

Novel Building Detection and Location Intelligence in Aerial Satellite Imagery

Sandeep Singh*, Christian Wiles, Ahmed Bilal

School of Computer Science, Georgia Institute of Technology, Atlanta, United States

Email address:

ssingh600@gatech.edu (Sandeep Singh), cwiles7@gatech.edu (Christian Wiles), abilal7@gatech.edu (Ahmed Bilal)

*Corresponding author

To cite this article:

Sandeep Singh, Christian Wiles, Ahmed Bilal. (2024). Novel Building Detection and Location Intelligence in Aerial Satellite Imagery. *American Journal of Computer Science and Technology*, 7(4), 183-194. <https://doi.org/10.11648/j.ajcst.20240704.16>

Received: 30 August 2024; **Accepted:** 26 September 2024; **Published:** 18 December, 2024

Abstract: The accurate detection and extraction of building information from aerial imagery is of paramount importance in urban planning, land use analysis, and disaster management. This study presents a comprehensive investigation into the development of a robust and efficient methodology for building detection in satellite imagery utilizing state-of-the-art deep learning techniques. We conducted a comparative analysis of three distinct semantic segmentation models based on the U-Net architecture: a baseline U-Net trained from scratch, a U-Net incorporating a pre-trained ResNet34 encoder, and a U-Net with custom architectural enhancements. Our methodological approach encompassed data augmentation strategies, transfer learning techniques, and ensemble methods to optimize model performance. The Inria Aerial Image Labelling Dataset served as the primary source for model training and validation. We explored the efficacy of various loss functions, including dice loss, focal loss, and weighted cross-entropy, to address class imbalance and enhance segmentation accuracy. Model performance was rigorously evaluated using a comprehensive set of metrics, including pixel-wise accuracy, Intersection over Union (IoU), and F1-score. Our highest-performing individual model achieved a dice score of 92 percent on the validation set, while the implementation of ensemble techniques further improved detection accuracy to 93 percent on the heldout test set. Post-processing algorithms, incorporating traditional computer vision methods, were applied to refine building polygon delineation. This research demonstrates the efficacy of deep learning-based segmentation approaches for building detection in aerial imagery and offers valuable insights into potential applications across various domains, including urban planning, construction monitoring, and disaster response. Future research directions may explore building classification, change detection analysis, and model optimization for real-time applications in dynamic urban environments.

Keywords: Building Detection, Aerial Imagery, Semantic Segmentation, U-Net, Deep Learning, Ensemble Methods, Satellite Image Analysis, Urban Planning

1. Introduction

We plan on reproducing semantic segmentation CNN models based on U-Net [2] and Res-U-net [3] algorithms, trained by transfer learning using the ImageNet dataset. In addition, we will optimize these different models with focal loss, dice loss, cross entropy loss, hierarchical loss and differently weighted intersection-over-union [IoU] loss to overcome issues of scale difference [3] in building detection. To clearly delineate each individual member's contribution, we will be organizing our paper by each member's contribution.

2. Methodology

We have decided to work on three different models in parallel, which are inherently different on following parameters:

1. Architecture of models: We have decided to use different variations of UNet architectures, which might help us focus of different patterns easily.
2. Pre-training usage of the models: We have employed training from scratch as well using pre-trained weight for the encoder layers across out 3 of the models.

3. Loss functions being employed by models: We have used different loss functions in all the different models.
4. Image sampling and augmentations employed: We have used totally different techniques to samples the images from the data sets available to introduce the component of randomness in data distribution across models. While model 1 and 2 have used random resized crops of size 224x224 from big 5000x5000 images and masks; model 3 has resorted to fixed size split of big 5000x5000 images in 512x512 tiles. Both these sampling techniques have again used totally different set of transforms with different quantifiers.

While we have details of all the models provided in the sections 3, 4 and 5, here is a quick overview of the ensemble technique exploited by us:

1. Get prediction from all 3 models. All predictions must of same size as input image. We output thresholded mask as tensor of softmax probabilities.
2. Chose the most confident pixel from each of these masks from 3 models for every pixel location's softmax probability. This is will be single merged mask of same size as input.
3. From merged mask predicted, drop anything that is not at least 0.75 confident. Rational behind doing this is that we have already chose most confident pixel locations from 3 models masks, So, all the softmax value at this stage must be pretty confident ones at least in one of the models.

As seen in the figure 1. We have final mask, which is without any less confident pixel.

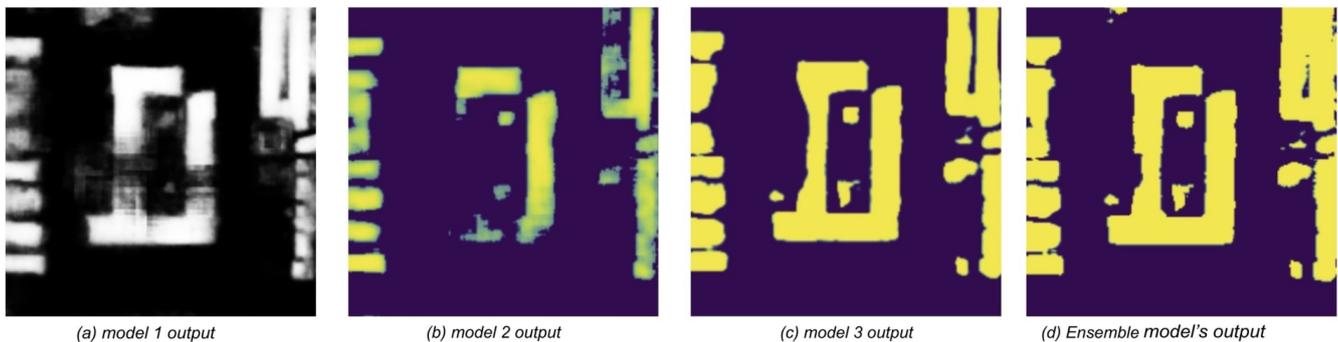


Figure 1. Ensemble method being employed. Output softmax probabilities are compared and chosen to select most confident pixels of 2 models and thresholded to be at least 75% confident to be considered for final mask.

3. Standard U-Net Architecture

As all three models are derived from U-net, we will begin with a discussion of the canonical U-net architecture. U-net is a modular model largely consisted of encoder blocks and decoder blocks. A graphical depiction of a u-net model can be found in Figure 5.

3.1. Encoder Blocks

The encoder layer is primarily responsible for detecting the 'what' elements of the images. The goal is to be able to extract features in the image at different scales and different levels of abstraction. As such, at every steps of the encoder, two 2D convolutional blocks are used to extract information from the image and double the size of the feature space. At each encoder layer, we used a maxpool layers of 2x2 kernel size and a stride of 2 for down sampling spatially. This allowed us to increase the number of filters at each of our encoder layers without being extremely computationally expensive and increase the receptive field of our filters with deeper layers allowing for segment detection at multiple scales.

3.2. Decoder Blocks

The decoder layers are the up sampling layers in the model. The primary purpose of these layers is to localize the features extracted in the encoder block. This information is essential in our semantic segmentation in order to be able to output an image with the buildings detected localized in the right spaces in our output mask. For up sampling, we used Transposed Convolution layers. This allows us to ultimately assign class labels to each pixel in our image as part of our semantic segmentation.

At each of our decoder layers, we also make use of skipped connection given to us by the respective encoder layer for our decoder layer. The skipped connections cross from same sized part in the encoders to the decoders. The skipped connections allow us to overcome problem of vanishing gradient, increasing dimensionality and help regain the initial spatial information that we lost during the encoding path.

3.3. Integration

A full u-net model is composed of N encoder blocks, and N-1 decoder blocks. The feature space of the first encoder block is a hyperparameter but seems to be often set to 64 or 128. Save for the last encoder block, the output of the final convolutional layer in each encoder block is cropped and concatenated with

the output of the transposed convolutional layer of the decoder block. At the end of the model, a 1x1 convolutional layer is used to create a classifier head with the same features space as the number of classes. This can be passed to a softmax layer to produce class probabilities.

4. Model 1 Approach [Christian]

4.1. Data Pipeline and Exploration

For this project, we used the Inria Aerial Image Labelling Dataset for training [4]. The dataset consists of 360 [180 train and 180 test] 5000x5000 pixel full-color images with corresponding masks indicating the presence of building or non-building pixels. Only the training set had ground-truth segmentation masks available. Images were taken from a variety of settings, including rural and urban cities from different continents. A few problems had to be solved to enable training with this dataset: data augmentation and quick random access.

4.1.1. Data Pre-processing and Augmentation

To generate more data for training, data augmentation was undertaken. Pytorch's standard transformation library does not make allowances for maintaining consistent transformations between an image and a segmentation mask, so the functional transforms library was used, which allows for the randomness to be provided by external variables, which can be held static between the ground truth and full-color images. For each of the 180 input training images, 350 224x224x3 image patches were created. This was intended to allow transfer learning for networks trained on ImageNet. A patch was taken from the image with random width [between 100 and 500 px], height [within +/- 10% of the random width], and image origin. This image was then randomly flipped horizontally and/or vertically and normalized by ImageNet standard deviation and mean. The intention of this was to teach the model scale and orientation invariant features. Once the incoming data was processed, it was split 80/20 into train and validation sets. Data from all 5 cities in the dataset was randomly selected for both validation and training set, as the test set consists of different cities and would be usable for testing how well the model can generalize. A manual seed was set such that this split was repeatable if restarting training from a checkpoint.

4.1.2. Caching

It was found that performing the loading of the full-color images and performing transformations on the fly was too computationally intensive. To alleviate this problem, after the first time the transformations were done, the resulting input and target tensors were saved to disk. This reduced the time to train significantly, as a 70 MB image did not have to be loaded and manipulated thousands of times per epoch, but instead a 600 KB tensor could be used.

4.1.3. Other Considerations

It was also deemed important to add support for visualization of training-related metrics. Tensorboard support was added to the project to track training and validation set loss and accuracy, precision-recall curves for the validation set, and visualization of the forward pass of the model on the validation set. One last consideration made was the use of a seed when splitting the training and validation set. It was noticed that when resuming training from a checkpoint that the validation set was not the same as before the checkpoint. By maintaining the constant seed, a barrier was maintained between the two sets.

4.1.4. Dataset Statistics

As is often the case with segmentation tasks, the dataset was not balanced between building pixels and non-building pixels. The training dataset was analyzed to determine the prevalence of each class. The findings are below.

4.2. U-net from Scratch in Pytorch

To test the hypothesis that building detection was a sufficiently specific domain to merit training from scratch, a u-net was created with random Xavier-initialized weights. 4 encoder and 3 decoder blocks were used, with the first encoder block having a hidden dimension of 64 features. Both the original U-net paper [2] and Johannes Schmidt's blog posts [5] were consulted in the creation of the model.

Two major deviations were attempted from the models mentioned above. Both u-nets resulted in a cropped image with every convolution due to the use of valid padding. By using same padding, on convolutions and transposed convolutions, we can return an image that is of equal size to the input. This may result in slightly worse accuracy in the extremities of the image due to the extrapolation employed by same padding, but does simplify some aspects of the analysis, as every mask pixel has a corresponding prediction.

Secondly as the output of a 2-class softmax classifier only has 1 degree of freedom, it was attempted to perform classification as a single-class regression, with the output of the regression put through a sigmoid function. This one-channel output can then be interpreted as p [building]. This approach was eventually discarded, as it had a very small impact on model size due to only affecting the final 1x1 convolutional layer, and adding a second classifier dimension increased model performance by a few percentage points.

Finally, batch normalization was added between the convolutional and activation layers in encoder and decoder blocks. These recenter the distribution of the output of the convolutional layers and add to stability in training as seen in [6].

To discourage overfitting, dropout was added between the output of the final decoder layer and the 1x1 convolution.

4.2.1. Loss Function

3 different loss functions were attempted with this model. With the regression-based approach, weighted mean square

error was used due to the class imbalance. On a per-batch basis, the effective number of building and non-building pixels was calculated on a per-batch basis, similar to the methodology in [7]. This weighting was weigh the loss on the minority class [buildings] more heavily.

Once the model had progressed to a two-class method, two losses were pursued: weighted cross-entropy loss and dice loss. Dice loss was pursued due to its background in segmentation tasks and invariance with respect to class imbalance [as dice loss is related to the size of the true positive region]. Binary cross-entropy weighted by the inverse of effective number was also explored. This provides a more convex loss function that should be easier and more stable to train.

4.2.2. Training

Training was performed over 20 epochs with the Adam optimizer. [8] No learning rate scheduler was used to govern learning rate as epochs progressed, as Adam should manage its own learning rates on a per-parameter basis [9]. A batch size of 20 was used to fit in GPU memory. The model state was saved to disk whenever validation accuracy exceeded the previous maximum to allow for training to be resumed later.

4.2.3. Results

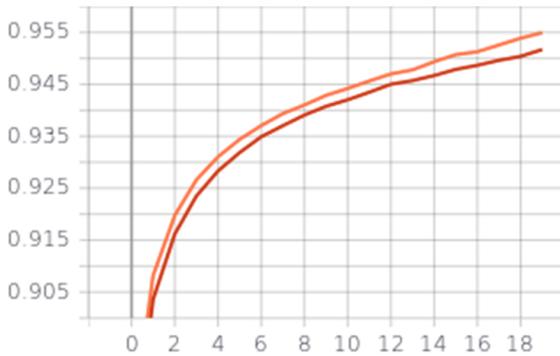


Figure 2. Training accuracy over 20 epochs for model 1. Weighted CE in orange, Dice Loss in red.

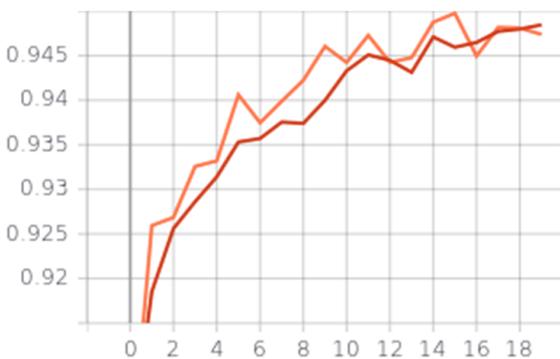


Figure 3. Validation accuracy over 20 epochs for model 1. Weighted CE in orange, Dice Loss in red.

Table 1. Model 1 Validation Set results after 20 Epochs.

Loss Fn	Accuracy	IOU Score	F1-Score
Dice	94.9%	0.717	0.836
BCE	95.0%	0.726	0.841

Weighted cross-entropy resulted in marginally better training efficiency and overall metrics, but by an almost negligible amount. For both approaches, overfitting does not appear to be a concern, as the validation and training accuracy are almost identical.

Though this model performed worse than Model 2, it is hard to say whether it is due to differences in pre-training or the squeeze-and-attention layers. It is likely, however, that due to learning features from scratch, it may provide diversity in the ensemble that can help in overall accuracy.

5. Model 2 Approach [Ahmed]

5.1. Model Specifications

5.1.1. Double Convolution Blocks

The Unet build consisted of a double convolution layer, where each convolution layer consisted of a kernel size of 3, stride and padding of 1. We set the bias to false in order to add a BatchNorm layer, which is then followed by a ReLU activation layer. [4] We settled on a small 3X3 kernel receptive field in our convolution layers in order to be able to detect very small edges and shapes in our aerial images. Doing so is especially relevant for our aerial images as there is a lot of noise in the images and our model needs to be able to use small edges and shapes to detect buildings as buildings appear in many different sizes in our input images.

5.1.2. Encoder Layers

The authors in *U-Net: Convolutional Networks for Biomedical Image Segmentation* [2] recommend encoding layers with output channels 64,128,256, 512 and 1024. However, we found more success with output channel layers 16,32,64,128 and 256. We believe this is because lower output channels of 16 and 32 in the start allow us to detect really small building segments with a small receptive field. In addition, the 512 and 1024 channel layers were not leading to any significant performance gains in our testing.

5.1.3. Decoder Layers

At each of our decoder layers, we also make use of skipped connection given to us by the respective encoder layer for our decoder layer. The skipped connections cross from same sized part in the encoders to the decoders. The skipped connections allow us to overcome problem of vanishing gradient, increasing dimensionality and help regain the initial spatial information that we lost during the encoding path [10].

5.2. Pre-trained ResNet34 Encoder Specifications

We now add ResNet 34 Encoder layers to the model. As such, we are now performing the Double Convolution blocks 3,4,6, and 3 times at each encoder layer level, using skipped connections between encoder layers, and using a higher stride to down sample instead of max pooling. These encoder layers are also pre-trained on image-net dataset.

5.2.1. Encoder Modifications

After finding success in our U-Net built with 16,32 and 64 output channel initial encoder layers, we replace the initial ResNet convolution, ReLU and Max Pool layers with our U-Net 16,32 and 64 output channel encoder layers with skipped connections in order to preserve a lot of the small shapes and edges information in our images.

5.2.2. Attention Mechanism

For the loss function, we will be using the Dice Loss to create cleaner mask segments to represent the buildings. In order to supplement our model in reducing the Dice Loss, we also include an attention mechanism using spatial and channel 'squeeze & excitation' Blocks. [11] This is done to aid our encoder layers in spatial encoding for more accurate mask prediction and better network flow. The authors in *Recalibrating Fully Convolutional Networks with Spatial and Channel 'Squeeze & Excitation' Blocks* [12] found a reduction of 4-9% in the Dice Loss. We see similar results in our testing.

5.2.3. Results

In our testing, we saw the pre-trained image-net backbone significantly increase the model performance. After 15 epochs, we saw the following results.

Table 2. Validation Set results after 15 Epochs.

Set	Acc	Loss	IOU Score	F1-Score
Train	0.965	0.116	0.749	0.856
Val	0.961	0.129	0.702	0.824

Detailed Model 2 results and mask outputs available in appendix Section F

6. Model 3 Distinctive Approach [Sandeep]

6.1. Data Sampling Strategy

For this model, we have used "Progressive Resizing." This is a training technique where we purposefully change the contents of image by resizing the images to contain more area. Instead of randomized crops of size 224x224, we created 512x512 non-overlapping and contiguous tiles, only then resizing them to 224x224 input images. [13] On average, each tile has almost 4 times more buildings in each tile compared to model 1 and 2. Hence, the model more easily learns smaller

buildings in more crowded areas [14].

6.2. Architectural Considerations

For model 3, we have evaluated 3 types of encoders [Resnet18/34/50] and choose ResNet34 as ResNet18 has shown to be struggling to encode features of smaller buildings successfully. ResNet34 and 50 have shown very similar, but resnet 50 slow performance in detected buildings without any marginal increase in performance. One more significant improvement in model 3 was the use of we used Pixel Shuffle up-sampling in the decoder blocks, as provided by shuffleblock implementation [15].

6.3. Training and Validation Split

For model 3, we have done the data split on the basis of geography, instead of random ratio split. Out of training data from 5 cities as: Austin, Chicago, Kitsap, Tyrol, Vienna. Different cities are included in each of the subsets. e.g., images over Chicago are included in the training set, but not on the test set. Also, images over San Francisco are included on the test set but not on the training set. At the same time, we have tried to include the training data all type of structures of building. e.g. low rise vs high rise vs community living buildings apartment complexes. [16]

6.4. Custom Loss Function Design

Model 3 did not used CrossEntropy loss. Instead, we have written our own custom loss function, which has helped us predict foreground pixel with higher softmax confidence. [17] We implemented Combined Loss of Dice Loss and Focal Loss with equal weights. Focal loss penalizing more confident wrong predictions more heavily. We have used gamma value of focal loss as 2. Also, Dice score has provided feed back to strive to keep precision and recall both highest possible. Also, for model 3, we have used the Dice Score metric.

6.5. Training Convergence

For model 3, we used a LR finder scheduler before starting to fine tune the pretrained weights of ResNet34 encoder. This has helped of find the most appropriate maximum LR value. The second innovative technique employed by us was "Fit-one-cycle" [18] to achieve super convergence. In this technique, we increase LR to maximum value in initial batches before start to anneal the learning rates. Please refer to figure for LR finder and fit one cycle both [8]. This technique is taken from Leslie Smith iconic Super Convergence paper. Also, We trained with total 40 epochs without over-fitting and saved the model only when better score on validation was seen without over-fitting, while Dice score was approaching 92%. [1]

Table 3. Model 3 Losses and Metrics Values.

Train.Loss	Valid.Loss	Dice.Score
0.102414	0.115400	0.920870

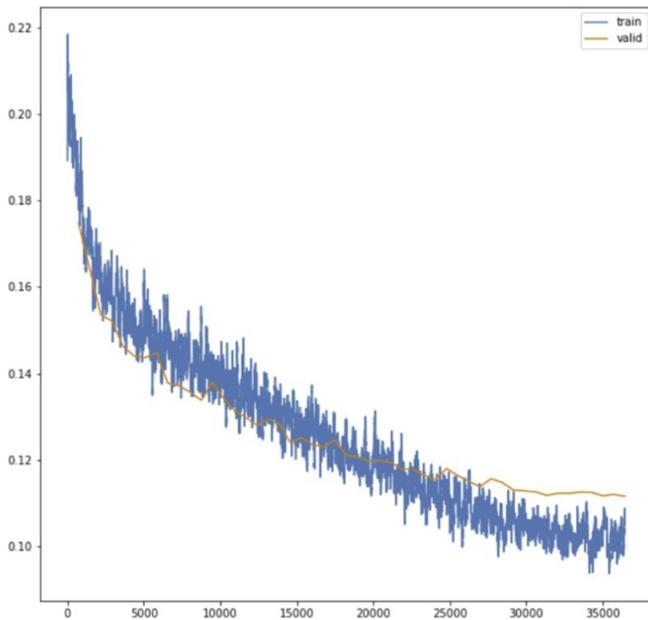
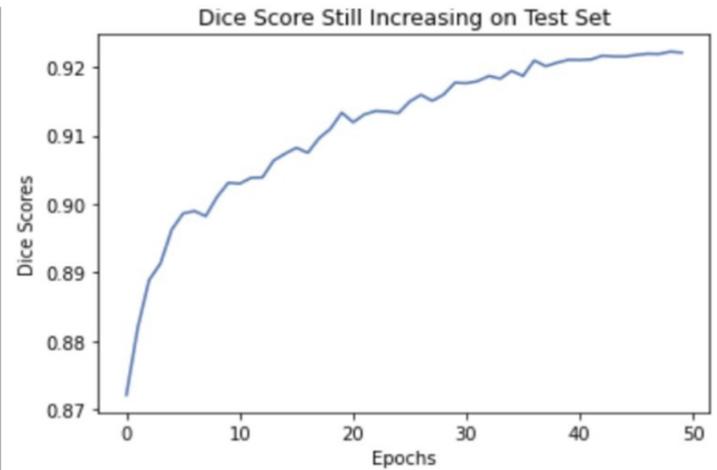


Figure 4. Model 3 Training Curves.



While training and validation loss start to diverge, our dice score still getting better on unseen test data. This was investigated and found to be because of bad labelled data. So, model is still learning and can be trained for few more epochs.

6.6. Post Processing Enhancements

As can be seen in some of our validation images, the masks created by this model are not always crisp and polygonal. Please refer to appendix figure for Model 3's traditional computer vision techniques of segmentation and attempt to fuse the results with Unet predicted mask. We have tried to experiment with Otsu's threshold, Watershed segmentation, SLIC Super pixel algorithm for segmentation. [15] We have selected all the coincidental mask segments from super pixel algorithms' output with Unet's mask and tried to shape correct polygon for buildings with sharp edges [see figure 10].

7. Experiences and Challenges

7.1. Challenges

Satellite images are very noisy and affected by many factors such as weather, zoom level, resolution, trees, and cost to obtain [19]. After evaluating, we used Google Static maps API for pulling additional data because of quality and ease of use.

7.2. Project Success Criterion

We were able to build model with Dice score more than 90 percent on test set and model 2 achieving detection of more 96 percent ground truth pixels in valid set. With help of Softmax based adaptive selection and ensemble, we have achieved detection of more 93 percent ground truth pixels in test set.

7.3. Conclusion and Future Aspirations

We have explored and confirmed that deep learning based segmentation is very effective in segmenting buildings.

We could extend these models for clustering similar buildings, classifying residential vs commercial, predicting future constructions or detecting illegal construction or activities. Potential technical improvements include learned polygonization, elastic transformations during training, and model compression for cheaper inference.

Abbreviations

CNN	Convolutional Neural Network
IoU	Intersection over Union
INRIA	Institut National de Recherche en Informatique et en Automatique
ResNet	Residual Network
SVG	Scalable Vector Graphics
GPU	Graphics Processing Unit
API	Application Programming Interface
LR	Learning Rate
SLIC	Simple Linear Iterative Clustering
ORCID	Open Researcher and Contributor ID
DOI	Digital Object Identifier
GIS	Geographic Information System
VHR	Very High Resolution
CRF	Conditional Random Field
DSM	Digital Surface Model

ORCID

0009-0002-6249-2541 (Sandeep Singh)
 0009-0006-4154-0419 (Christian Wiles)
 0009-0006-5940-2428 (Ahmed Bilal)

Author Contributions

Sandeep Singh: Conceptualization, Data curation, Formal Analysis, Funding acquisition, Investigation, Methodology, Project administration, Resources, Supervision, Validation, Visualization, Writing - original draft, Writing - review & editing

Christian Wiles: Formal Analysis, Investigation,

Methodology, Validation, Visualization

Ahmed Bilal: Formal Analysis, Funding acquisition, Investigation, Project administration, Software, Validation, Visualization, Writing - review & editing

Conflicts of Interest

The authors declare no conflicts of interest.

Appendix

A. Code Repository

We have work on total of 4 repositories during our project life cycle. 3 repositories were used by each of us individually and 1 final repository was created as a place to perform ensemble and integration of all models together and do all the needed post-processing. Here the repositories as below:

1. Integrated Final repo:
https://github.com/sandeepsign/building_footprint_ensemble
2. Sandeep's repo:
https://github.com/sandeepsign/building_footprints_cs7643
3. Ahmed's repo:
https://github.com/abilal19/DL_FinalProject_Draft
4. Christian's repo:
<https://github.com/cswksu/aerialDetection>

B. Individual Contributions

Table 4. Individual team member contributions.

Contributor	Contribution
Ahmed	Full Model 2 [best performing], Project Report
Christian	Full Model 1, data preprocessing and visualization for models 1 and 2, Project Report
Sandeep	Full Model 3, ensembling, post processing, Geo Coding, VM Setup, Clustering of polygons, Project Report

C. Data Source

We have used INRIA's spacenet challenge data from:

<https://project.inria.fr/aerialimagelabeling/>

D. Train Infrastructure

We have used combination of techniques to execute this project.

Each had used individual hardware for setup and eventually to run long time training on google cloud VM instance.

GPUs Used are:

nVIDIA QUADRO RTX 5000 16GB

nVIDIA T4 16GB

nVIDIA GTX 1080ti 11GB

E. Enlarged Figures

Table 5. Class distribution of pixels in training set images.

Class	Number of Pixels	Percentage	Effective Number [$\beta = 1 - 10E-9$]
Building	7.1E8	1.58%	5.08E8
Not building	4.4E10	98.4%	1.00E9
Total	4.5E10	100%	1.00E9

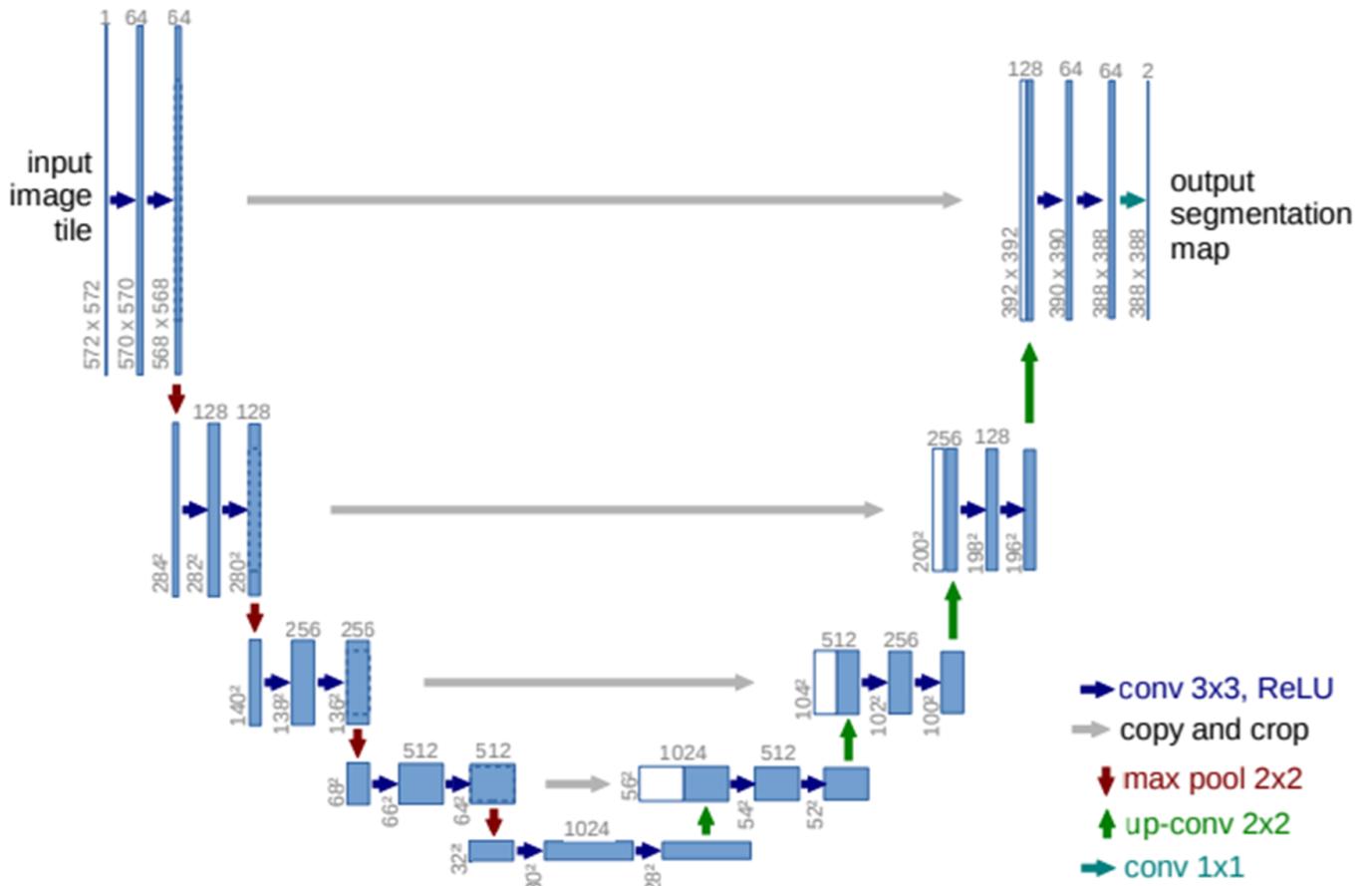


Figure 5. Example of overall organization of u-net model from [2].

F. Model 2 Results

Model 2 Validation Set Mask Samples

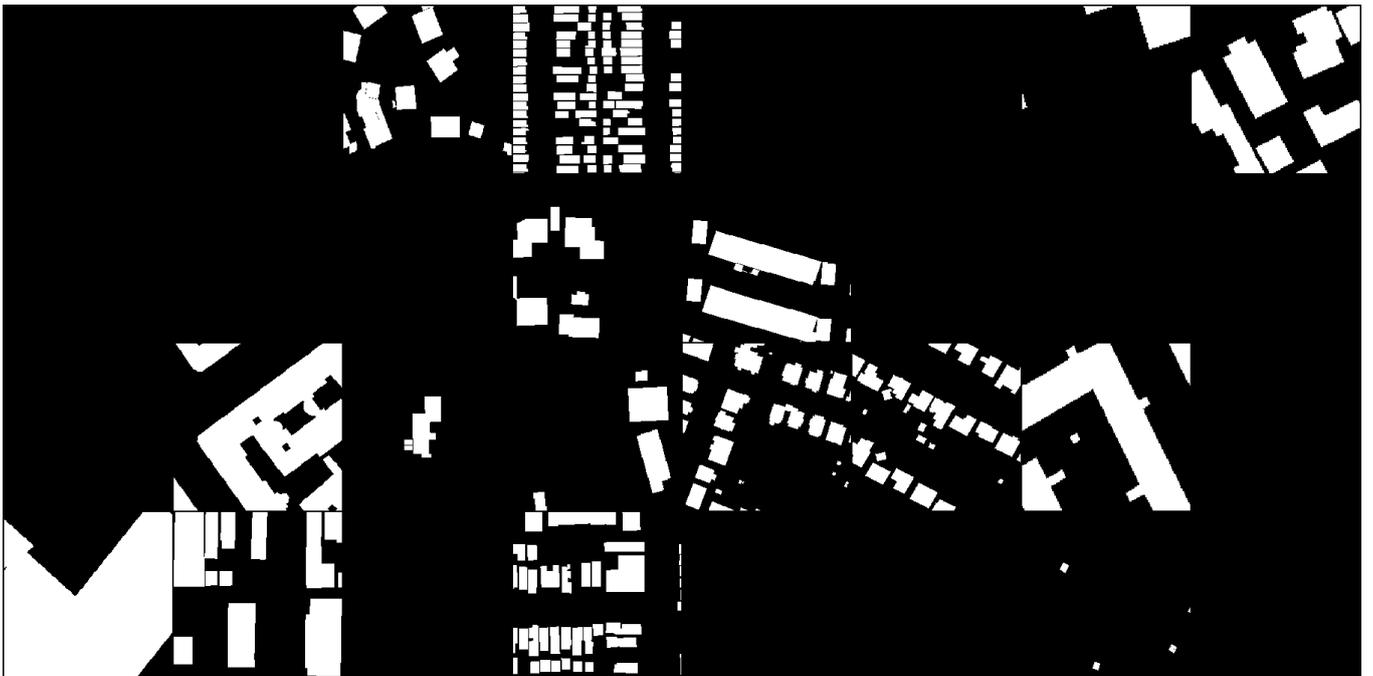


Figure 6. Validation Set Target.

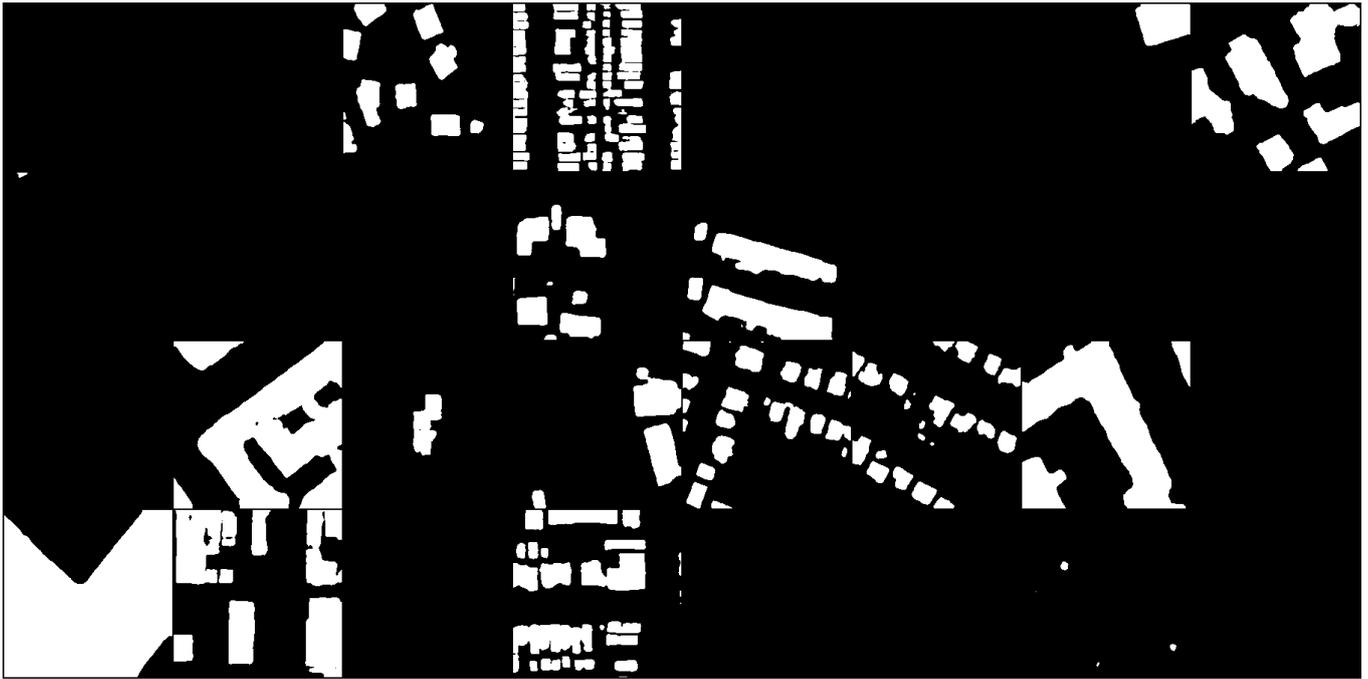


Figure 7. Validation Set Output.

Accuracy Results Per Epoch

accuracy - train
tag: accuracy - train

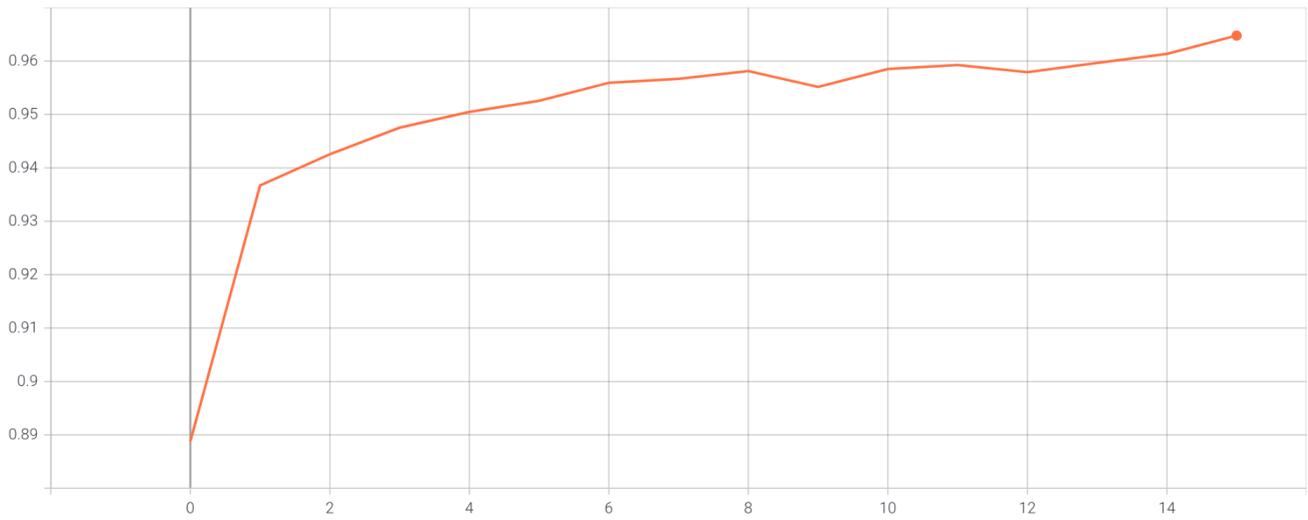


Figure 8. 'Model 2 Train Accuracy'.

accuracy - valid
tag: accuracy - valid

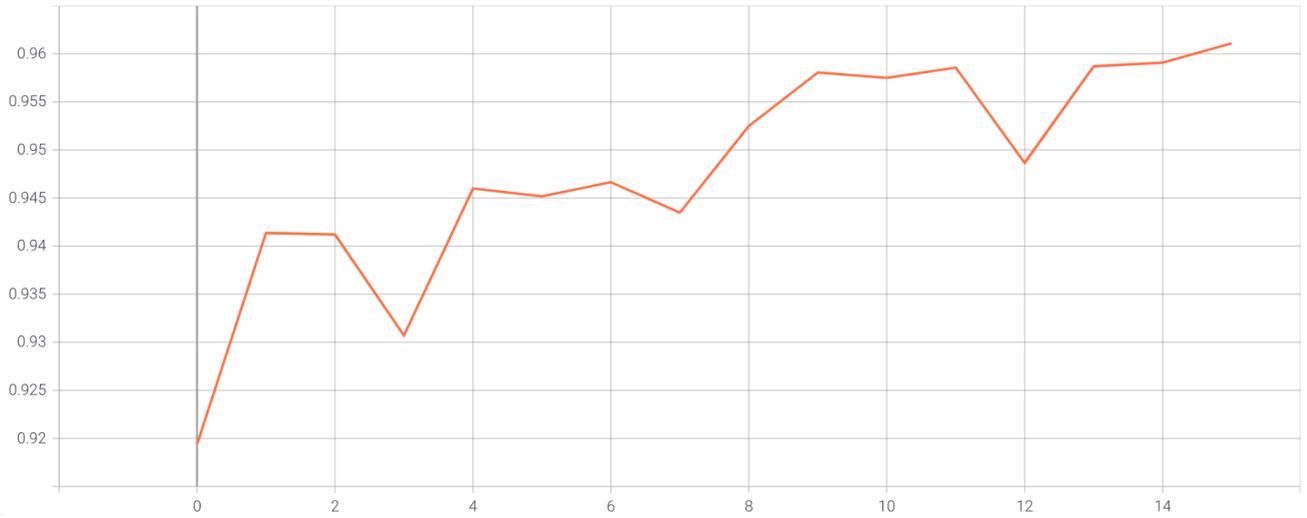


Figure 9. 'Model 2 Valid Accuracy'.

G. Model 3 Figures

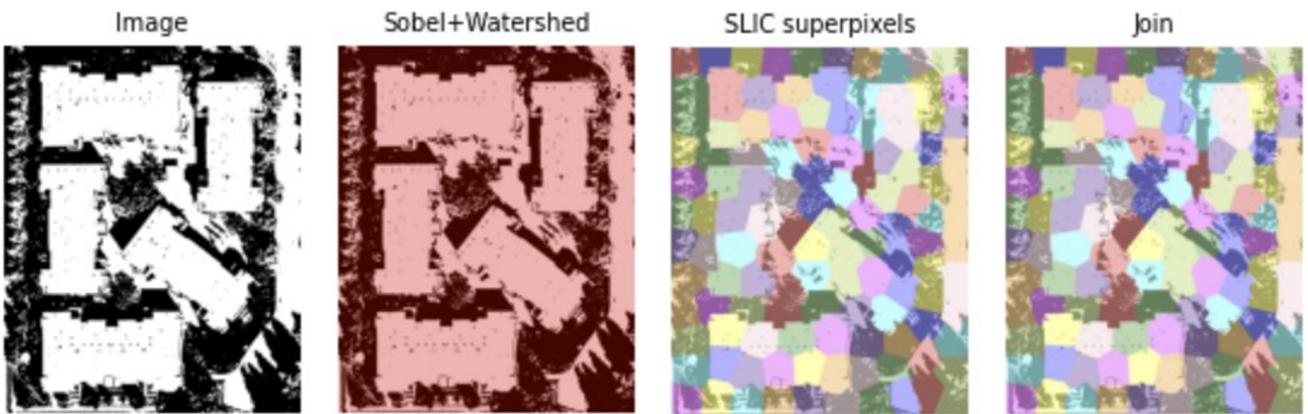


Figure 10. Post processing and merging of traditional segmentation techniques with Unet's mask and shape correcting polygons for buildings.

SuggestedLRs (valley=5.248074739938602e-05)

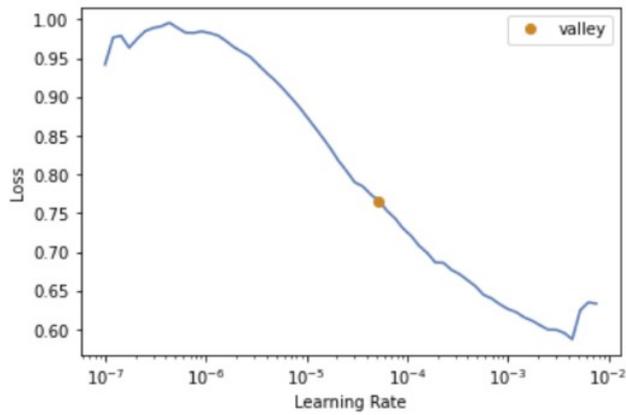


Figure 11. Model 3 learning rate finder.

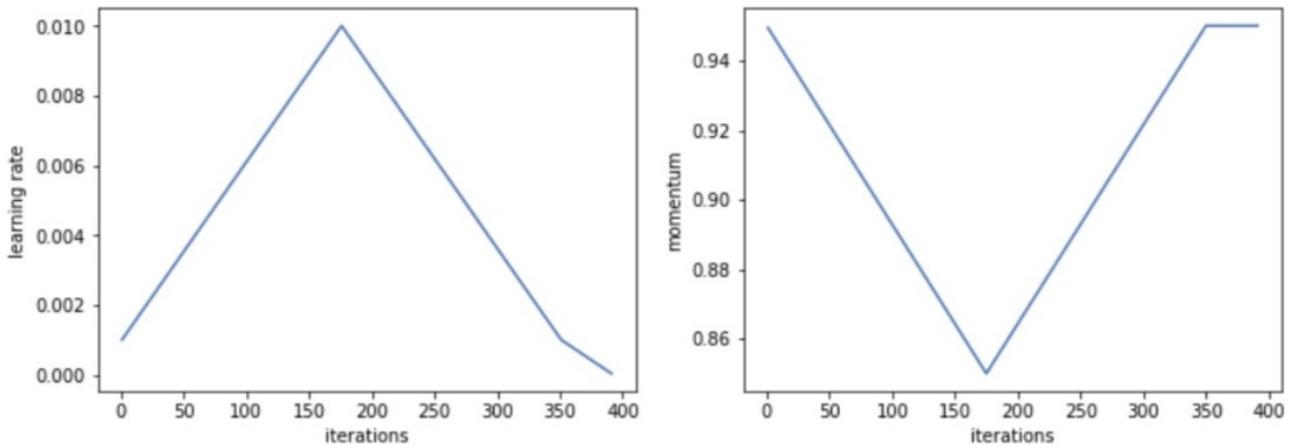


Figure 12. Model 3 fit one cycle scheduler: Momentum is moved opposite to learning rate during increase or annealing phase.

References

- [1] K. Zhao, J. Kang, J. Jung, and G. Sohn, "Building extraction from satellite images using mask r-cnn with building boundary regularization," in Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition Workshops, pp. 247-251, 2018.
- [2] O. Ronneberger, P. Fischer, and T. Brox, "U-net: Convolutional networks for biomedical image segmentation," 2015.
- [3] F. I. Diakogiannis, F. Waldner, P. Caccetta, and C. Wu, "Resunet-a: A deep learning framework for semantic segmentation of remotely sensed data," ISPRS Journal of Photogrammetry and Remote Sensing, vol. 162, p. 94-114, Apr 2020.
- [4] E. Maggiori, Y. Tarabalka, G. Charpiat, and P. Alliez, "Can semantic labeling methods generalize to any city? the inria aerial image labeling benchmark," in IEEE International Geoscience and Remote Sensing Symposium (IGARSS), IEEE, 2017.
- [5] J. Schmidt, "Creating and training a u-net model with pytorch for 2d & 3d semantic segmentation: Dataset...," May 2021.
- [6] S. Santurkar, D. Tsipras, A. Ilyas, and A. Madry, "How does batch normalization help optimization?" 2019.
- [7] Y. Cui, M. Jia, T.-Y. Lin, Y. Song, and S. Belongie, "Class-balanced loss based on effective number of samples," 2019.
- [8] J. H. Sylvain Gugger, "Adamw and superconvergence is now the fastest way to train neural nets," 2018.
- [9] D. P. Kingma and J. Ba, "Adam: A method for stochastic optimization," 2017.
- [10] E. Li, J. Femiani, S. Xu, X. Zhang, and P. Wonka, "Robust rooftop extraction from visible band images using higher order crf," IEEE Transactions on Geoscience and Remote Sensing, vol. 57, no. 8, pp. 5561-5575, 2019.
- [11] K. Bittner, F. Adam, S. Cui, M. Körner, and P. Reinartz, "Building footprint extraction from vhr remote sensing images combined with normalized dsms using fused fully convolutional networks," IEEE Journal of Selected Topics in Applied Earth Observations and Remote Sensing, vol. 11, no. 8, pp. 2615-2629, 2018.
- [12] A. G. Roy, N. Navab, and C. Wachinger, "Recalibrating fully convolutional networks with spatial and channel 'squeeze & excitation' blocks," CoRR, vol. abs/1808.08127, 2018.
- [13] Z. Zhang, Y. Wang, W. Gan, and X. Jia, "Refined unet: Unet-based refinement network for building extraction from optical remote sensing imagery," IEEE Geoscience and Remote Sensing Letters, vol. 18, no. 9, pp. 1620-1624, 2020.
- [14] J. Howard, "Progressive resizing for better generalization of the computer vision models," 2018.
- [15] A. Aitken, C. Ledig, L. Theis, J. Caballero, Z. Wang, and W. Shi, "Checkerboard artifact free sub-pixel convolution: A note on sub-pixel convolution, resize convolution and convolution resize," 2017.
- [16] Q. Chen, L. Wang, Y. Wu, G. Wu, Z. Guo, and S. L. Waslander, "Aerial imagery for roof segmentation: A large-scale dataset towards automatic mapping of buildings," ISPRS journal of photogrammetry and remote sensing, vol. 147, pp. 42-55, 2019.
- [17] W. Li, C. He, J. Fang, J. Zheng, H. Fu, and L. Yu, "Semantic segmentation-based building footprint extraction using very high-resolution satellite images and multi-source gis data," Remote Sensing, vol. 11, no. 4, p. 403, 2019.

- [18] L. N. Smith, "A disciplined approach to neural network hyper-parameters: Part 1 - learning rate, batch size, momentum, and weight decay," 2018.
- [19] Wikipedia, "Satellite imagery and challenges associated with them," 2018.