Apple Maps Scraper, a.k.a. jimutmap.

Datasets used (cont.)

- We have downloaded 236606 image-mask pairs (2.45 GB) from latitude 22.35 to 23.1 and longitude 88.0 to 88.6.
- This consists of the major Kolkata area
- For a cleaned and smaller version of the dataset we have discarded the tiles which doesn't have any road mask, so now the JIMUT_MAPS.zip dataset has about 61950 tiles (1.07 GB).
- A sample of 256x256 map tile is shown below, occluded region looks challenging, wide range of color values.



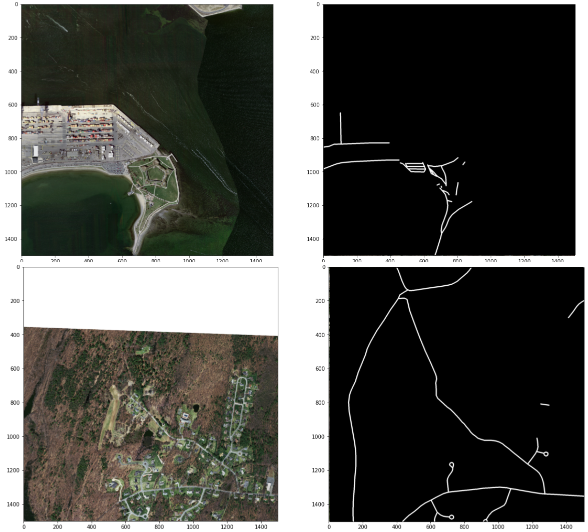
Datasets used

Roads dataset

- **Massachusetts Roads Dataset**
  - This dataset was introduced by Volodymyr Mnih, in his Ph.D. thesis.
  - It contains about 1120 files of size 1500x1500x3 with single channel mask.
  - The size of the dataset is about 4.8 GB.
  - Tiles of the dataset has a higher scale, i.e., zoomed out version.
  - Few sample of the data is shown in the next page, which shows the quality of the dataset.

Datasets used

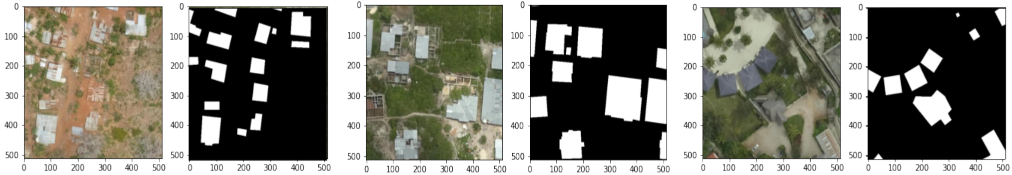
- From the data sample we can see that some tiles are corrupted.
- There are several tiles which are corrupted, for instance the regions are occluded in the map while there are segmentation masks and vice versa.
- This dataset is very challenging, and augmentation of data won't help much.



Building dataset

Zanzibar OpenAI Building Footprint Mapping - Kaggle

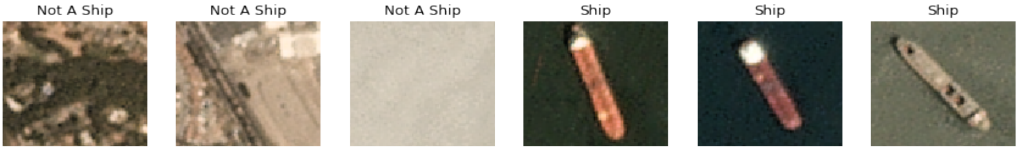
- This dataset was taken from Kaggle.
- It consists of 2691 building mask pair. Size of dataset is 471.86MB.
- Each tile consists of 500x500x3 dimension, mask are of single channel.
- Sample from the dataset is shown below



Ships dataset

Ships in Satellite Imagery dataset (Kaggle)

- This dataset was taken from Kaggle.
- It consists of 4000 data tiles containing ship and not containing ship
- Each tile consists of 80x80x3 dimension. Size of dataset is 185.45MB.
- Sample from the dataset is shown below (a classification problem),



Here are some of the metrics used for analyzing the performance of the segmentation models.

Notations:

True Postive (**TP**), True Negative (**TN**), False Postive (**FP**), False Negative (**FN**), Accuracy (**AC**), Recall or Sensitivity (**SE**), Specificity (**SP**), Precision (**PC**), F1-score (**F1**), Jaccard Similarity (**JS**), Dice Coefficient (**DC**), Ground truth (**GT**), and segmented result (**SR**).

- $$AC = \frac{TP + TN}{TP + TN + FP + FN}$$
- $$SE = \frac{TP}{TP + FN}$$
- $$SP = \frac{TN}{TN + FP}$$
- $$PC = \frac{TP}{TP + FP}$$
- $$F1 = \frac{2 \times (PC \times SE)}{PC + SE}$$
- $$JS = \frac{GT \cap SR}{GT \cup SR}$$
- $$DC = 2 \times \frac{GT \cap SR}{GT + SR}$$

- Loss Function - Binary Cross-Entropy

$$L_{y'}(y) := -\frac{1}{N}\sum_{i=1}^{N}(y_i' \log(y_i) + (1 - y_i') \log(1 - y_i)) \tag{1}$$

- Where, y_i is the predicted class per pixel, y_i' is the original pixel value of segmentation mask.
- All the pixels in the segmentation mask are averaged to get the overall loss.
- Optimiser - ADAM (Combination of RMSprop and Stochastic Gradient Descent with momentum).

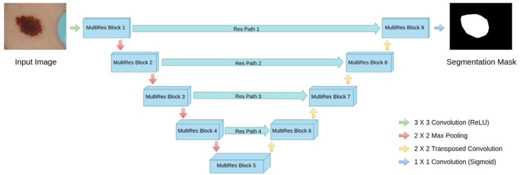


Figure 8: MultiResU-Net model.

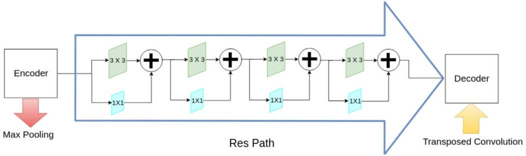
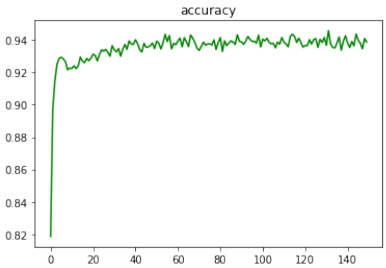
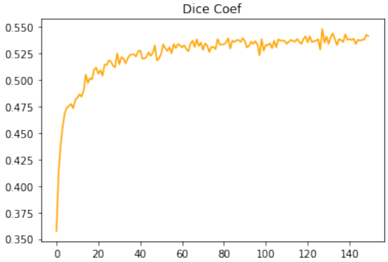


Figure 9: Residual block.

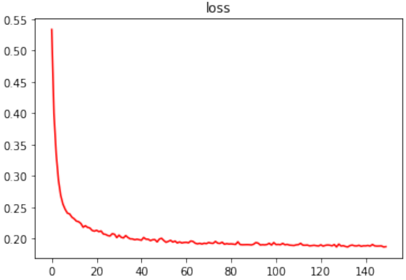
- Can be considered as the potential successor to the UNet model.
- Uses multiple residual blocks for better accumulation of gradients.
- Residual blocks helps in faster learning of features.
- Instead of passing the encoder feature maps directly to the decoder, they use a sequence of convolutional layers (non-linear operations) which reduce the gap between encoder and decoder features.

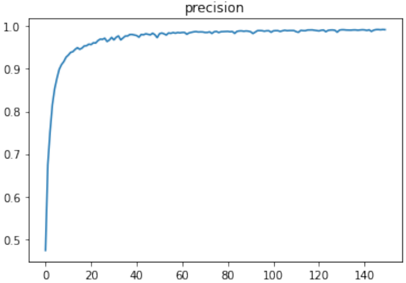
- The MultiResU-UNet model is applied to the modified pix2pix dataset for 150 epochs.
- Adam optimizer is used, with a learning rate of 10e-05.
- Got a test loss of 0.1872, test dice coefficient of 0.5409, jaccard index of 0.3735, test recall of 0.6399 and a test precision of 0.9919 and accuracy of 0.9386.
- The dataset is challenging and the mask generated is quite good.

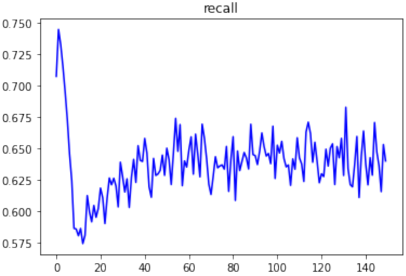




MultiResU-Net model applied to modified pix2pix dataset



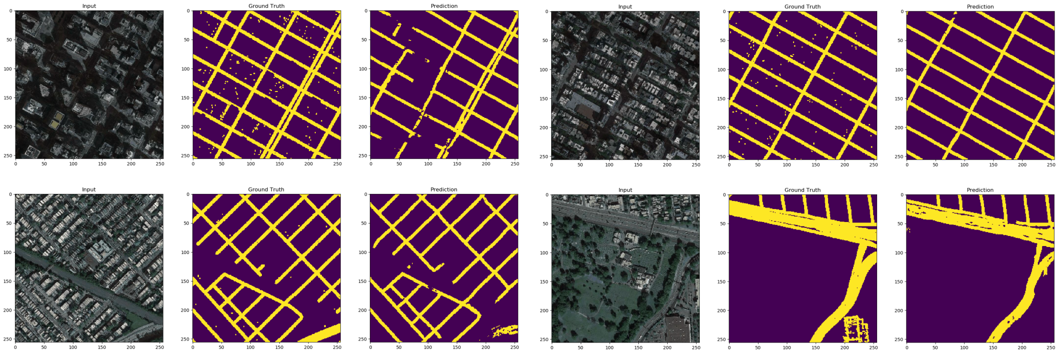




- The graph of change of accuracy, dice coefficient, loss precision and recall is shown.
- The ground truth and segmentation masks are shown in the next page.

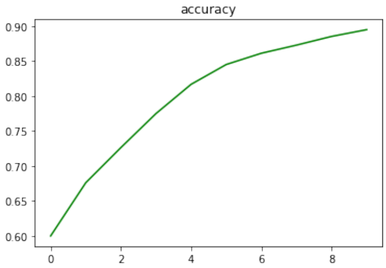
MultiResU-Net model applied to modified pix2pix dataset

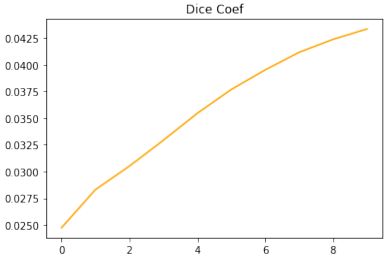
Sample of the ground truth along with the mask generated is shown below.

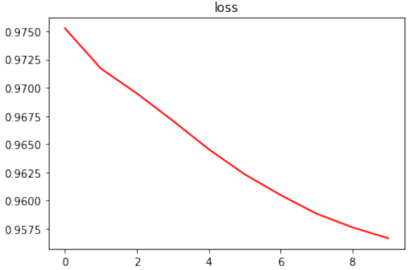


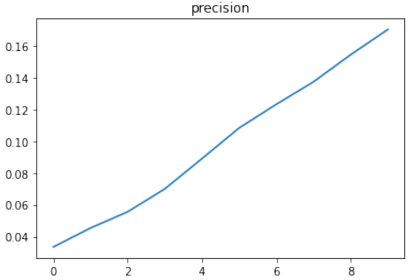
MultiResU-Net model applied to jimutmap dataset

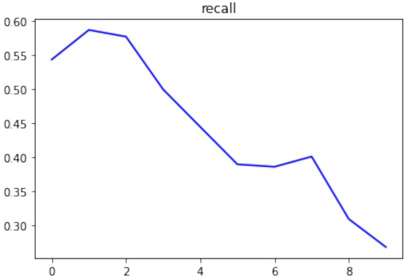
- The MultiResU-UNet model is applied to the jimutmap dataset for 10 epochs.
- Nadam optimizer is used, with a learning rate of 10e-05.
- Got a test loss of 0.9745, test dice coefficient of 0.0255, jaccard index of 0.0130, test recall of 0.2721 and a test precision of 0.0948 and accuracy of 0.9266.
- The dataset is challenging and the mask generated is not so good.





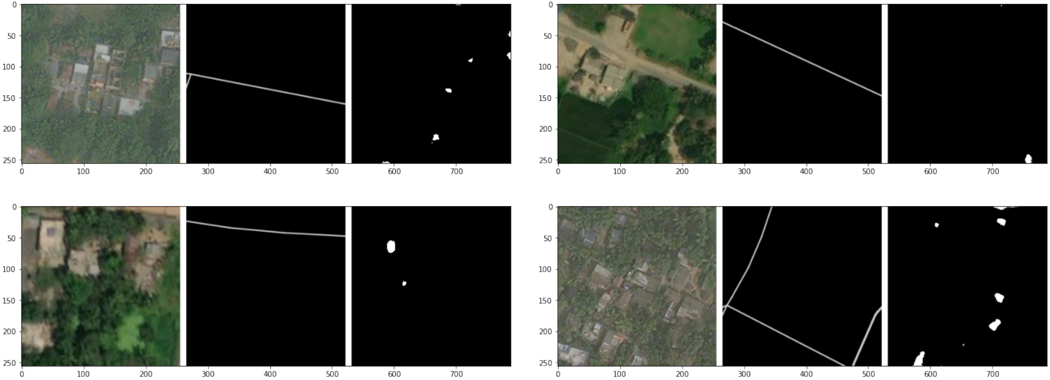






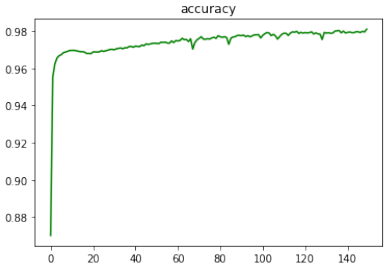
- The graph of change of accuracy, dice coefficient, loss precision and recall is shown.
- The ground truth and segmentation masks are shown in the next page.

Sample of the ground truth along with the mask generated is shown below.

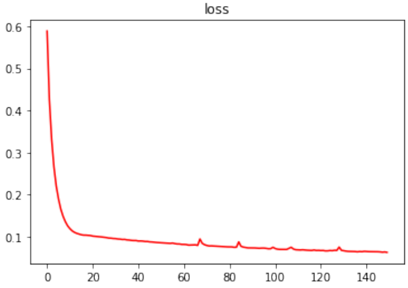


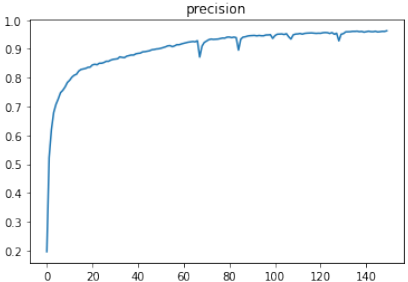
MultiResU-Net model applied to Massachusetts Roads dataset

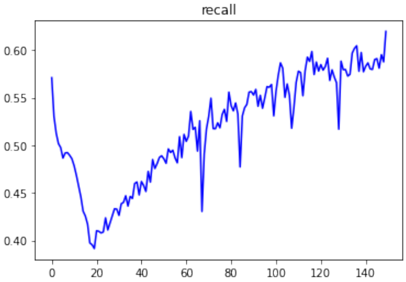
- The MultiResU-UNet model is applied to the Massachusetts Roads dataset for 150 epochs.
- Adam optimizer is used, with a learning rate of 10e-05.
- Got a test loss of 0.0620, test dice coefficient of 0.5505, jaccard index of 0.3841, test recall of 0.6195 and a test precision of 0.9628 and accuracy of 0.9810.
- The dataset is challenging and the mask generated is quite good.



MultiResU-Net model applied to Massachusetts Roads dataset



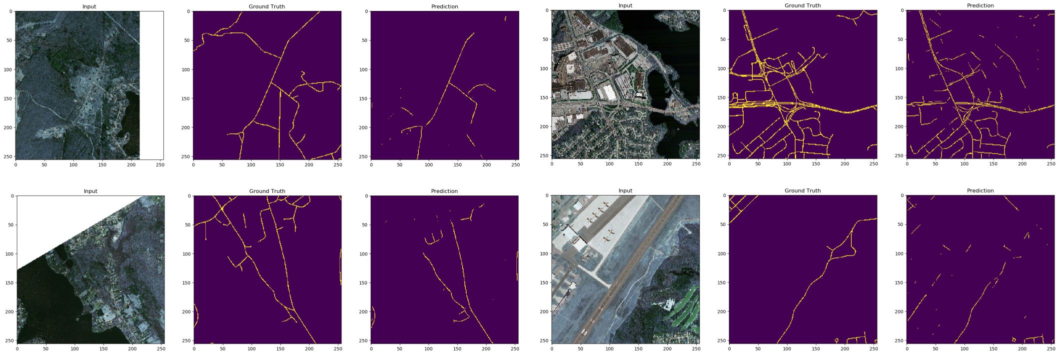




- The graph of change of accuracy, dice coefficient, loss, precision and recall is shown.
- The ground truth and segmentation masks are shown in the next page.

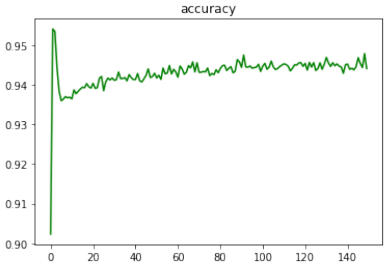
MultiResU-Net model applied to Massachusetts Roads dataset

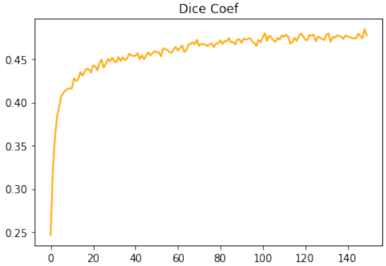
Sample of the ground truth along with the mask generated is shown below.



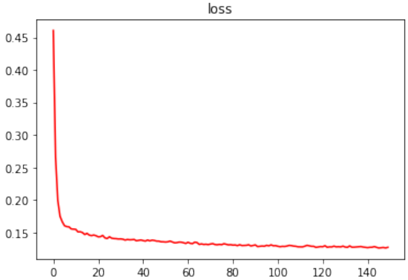
MultiResU-Net model applied to Zanzibar OpenAI Building dataset

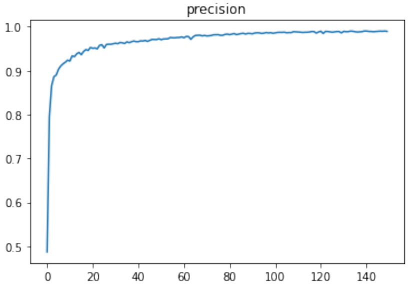
- The MultiResU-UNet model is applied to the Massachusetts Road for 150 epochs.
- Adam optimizer is used, with a learning rate of 10e-05.
- Got a test loss of 0.1274, test dice coefficient of 0.4773, jaccard index of 0.3227, test recall of 0.4110 and a test precision of 0.9894 and accuracy of 0.9440.
- The dataset is challenging and the mask generated is quite good.

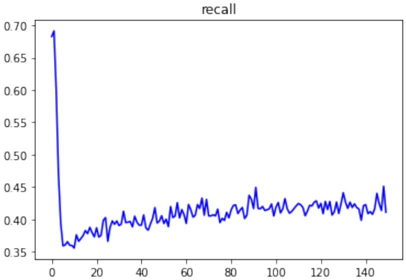




MultiResU-Net model applied to Zanzibar OpenAI Building dataset

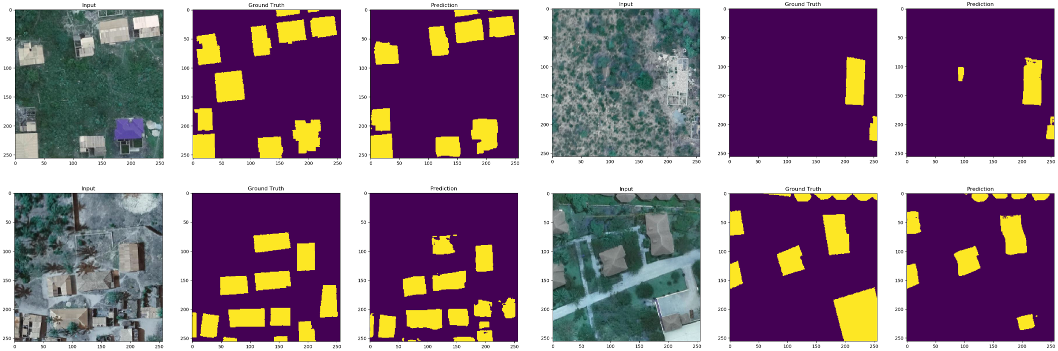






- The graph of change of accuracy, dice coefficient, loss, precision and recall is shown.
- The ground truth and segmentation masks are shown in the next page.

Sample of the ground truth along with the mask generated is shown below.



Results on Application of MultiResU-Net on the datasets

- Result of 5 fold cross validation applied to each of the datasets to record the variations. Those represented by **bold** performs better than it's corresponding models.

- The results are presented as $\mu \pm \sigma$, where μ is the mean and σ is the standard deviation of the five folds, $\sigma = \sqrt{\frac{\sum(x_i - \mu)^2}{N}}$

	Dice Coeff	Jaccard	Recall	Precision	Accuracy
modified pix2pix	53.09 ± 2.34	38.35 ± 2.78	63.99 ± 1.89	$\mathbf{98.19 \pm 3.11}$	92.86 ± 2.34
jimutmap	20.33 ± 2.71	25.15 ± 1.37	36.13 ± 2.31	43.21 ± 3.27	53.72 ± 3.21
Massachusetts Roads	$\mathbf{54.03 \pm 2.73}$	36.57 ± 0.53	$\mathbf{62.57 \pm 3.23}$	$\mathbf{96.28 \pm 2.32}$	97.81 ± 2.73
Zanzibar OpenAI Building	47.73 ± 3.21	32.27 ± 3.42	42.10 ± 3.31	$\mathbf{97.91 \pm 3.32}$	93.31 ± 2.31

Proposed Modified U Net model

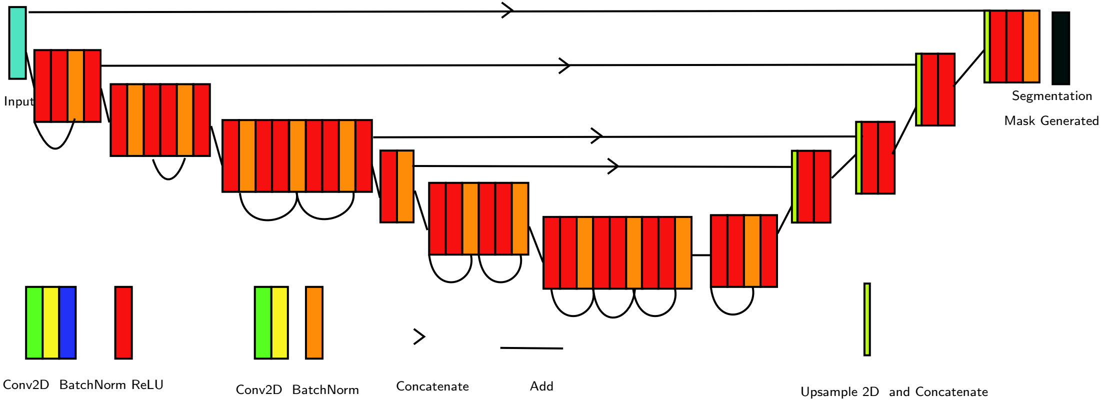
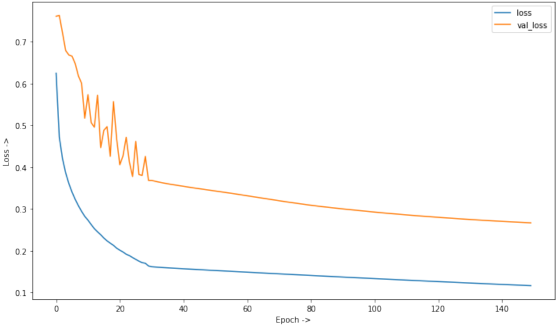


Figure 18: A very Deep UNet Model is Proposed.

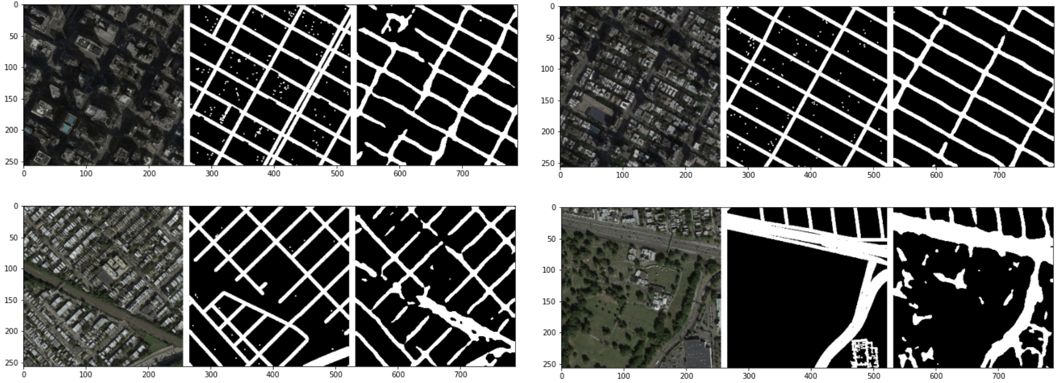
Proposed model when applied to modified pix2pix dataset

- The UNet model is applied to the pix2pix dataset for 150 epochs.
- Nadam optimizer is used, with a learning rate of 10e-05.
- Got a test loss of 0.2616, test dice coefficient of 0.7392, test recall of 0.6537 and a test precision of 0.8173.
- The dataset is challenging and the mask may differ here and there.
- The graph for loss is shown on the right.



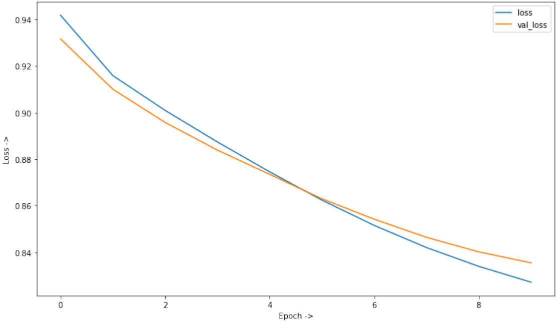
Proposed model when applied to modified pix2pix dataset

Sample of the ground truth along with the mask generated is shown in below.



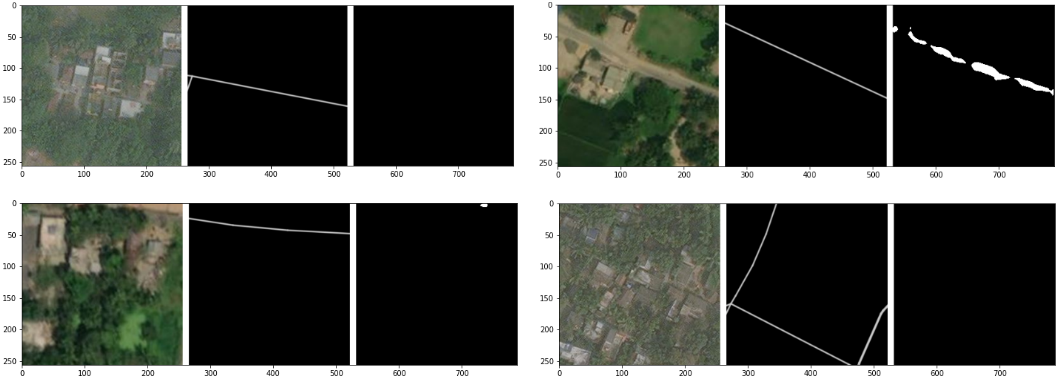
Proposed model when applied to jimutmap dataset

- The UNet model is applied to the jimutmap dataset for 10 epochs.
- Nadam optimizer is used, with a learning rate of 10e-05.
- Got a test loss of 0.8332, test dice coefficient of 0.1667, test recall of 0.2769 and a test precision of 0.2379.
- The dataset is very challenging and the mask generated is not up to the mark.
- Sample of the mask generated along with the ground truth is shown in the next page.

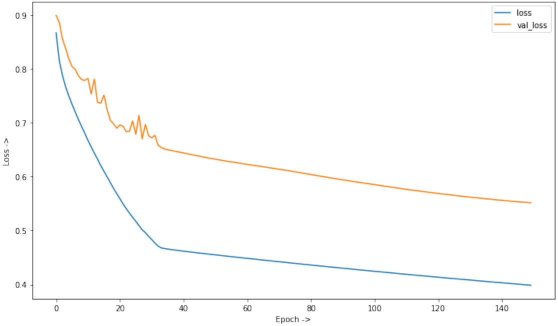


Proposed model when applied to jimutmap dataset

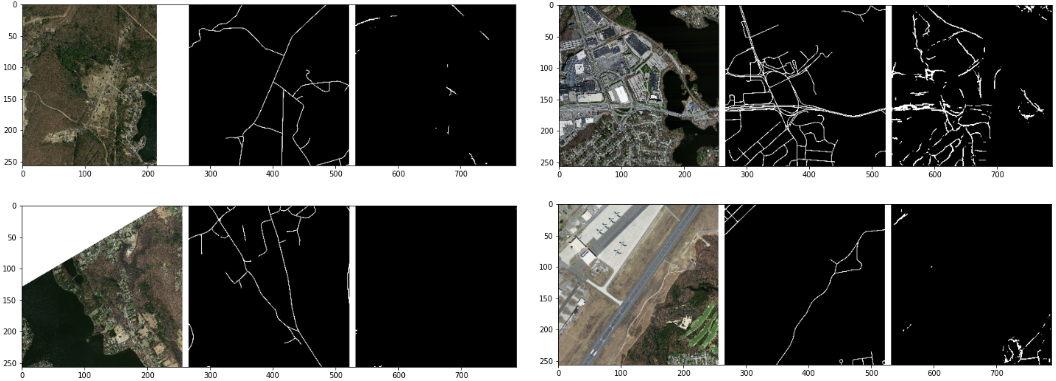
Sample of the ground truth along with the mask generated is shown in below.



- The UNet model is applied to the Massachusetts Road for 150 epochs.
- Nadam optimizer is used, with a learning rate of 10e-05.
- Got a test loss of 0.5539, test dice coefficient of 0.4456, test recall of 0.4525 and a test precision of 0.5323.
- The dataset is challenging and the mask generated is not up to the mark since the ground truth is thinner.

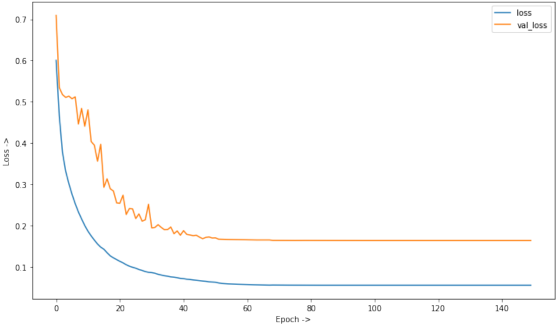


Sample of the ground truth along with the mask generated is shown in below.

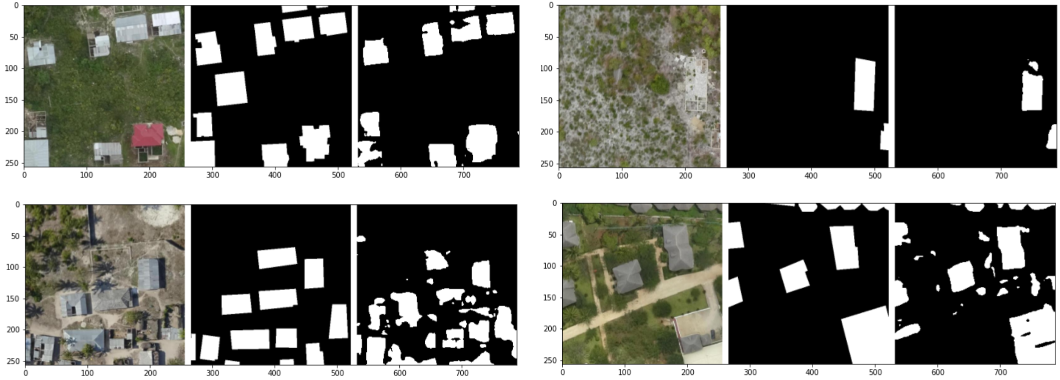


Proposed model when applied to Zanzibar OpenAI Building dataset

- The UNet model is applied to the Massachusetts Road for 150 epochs.
- Nadam optimizer is used, with a learning rate of 10e-05.
- Got a test loss of 0.1873, test dice coefficient of 0.8121, test recall of 0.7806 and a test precision of 0.8624.
- The dataset is challenging and the mask generated is quite good.



Sample of the ground truth along with the mask generated is shown in below.



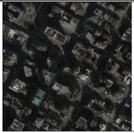
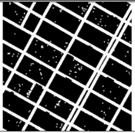
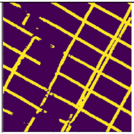
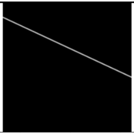
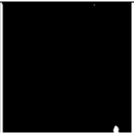
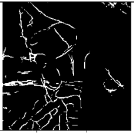
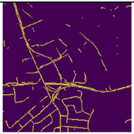
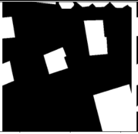
Results on Application of proposed model on the datasets

- Result of 5 fold cross validation applied to each of the datasets to record the variations. Those represented by **bold** performs better than it's corresponding models.

- The results are presented as $\mu \pm \sigma$, where μ is the mean and σ is the standard deviation of the five folds, $\sigma = \sqrt{\frac{\sum(x_i - \mu)^2}{N}}$

	Dice Coefficient	Jaccard	Recall	Precision	Accuracy
modified pix2pix	**72.93** $\pm$ **2.31**	**63.17** $\pm$ **2.56**	**64.37** $\pm$ **1.12**	81.73 $\pm$ 1.78	**98.18** $\pm$ 3.21
jimutmap	**31.27** $\pm$ **3.31**	**37.17** $\pm$ **1.93**	**46.53** $\pm$ **3.11**	**63.73** $\pm$ **3.11**	**97.83** $\pm$ **3.18**
Massachusetts Roads	43.51 $\pm$ 2.73	32.17 $\pm$ 3.15	44.25 $\pm$ 2.31	53.23 $\pm$ 2.27	92.17 $\pm$ 2.19
Zanzibar OpenAI Building	**81.11** $\pm$ **3.21**	**75.11** $\pm$ **1.31**	**79.30** $\pm$ **2.70**	85.24 $\pm$ 3.21	**97.52** $\pm$ **2.98**

Comparison of the Results.

	Image	GT	Our	MRUNet
pix2pix dataset				
jimutmap dataset				
Massachusetts Road dataset				
Zanzibar OpenAI Building dataset				

Model for detecting Ships from Above

- A model is proposed which is able to classify if a given image is a ship or not.

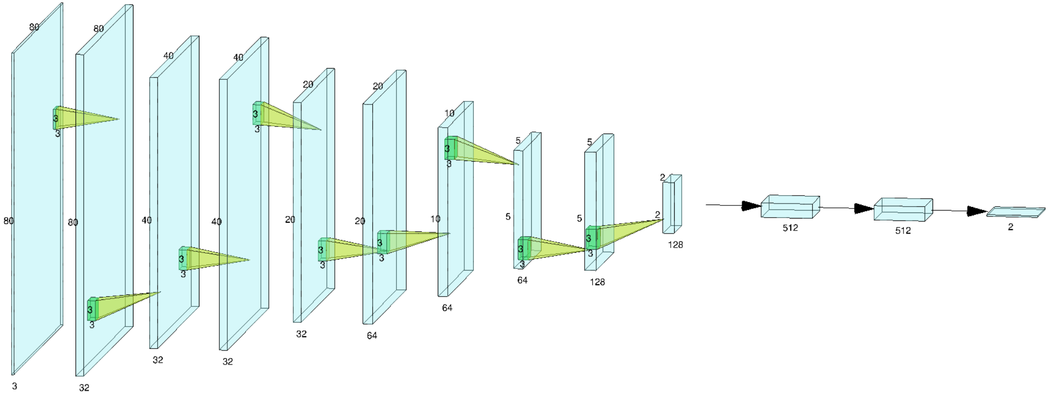
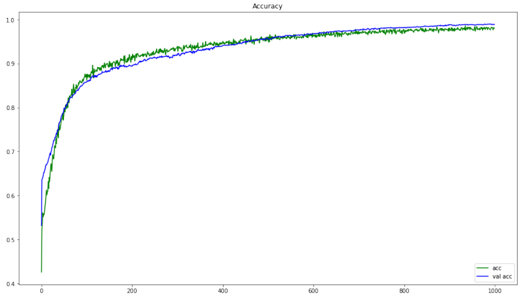
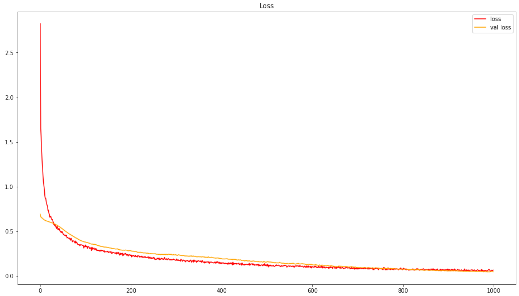


Figure 27: The proposed model.

- Uses Nadam optimizer with a learning rate of 0.001, and a loss function of binary crossentropy.

- The model is trained for 1000 epochs and we get a loss of 0.0640, an accuracy of 0.9802, a validation loss of 0.0457 and a validation accuracy of 0.9889.





Model for detecting Ships from Above

- All the six ships are detected along with some flase positives.
- We can see that there are only 5 non-ships which are classified as ships.
- In case of video feed, we could easily track the ships faster, but this took 3 mins to search sequentially.

Model for detecting Ships from Above

- All the 13 ships are detected.
- We can see that there are only 2 non-ships which are classified as ships.

Model for detecting Ships from Above

- We can see all the 13 ships are detected.
- We can see that there are only 6 non-ships which are classified as ships.
- The model is able to detect all the ships which is good, and not missing any trespassing ships.

Model for detecting Ships from Above

- All the 8 ships are detected.
- This model performs well in the open water, and can able to detect all the ships that are present in the wild sea.
- The model perhaps gets confuse when there are ship like objects in the sea. We have tried different types of model, even nested and composite model, but this model is the best.
- We may try to pass the images to GPU for parallel processing and faster detection of ships.

Bonus Section - Mask RCNN

- Gives object **bounding boxes**, **classes** and **masks** for an image.
- Convolutional backbone - **Feature Pyramid Network (FPN)** for preserving features at different scales - uses a Faster R-CNN backbone.
- **Region Proposal Networks (RPN)** contains bounding boxes, known as anchors, which helps to detect objects faster.
- Uses **ROIAlign** to align the features at different scales using **Bilinear Interpolation**, which helps to **remove location misalignment** caused due to **ROI pooling**.

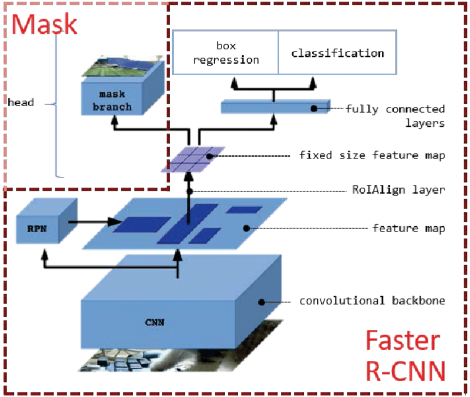


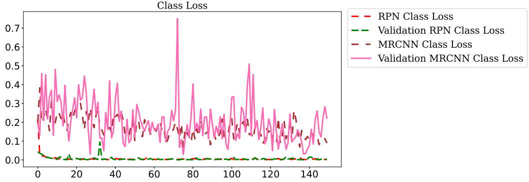
Figure 29: Mask RCNN (Picture Courtesy: Research Gate)

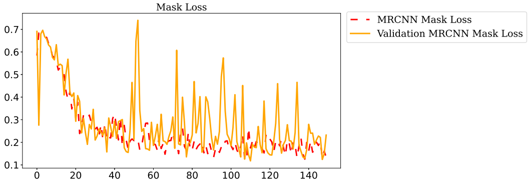
Mask RCNN - about the data

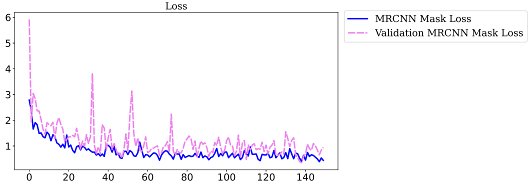
- Taken from Incubit Challenge Dataset.
- There are 3 classes - **Buildings**, **Houses** and **Sheds/Garages**.
- Task is to find instances of these objects.
- 50 images from training dataset, 22 from validation dataset 8 from test set.
- The images are of different shapes and sizes, but were resized to 512x512 before passing it to RPN with a batch size of 2.

Mask RCNN - Loss

$$L_{total} = L_{box} + L_{class} + L_{mask}$$







Mask RCNN - Predictions

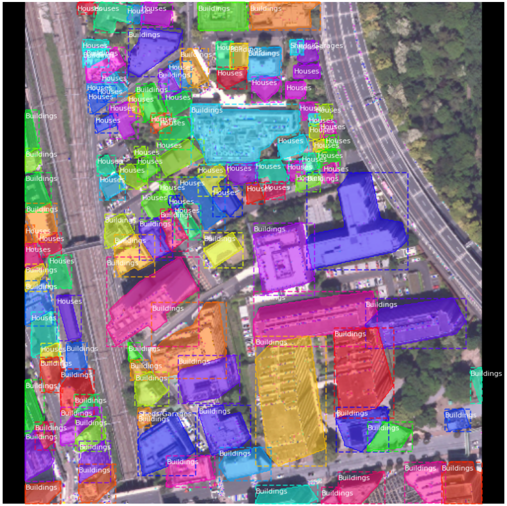


Figure 31: GT instance masks.

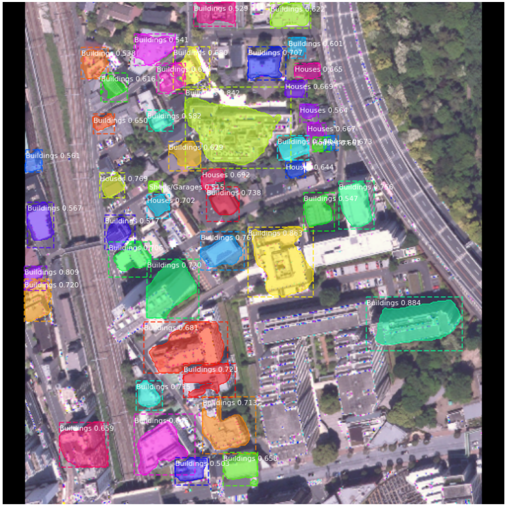
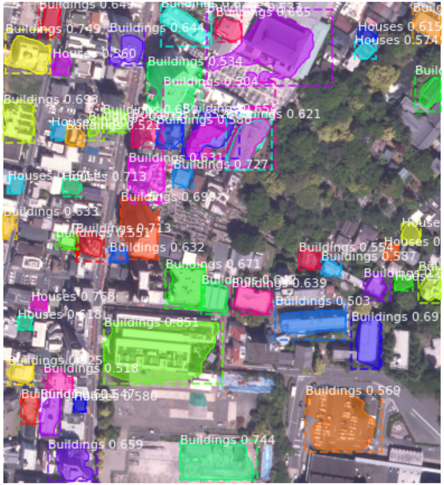


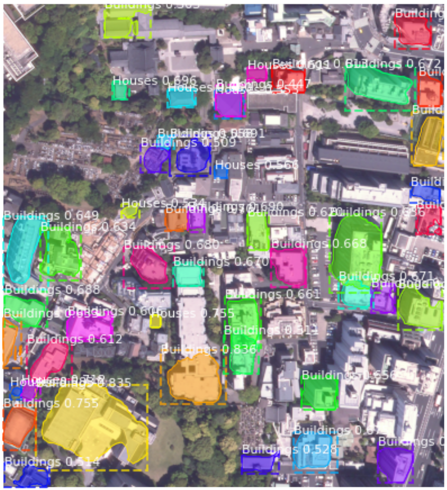
Figure 32: Prediction using Mask RCNN.

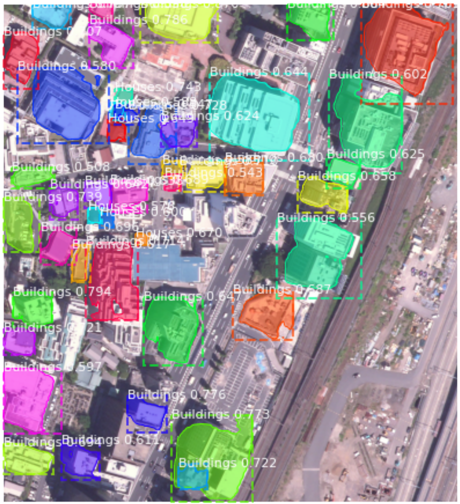
Mask RCNN - Predictions (Test Set)

Mask RCNN - Predictions (Test Set)



Mask RCNN - Predictions (Test Set)





Conclusions

- We have seen that different style of U-Net architectures perform differently for datasets, i.e., they are dataset dependent.
- Our main goal was to design a robust type of model which performs well on different datasets when trained.
- Our proposed U Net variant model gives better dice coefficient and recall on jimutmap, pix2pix and zanzibar openAI dataset, which is a great feat.
- Our segmentation model may be used in medical datasets also, since this performs well on a range of satellite segmentation datasets.
- The only modification that can be done to ship segmentation dataset is by passing the extracted images to GPU for faster processing.
- It is only giving false positive to those stationary samples which looks like ship, and in the case of video feeds, we might get over this problem.

Wikipedia contributors. Image segmentation - Wikipedia, the free encyclopedia. https://en.wikipedia.org/wiki/Image_segmentation, 2020. Last accessed on 27.10.2020.

Wikipedia contributors. Haversine formula - Wikipedia, the free encyclopedia. https://en.wikipedia.org/wiki/Haversine_formula, 2020. last accessed on 27.10.2020.

Jimut Bahan Pal. jimutmap. https://github.com/Jimut123/jimutmap, 2019.

Volodymyr Mnih. Machine Learning for Aerial Image Labeling. PhD thesis, University of Toronto, 2013.

Nabil Ibtehaz and M Sohel Rahman. Multiresunet: Rethinking the u-net architecture for multimodal biomedical image segmentation. Neural Networks, 121:74–87, 2020

References (cont.)

- Olaf Ronneberger, Philipp Fischer, and Thomas Brox. U-net: Convolutional networks for biomedical image segmentation. In Nassir Navab, Joachim Hornegger, William M. Wells, and Alejandro F. Frangi, editors, Medical Image Computing and Computer-Assisted Intervention – MICCAI 2015, pages 234–241, Cham, 2015. Springer International Publishing.
- Vladimir Iglovikov and Alexey Shvets. Ternausnet: U-net with vgg11 encoder pre-trained on imagenet for image segmentation, 2018.
- Guillaume Chhor and Cristian Bartolome Aramburu. Satellite image segmentation for building detection using u-net. 2017.
- Andrew Khalel and Motaz El-Saban. Automatic pixelwise object labeling for aerial imagery using stacked u-nets. CoRR, abs/1803.04953, 2018.

- B. Bischke, P. Helber, J. Folz, D. Borth, and A. Dengel. Multi-task learning for segmentation of building footprints with deep neural networks. In 2019 IEEE International Conference on Image Processing (ICIP), pages 1480–1484, 2019.
- Sébastien Ohleyer and ENS Paris-Saclay. Building segmentation on satellite images. 2018.
- user rhammell. Ships in satellite imagery, 2018.
- Sayantan Das. Zanzibar openai building footprint mapping, 2019.
- Kaiming He, Georgia Gkioxari, Piotr Dollár, and Ross Girshick. Mask r-cnn. In 2017 IEEE International Conference on Computer Vision (ICCV), pages 2980–2988, 2017.

Acknowledgements

It is ritual that scholars express their gratitude to their supervisors. This acknowledgement is very special to me to express my deepest sense of gratitude and pay respect to my supervisor, **Tamal Maharaj**, Department of Computer Science, for his constant encouragement, guidance, supervision, and support throughout the completion of my project. His close scrutiny, constructive criticism, and intellectual insight have immensely helped me in every stage of my work. I would like to thank him for patiently answering my often-naive questions related to Computer Vision.

I would like to thank all the professors of Department of Computer Science for creating an intellectually stimulating environment that enabled me to think more carefully and methodically about my research than I had ever done before. I would like to thank **Swathy Prabhu Maharaj** for arranging free compute resources, **Dripta Maharaj** for the stimulating discussions that he had shared with me during the Advanced Machine Learning Course.

I'm grateful to my father, Dr. **Jadab Kumar Pal**, Deputy Chief Executive, Indian Statistical Institute, Kolkata for constantly motivating and supporting me to develop this documentation along with the proofreading. I will also mention about my brother **Jisnoo Dev Pal** and my mother **Sumita Pal**, for supporting me.

Any Questions?

Thank You

jimutbahanpal@{yahoo,gmail,outlook}.com