Generating 3D City Models Based on the Semantic Segmentation of LiDAR Data using Convolutional Neural Networks

Amgad Agoub[*], Valentina Schmidt and Martin Kada

Institute of Geodesy and Geoinformation Science (IGG), Technische Universität Berlin, Straße des 17. Juni 135, 10623 Berlin, Germany, firstname.lastname@tu-berlin.de

Commission VI, WG VI/10

KEY WORDS: Deep Learning, Semantic Segmentation, Building Footprint Extraction, 3D Building Reconstruction, CNN.

ABSTRACT:

Virtual city models are important for many applications such as urban planning, virtual and augmented reality, disaster management, and gaming. Urban features such as buildings, roads, and trees are essential components of these models and are subject to frequent change and alteration. It is laborious to manually build and update virtual city models, due to a large number of instances and temporal changes on such features. The increase of publicly available spatial data provides an important source for pipelines that automate virtual city model generation. The large quantity of data also opens an opportunity to use Deep Learning (DL) as a technique that minimizes the need for expert domain knowledge. In addition, many Deep Learning models calculations can be parallelized on modern hardware such as graphical processing units, which reduces the computation time substantially.
We explore the opportunity of using publicly available data in computing multiple thematic data layers from Digital Surface Models (DSMs) using an automatic pipeline that is powered by a semantic segmentation network. To evaluate this design, we implement our pipeline using multiple Convolutional Neural Networks (CNN) with an encoder-decoder architecture. We produce a variety of two and three-dimensional thematic data. We focus our evaluation on the pipeline's ability to produce accurate building footprints. In our experiments we vary the depths, the number of input channels and data resolutions of the evaluated networks. Our experiments process public data that is provided by New York City.

1. INTRODUCTION

Virtual 3D city models are used in many planning, analysis, simulation, and visualization applications in an urban context especially ones that relate to the environment, renewable energy, natural hazards, mobility (including navigation and autonomous driving), city marketing and cultural heritage (Biljecki et al., 2015). Buildings are one of the most prominent features in an urban scene. But not all applications require 3D building models in the same level of detail. For example, buildings models with differentiable roof structures are suitable for projects on the scale of city districts such as shadowing, mobile signal, and line-of-sight analysis (Kolbe et al., 2005). Within this paper, we focus on generating so-called block models of buildings with discretized roof structures as a geometric approximation of building roof shapes.

Large cities and metropolitan areas consist of an enormous number of objects that belong to a variety of thematic feature types such as buildings, trees, streets, etc. These objects can change over time as new construction and development projects happen on a frequent basis. Therefore, generating up-to-date 3D city models is laborious, time-consuming as well as expensive, and raises the need for automatic methods for generating such large area 3D models. A further necessity is the availability of an up-to-date data basis, e.g. 3D point cloud data. Fortunately, more and more geo data are being made open for public use and they are rich in geometric and thematic information.

On the methodological side, Deep Learning recently achieved human-level performance on many challenging computer vision tasks such as image classification (Krizhevsky et al., 2012), object detection (Liu et al., 2016; Redmon et al., 2016), instance segmentation (He et al., 2017), sequence-to-sequence translation (Sutskever et al., 2014) and data synthesis (Goodfellow et al., 2014). This recent comeback of neural networks can be credited by the abundance of data available for training and the increase in computing power. We, therefore, see an opportunity to design a processing pipeline using recent Deep Learning techniques that is able to generate large area 3D building representations. The parameters (i.e. weights) of the mentioned neural network models can be trained through an optimization process on publicly available data. Hence this approach minimizes the need for expert domain knowledge (e.g. deriving normals and designing morphological filters) required in approaches such as in (Nex et al., 2012; Poullis et al., 2009).

We explore this new technology by designing a simple pipeline that can generate multiple thematic data layers; such as buildings, trees, and roads; from digital surface models (DSMs). To evaluate our method, we implemented this pipeline with multiple Deep Learning semantic segmentation networks. We trained these networks on an open dataset of New York City and designed a data augmentation module specific for DSMs. Our evaluation of the neural networks' performances is focused on building footprints and includes neural network models of different depths (i.e. the number of layers), multiple inputs and output resolutions, and varied thematic output layers. To demonstrate the pipelines' potentials, we reconstructed the 3D city scene of the Manhattan area of interest (AOI) as shown in Figure 1.

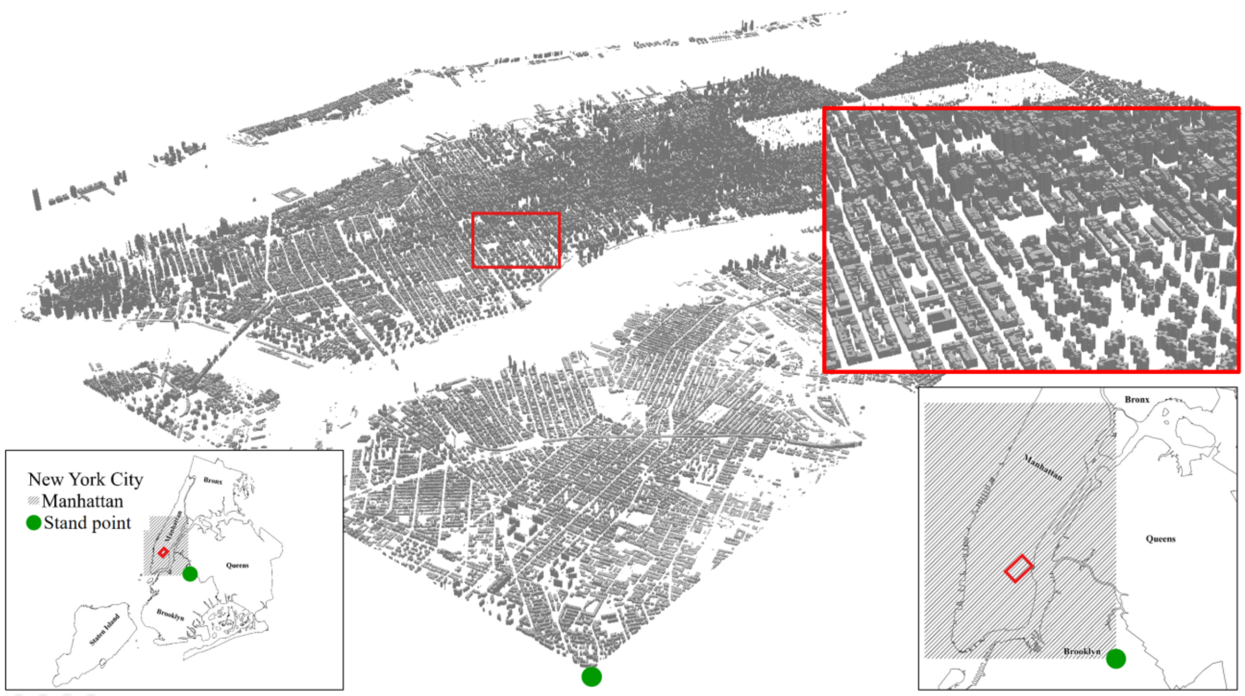


Figure 1. Results from our proposed 3D city reconstruction pipeline depicting the 3D city block models of the Manhattan area. The semantic segmentation was conducted with model denoted as b1_2_128 in Table 1.

2. RELATED WORK

Deep Learning techniques have achieved impressive results in solving object recognition problems such as object detection, semantic segmentation, and instance segmentation.

2.1 Object detection

Approaches that deal with object detection problem can be divided into two categories: region-based and single-pass. Predictions produced by region-based DL models usually go through two stages. In the first stage, the model identifies minimum bounding box proposals of possible objects in the input image or feature maps. In the second stage, the model evaluates the proposals with the help of a predicted confidence score (e.g. using a classifier) and further refines the bounding box to match object boundaries (e.g. using a regressor). R-CNN (Girshick, et al. 2014) is an example of such an approach, where a selective search algorithm (Uijlings, 2013) is used to produce a large number of object proposals (~2000). Each of these proposals is then inferred by a convolutional neural network (CNN) to produce a classification for each region. Compared to its predecessor R-CNN, Fast R-CNN (Girshick, 2015) reduces the model's overall prediction time by inferring each input image once then sharing the CNN calculations between all predicted regions. This improvement leaves the heuristic selective search algorithm to be the most time-consuming component in this approach. Faster R-CNN (Ren et al., 2015) replaces this component with a Region Proposal Network (RPN), which dramatically increases the overall speed.

Contrary to region-based approaches, single-pass approaches view the objection detection as a regression problem and not as a classification problem. Prediction tensors mimic a grid that divides the input image into sections. After training a single-pass network, it produces high score predictions for tensors that correspond to grid cells that contain objects and refinements to anchor box coordinates. Such anchor boxes are typically added to each grid cell to simulate common width to height ratio in the data samples. Examples for such approaches are Single Shot MultiBox Detector (SSD) (Liu et al., 2016) and You Only Look Once (YOLO) (Redmon et al., 2016).

2.2 Semantic segmentation

Object detection approaches are successful in determining a minimum bounding box around each object instance. However, precise pixel-wise object boundaries are also necessary for many applications. Many Deep Learning models are used in approaches that address the semantic segmentation task, e.g., fully convolutional networks (FCN) (Long, 2015), auto-encoders, and conditional generative adversarial networks (cGANs) (Isola, et al. 2017). Such networks are strong candidates to use for the task of building detection and outline generation. For example, the winning solution in the second SpaceNet challenge (Etten et al., 2018) for the task of automatic footprint detection from satellite images is based on U-Net (Ronneberger, 2015), an established semantic segmentation network.

2.3 Instance Segmentation

Semantic segmentation predictions require post-processing to calculate object instances from predicted segmentation maps. A combination of object detection and semantic segmentation can eliminate the need for many of these steps. For example, Mask R-CNN (He et al., 2017) is a network that adds an FCN segmentation branch to Faster R-CNN. This branch allows for pixel-wise segmentation for each detected object separately. Object boundaries produced by Mask R-CNN contain many points that follow the segmentation map pixel corners. Zhao et al. (2018) extend the Mask R-CNN model with a building boundary regularization technique to counter this issue. In another view of the building detection problem, Marcos et al. (2018) propose Deep Structured Active Contours (DSAC) and

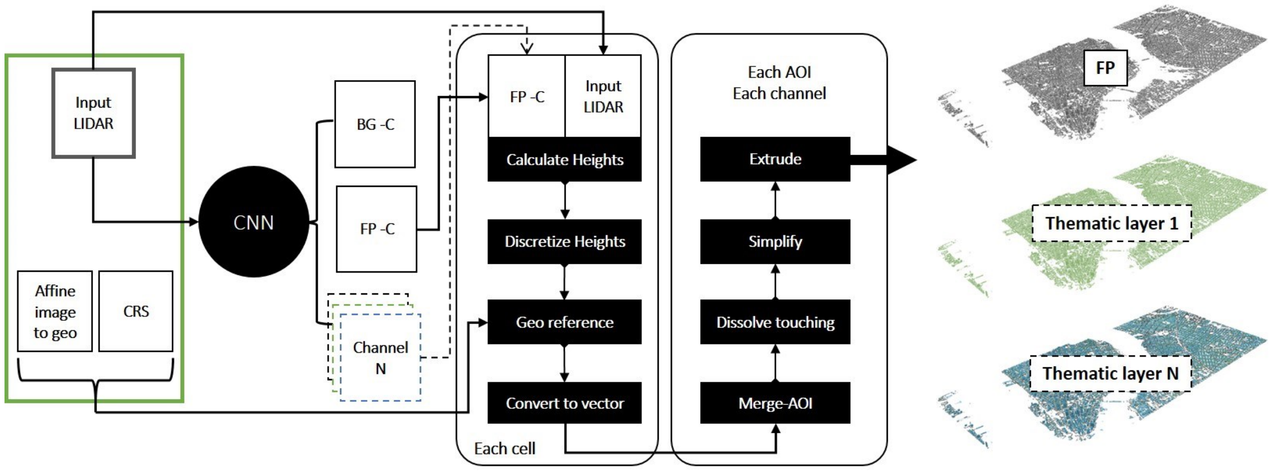


Figure 2. Proposed pipeline. BG Background channel. FP Footprint channel. CNN Convolutional Neural Network. The word vector refers to a specific data representation that is used in the context of geographical information systems (GIS).

calculate building boundaries from initial polygons with promising results.

3. METHODOLOGY

We propose a pipeline that uses a CNN to process DSMs and produces different thematic features. The complete design is shown in Figure 2.

3.1 CNN architecture

Mask R-CNN network can perform instance segmentation from image or raster data and is a strong candidate for this experiment. However, this network has a number of modules that add many unknowns to the experiment. For example, the network contains a Region Proposal Network (RPN), Region of interest (ROI) pooling and the complete architecture of Faster R-CNN. Increasing the number of unknowns might hinder the evaluation process. Therefore, as the core of our pipeline, we implement a simple encoder-decoder architecture that is inspired by (Ronneberger, 2015). However, to preserve locality, we decrease feature maps size using strided convolution rather than pooling.

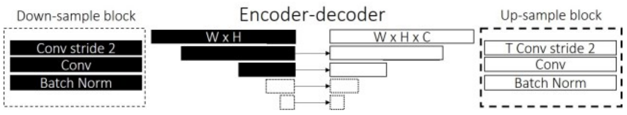


Figure 3. Encoder-decoder with mirrored architecture. Arrows depicted in the figure represent concatenation connections. The depth of the model changes according to the number of blocks. For each down-sample block, an up-sample block is added. W, H, C are the width, height, and the number of output channel respectively.

We argue that the encoder-decoder architecture represents a compromise between complexity and performance. In addition, the simplicity of the encoder-decoder architectures supports the incorporation of additional techniques such as in (Isola, et al. 2017; Wang, et al. 2018). The suggested encoder-decoder network shown in Figure 3 and consists of two mirrored reparative blocks (down-sampling and up-sampling) that are connected through concatenation. The use of concatenation allows the network to pass important information between feature maps at multiple scales regardless of the network depth. The number of the encoder-decoder output feature maps in our pipeline reflects the number of thematic data layers that were included in the training process explained in section 5. The minimum number of output channels in our experiments is two, corresponding to building footprints and background. The background feature map is a binary mask featuring non-objects locations set to 1.

3.2 Geodata inputs and pre-processing

The input size of semantic segmentation networks is usually fixed and determined as a design choice. Increasing the geographical extent of the input to cover an area of interest while preserving pixel count means a lower spatial resolution. Therefore, as an initial step, large areas of interest (AOIs) are divided using a regular grid as shown in Figure 9. Each grid cell span a small geographical area which increases the spatial resolution of each pixel.

CNN's predictions are typically represented as a multi-dimensional array of numeric values (images) which lacks geographical information. In order to enable spatial operations that are performed in the post-processing step on large AOIs, the pipeline stores an affine matrix for each input sample. This matrix represents projection parameters from local image space to a predefined coordinate system (CRS). And it is used to project the model’s output to the correct geographical context.

3.3 Post processing

In order to pair height information and model prediction, input depth maps from the DSM and predicated features are stacked vertically. The depth map is then masked by the predicted segmentation map. Depth map pixels that correspond to the value of ‘one’ prediction at the segmentation map keep their intensity values. All other pixels are given the value zero. Figure 4 shows that many points on building roofs tend to cluster within small height ranges. Using this observation, values that belong to a certain range (bin) are mapped to a specific integer as a rough approximation of building roof shapes. These bins are calculated empirically in our experiments.
The vector data representation is well supported in many geographical information systems (GIS). For example, simple feature access, a well-known standard, (Herring, 2006) defines

many geometries that can be used as a base to spatially model real-world features as vector data structures. Therefore, we convert the raster output of the model to a boundary model representation (vector) by using a 2D version of the marching cubes algorithm (Lorensen and Cline, 1987).

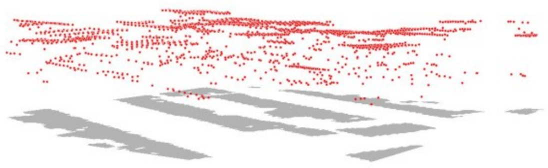


Figure 4. Input LiDAR data in red masked by the corresponding semantic segmentation network prediction in grey.

The marching cubes algorithm produces vector data that follows pixel edges and yields discretized vector data with a large number of boundary points. The boundary point count increases when increasing the input raster resolution. To counter this challenge, a simplification algorithm is used to eliminate the intermediate points (Douglas and Peucker, 1973). In order to acquire the 3D block city model, predicted building footprints are extruded to the individual height calculated in the previous steps.

4. EXPERIMENT

New York City offers a variety of open datasets that covers the entire city area. To label the data needed for our experiments, we prepare and process a LiDAR survey that contains approx. thirteen billion points, over a million building footprints, roadbeds, and street tree locations. We use these datasets in our experiments to train the semantic segmentation network and evaluate the suggested pipeline.

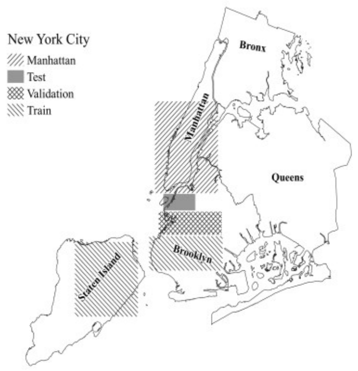


Figure 5. Districts of New York City and the experiment dataset geographical extent. Original data source (New York City Open Data Portal).

We divide the dataset into three non-overlapping locations that correspond to training, validation, and testing AOIs as shown in Figure 5. Each AOI is divided further using a grid. Each grid cell represents a data point (input raster). Training, validation, and testing AOIs has 6000, 1200, 420 cells respectively. Each cell spans a geographical area of 150x150 meters as shown in Figure 6. In addition, to experiment with models ability to generalize, we use a separate area 'Manhattan' as an additional test. This area is not included in the training or validation process, nor in the calculation of the evaluation of the results. The buildings of the Manhattan area are different from most of the buildings in the rest of NYC. The area contains dense building distribution, many skyscrapers, and large buildings. Therefore, we argue that a model's good prediction on this area gives a strong indicator of its ability to generalize on other geographical locations.

4.1 Training data

To leverage the use of well-established 2D CNNs techniques, we convert the input LiDAR data to a raster format (DSM). LiDAR data that corresponds to the cell location is converted to a depth map by an inverse weighted interpolation process. Each pixel is given the weighted average height value of the nearest three LiDAR points to the two-dimensional location of the pixel's center. This depth map is used as the input to the semantic segmentation network. For each of the thematic features used in the training process, we prepare a binary (label) mask as shown in Figure 6. Pixels that correspond to feature locations are given the value one while all other pixels are given the value zero.

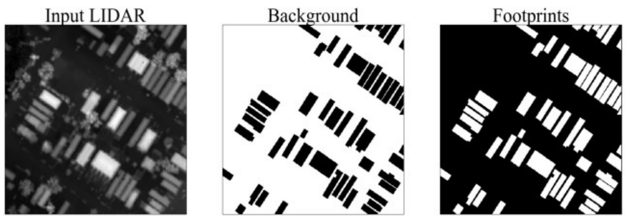


Figure 6. Example of one training data sample. The geographical extent of the input and output raster corresponds to the extent of each grid cells introduced in Section 4.

5. TRAINING

5.1 Network selection

We experiment with multiple encoder-decoder configurations that follow the design mentioned in subsection 3.1. In addition, we include a test on a well-known semantic segmentation network U-Net as demonstrated in Table 1 as a baseline. For the purpose of this experiment each model is trained on the training AOI and validated on the validation AOI. The used hyperparameters are the same for all tested models. We use Adam optimizer (Kingma & Ba, 2014) with a learning rate of 10^{-3}.

5.2 Loss function

In our experiments, we vary the CNN output labels. An example is shown in Figure 7.

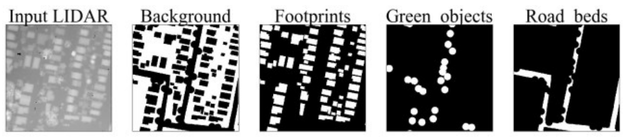
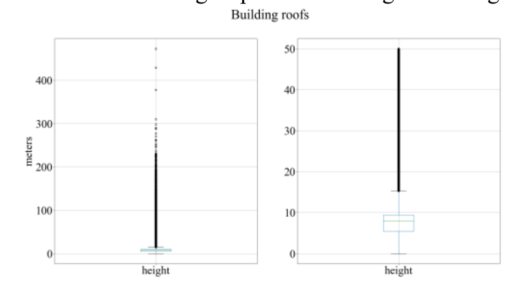


Figure 7. Class imbalance. The number of background pixels is much higher than those of corresponding thematic features. This imbalance increases when the image resolution is higher (sparsity problem).

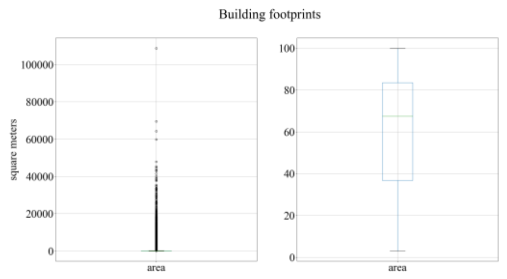
The number of pixels in the output is unbalanced. Weights are calculated through a spatial analysis to collect the ratio of pixel count imbalance on each resolution. For example, we estimate that the number of pixels representing trees in the output channel is one to ten compared to the background channel on a data sample with 128 by 128 resolution. For this reason, we chose weighted categorical-cross-entropy as the loss function for this experiment.

5.3 Challenges

The geographical extent of the training AOI spans Brooklyn and Staten Island. Brooklyn has a grid-like building footprints pattern. While the eastern side Staten Island building footprints have a combination of a grid pattern and single standing houses.



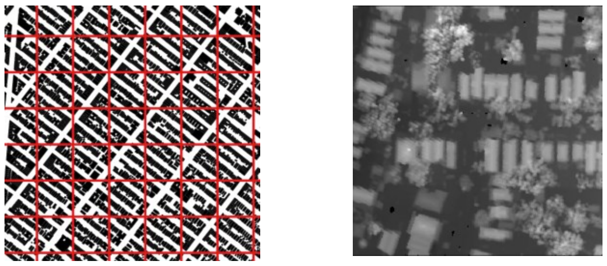
(a)



(b)

Figure 8. (a) left-hand side plot shows a boxplot of roof heights of all buildings in NYC. (a) right-hand side plot shows a boxplot of roof heights without including outliers. (b) left-hand side plot shows a boxplot of footprint areas of all buildings in NYC. (b) right-hand side plot shows a boxplot of footprint areas without outliers.

While some variation exists in the training datasets, both these locations have building footprints and heights that differ from the test and Manhattan datasets. Figure 8 shows a box plot of the height and area ranges of all buildings in the New York City dataset. According to these plots, most of the buildings have heights under 10 meters and most of the building footprints have an area between 50 and 125 meters.



(a) (b)

Figure 9. (a) shows the building footprints grid pattern in Brooklyn overlaid with the training cell grid. (b) shows LiDAR values on one data sample where height values for all buildings are almost identical.

In addition, due to the grid-like pattern planning of Brooklyn area, buildings of the same size and height follow the same orientation yielding a training dataset with a low degree of variation. See Figure 9. There is a risk of overfitting when training a Deep Learning model on such a dataset that has strong similarity between its samples. In this case, the model might not generalize when making predictions on new datasets. For example, our initial experiments showed that skyscrapers were not recognized as buildings by a CNN trained solely on the training AOI. Hence producing a city model without many of the recognizable landmarks. To counter this issue, we implemented a specialized data augmentation module.

5.4 Data augmentation

During training, random values are added on each input depth map for each building footprint location. This produces depth maps with exaggerated height differences between buildings. Hence, increasing the building roof height variation of the training dataset. Results of this process on a normalized data sample are shown in Figure 10.

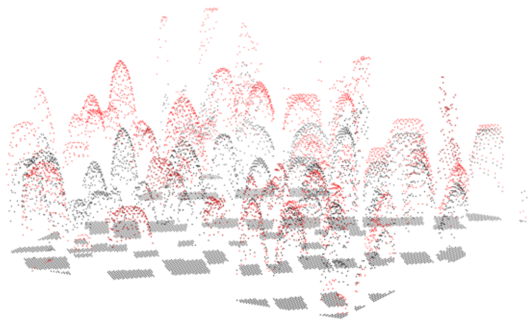


Figure 10. Comparisons between original and augmented data both examples are normalized. Each point's coordinates are the pixel locations on a 2D image elevated to the intensity value at that pixel. Grey 3D points represent the normalized values and the red points represent the augmented data. Both are masked by building footprints feature map for demonstration purposes.

To simulate different footprint sizes, we use multiple factors to scale each training and validation sample. Additionally, we add a random skewing and mirroring that is applied to both horizontal axes of the input and output as well as scaling. The complete process is shown in Figure 11.

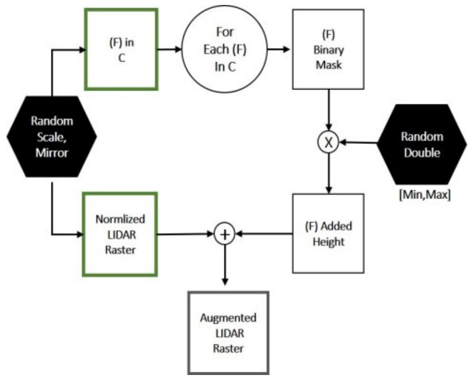


Figure 11. Data augmentation module applied on each cell for each building footprint F that is located with this cell C.

5.5 Early stopping condition

Each of the models in our experiments was trained with an early-stopping condition which is a common technique used when training CNNs. We believe that this condition is a good measure of the model's ability to generalize since our data augmentation module produces novel data samples on each iteration. The value of the loss function is calculated both on training and validation AOI. The training is stopped when the validation loss value stagnates or drops compared to previous values or training loss value with a patience value of three epochs. Only the model with the minimum loss on validation is selected for further tests.

5.6 Segmentation networks results

The semantic segmentation CNN produces multiple feature maps of floating point number predictions. An equality test of pixel intensity values is necessary in order to group pixels and identify feature instances using the marching squares algorithm and floating point numbers are, therefore, not suitable for this procedure. To avoid this problem, the predictions in the feature maps are converted to binary masks. The feature map with the highest confidence score at each pixel location is given the value of one. The remaining feature maps are given the value of zero at that location. Figure 12 shows some example results.

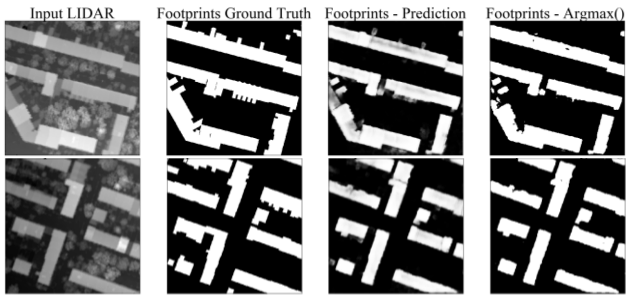


Figure 12. Semantic segmentation network prediction using model b5_2_512 in Table 1. The binary conversion process is expressed in the figure as Argmax().

6. EVALUATION

Due to the class imbalance between background and building footprints, per pixel accuracy is not a sufficient metric to evaluate the overall model performance. For example, positive prediction for all pixels within a channel that contains a small building yields a high accuracy score. We use the F1 score as the main evaluation metric (Chinchor, 1992) for this experiment. The F1 score is used as an evaluation metric in the SpaceNet (Etten et al., 2018) challenge and International Society for Photogrammetry and Remote Sensing (ISPRS) benchmark on object classification and 3D building reconstruction (Rottensteiner et al., 2012).
We follow a similar evaluation approach to the one suggested by (Etten et al., 2018). In order to calculate this score, we made the following assumptions: We evaluate the vector data that resulted from the proposed pipeline. Building footprints are considered to be correctly detected if they exceed a pre-determined threshold to the Jaccard distance (Jaccard, 1908) or Intersection Over Union (IOU) compared to the ground truth data. We set the IOU threshold to 0.5 following the suggested value by Etten et al. (2018) and if the ratio of building footprint prediction intersection area with the ground truth area is larger than (0.5) then the footprint is considered to be detected correctly and one is added to the true positive count (TP). False Negative (FN) count is increased by one if a ground truth sample had no intersection with the prediction or in case the IOU value of this sample is less than 0.5 as shown in Figure 13. False Positive (FP) count is increased by one if predictions have no intersection with the ground truth.
The ground truth data differentiate between building footprint instances not only based on building geometry but also on cadastre information. Information that relates to cadastre ownership cannot be derived from the LiDAR data alone. Hence using ground truth data in this way will yield many FN even if the segmentation prediction where correct. Therefore, in this experiment, building footprints with touching geometries are dissolved in the ground truth and prediction data.

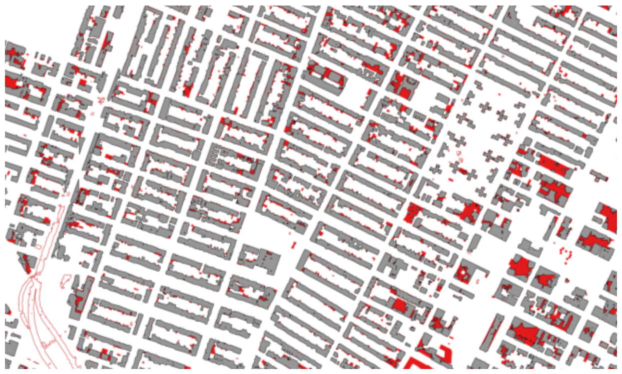


Figure 13. Pipeline predictions are depicted in grey and ground truth in red. FP outlines are shown in red using model b1_2_128 in Table 1.

6.1 Discussion and limitation

Table 1 shows the results of our experiment. Most models resulted in a high TP score.

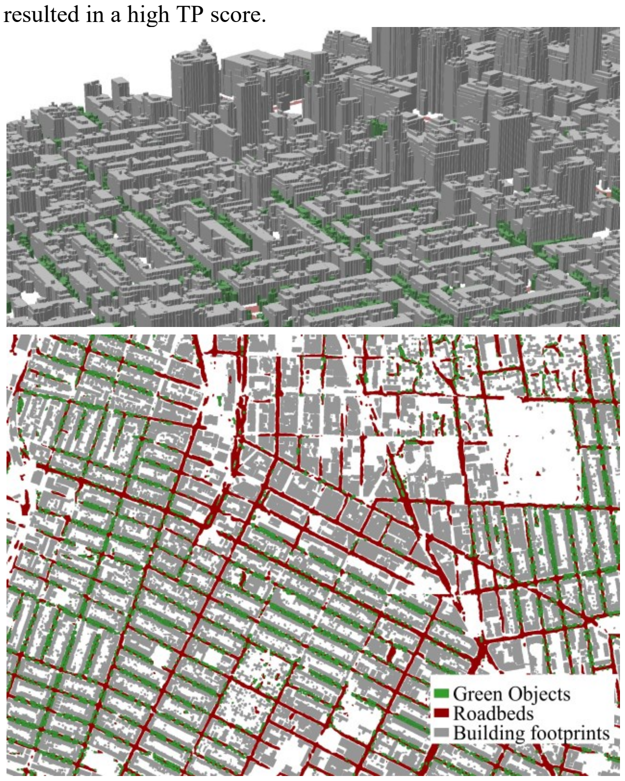


Figure 14. Shows a 2D map of thematic features predicted using model b3_4_256 with the extruded predictions.

The best performing models are the ones with the highest input resolution and depth and are able to achieve a low FP score. Adding labels that indicate the location of green objects was expected to reduce the number of FP predictions. However, our experiment showed that this addition decreased the models'

Model	#layers	#params	TP	FP	FN	Precision	Recall	F1 score
b1_2_128	21	86,626	1204	353	752	0.77	0.62	**0.69**
b1_4_128	21	87,204	1227	1166	729	0.51	0.63	0.56
b2_2_128	31	136,226	1182	2970	774	0.28	0.60	0.39
b2_4_128	31	136,804	1211	1419	745	0.46	0.62	0.53
b3_2_128	41	185,826	1218	402	738	0.75	0.62	0.68
b3_4_128	41	186,404	1268	1271	688	0.50	0.65	0.56
u_2_128	49	1,943,714	1042	290	914	0.78	0.53	0.63
u_4_128	49	1,943,748	1259	853	697	0.60	0.64	0.62
b1_2_256	21	86,626	1284	678	672	0.65	0.66	0.66
b1_4_256	21	87,204	1220	898	736	0.58	0.62	0.60
b2_2_256	31	136,226	1287	560	669	0.70	0.66	0.68
b2_4_256	31	136,804	1366	893	590	0.60	0.70	0.65
b3_2_256	41	185,826	1340	662	616	0.67	0.69	0.68
b3_4_256	41	186,404	1311	505	645	0.72	0.67	**0.70**
b4_2_256	51	235,426	1343	850	613	0.61	0.69	0.65
b4_4_256	51	236,004	1325	646	631	0.67	0.68	0.67
u_2_256	49	1,943,714	1201	336	755	0.78	0.61	0.69
u_4_256	49	1,943,748	1329	1212	627	0.52	0.68	0.59
b1_2_512	21	86,626	1232	792	724	0.61	0.63	0.62
b2_2_512	31	136,226	1372	457	584	0.75	0.70	0.72
b3_2_512	41	185,826	1364	329	592	0.81	0.70	0.75
b4_2_512	51	235,426	1278	319	678	0.80	0.65	0.72
b5_2_512	61	285,026	1339	208	617	0.87	0.68	**0.76**
u_2_512	49	1,943,714	1240	654	716	0.65	0.63	0.64

Table 1. Evaluation results. Models that have the name starting with the letter b are implemented to follow the design suggested in subsection 3.1. Model names are arranged in this way b{#blocks}_{#label channels}_{input shape}. Models that have the name starting with the letter u are implemented according to U-net proposed in (Ronneberger, 2015). Model names are arranged in this way u_{# label channels}_{input shape}. For all implementations, if #label channels is two, then included label channels are building footprints and background; if three, green objects channel is included and if four, roadbeds channel is included.

performances and binary classification models are with the highest performance.

Adding roadbed labels, on the other hand, improved the model's performance compared to when adding only green object labels. The green object training data only depicts a buffer around the centroid location street trees; Many trees are missing in the dataset. Therefore, this is not sufficient to draw a conclusion about the correlation between adding tree labels and the poor performance of a CNN.

Many hyperparameters and thresholds were used in our experiment. These decisions might have a large impact on the outcome of the network and the pipeline. Hence changing these thresholds might yield different evaluation metrics than conducted in the experiment. These parameters are: in the pre-processing step the number of neighboring points, input image size, spatial resolution; in the CNN training step the number of convolutional layers and filters, activation functions, weight initialization, and batch normalization frequency, stop condition, data augmentation random ranges and loss function; in the post-processing step the number of minimum pixels forming a polygon, the minimum area of an accepted building.
Another limitation is that predictions in our experiment were calculated on small regions (cells) then merged. Buildings or other objects of interest that are located on the boundaries of these cells might be divided into smaller parts and get ignored due to certain thresholds in the post-processing steps.

The post-processing step is sensitive to the outliers in the input LiDAR data. Any erroneous continuous batch will be detected as a building and our visual inspection showed that many elevated structures such as bridges were falsely predicted by the CNN as buildings.

To demonstrate the pipeline's ability to parallelize computations we select use CNN b3_4_256 referenced in Table 1 to compute green objects, roads in addition to building footprints as predictions from the input DSM. The same post-processing pipeline is used on green objects and building footprints predictions to acquire the 3D extruded representation. Results are shown in Figure 14.

7. CONCLUSION AND OUTLOOK

This paper suggests a pipeline for generating block-like city models from depth maps. For testing purposes, the pipeline was implemented with encoder-decoder networks at its core and evaluated on a dataset that spans the geographical extent of New York City. Even with a comparably small dataset count, we were able to train this model from scratch purely on the mentioned data. However, although the pipeline shows good results in detecting accurate building footprints, it includes many post-processing steps in order to detect individual building instances. Further research aims toward end-to-end Deep Learning 3D block city model generation by substituting the post-processing steps with DL modules in a similar approach to (Gkioxari et al., 2017) while preserving the simplicity of the pipeline.

8. ACKNOWLEDGEMENT

We are grateful to the city of New York for providing all data necessary for this research through the New York City Open

Data Portal (https://opendata.cityofnewyork.us) and to the U.S. Geological Survey (USGS) for providing LiDAR point clouds of New York.

REFERENCES

Biljecki, F., Stoter, J., Ledoux, H., Zlatanova, S. and Çöltekin, A., 2015. Applications of 3D city models: State of the art review. *ISPRS International Journal of Geo-Information,* 4(4), pp.2842-2889.

Douglas, D.H. and Peucker, T.K., 1973. Algorithms for the reduction of the number of points required to represent a digitized line or its caricature. *Cartographica: the international journal for geographic information and geovisualization*, 10(2), pp.112-122.

Van Etten, A., Lindenbaum, D. and Bacastow, T.M., 2018. Spacenet: *A remote sensing dataset and challenge series. arXiv preprint arXiv:1807.01232*.

Girshick, R., 2015. Fast r-cnn. *In Proceedings of the IEEE international conference on computer vision* (pp. 1440-1448).

Goodfellow, I., Pouget-Abadie, J., Mirza, M., Xu, B., Warde-Farley, D., Ozair, S., Courville, A., & Bengio, Y. (2014). Generative adversarial nets. *Advances in neural information processing systems (pp. 2672-2680).*

He, K., Gkioxari, G., Dollár, P. and Girshick, R., 2017. Mask r-cnn. *In Proceedings of the IEEE international conference on computer vision* (pp. 2961-2969).

Herring, J.R., 2006. OpenGIS implementation specification for geographic information-Simple feature access-Part 1: Common architecture. *Open Geospatial Consortium,* p.95.

Isola, P., Zhu, J.Y., Zhou, T. and Efros, A.A., 2017. Image-to-image translation with conditional adversarial networks. *In Proceedings of the IEEE conference on computer vision and pattern recognition* (pp. 1125-1134).

Jaccard, P., 1908. Nouvelles recherches sur la distribution florale. Bull. *Soc. Vaud. Sci. Nat., 44, pp.223-270.*

Kingma, D.P. and Ba, J., 2014. Adam: A method for stochastic optimization. *arXiv preprint arXiv:1412.6980.*

Krizhevsky, A., Sutskever, I., & Hinton, G. E. (2012). Imagenet classification with deep convolutional neural networks. *Advances in neural information processing systems* (pp. 1097-1105).

Liu, W., Anguelov, D., Erhan, D., Szegedy, C., Reed, S., Fu, C.Y. and Berg, A.C., 2016. Ssd: Single shot multibox detector. *In European conference on computer vision* (pp. 21-37). Springer, Cham.

Long, J., Shelhamer, E. and Darrell, T., 2015. Fully convolutional networks for semantic segmentation. *In Proceedings of the IEEE conference on computer vision and pattern recognition* (pp. 3431-3440).

Lorensen, W.E. and Cline, H.E., 1987, August. Marching cubes: A high resolution 3D surface construction algorithm. *In ACM siggraph computer graphics* (Vol. 21, No. 4, pp. 163-169). ACM.

Marcos, D., Tuia, D., Kellenberger, B., Zhang, L., Bai, M., Liao, R. and Urtasun, R., 2018. Learning deep structured active contours end-to-end. *In Proceedings of the IEEE Conference on* Computer Vision and Pattern Recognition (pp. 8877-8885).

Nancy Chinchor, MUC-4 Evaluation Metrics*, in Proc. of the Fourth Message Understanding Conference,* pp. 22–29, 1992.

Girshick, R., Donahue, J., Darrell, T. and Malik, J., 2014. Rich feature hierarchies for accurate object detection and semantic segmentation. *In Proceedings of the IEEE conference on computer vision and pattern recognition* (pp. 580-587).

Nex, F., & Remondino, F. (2012). Automatic roof outlines reconstruction from photogrammetric DSM. *ISPRS Annals of the Photogrammetry, Remote Sensing and Spatial Information Sciences*, 1(3), 257-262.

Poullis, C., & You, S. (2009). Automatic reconstruction of cities from remote sensor data. *In 2009 IEEE conference on computer vision and pattern recognition* (pp. 2775-2782). IEEE.

Redmon, J., Divvala, S., Girshick, R. and Farhadi, A., 2016. You only look once: Unified, real-time object detection. *In Proceedings of the IEEE conference on computer vision and pattern recognition* (pp. 779-788).

Ren, S., He, K., Girshick, R. and Sun, J., 2015. Faster r-cnn: Towards real-time object detection with region proposal networks. *In Advances in neural information processing systems* (pp. 91-99).

Ronneberger, O., Fischer, P. and Brox, T., 2015. U-net: Convolutional networks for biomedical image segmentation. *In International Conference on Medical image computing and computer-assisted intervention (pp. 234-241). Springer, Cham.*

Rottensteiner, F., Sohn, G., Jung, J., Gerke, M., Baillard, C., Benitez, S. and Breitkopf, U., 2012. *The ISPRS benchmark on urban object classification and 3D building reconstruction. ISPRS Ann. Photogramm. Remote Sens. Spat.* Inf. Sci, 1(3), pp.293-298.

Sutskever, I., Vinyals, O., & Le, Q. V. (2014). Sequence to sequence learning with neural networks. *Advances in neural information processing systems* (pp. 3104-3112).

Uijlings, J.R., Van De Sande, K.E., Gevers, T. and Smeulders, A.W., 2013. Selective search for object recognition. *International journal of computer vision,* 104(2), pp.154-171.

Wang, T.C., Liu, M.Y., Zhu, J.Y., Tao, A., Kautz, J. and Catanzaro, B., 2018. High-resolution image synthesis and semantic manipulation with conditional gans. *In Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (pp. 8798-8807).*

Zhao, K., Kang, J., Jung, J. and Sohn, G., 2018, June. Building Extraction from Satellite Images Using Mask R-CNN with Building Boundary Regularization. *In 2018 IEEE/CVF Conference on Computer Vision and Pattern Recognition Workshops (CVPRW)* (pp. 242-2424). IEEE.