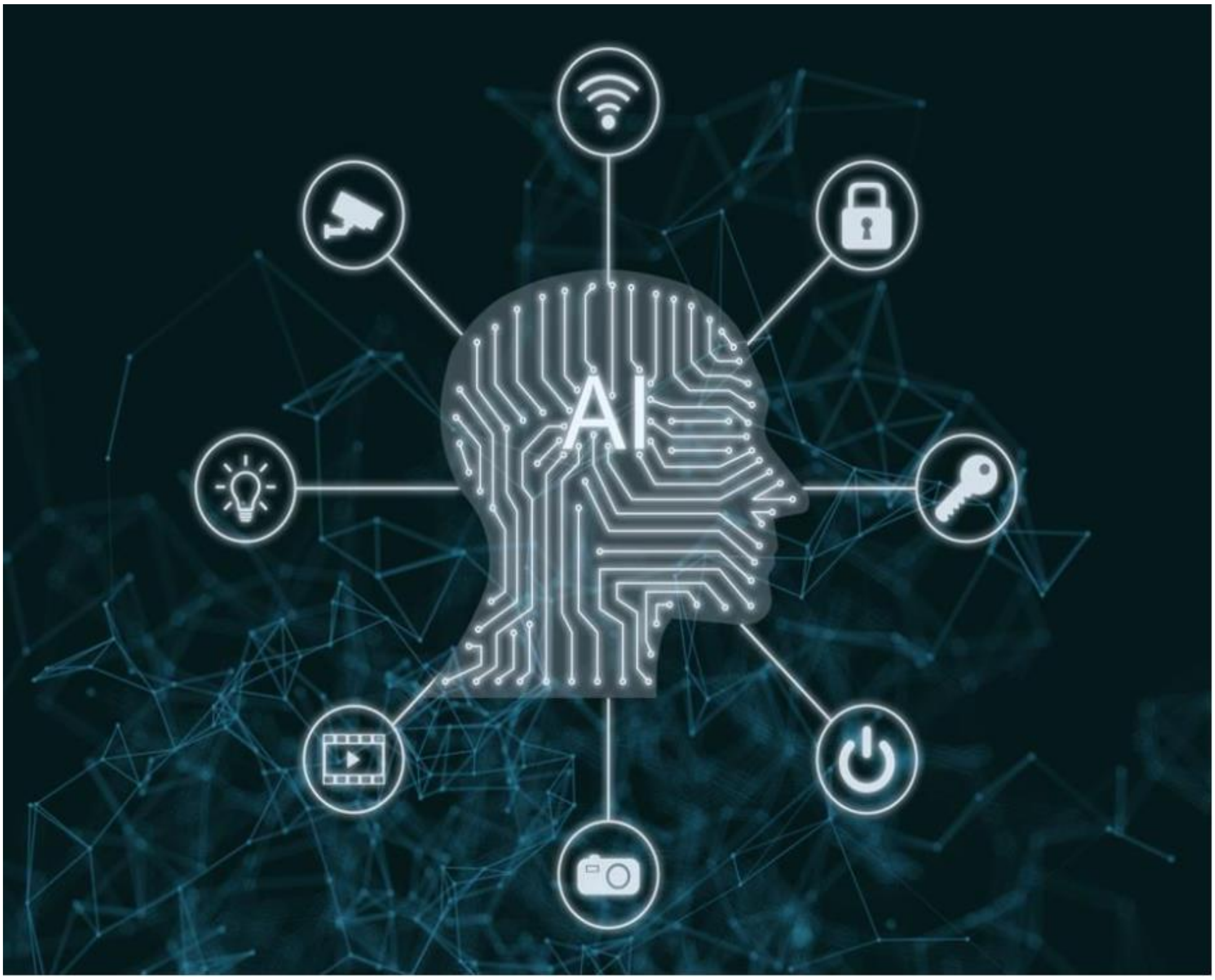

Deep Learning to Foster Building Footprint Extraction

Kushanav Bhuyan 4/16/21 Programming Solutions

Table of Contents

List of Figures

List of Tables

1. Project Introduction

The detection of buildings is crucial in recognizing them as an essential component of elements-at-risk (EaR) for disaster risk assessment. Implementing approaches for a hazard impact assessment on EaR is important but cannot be executed without proper data sets. Databases with updated information about elements exposed to hazards are fundamental for response activities and support crisis preparedness (Eshrati, Mahmoudzadeh, & Taghvaei, 2015). Hence, the project would aim to detect buildings through deep learning algorithms from Open Street Maps (OSM) data. OSM data has huge implications for disaster response and relief missions when certain disasters occur by establishing humanitarian projects aimed at mapping the vulnerable places of the world. The need for accurate geographical information of human settlements' spatial distribution has become quintessential with the ever-growing demand for disaster response (Li, Herfort, Huang, Zia, & Zipf, 2020). Plenty of research (Barrington-Leigh & Millard-Ball, 2017; Grippa et al., 2018; Zhao et al., 2019) have explored crowdsourced and open-sourced data like OSM in mapping human settlements that aid in generating accurate data sets in disaster management, habitat and ecological system conservation, and public health monitoring (Herfort, Li, Fendrich, Lautenbach, & Zipf, 2019). Hence, the project will address the same for a commonly disaster affected city of Palakkad, Kerala, which witnesses landslide occurrences due to the monsoonal rainfalls during the monsoon seasons in India.

2. Project Methodology

The project methodology has been divided into three sections that discuss (1) data acquisition and preparation, (2) the deep learning model, and (3) the assessment of test data set accuracy in examining the trained model's applicability. Figure 1 describes the flow chart of the entire methodology.

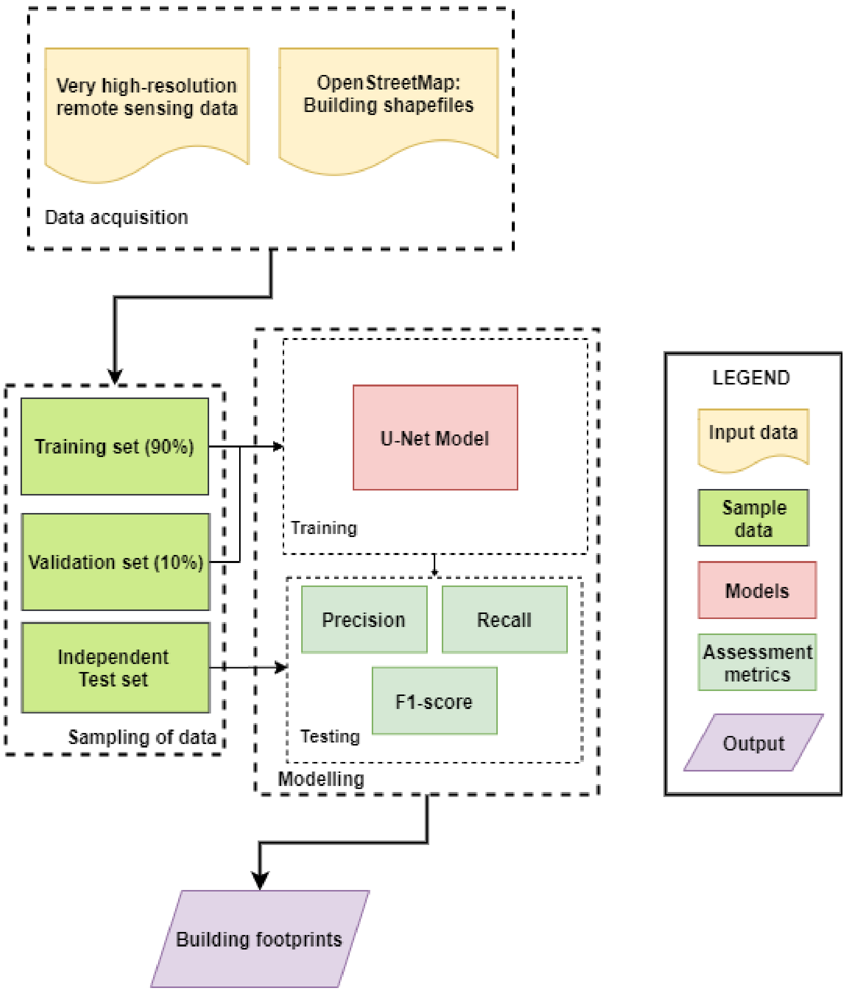

Figure 1: Flowchart of the methodology

2.1 Data Acquisition and Preparation

2.1.1 *Satellite Imagery*

Satellite imagery was download from the *SAS Planet* software of 0.6-metre resolution Google Earth images (from Quick bird) containing only three-channels of Red, Green and Blue (figure 2).

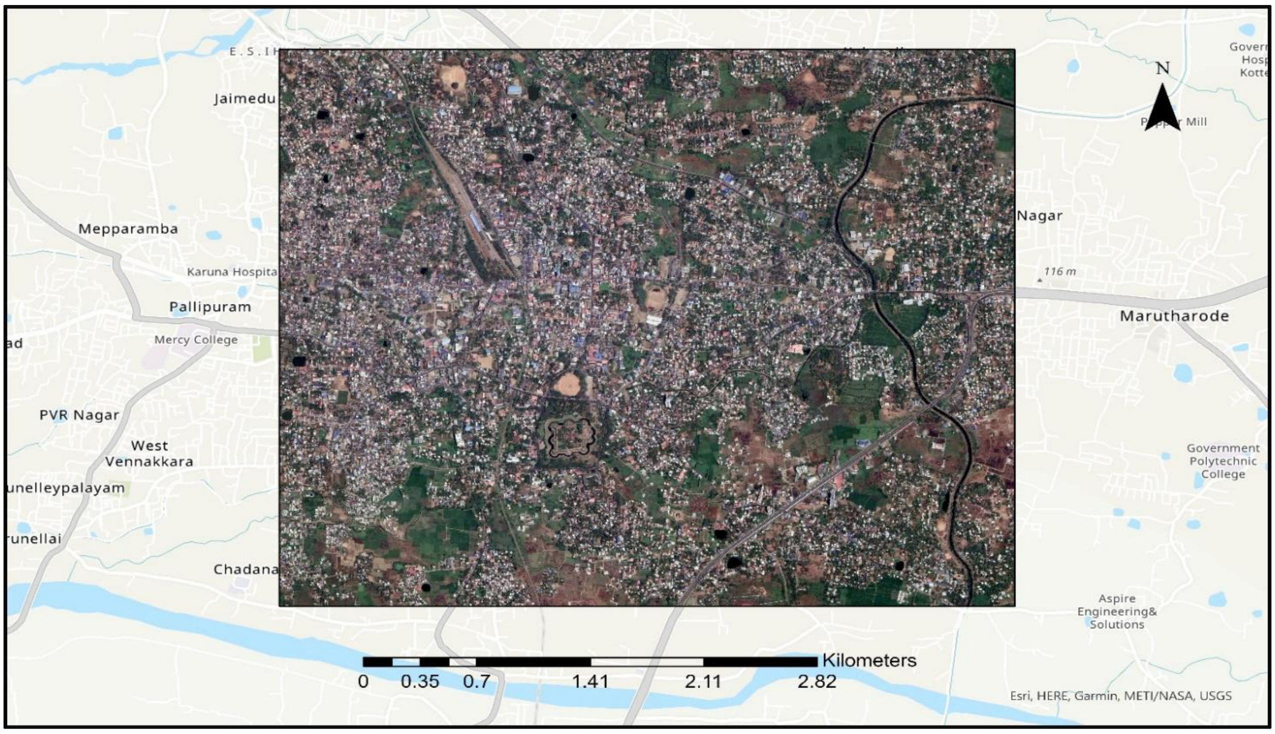

Figure 2: Satellite image of the study area Palakkad, Kerala, India

2.1.2 *Open Street Map*

The data set of buildings for labels as training is downloaded from OSM for the city of Palakkad in Kerala, India. The data set contains approximately 3000 building polygons which were used for training the model (figure 3).

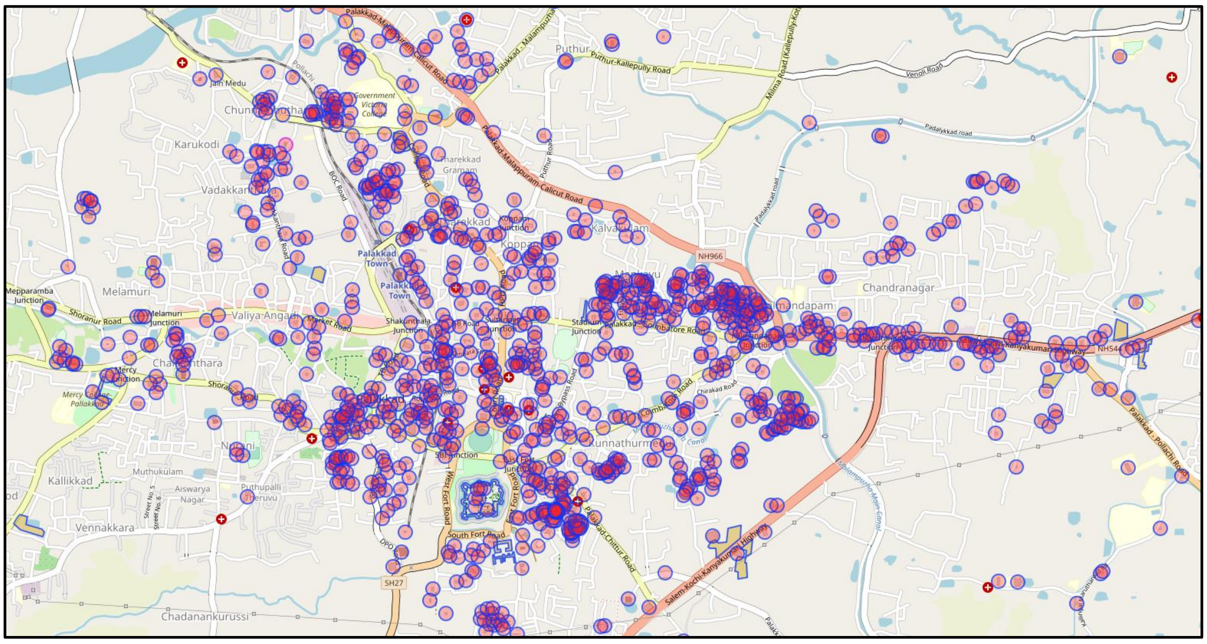

Figure 3: Building footprints available in Open Street Map

2.1.3 Manual Digitization

The existing 3000 building polygons were not enough for training the model properly; hence, an additional 5000 buildings were digitized manually to increase the model's classification accuracy.

2.1.4 Preparation of data

A series of steps are taken into consideration to prepare the data set before training the model. The satellite images acquired are georeferenced and atmospherically corrected.

- After manually digitizing the buildings and obtaining the resultant footprints, the polygons were converted into raster images with the satellite image's environments (Figure 4). This step assured the spatial extent, coordinate system and cell size of the rasterized building footprints to adhere to the satellite image cell size and spatial extent.
- Followed by this, the rasterized building footprints are then reclassified as "0" and "255" where "0" indicates a non-building class and "255" indicates a building class.
- This led to the generation of the labelled data that referred to the building and non-building classes.

Data	Information	Remarks
Satellite Image – Quick Bird	0.6-metre resolution	The resolution was quite optimal for building detection.
OSM shapefile	3000 building polygons + 5000 manually digitized	Data is quite lacking hence needed to digitize more buildings manually.

Table 1: Data set used for training the model.

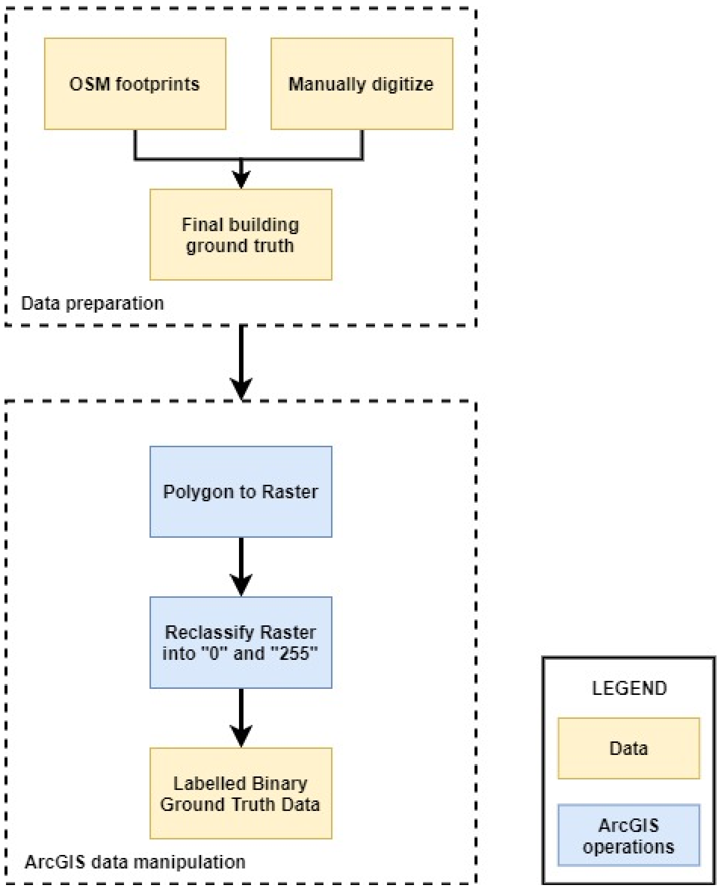

Figure 4: Ground truth data preparation.

2.2 Deep Learning Model

2.2.1 Data splitting strategy

The data set is split between training and validation sets. The splitting was done over 11 image tiles spread strategically over the study area to encompass the complex environments, further patched into 512x512 sized image patches. So, in total, 5632 image patches were used in training and validation. The data set split was 9:1, meaning only 10 per cent of the image patches were used for tuning the model. In the figure below, the tiling of the image into 11 tiles can be observed.

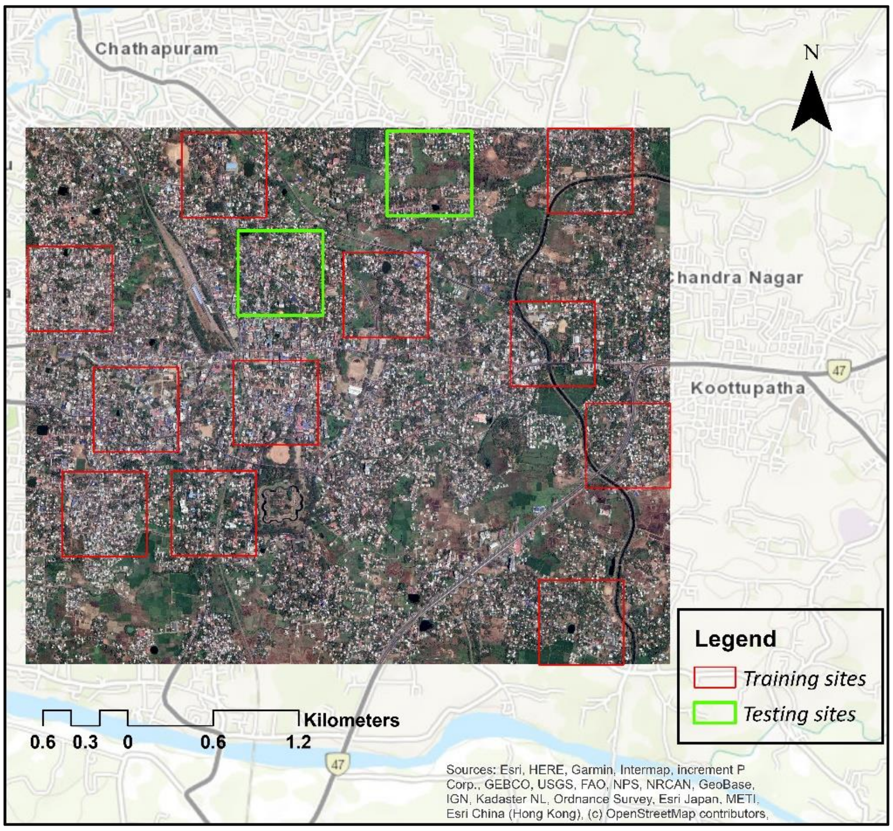
Chathapuram
N
Chandra Nagar
Koottupatha
47
Legend
🟥 Training sites
🟩 Testing sites
Kilometers
0.6 0.3 0 0.6 1.2
Sources: Esri, HERE, Garmin, Intermap, increment P Corp., GEBCO, USGS, FAO, NPS, NRCAN, GeoBase, IGN, Kadaster NL, Ordnance Survey, Esri Japan, METI, Esri China (Hong Kong), (c) OpenStreetMap contributors,

Figure 5: 11 tiles spread strategically over the study area.

2.2.2 Experiment Setup

Under the framework of Keras[1], a U-Net model is employed to classify the satellite image into "building" and "non-building" classes in Google Colab[2]. According to Ronneberger, Fischer, and Brox (2015), the U-Net architecture (Figure 6) comprises of layers and special operations summarised as follows:

1. Conv2D – Simple convolutional layer with a 3x3 kernel.
2. MaxPooling2D – Simple max pooling with a 2x2 kernel.
3. Cropping2D – Cropping layer to crop features and concatenate.
4. Concatenate layer that concatenates multiple feature maps from different stages of training.
5. UpSampling2D – To increase the size of the feature map.
6. SoftMax layer that generates a final segmentation map.

[1] Deep learning API built on top of TensorFlow.
[2] Free online cloud based Jupyter environment.

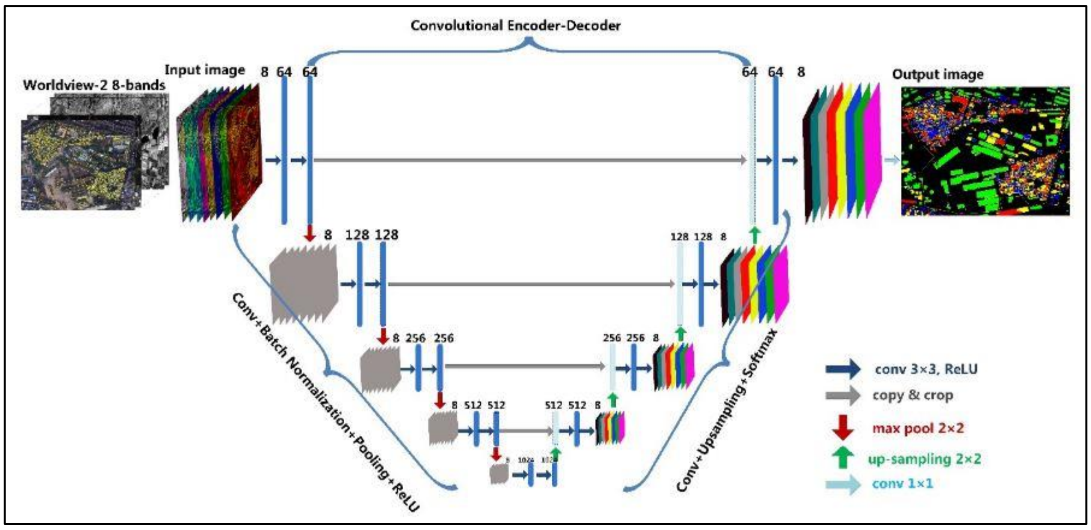

Figure 6: Schematic diagram of the U-Net model based on Ronneberger et al. (2015)

A binary image classification task aiming to distinguish building footprints with non-building pixels in input raster data is employed. For model training using the U-Net model, the whole process was run on an NVIDIA P100 GPU (16 GiB VRAM) and 25 GiB of RAM in Google Colab.

2.2.3 *Setting up hyper-parameters*

In U-Net, a series of hyper-parameters (HP) were chosen, which are summarised below:

- Number of Epochs
- Loss Function
- Learning Rate
- Optimizer

The HPs play an important role in training the model. With the right combination of HPs, the highest possible accuracy can be obtained; however, finding the optimal combination is quite difficult as it is iterative by nature. Experimenting with possible combinations of HPs, various classification outputs can be obtained, as discussed in *section 3*. To handle the computational load, the batch size was set to 8 that led to the optimal speed in training the model.

2.3 Accuracy Assessment

2.3.1 *Metric Evaluation*

Using metrics of True Positives (TP), False Positives (FP) and False Negatives (FN), standard accuracy assessment like Precision, Recall, and F1 score can be calculated for the results. Precision (1) indicates the proportion of buildings that are correctly identified as buildings. Recall (2) indicates the proportions of buildings in the labelled data that were correctly detected. F1-score (3) is used to balance the Precision and Recall metrics.

$$Precision = \frac{TP}{TP+FP} \quad (1) \qquad\qquad Recall = \frac{TP}{TP+FN} \quad (2)$$

$$F1 - score = 2 \times \frac{Precision \times Recall}{Precision + Recall} \quad (3)$$

2.3.2 *Test set evaluation*

The test set is an “un-seen” data that the model was not trained on, making it a useful candidate data to test the model’s accuracy. The model was used to predict on the test set, and then the metrics are evaluated accordingly, indicating how the model predicts un-seen data.

3. Results and Discussions

3.1 Experiments with the hyper-parameters

The current method of U-Net for detecting and classifying satellite images of Palakkad city between "buildings" and "non-buildings" provides a good classification accuracy throughout the whole study area. Initial experiments with the HPs witnessed decent results of F1-score of 65% with an overall accuracy of 89%. However, keeping in mind that a better combination of HPs could provide better results, multiple HP combinations were tried out.

Hyper-parameters	**Record**	**Remarks**
Number of Epochs	100, 120,150	The accuracy seems to remain relatively close to 88% with higher epochs, revolving between 87% and 88%.
Learning Rate	1e-4, 1e-5	The learning rate did not improve any metrics significantly.
Optimizer	Adam, Ada Delta	Adam seems to improve the F1-score from 73% to 74%
Loss function	Tversky loss, Binary Cross-entropy	Tversky loss proves better in all metrics as it is better at addressing unbalanced classes.

Table 2: Hyper-parameter combinations

Based on the table seen above, a range of combinations of hyper-parameters resulted in different accuracies. With the initial combination of different loss functions, the Binary Cross-entropy did not perform well when compared to the Tversky loss function. This can be because the Tversky loss allows controlling the False Positives and False Negatives with the *alpha* and *beta* parameters. These parameters control how the loss is tuned to specifically reduce the False Positives and, thus, can help control imbalance in class much better than the Binary Cross-entropy loss function (Abraham & Khan, 2019).

3.1.1 Experimentation with the batch size

Kandel and Castelli (2020) mention that batch size influences overall accuracy, although at a minute level but does affect the accuracy of the metrics significantly. Hence, experiments with the batch size were carried out with 8, 12, 16 and 32 to understand the same.

Batch size	Accuracy	Precision	Recall	F1-score
8	**89.18%**	**73%**	63%	67%
12	87%	64%	72%	**68%**
16	88%	70%	65%	67%
32	87%	62%	**74%**	67%

Table 3: Batch size experiments

As we can observe, the overall accuracy decreases as we increase the batch size from 8 to 32, but there is a range of changes in the Precision and Recall that eventually affect the F1-score. Although the F1-score relatively remains the same, the main issue that would ultimately lead to select the batch size would be a higher Precision value. This reason is that a higher Precision translates to lower false positives, meaning fewer falsely predicted buildings. Hence, a batch size of 8 seems more suitable.

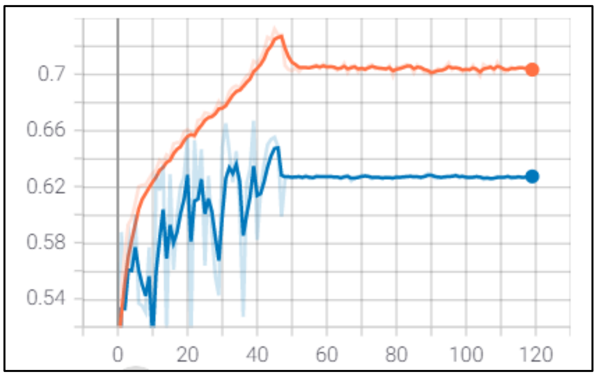

Figure 7: Ada delta optimizer (issue with stagnancy after 40th epoch in F1-score)

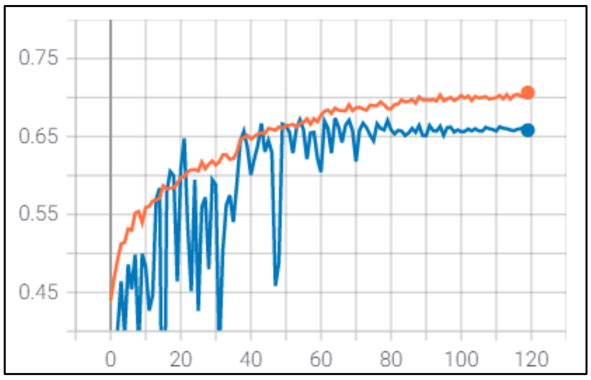

Figure 8: Adam optimizer with learning rate of 1e-3

3.1.2 Experimentation with the optimizers

Investigations suggested that the Adam optimizer resulted in higher scores in the evaluated metrics than the Ada Delta optimizer. This observation can result from the fact that Adam converges faster than Ada Delta; moreover, as we observe in Figure 67, after the 45th training epoch, F1-score plateaus and remains stagnant so forth up until the end of the training till 120 epochs. Furthermore, the average F1-score is relatively higher with the Adam optimizer (65%) than the Ada delta optimizer (62.5%).

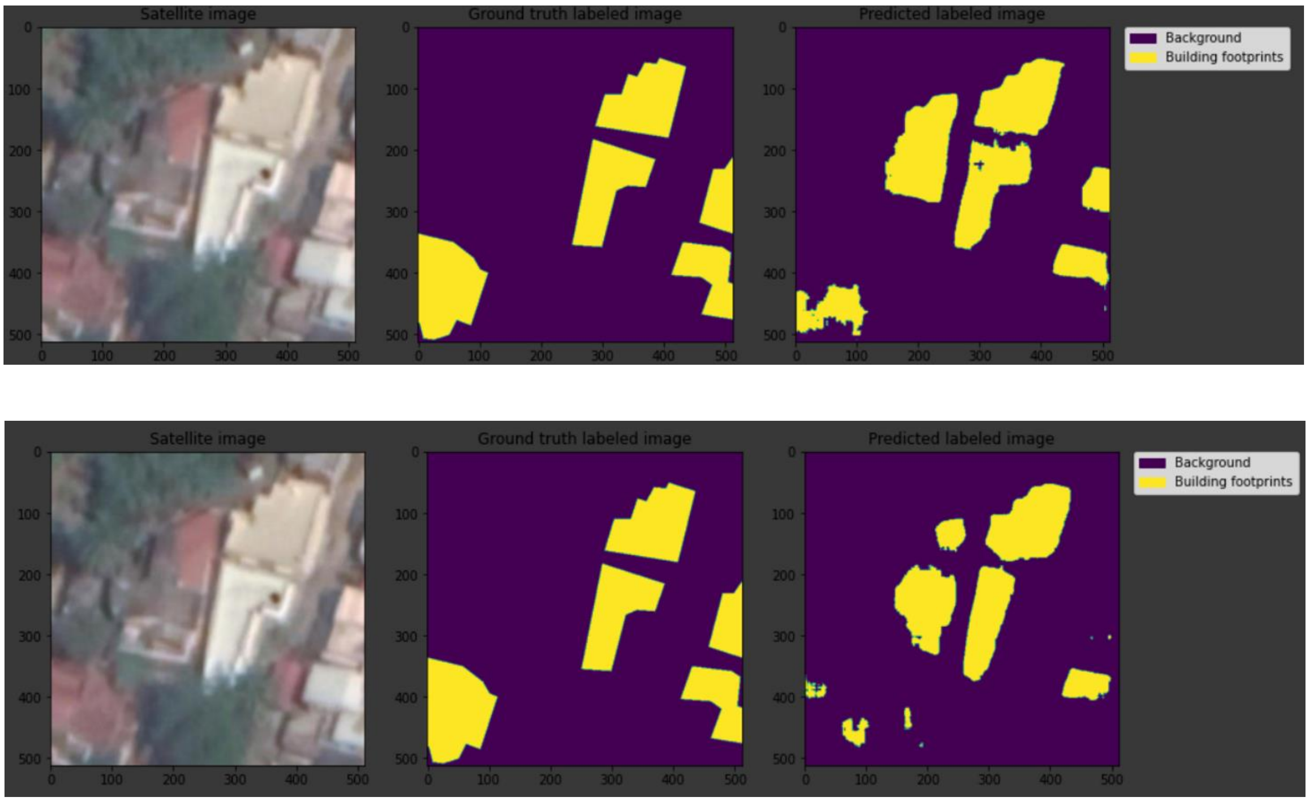

Figure 9: (Above) Effect of Adam optimiser with 1e-3 learning rate. (Below) Effect of Ada Delta optimiser

Figure 9 (below) shows that the buildings were not correctly classified on the lower left of the predicted labelled image. Comparing the middle section of both predictions in figure 10 (above and below), where the building does not have any label (discussed in the limitation sub-section), the model trained with Adam optimizer predicts the building footprint far better than the model trained with the Ada Delta optimizer. Hence, the choice of Adam optimizer proved more promising in this scenario.

Based on the observations made on the experiments shown above, the U-Net model was finally trained using a batch size of 8, a learning rate of 0.001 on Adam optimizer with a Tversky loss function. F1-score of 68% was obtained after training the model with a 9:1 training-validation split. The summary of the metrics is shown below.

Metrics	***Scores (%)***
Accuracy	94.62
Precision	72.66
Recall	66.16
F1-score	68.78

Table 4: Summary of accuracy metrics

The table above clearly depicts the best selection of the hyper-parameters, which resulted in a high accuracy rate of 94.62 per cent of the classification. This underlying experiment proves that with the help of open-source data (OSM), it is possible to obtain decent building footprints in data-scarce regions that are frequently devastated by dangerous natural hazards.

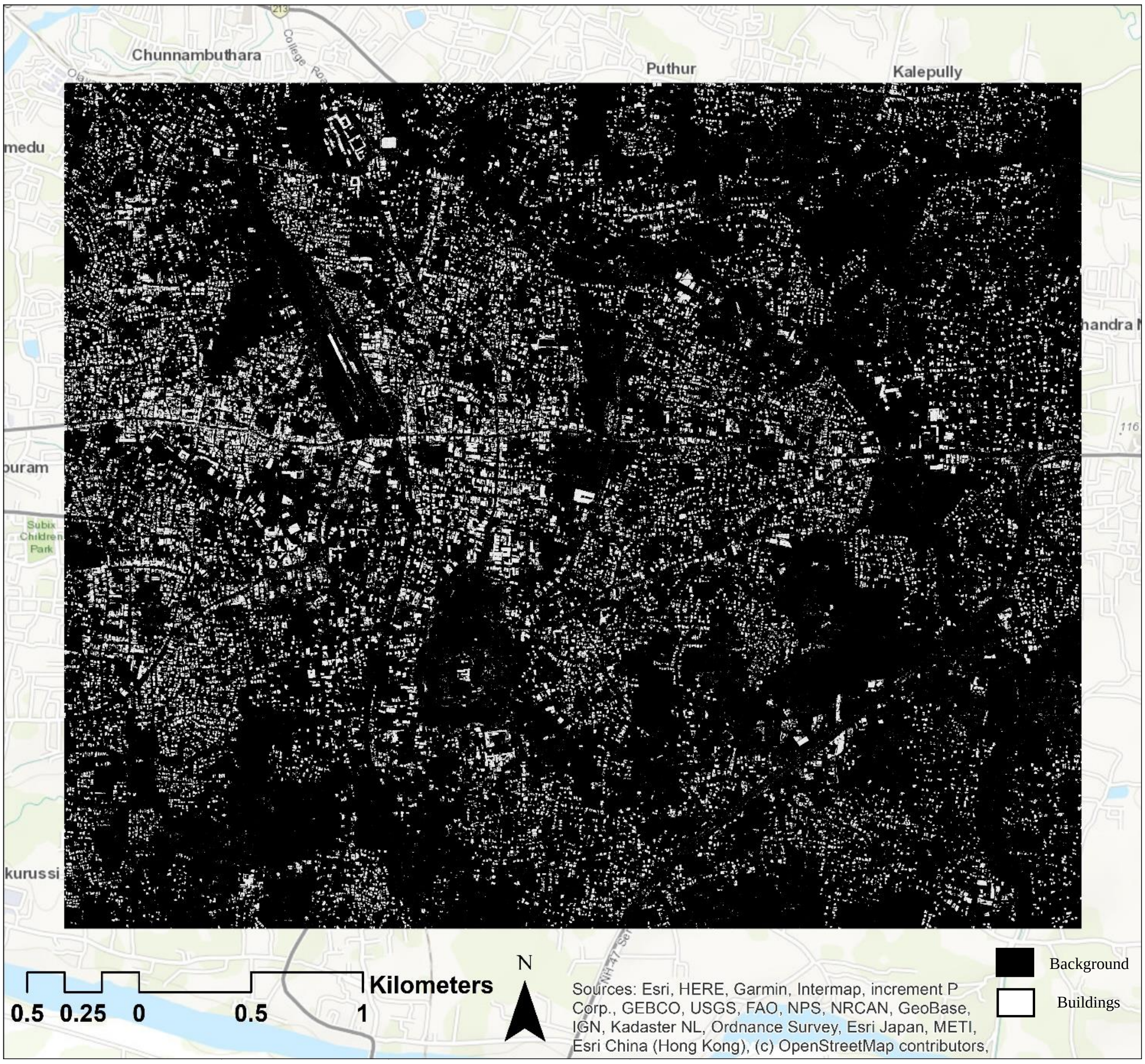
Chunnambuthara
Puthur
Kalepully
medu
handra
puram
Subix Children Park
kurussi
0.5 0.25 0 0.5 1 Kilometers
N
Sources: Esri, HERE, Garmin, Intermap, increment P Corp., GEBCO, USGS, FAO, NPS, NRCAN, GeoBase, IGN, Kadaster NL, Ordnance Survey, Esri Japan, METI, Esri China (Hong Kong), (c) OpenStreetMap contributors,
■ Background
□ Buildings

Figure 10: Final prediction over Palakkad study area.

4. Conclusion

The study depicts the importance of open-source data in creating footprints of buildings in regions with a scarcity of good quality data. Although the study area suffered from a lack of building polygons, based on the U-Net model's effectiveness, which is praised for giving good results with less training data (Wu et al., 2018), enough building footprints were detected to be used for humanitarian purposes in times of disasters. The purposes can be relief missions, emergency planning, disaster response, disaster reduction and mitigation, and hazard risk assessment.

Certain drawbacks were also discussed that was expected due to certain limitations of access to good computing power and memory. Furthermore, the disadvantage of class imbalances is very detrimental to properly detect True Positives and separating False Negatives from True Negatives. This disadvantage further contemplates with the resolution of the image, which although is very high but not enough to accurately distinguish between certain building spectral signatures from roads, leading to many False Positives. However, despite all the constraints and limitations faced, the results produced were genuinely good in terms of metric accuracies and the general prediction of buildings in the study area. The resulting outcomes of the model can be suggested to be used in disaster response scenarios and hazard exposure of such elements-at-risk.

Future improvements and suggestions can include using better computer hardware in order to be able to load and process more data in training. The use of deeper networks and complex architectures like Residual Networks as an encoder for the encoder part U-Net or nested U-Nets like U-Net++ can provide better accuracies (Chakraborty, Shaw, Aich, Bhattacharya, & Parui, 2018) as deeper networks usually help in learning more intricate features and produce better feature maps.

References

Abraham, N., & Khan, N. M. (2019). A novel focal tversky loss function with improved attention u-net for lesion segmentation. *Proceedings - International Symposium on Biomedical Imaging*, *2019-April*, 683–687. https://doi.org/10.1109/ISBI.2019.8759329

Barrington-Leigh, C., & Millard-Ball, A. (2017). The world's user-generated road map is more than 80% complete. *PLOS ONE*, *12*(8), e0180698. https://doi.org/10.1371/journal.pone.0180698

Chakraborty, B., Shaw, B., Aich, J., Bhattacharya, U., & Parui, S. K. (2018). Does deeper network lead to better accuracy: A case study on handwritten devanagari characters. *Proceedings - 13th IAPR International Workshop on Document Analysis Systems, DAS 2018*, 411–416. https://doi.org/10.1109/DAS.2018.72

Eshrati, L., Mahmoudzadeh, A., & Taghvaei, M. (2015). Multi hazards risk assessment , a new methodology. *International Journal of Health System and Disaster Management*, *3*(2), 79. https://doi.org/10.4103/2347-9019.151315

Grippa, T., Georganos, S., Zarougui, S., Bognounou, P., Diboulo, E., Forget, Y., … Wolff, E. (2018). Mapping urban land use at street block level using OpenStreetMap, remote sensing data, and spatial metrics. *ISPRS International Journal of Geo-Information*, *7*(7), 246. https://doi.org/10.3390/ijgi7070246

Herfort, B., Li, H., Fendrich, S., Lautenbach, S., & Zipf, A. (2019). Mapping human settlements with higher accuracy and less volunteer efforts by combining crowdsourcing and deep learning. *Remote Sensing*, *11*(15). https://doi.org/10.3390/rs11151799

Kandel, I., & Castelli, M. (2020). The effect of batch size on the generalizability of the convolutional neural networks on a histopathology dataset. *ICT Express*, *6*(4), 312–315. https://doi.org/10.1016/j.icte.2020.04.010

Li, H., Herfort, B., Huang, W., Zia, M., & Zipf, A. (2020). Exploration of OpenStreetMap missing built-up areas using twitter hierarchical clustering and deep learning in Mozambique. *ISPRS Journal of Photogrammetry and Remote Sensing*, *166*, 41–51. https://doi.org/10.1016/j.isprsjprs.2020.05.007

Ronneberger, O., Fischer, P., & Brox, T. (2015). U-net: Convolutional networks for biomedical image segmentation. *Lecture Notes in Computer Science (Including Subseries Lecture Notes in Artificial Intelligence and Lecture Notes in Bioinformatics)*, *9351*, 234–241. https://doi.org/10.1007/978-3-319-24574-4_28

Wu, G., Shao, X., Guo, Z., Chen, Q., Yuan, W., Shi, X., … Shibasaki, R. (2018). Automatic building segmentation of aerial imagery usingmulti-constraint fully convolutional networks. *Remote Sensing*, *10*(3). https://doi.org/10.3390/rs10030407

Zhao, W., Bo, Y., Chen, J., Tiede, D., Thomas, B., & Emery, W. J. (2019). Exploring semantic elements for urban scene recognition: Deep integration of high-resolution imagery and OpenStreetMap (OSM). *ISPRS Journal of Photogrammetry and Remote Sensing*, *151*, 237–250. https://doi.org/10.1016/j.isprsjprs.2019.03.019

Appendix

GitHub link for the code: https://github.com/kushanavbhuyan/Building-Footprint-Extraction