# Deep Learning Project Proposal

**Written by**
**Sayam Dhingra(sd5292), Yami Naik(yn2224), Bhautik Sudani(bgs8255)**[1]

[1]Tandon School of Engineering, New York University

### Abstract

Satellite images are widely used for various applications such as urban planning, environmental monitoring, and disaster response. In this project, we aim to develop a deep learning model based on U-Net to detect and classify objects in satellite images. Our code can be found at https://github.com/yaminaik/ObjectDetection

## Introduction

Applications for human development and catastrophe response are numerous for satellite imagery analytics, especially when time series methods are used. For instance, 67 of the 232 Sustainable Development Goals depend on measuring population figures, yet the World Bank estimates that more than 100 nations currently do not have efficient civil registration systems. In recent years, the application of deep learning techniques has revolutionized the field of computer vision, enabling significant advancements in various domains, including satellite imagery analysis. With the availability of high-resolution satellite imagery, the extraction and detection of buildings from such datasets have become essential for urban planning, disaster response, and environmental monitoring. This technical introduction presents a report on the development and training of a mask-based neural network specifically designed to detect buildings in the SpaceNet 7 dataset.

The primary objective of this project is to develop an accurate and efficient solution for building detection by leveraging the power of deep learning and utilizing the SpaceNet 7 dataset. The utilization of a mask-based neural network architecture allows us to generate pixel-wise segmentation masks, providing fine-grained information about building boundaries and spatial extents.

The SpaceNet 7 dataset is a widely used benchmark dataset in the field of remote sensing and computer vision. It encompasses a large collection of high-resolution satellite imagery with corresponding pixel-level annotations for building footprints. The dataset presents significant challenges due to the diversity of building types, varying lighting conditions, occlusions, and complex urban environments. This project aims to leverage this rich dataset to train a neural network model capable of accurately detecting buildings.

To address the task of building detection, we employed a mask-based neural network architecture, specifically designed for pixel-wise semantic segmentation. This architecture enables us to generate binary masks where each pixel is classified as building or non-building. Due to the significant class imbalance, a standard accuracy metric alone is not informative. We need a metric that evaluates the model's ability to accurately segment buildings in an image. Therefore, we focus on the Dice metric, which is commonly used for segmentation tasks.

Building masks serve as valuable tools for visualization and training deep learning segmentation algorithms. However, the vector labels provided in the SpaceNet 7 dataset offer an additional advantage by allowing for the assignment of a unique identifier, such as an address, to each building. This unique identification enables the matching of building addresses across different time steps, which constitutes a fundamental aspect of the SpaceNet 7 challenge (7). Because of this we changed to a mask based network instead of a bounding box based network.

## Related Work

Several papers have contributed to the field of object detection in satellite imagery, addressing various challenges and proposing novel approaches. In this section, we discuss relevant papers that discuss different aspects of object detection in satellite imagery.

In a study by Cheng G. et al (5) they provide a comprehensive review of the recent progress in generic object detection in optical remote sensing images. Unlike previous surveys that focus on specific object classes, this review covers a wide range of object categories, including roads, buildings, trees, vehicles, ships, airports, and urban areas. The paper surveys different approaches such as template matching, knowledge-based methods, object-based image analysis, and machine learning-based methods. Additionally, publicly available datasets and standard evaluation metrics are discussed. The paper also identifies two promising research directions: deep learning-based feature representation and weakly supervised learning-based geospatial object detection.

Similarly, Etten(6) addresses the challenges of detecting

small objects in large satellite imagery. The authors propose a pipeline called YOLT, which enables the evaluation of satellite images of arbitrary size at a high rate. The pipeline utilizes deep learning techniques and demonstrates the ability to detect objects of varying scales with relatively little training data across multiple sensors. The evaluation results show high localization scores for vehicles, even for objects as small as five pixels in size. The paper highlights the efficiency and effectiveness of the YOLT pipeline in object detection tasks.

FrRNet-ERoI is a deep learning model for object detection in satellite images. It is a combination of two models: FrRNet (Feature Pyramid Refined Residual Network) and EROI (Efficient Region Proposal Generation Network with Object Interaction). Therefore, Pazhani et al. (1) suggest these techniques to train these models to detect objects and classify them.

In other works, Tahir et al. (2) have done a comparative study between different CNN's like faster RCNN (faster region-based convolutional neural network), YOLO (you only look once), SSD (single-shot detector) and SIMRDWN (satellite imagery multiscale rapid detection with windowed networks). Their study concluded that SIMRDWN had the highest accuracy and YOLOv3 had the fastest speed and efficiency.

## Methodology

We have developed and implemented a neural network-based solution to detect building footprints on the SpaceNet 7 dataset. In our approach, we primarily focus on performing segmentation to identify buildings within individual images, disregarding the temporal aspect of the original challenge. To facilitate the development process, we utilize fastai, a PyTorch-based deep learning library. This powerful framework offers advanced functionality for training neural networks, incorporating modern best practices, and minimizing the need for cumbersome boilerplate code.

### Pre-processing

We create binary masks using the dataset provided. Binary masks refer to a binary representation that indicates the presence or absence of a specific object or feature in an input image. We create this using the geojson files in the spacenet7 dataset.

The dataset exhibits a notable degree of similarity among the approximately 24 images per scene. Although there are seasonal variations in vegetation and sporadic instances of building activity, the overall similarities outweigh the differences. Consequently, a decision was made to disregard the majority of the images. Initially, the intention was to retain every fifth image from each scene, capturing the variability across seasons such as January, June, November, April, and September. However, it was discovered that achieving comparable results could be accomplished by selecting only one image per scene, substantially reducing the required training time.

To accommodate the large size of the images, a strategy was implemented to divide them into smaller tiles measuring 255x255 pixels, and subsequently store them on disk. This approach was chosen based on the observation that most structures within the scenes are relatively small compared to the overall image. Consequently, the division into smaller tiles is not expected to significantly hinder the training process. Additionally, the utilization of smaller tiles offers the advantage of enabling larger batch sizes and accommodating deeper models within the available GPU memory.

It is important to note that while the majority of the images have dimensions of 1024x1024 pixels, a subset of images possesses dimensions of 1023 pixels in one dimension. As a result, a tile size of 255 was specifically chosen instead of 256 to maintain uniformity across all images.

As an improvement for future work, it is recommended to explore the creation of overlapping tiles. By incorporating overlapping regions, the issue of buildings being severed in half and consequently not fully captured by the model can be mitigated. This would enhance the model's ability to comprehend the complete shape and structure of buildings during training.

Finally we can now see the tiles with their corresponding binary masks in the Fig.1

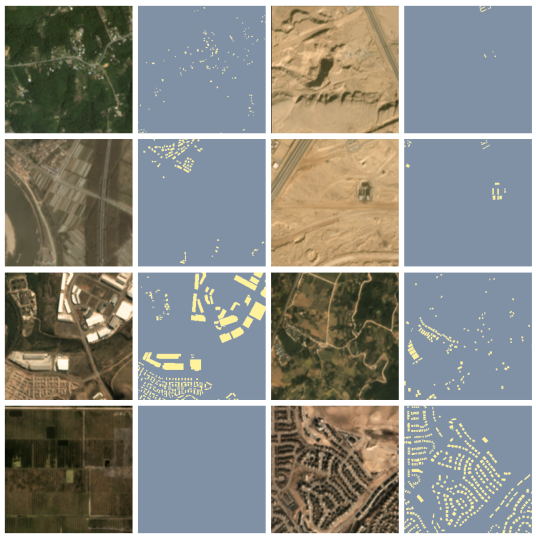

Figure 1: Tiles with their mask after pre-processing

### Dataset Challenges

The visualization of the dataset reveals several challenges that need to be addressed.

1. The dataset presents the difficulty of dealing with buildings that are often tiny, consisting of only a few pixels, and located in close proximity to one another.
2. Conversely, there are large structures within the dataset that occupy significantly greater areas compared to the small buildings.
3. Certain buildings pose challenges for recognition, even for the human eye, making it crucial to develop robust algorithms.

4. The dataset exhibits significant variations in building density. Some tiles contain no buildings at all, while others portray urban scenes with numerous buildings.
5. The images in the dataset exhibit remarkable diversity, encompassing variations in topography, vegetation, and urbanization levels.
6. A subset of tiles is partially or completely covered with UDM (User-Defined Mask) masks, necessitating strategies to handle and interpret this partial or missing information effectively.

To thoroughly examine the data's class imbalance, we conducted an analysis focusing on the percentages of building pixels within each tile. To facilitate this analysis, we developed a straightforward dataloader specifically designed for loading and evaluating the masks.

The results reveal a significant prevalence of tiles that either lack buildings entirely or contain an exceedingly small number of building pixels. Notably, 75 images, constituting nearly 10% of the dataset, do not possess a single pixel assigned to the building class. These instances can be attributed to areas characterized by vacant land, bodies of water, or tiles obscured by cloud cover.

On average, a mere 6.5% of a tile's pixels represent the building class. Moreover, the median value is merely 3.4%. Consequently, it is evident that this dataset exhibits a substantial class imbalance, further underscoring the need for careful consideration and effective handling of this inherent imbalance.

For examination we can see that the tile with the highest percentage of buildings is represented in Fig.2

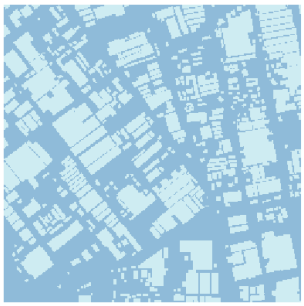

Figure 2: Tiles with the highest percentage of buildings

To help mitigating the imbalanced classes, we remove all tiles that contain no buildings at all from the training set. This reduces the amount of samples by 10%, thereby accelerating the training while helping the model perform better.

## Data Loading

The chosen transformations for satellite images are deemed reasonable and appropriate. These transformations encompass vertical and horizontal flipping, rotation, slight adjustments to brightness, contrast, and saturation. Furthermore, normalization is applied based on ImageNet statistics, facilitating the utilization of pretrained models in subsequent stages.

To streamline the dataset creation process, we leverage the convenient DataBlock API provided by fastai. Notably, only 16 tiles per scene are loaded, resulting in the inclusion of just one image per region.

The training set consists of 741 tiles, while the validation set comprises 144 tiles.

As anticipated, the dimensions conform to the following specifications:

1. Each batch contains 12 images.
2. The input images consist of 3 channels.
3. The target masks lack color channels.
4. The image size is set to 255x255 pixels.

## Model

The current task involves tackling an image segmentation problem. In the original competition, the objective entails assigning distinct labels to individual buildings to track their evolution over time (known as instance segmentation). However, in this particular approach, the focus is shifted towards semantic segmentation, which involves classifying each pixel as either belonging to a building or not.

The fastai library offers a remarkably straightforward solution through the utilization of a U-Net, a well-established architecture for image segmentation tasks. The DynamicUNet (8) module, coupled with an encoder architecture, automatically constructs a decoder and establishes cross connections. This streamlined approach enables the creation of a U-Net using various pretrained architectures, affording more time for experimentation as opposed to writing code from scratch. Several aspects were considered in this decision-making process:

1. Encoder: A xResNet34 model pretrained on ImageNet was chosen as the encoder. This 34-layer encoder strikes a favorable balance between accuracy and memory/compute requirements.
2. Loss function: The selection of an appropriate loss function is crucial for segmentation problems. In this case, a weighted pixel-wise cross-entropy loss will be employed. The incorporation of weights is of paramount importance due to the imbalanced nature of the dataset.
3. Optimizer: The default optimizer, Adam, will be utilized for training the model.
4. Metrics: Given the highly imbalanced classes, a simple accuracy metric would not be meaningful. For instance, a model could predict "no building" for every pixel in an image with only 3% buildings and still achieve 97% accuracy. As an alternative, the focus will be on the Dice metric, which is commonly used for segmentation tasks. This metric, equivalent to the F1 score, measures the ratio of $\frac{2TP}{2TP+FP+FN}$. Additionally, the foreground_acc metric provided by fastai will be included, measuring the percentage of correctly classified foreground pixels (in this case, the building class), akin to Recall.

By considering these factors and making informed decisions, we aim to address the challenges of the image segmentation problem effectively.

## Training the model

First we need to decide the learning rate. To do this we can use fastai's learning rate finder to pick a reasonable learning rate. We can see the resulting graph in the Fig.3

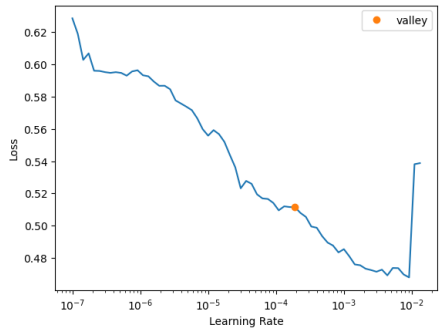


Figure 3: Learning rate discovery graph

After reviewing the graph we can see that the suggested learning rate is somewhere around 1e-4, where the loss decreases steadily.

To train both the encoder and decoder simultaneously, we unfreeze the model. It is important to note that the pretrained features within the encoder should be altered minimally. To achieve this, we set a lower learning rate specifically for the encoder, ensuring that the fine-tuning process does not excessively modify the pretrained features. This is accomplished by defining the learning rate as 'slice(lr_max/10, lr_max)'.

For the training process, we employ the 'fit_one_cycle' method, which encompasses a progressive learning rate schedule. Initially, the learning rate starts at a low value during a warm-up period, gradually increases to a maximum value of 'lr_max', and then anneals to 0 towards the end of the training process. This approach facilitates a balanced and effective optimization process for the model.

In the Fig.4 we can see the momentum graph. In the momentum plot the Y-axis represents the momentum and the x-axis represents the steps/epochs.

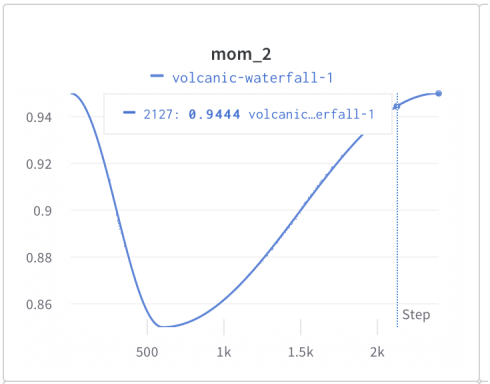


Figure 4: Momentum plot

The ADAM optimizer in neural networks incorporates a form of momentum through the use of exponential moving averages for estimating the first moment of the gradients. This allows ADAM to have momentum-like behavior without a separate momentum parameter. Additionally, ADAM combines adaptive learning rates based on the estimates of the first and second moments of the gradients.

In the second graph in Fig.5 we can see the loss value. Y-axis shows the loss value and the X-axis shows the steps/epochs. This represents the loss on the validation stage.

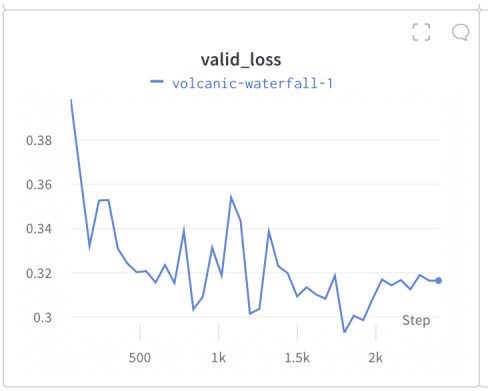


Figure 5: Valid Loss Plot

Next in the Fig. 6 we can see the leraning rate plot. On the Y-axis we have the learning rate vlaue and the X-axis has the steps/epochs. The learning rate controls the magnitude of these parameter updates.

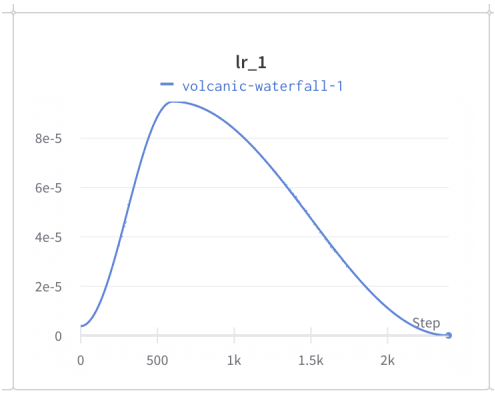


Figure 6: Learning Rate Plot

We can observe that the epsilon value and weight decay values remain constant trhought the training process for all steps and epochs. Epsilon at 0.00001 and weight decay at 0.01

In the training table which appears after we start training we can see the train_loss, valid_loss, dice score, foreground_acc and time for each epoch. The best model at epoch 29 has a Dice score of '0.539882', which is a good score for us to be able to successfully detect the buildings. We can also observe the valid_loss and train_loss decrease as the epochs increase. They start to even out at epoch 33.

Therefore, it was safe to stop after epoch 39.

Now, since the training has finished we can view the loss plot graph. In Fig.7. On the X-axis we can see the steps/epochs and on the Y-axis we can see the loss values.

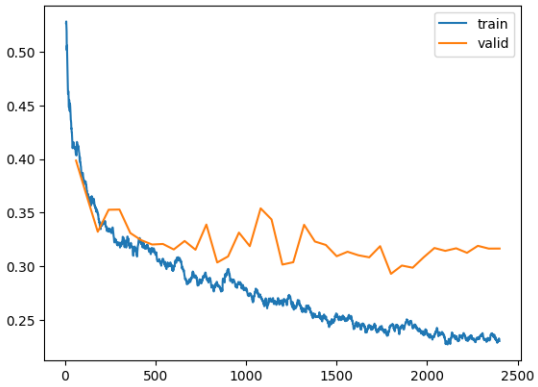


Figure 7: Loss value plot

## Results

In this section we will show the results of the training. We will also show the result images after testing our model on the test dataset. We store the values of the outputs in descending value of loss.

### Samples with highest value of loss

The analysis of images exhibiting the highest losses reveals a consistent pattern of dense urban areas. It is evident that the model encounters challenges in accurately identifying and segmenting large buildings, as they are frequently overlooked or disregarded. Additionally, it is notable that the model tends to merge very small buildings, resulting in conglomerations or ”blobs” of multiple buildings. Consequently, the task of tracking individual buildings within this context is expected to be inherently difficult. We can observe Fig.8

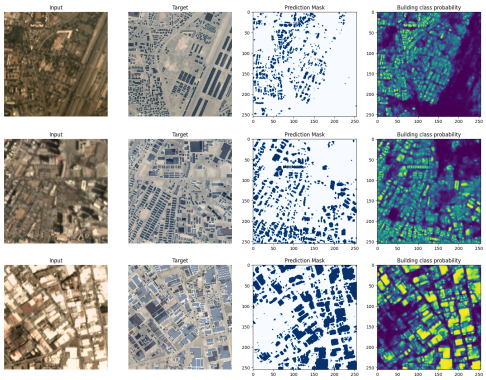

Figure 8: High Loss

### Samples with medium value of loss

The model exhibits a tendency to merge small buildings into larger conglomerations, leading to the formation of indistinct blobs. Additionally, there are instances where false positives occur, indicating the misclassification of non-building elements as buildings. However, it is worth noting that amidst these limitations, the model demonstrates noteworthy performance by accurately identifying certain buildings that pose challenges even for human observers. We can observe this in Fig.9

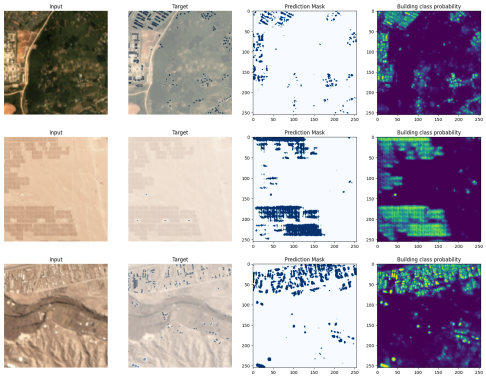

Figure 9: Medium Loss

### Samples with low values of loss

In images with a sparse distribution of buildings, the model exhibits varied performance. Generally, the accuracy in such cases appears to be comparatively better than in densely populated areas. However, it is important to note that the model still generates false positives, erroneously classifying non-building elements as buildings. Furthermore, certain tiles exhibit peculiar artifacts, particularly in the corners. This suggests that the model may interpret the corners themselves as buildings, particularly on tiles encompassing water areas. We can observe this in the Fig.10

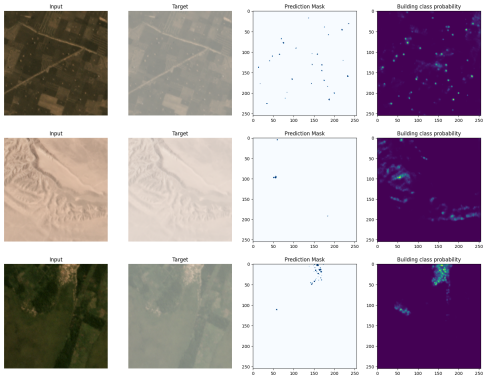

Figure 10: Low Loss

Figure 11: Full Scene 1

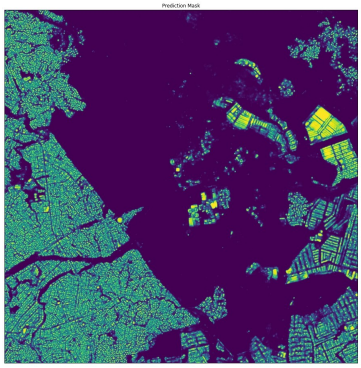

Figure 12: Full Scene 2

### Full image results

Previously we showed the results for each tile image. In this section we will discuss the entire image and show the figures with their prediction masks. We can observe one entire image in the Fig.11 and Fig. 12.

### What worked?

1. Pretrained encoder utilization: Using a pretrained xResNet34 model from ImageNet improved performance by leveraging learned features.
2. Selective image utilization: Focusing on one image instead of five yielded comparable results, reducing training time without sacrificing outcomes.
3. Standard data augmentations: Applying common augmentations prevented overfitting, enhancing the model's ability to handle diverse and unseen images.
4. Weighted cross-entropy loss: Incorporating weights into the loss function notably improved performance by addressing dataset imbalance and bias towards the background class.

Additionally, although undersampling had a modest impact, it facilitated a slight improvement in training efficiency and contributed to enhanced accuracy.

Overall, these strategies and techniques demonstrated their effectiveness in improving the model's performance and addressing specific challenges encountered in the image segmentation task.

## Conclusion

Evaluating the results is challenging due to the lack of direct comparison. Despite a Dice score of 0.57, considering dataset difficulties and minimal customization, the result is satisfactory. The approach successfully classified pixels as building or non-building, even identifying challenging buildings. The segmentation framework effectively captured relevant features. Future work aims to recognize and track individual buildings over time, as in the original SpaceNet7 challenge, for building detection and tracking in satellite imagery analysis. To improve results, strategies include using overlapping tiles and scaling up images for finer details, dynamic thresholding for better differentiation, advanced segmentation models, deeper models with more resources, cross-validation, and ensemble learning. These strategies show promise for future research, addressing limitations and improving segmentation accuracy and robustness.

## References


[1] Pazhani, A.A.J., Vasanthanayaki, C. Object detection in satellite images by faster R-CNN incorporated with enhanced ROI pooling (FrRNet-ERoI) framework. Earth Sci Inform 15, 553–561 (2022). https://doi.org/10.1007/s12145-021-00746-8

[2] Tahir, A.; Munawar, H.S.; Akram, J.; Adil, M.; Ali, S.; Kouzani, A.Z.; Mahmud, M.A.P. Automatic Target Detection from Satellite Imagery Using Machine Learning. Sensors 2022, 22, 1147. https://doi.org/10.3390/s22031147

[3] Avanetten GitHub user https://github.com/avanetten/CosmiQ_SN7_Baseline

[4] Rim-Chan GitHub user https://github.com/Rim-chan/SpaceNet7-Buildings-Detection

[5] Gong Cheng, Junwei Han, A survey on object detection in optical remote sensing images, ISPRS Journal of Photogrammetry and Remote Sensing, Volume 117, 2016, Pages 11-28, ISSN 0924-2716, https://doi.org/10.1016/j.isprsjprs.2016.03.014

[6] https://doi.org/10.48550/arXiv.1805.09512

[7] Blog: Adam Etten The SpaceNet 7 Multi-Temporal Urban Development Challenge: Dataset Release https://shorturl.at/rSUV5

[8] https://docs.fast.ai/vision.models.unet#DynamicUnet