UNIVERSITY OF CALIFORNIA

Los Angeles

Application of Deep Neural Network and 3D Data for Seismic Risk Mitigation

A thesis submitted in partial satisfaction
of the requirements for the degree
Master of Science in Statistics

by

Peng-Yu Chen

2021

ABSTRACT OF THE THESIS

Application of Deep Neural Network and 3D Data for Seismic Risk Mitigation

by

Peng-Yu Chen
Master of Science in Statistics
University of California, Los Angeles, 2021
Professor Ertugrul Taciroglu, Co-Chair
Professor Ying Nian Wu, Co-Chair

Seismically vulnerable, especially collapse-prone, buildings often represent the greatest life-safety hazard worldwide. Identifying these buildings is the first step in seismic risk mitigation efforts for a given urban region's resilience. This thesis aims to devise a workflow for the application of state-of-the-art deep neural networks (DNNs) for detecting and classifying seismically vulnerable buildings using three-dimensional point clouds. A number of prior studies have focused on using 2D image data in the field of structural health monitoring for damage recognition. The performance of those approaches for building classification at regional scales is highly dependent on well-controlled imagery data and may not be guaranteed when applied to real-world data. The present study, therefore, differs from prior studies in that it uses 3D point clouds, which implicitly contain depth information that can become a highly useful feature for training DNNs. Here, a specific DNN, namely, PointNet, is used for detecting soft-story buildings, which are ubiquitous in west-coast cities of the US. A workflow is devised for binary classification of point clouds into soft-story and non-soft-story points as well as the segmentation and association of classified point clouds with specific addresses. Over 10 billion point clouds obtained from the city of Santa Monica in California are manually labeled and split into training, validation, and testing sets, and the sensitive ranges of DNN hyperparameters are investigated to obtain the good performance.

The thesis of Peng-Yu Chen is approved.

Chad Hazlett

Ying Nian Wu, Committee Co-Chair

Ertugrul Taciroglu, Committee Co-Chair

University of California, Los Angeles

2021

To my parents . . .

who—among so many other things—

encouraged me discover science

since my childhood

TABLE OF CONTENTS

1 Introduction **1**

1.1 Background and Motivation 1

1.2 Prior Studies: Convolutional Neural Networks 3

1.3 Prior Studies: CNN based on 3D data 5

1.4 Objectives 6

2 Proposed Workflow **8**

2.1 Data Preparation 8

2.2 3D Deep Learning: PointNet 14

2.3 Locations Identification: Clustering Methods 18

2.4 Chapter Summary 20

3 Results Analysis and Discussion **21**

3.1 Performance of PointNet and Hyperparameter Tuning 21

3.2 Evaluation of Clustering Methods 23

3.3 Chapter Summary 25

4 Conclusions and Suggestions of Future Work **26**

4.1 Conclusions 26

4.2 Potential Impacts 27

4.3 Suggestions of Future Work 27

References **29**

LIST OF FIGURES

1.1 Typical soft-story buildings. 2

1.2 Soft-story detection with (a) non-blocked, (b) partially blocked, and (c) fully blocked images. (adapted from [40, 41]) 3

1.3 Scheme of the proposed workflow. 7

2.1 Scheme of the proposed workflow. 9

2.2 Selection of target regions. 11

2.3 Heuristic labeling flowchart. 12

2.4 Semantic3D segmentation: (a) example[13], and (b) segmented result in selected region. 13

2.5 Segmentation using building polygons. 14

2.6 An example of soft-story building points. 14

2.7 PointNet Architecture: (a) full structure; (b) T-Net (adapted from [28]). 17

2.8 Implementation of PointNet structure. 18

2.9 PointNet prediction: soft-story points. 18

3.1 Variation in performance with (a) block size, (b) stride, and (c) number of points in a block. 22

3.2 Clustering points to buildings. 23

3.3 Evaluation of the predicted addresses. 24

LIST OF TABLES

2.1 Statistics of point clouds. 9

3.1 Performance of PointNet on test set. 23

3.2 Performance of clustering on test set. 24

ACKNOWLEDGMENTS

I would like to acknowledge and thank all the professors I have been taking classes and discussed with at Statistics department of UCLA who not only delivers splendid lectures but also introduces deep insights, Prof.Chad Hazlett, Prof.Jingyi Li, and my thesis committee chair Prof. Ying Nian Wu for his patience and guidance. I would also like to appreciate my respectable advisor in Civil Engineering department, Prof. Ertugrul Taciroglu who greatly supports my study in the articulated M.S. degree in Statistics.

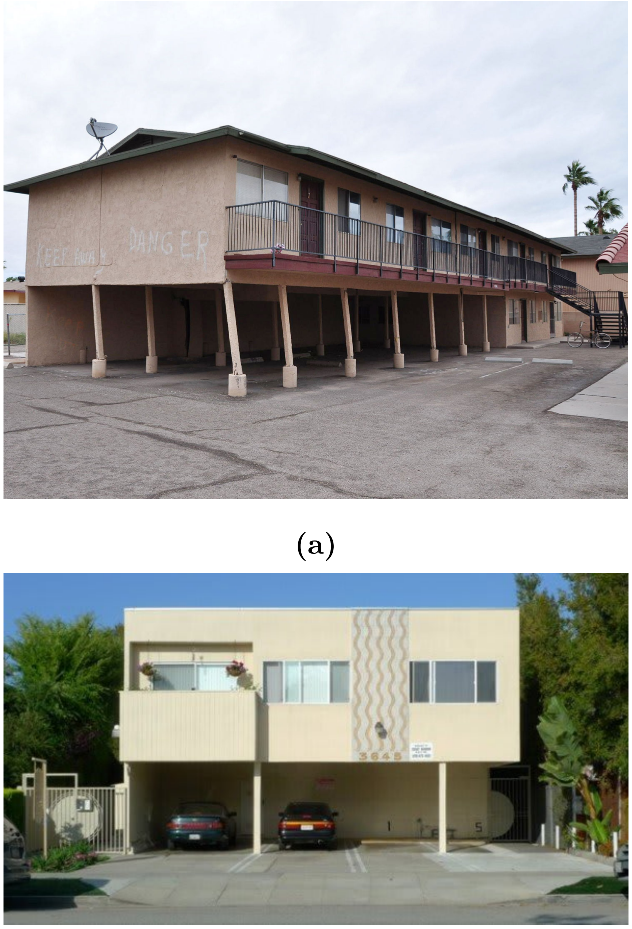
CHAPTER 1

Introduction

1.1 Background and Motivation

The vulnerability of soft-story buildings has been identified in past earthquakes including the 1971 San Fernando, 1989 Loma Prieta, and 1994 Northridge earthquakes. A typical weakness of soft-story buildings is the open-space configuration on lower story, which is usually designated for parking or commercial space. A lower wall density is designed in these spaces to improve accessibility, leading to inconsistent structural stiffness and hence called a "soft-story" building (shown in Fig1.1). The existence of these seismically vulnerable buildings may affects the losses (e.g., repair, replacement), post-earthquake occupancy rates, economic recovery, and even the number of fatalities during an earthquake, all of which are important for quantifing seismic resilience [3].

To enhance seismic resilience, many cities have established mandatory policies to retrofit soft-story buildings to mitigate their seismic risk (San Francisco, 2013; Los Angeles, 2015; Santa Monica, 2017). Burton et al.[4] evaluated the seismic performance of soft-story buildings in Los Angeles (LA) and compared the collapse capacity of existing buildings and retrofitted buildings following the LA ordinance. The average collapse capacity of archetype soft-story buildings was 50% after the retrofit. A follow-up study done by Kang et al.[17] showed that the LA ordinance led to a 64% reduction in the time required to recover occupancy. These studies demonstrate the importance of identification and retrofitting of seismically vulnerable buildings at a city scale.



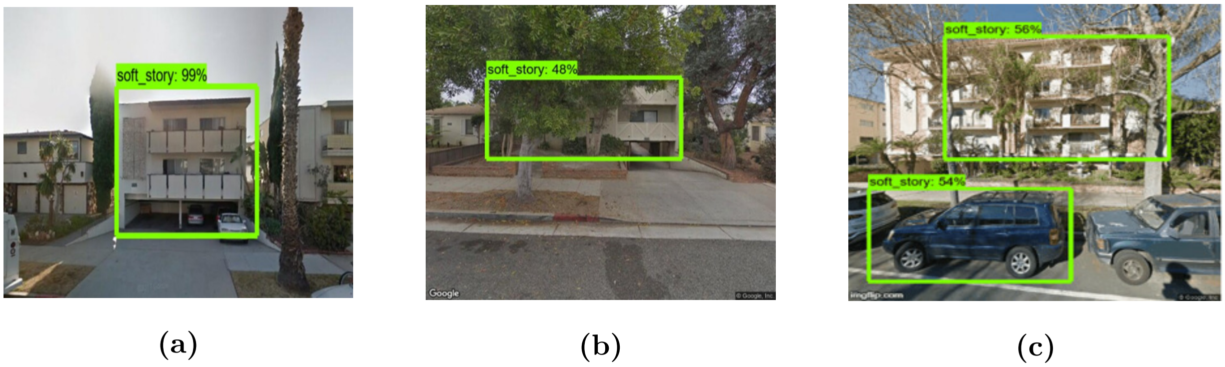
(a)

(b)

Figure 1.1. Typical soft-story buildings.

Although the nominal demand to identify soft-story buildings continues to increase, the process remains laborious and time consuming. The first step is the screening process to identify buildings of concern. Licensed professional engineers then conduct onsite inspections and file a report to the government authorities. This process is cost- and time-intensive for both the residents and the government. An automated process for the identification of soft-story buildings is hence indispensable.

The open space at the lower floor is an obvious feature that can be used to rapidly identify soft-story buildings. Because it is easy to get imagery data through public-access map services (i.e., Google street map), many studies hence tried to develop automatic methods that use these street-view images to identify soft-story buildings and reduce the labor-intensive site investigation. In recent years, automatic detection has been greatly implemented due to the development of artificial intelligence (AI) and deep learning (DL) in computer vision technology. For example, recent studies have applied DL models to identify soft-story buildings in street-view images (Wu et al.[40, 41]; Yu et al.[42]). These studies, in which DL models were trained on Google Street View images from several cities, reported promising results if the target building was not occluded by objects such as trees and cars. The detection confidence and accuracy decrease the more the buildings are blocked by objects, as shown in Fig1.2. To overcome this limitation of street-view image-based methods, 3D point-cloud data can be used to capture the soft-story features of buildings that are blocked or do not face the street.



(a) **(b)** **(c)**

Figure 1.2. Soft-story detection with (a) non-blocked, (b) partially blocked, and (c) fully blocked images. (adapted from [40, 41])

1.2 Prior Studies: Convolutional Neural Networks

Convolutional neural networks (CNNs) are at the core of recent DL development. Distinguished from traditional computer vision and machine learning (ML) techniques, CNNs have the ability to capture millions of parameters learned from imagery data (pixel matrix) and to avoid selection of features. Many well-designed CNN architectures, such as VGGNet [34], GoogLeNet [37], and ResNet [15] have demonstrated their dominance in image-recognition tasks. VGGNet was the winner of ImageNet[9] in 2014. There are two versions, VGG16 and VGG19, which consist of 16 and 19 hidden layers, respectively. The numbers of convolutional blocks and fully connected layers are implemented within the hidden layers. The VGGNet filters are all 3 × 3, and approximately 138 million parameters are trained in VGGNet. Inception net was named after the movie "Inception", which includes the line "we need to go deeper." While Inception has 27 layers, it only contains 6.4 million trainable parameters due to its extensive use of 1 × 1 filters, which limits the number of input channels. Inception dominates the classification task in ImageNet after VGGNet. ResNet was developed to address the vanishing gradient problem in back-propagation, which is triggered when deep CNN architecture is used. With the introduction of residual blocks with implemented identification functions, it is easier to find the derivative in the gradient descent method. However, this again increases the computational cost. For example, 60.3 million parameters are trainable in ResNet with 152 residual blocks.

In civil engineering, several studies have applied CNN-based models at detection and localization of structural damage. For example, Zhang et al. [44, 43] developed a CNN-based pixel-level crack detection model, CrackNet, which outperforms traditional ML methods for detecting cracks on asphalt pavement. Alipour et al. [2] proposed a CNN-based model, CrackPix, to detect crack damage based on patch-level images, which allowed random-size input images. Cha et al. [5] designed a CNN architecture to detect crack damage in concrete structures, and their model was less-sensitive to real-world situations (e.g., lighting, shadow changes). In addition to the implementation in imagery data, Kumar et al. [21] proposed a DL-based object-detection model for automated defect classification and localization in a closed-circuit television video of sewers. Three object-detection models, SSD [24], YOLOv3 [30], and Faster R-CNN [31] were evaluated for accuracy and speed in defect detection. The Faster R-CNN model demonstrated the highest accuracy in sewer inspection but required the most computational time.

While above-mentioned CNN models dominate many tasks in image classification, object detection, and semantic segmentation, the training of them from scratch is time-consuming and requires large amounts of data. To address this issue, transfer learning (TL) was developed to improve the training process and performance of CNN-based models (Pan et al.,[26]). In TL, knowledge learned from well-trained models on a base data set is transferred to a new model for retraining or fine-tuning on a smaller data set and task. Gao et al.[12] implemented TL for vision-based damage recognition at the pixel, component, and structural levels. In their study, the VGGNet model was retrained and fine-tuned for multi-label structural-damage classification tasks. [20] established a region-based fully convolutional network to detect construction equipment for site management, where the ResNet model was implemented as the base model in TL. Kalfarisi et al.[16] developed two DL-based approaches for crack detection and segmentation. The first approach integrated Faster R-CNN with structured random forest edge detection[10], whereas the second approach applied Mask R-CNN[14] with pretrained CNN models. Their results showed that Mask R-CNN with Inception-ResNet-V2[36] network architecture obtained higher crack detection and segmentation performance. In addition to component damage recognition and detection, some studies have implemented DL techniques for classification and detection at the regional scale. Kang et al.[18] trained VGGNet, GoogLeNet, and ResNet based on street-view images to classify building land use, and Srivastava et al.[35] used CNN models to classify land use based on aerial and street-view images.

1.3 Prior Studies: CNN based on 3D data

Compared to studies using CNN for 2D data, limited researches have been done in civil engineering for 3D data. For example, three-dimensional point-cloud data have been recently studied to reconstruct infrastructural models for facility management [8, 6], and post-event damage evaluation [19, 45]. Unlike the dominance of 2D CNN models, deep CNNs based on point clouds still have room to to be improved. One main limitation is the sparsity of point cloud data. An unordered characteristic and a highly variable point density make it difficult to be implemented.

Most often approaches to address this issue including:

- Transfer 3D point clouds into 2D images, and apply state-of-the-art CNN models [33, 39]. However, this approach results in loosing 3D information of objects. For example, the relative location of objects is highly depending on the perspective of where images are taken.
- Convert point-cloud data to volumetric forms like voxel grids and generate images [11, 25]. Nevertheless, the computational and memory costs will cubically grow with the resolution of voxels.
- Another method to detect 3D objects is inferring 3D bounding boxes directly from 2D images [7], yet the accuracy of 3D detection based on 2D images is unreliable.

To overcome these disadvantages, a 3D CNN structural that directly deals with point-cloud data, PointNet [28], has recently been developed. With a simple architecture, PointNet can process raw point clouds without transforming them into other forms, work in real-time, and provide promising results for 3D object classification and semantic segmentation. It is hence selected in this study to identify seismically vulnerable buildings.

1.4 Objectives

In this study, a workflow is proposed aiming to use PointNet to identify and segment soft-story buildings at a city scale. Besides, the segmented point-cloud sets are converted into geolocations so that real-world addressed can be obtained for further investigation. The overall scheme of this study is illustrated in Fig 1.3. The contributions of this study are improving the limitations of using 2D street-view images for identifying soft-story buildings, and the first attempting of using point-cloud data for regional segmentation and classification.

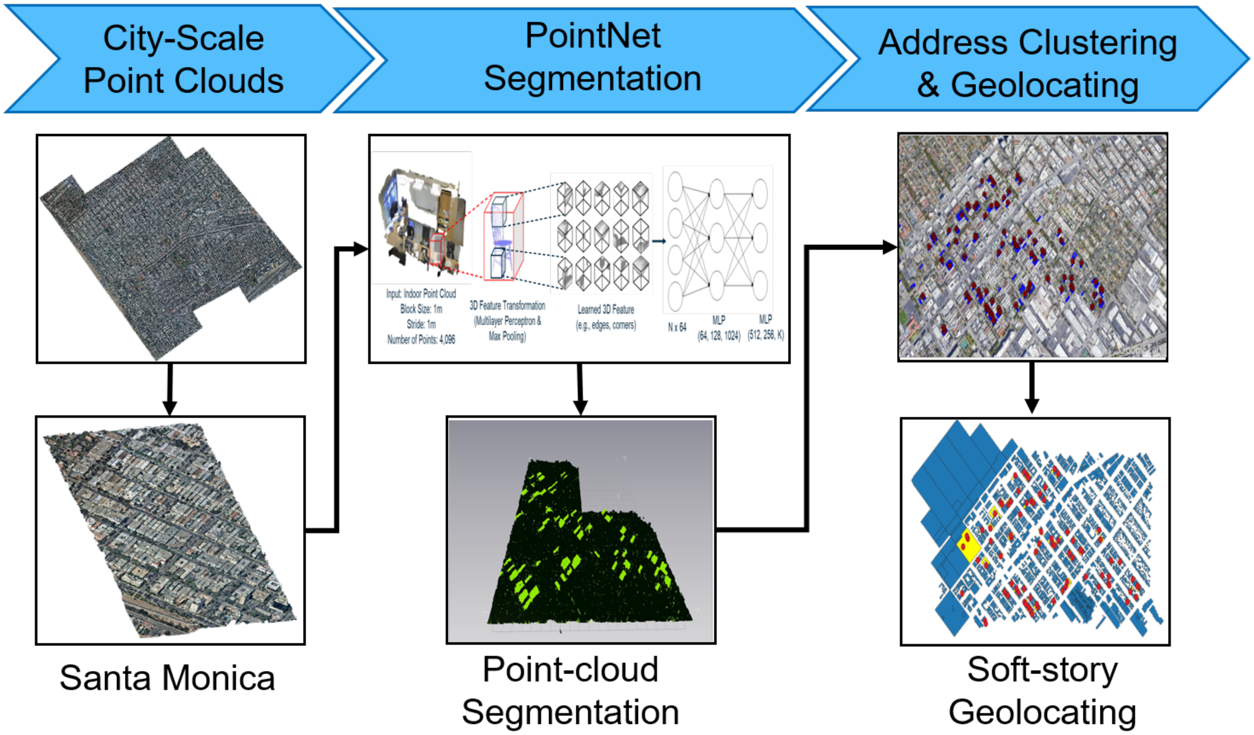


Figure 1.3. Scheme of the proposed workflow.

CHAPTER 2

Proposed Workflow

In this chapter, a city-scale point-cloud data with ground-truth labeling is first elaborated and followed by a detailed introduction of the 3D deep learning model, PointNet. The training, validation, and testing process are then described. Clustering methods used to predict building point clouds are then discussed. The final ingredient of the proposed workflow, namely, the geolocating address of soft-story building is explained at the end.

2.1 Data Preparation

The city of Santa Monica in southern California is selected as the target city in this study. It has been studied by many researchers [40, 41, 42] because the city government has done many works to identify seismically vulnerable buildings and retrofit them since 2017. A publicly accessible geographic information system (GIS) database (`https://gis-smgov.opendata.arcgis.com/app/seismic-retrofit-map`) that records all addresses is hence usually used as the ground-truth location. In this study, the ground-truth address and the building footprint (i.e., geometric boundary of the building property) are used to label point-cloud data into two classes, namely soft-story and non-soft-story points.

The point-cloud data used in this study was generated through the photogrammetry due to its affordability. In photogrammetry, evenly distributed points on images that depict an object were used to estimate the camera position with known and georeferenced 3D coordinates [38], and the output is a set of point clouds with coordinates and appearance textures (e.g., R, G, B). For Santa Monica, 1.1 billion point-cloud data points were generated, covering 18.16 km^2 at a density of 60 points/m^2. An example of the generated point clods is shown in Fig 2.1, and the statistical data is shown in Table 2.1

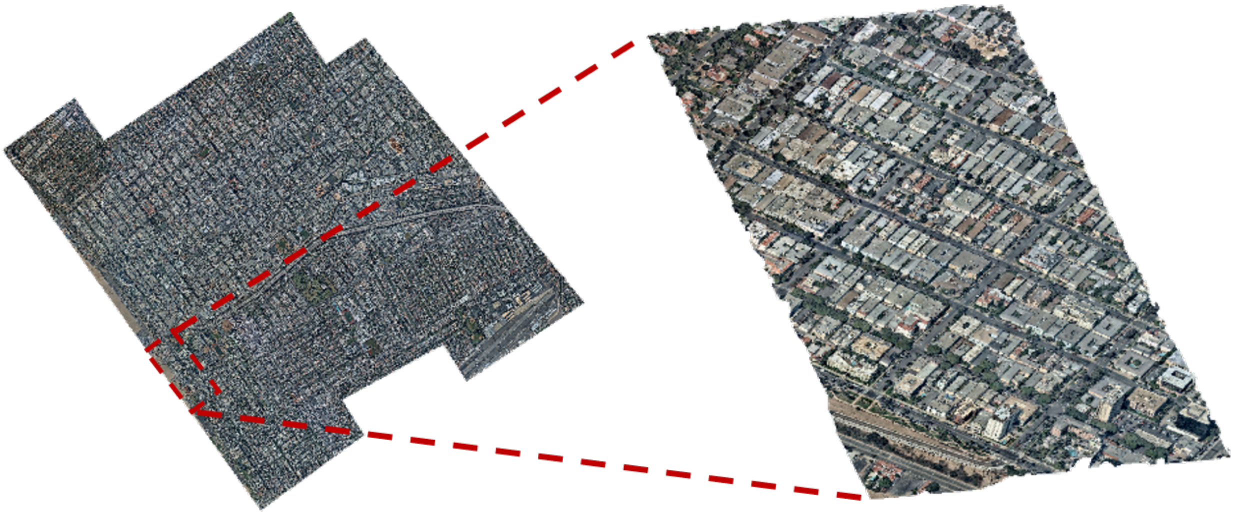


Figure 2.1. Scheme of the proposed workflow.

Table 2.1. Statistics of point clouds.

Soft-story points	51,377,865
Non-soft-story points	1,056,267,016
Total points	1,107,644,881
Number of soft-story buildings	1,453

As the statistical data shows, the number of soft-story points takes less than 5% of total points. If these data is directly used to train a deep learning model, the overfitting will occur and the prediction will be very unreliable. Hence, small sets of regions are selected to narrow down the problem scale. As shown in Fig 2.2, 13 regions (covered by red polygons) including 1048 soft-story buildings are selected, which has around 3.25 km^2 and is later split into training, validation, and testing sets for the PointNet. Meanwhile, to reduce the labor-intensive work of labeling, a heuristic algorithm is developed to collect and label all soft-story points as shown in Fig 2.3.

In the heuristic algorithm, all points in a selected regions are first classified into either building or non-building class. The training set for this classifier is different from the target region of the City of Santa Monica. It comes from a large scale out-door point-cloud data set, namely, Semasntic3D.Net (`http://semantic3d.net/`)[13], which consists of over four billion manually labelled points. The objective of Semantic3D was to build a large-scale data set similar to ImageNet[9] and specifically for point-cloud data.

In their study, CNN-based model used voxel grids to classify point cloud into 8 categories including: (1) man-made terrain: mostly pavement; (2) natural terrain: mostly grass; (3) high vegetation: trees and large bushes; (4) low vegetation: flowers or small bushes which are smaller than 2m; (5) buildings; (6) remaining hard scape: a clutter class with for instance garden walls, fountains, banks; (7) scanning artifacts: artifacts caused by dynamically moving objects during the recording of the static scan; (8) cars and trucks. An example of Semantic3D.Net is shown in Fig 2.4(a). Their model was trained for a binary classification task, namely, building or not. The heuristic algorithm goes through each selected region and collect points that belong to buildings as shown in Fig 2.4(b).

Once all building points are collected, the algorithm then checks whether the building points locate in soft-story polygon or not. The ground-truth polygons obtained from Santa Monica GIS database contain geolocations (i.e., latitude, longitude) of building boundaries which can be used to check if those building points are within boundaries. If the point is within a polygon, it will be segmented and labeled as soft-story point (as shown in Fig 2.5). An example of a segmented soft-story building is shown in Fig 2.6. The total number of soft-story points is 7,835,315, and the number of non-soft-story is 13,512,918. This data will be further discussed in the next section for training the 3D deep learning model.

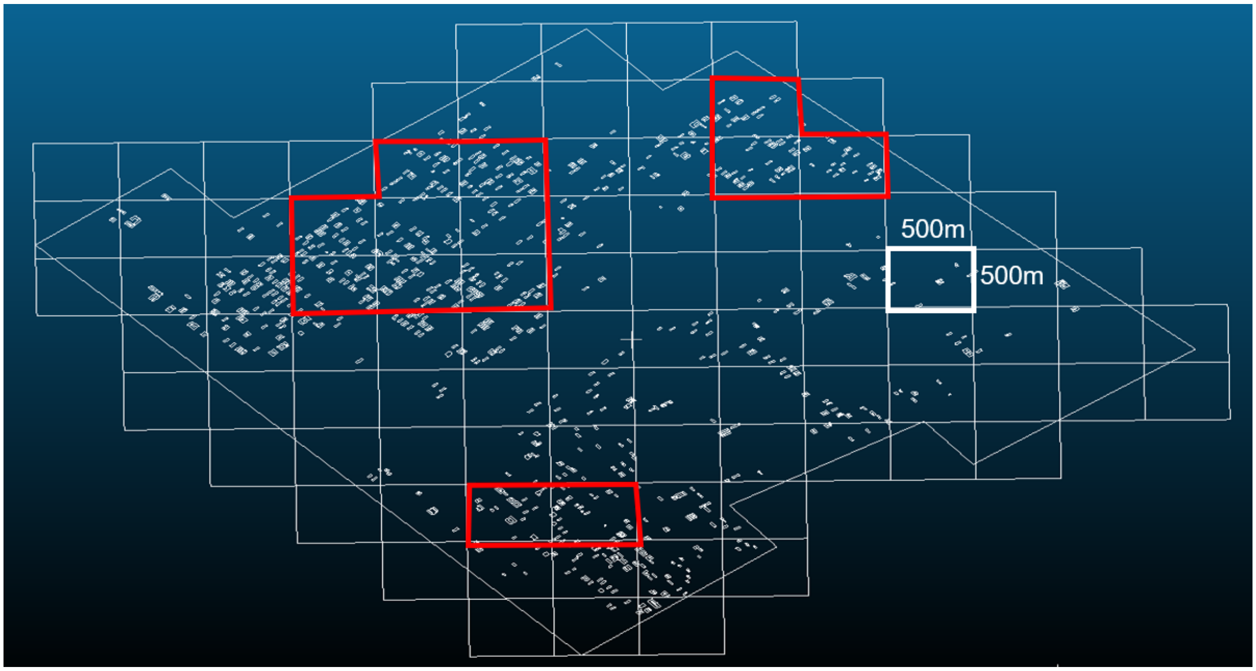


Figure 2.2. Selection of target regions.

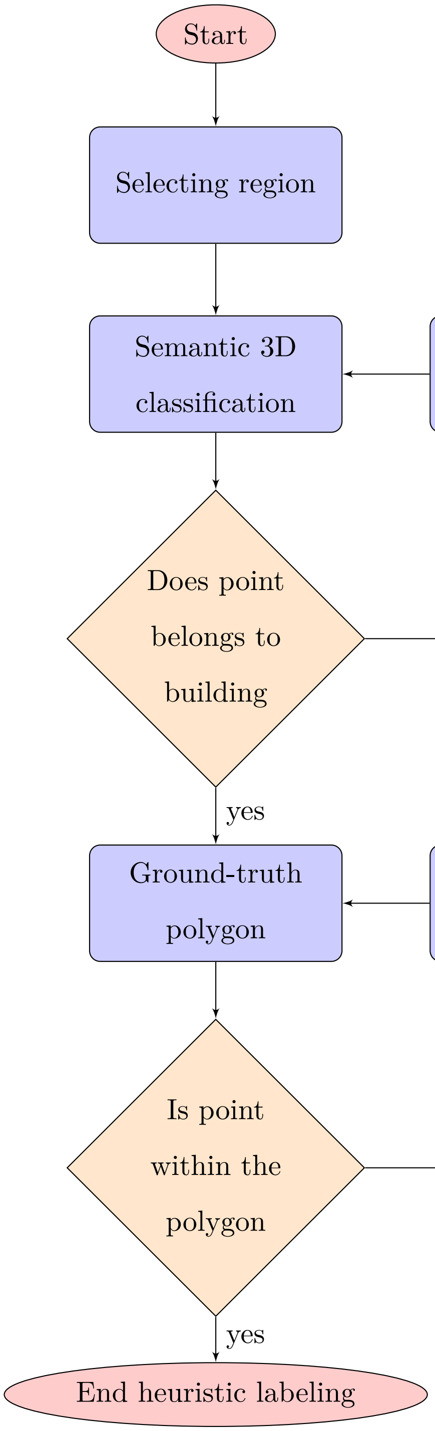
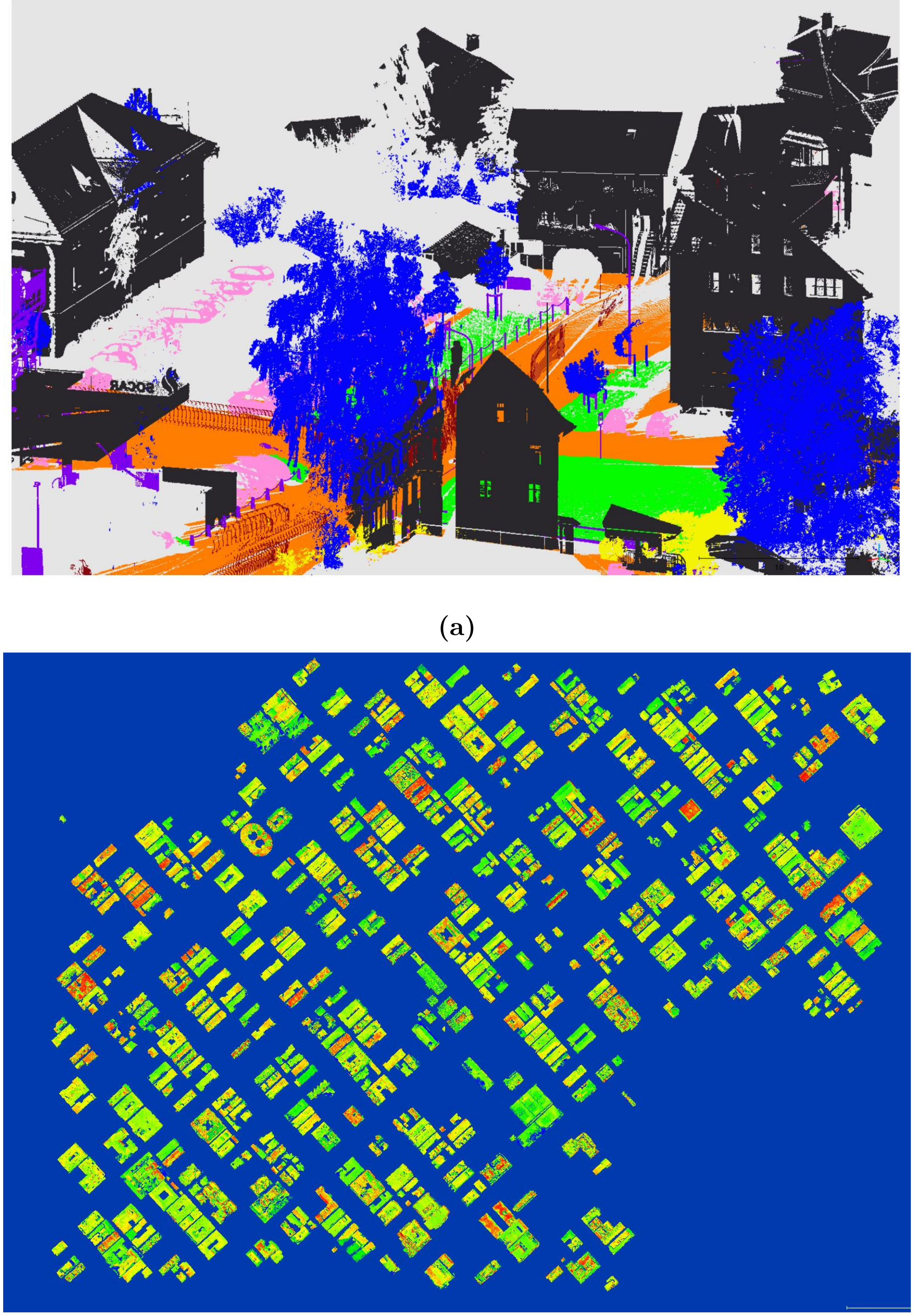


Figure 2.3. Heuristic labeling flowchart.



(a)

(b)

Figure 2.4. Semantic3D segmentation: (a) example[13], and (b) segmented result in selected region.

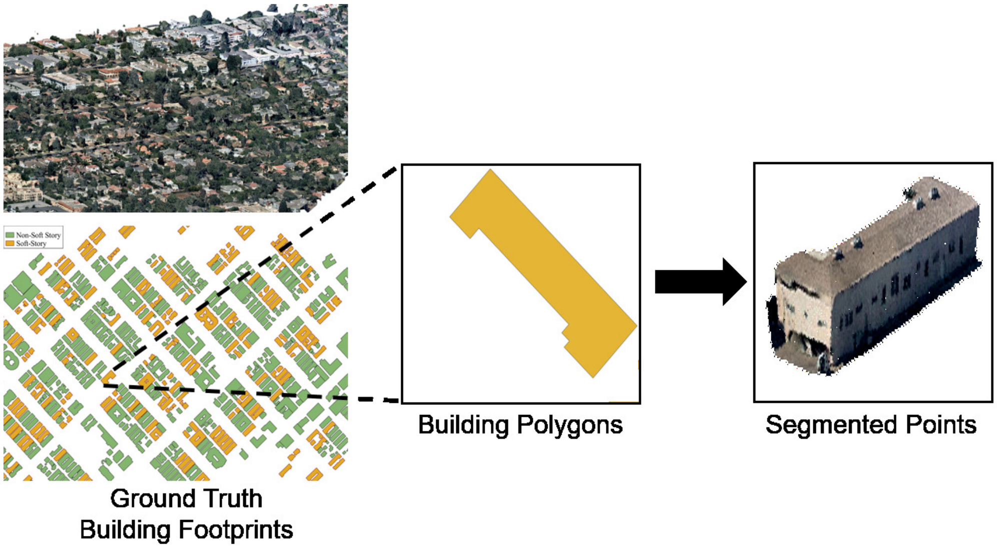


Figure 2.5. Segmentation using building polygons.

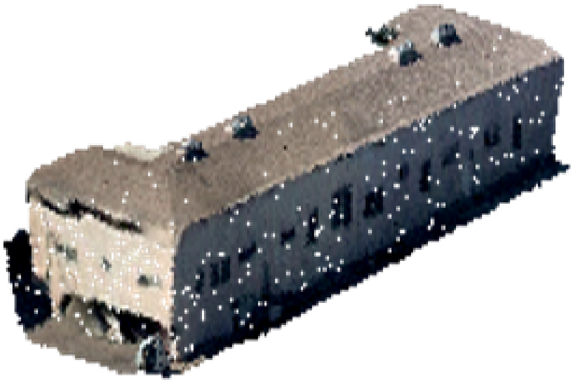


Figure 2.6. An example of soft-story building points.

2.2 3D Deep Learning: PointNet

This section introduces the 3D deep learning model PointNet that used for detecting and identifying soft-story point clouds. As shown in Fig 2.7 (a), PointNet consists of several multilayer perceptrons (MLPs) and fully connected networks (FCs). The input data is raw point clouds with 3 coordinates (i.e., x, y, z), and it is followed by a mapping to high dimensions (e.g., 64 and 1024) through shared MLPs. An invariant function is used to capture the global features for classification, namely, the maximal pooling. The reason of using invariant function is because point clouds do not like image data where the order of pixels is correlated with the features learned by the model. These global features are then followed by FCs and transformed into k categorical scores. For the segmentation, the local features of point clouds (i.e., relative location of objects) are considered through the concatenation with the global features (i.e., 64 local + 1044 global).

In MLP network, layers of artificial neurons are connected by weighted edges and are denoted as n_{ij} for the $j-th$ neuron in the $i-th$ layer. The input to a neuron n_{ij} as known as the "net" denoted v_{ij}, is the weighted sum of all incoming edges plus an optional bias. Its output $f_i(v_{ij}) = o_{ij}$ is computed by applying the $i-th$ layer's activation function f_i to v_{ij}. By arranging the weights of each layer into a matrices, the output of the $i-th$ layer of a MLP can be computed as Eq. 2.1

$$o_i = f_i(v_i) = f_i(W_i^T o_{i-1} + b_i) \tag{2.1}$$

During the learning process, back-propagation is often used with gradient descent to minimize the network error expressed in Eq.2.2 where $\hat{y}^{(l)}$ is the $l-th$ prediction made by the network.

$$\sum_{l=1}^{m} E^{(l)} = \sum_{l=1}^{m} \frac{1}{2}||y^{(l)} - \hat{y}^{(l)}||^2 \tag{2.2}$$

To update the weights and biases of the network, the chain rule is implemented to propagate the error from the output layer to the $j-th$ neuron in the $i-th$ layer. As shown in Eq. 2.3, $v_{ik}^{(l)}$ is the $k-th$ neuron's net in the $i-th$ layer, and $o_{(i-1)j}^{(l)}$ is the $j-th$ neuron's output in layer $i-1$. γ is the learning rate.

$$\Delta w_{ijk}^{(l)} = -\gamma \frac{\partial E_k^{(l)}}{\partial v_{ik}^{(l)}} o_{(i-1)j}^{l} \tag{2.3}$$

Take the $k-th$ component as an example, the $o_{(i-1)j}^{(l)}$ can be expressed as Eq.2.4. The update rule for the output neuron is then changed to Eq.2.5.

$$o_{(i-1)j}^{(l)} = \frac{\partial v_{ik}^{(l)}}{w_{ijk}} = \frac{\partial}{\partial w_{ijk}}(w_{i1k} o_{(i-1)1}^{(l)} + \cdots + w_{ijk} o_{(i-1)j}^{(l)} + \cdots + w_{iNk} o_{(i-1)N}^{(l)}) \tag{2.4}$$

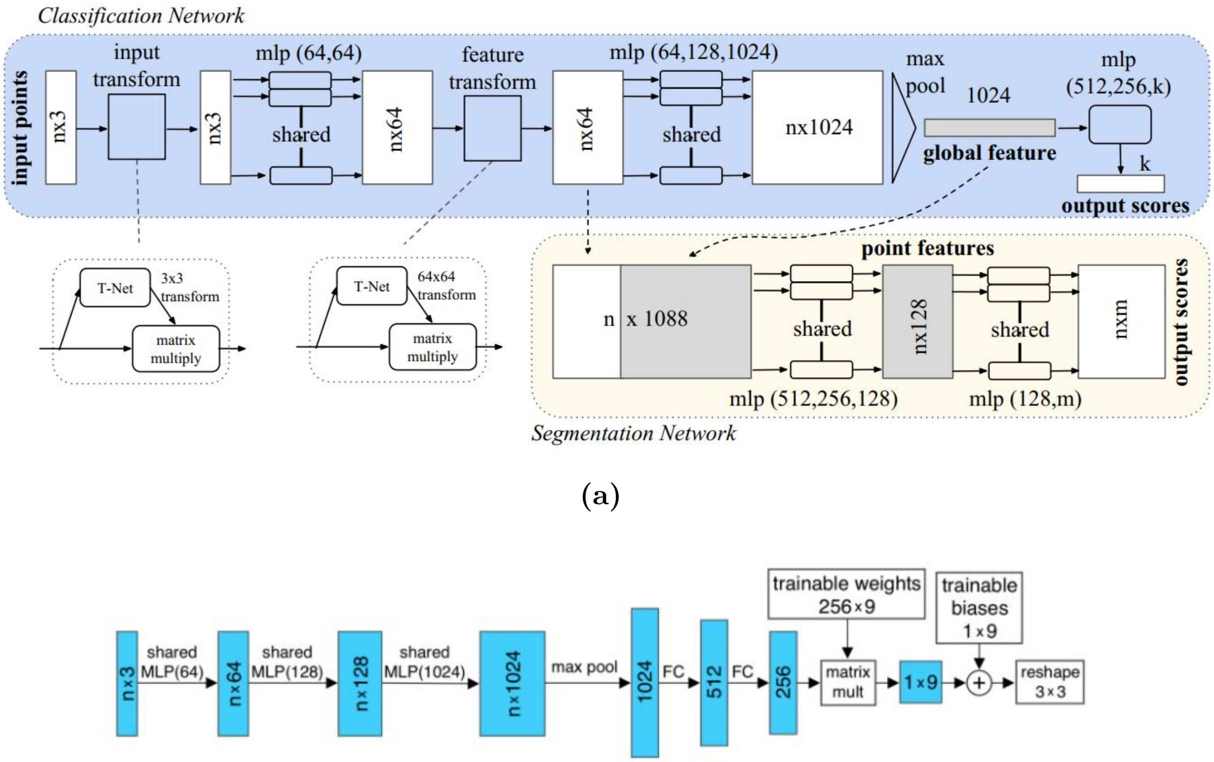
$$\Delta w_{njk}^{(l)} = -\gamma (y_k^{(l)} - \hat{y}_k^{(l)}) f_i'(v_{nk}^{(l)}) o_{(n-1)j}^{(l)} \tag{2.5}$$

As for hidden layer, the update is shown in Eq.2.6. To update the bias weights, $o_{i-1}^{(l)}$ needs to be replaced with 1.

$$\Delta w_{ijk}^{(l)} = -\gamma \sum_{c=1}^{N_i} \frac{\partial E_c^{(l)}}{\partial o_{ik}^{(l)}} f_i'(v_{ik}^{(l)}) o_{(i-1)j}^{(l)} = -\gamma \sum_{c=1}^{N_i} \frac{\partial E_c^{(l)}}{\partial v_{(i+1)c}^{(l)}} \frac{\partial v_{(i+1)c}^{(l)}}{\partial o_{ik}^{(l)}} f_i'(v_{ik}^{(l)}) o_{(i-1)j}^{(l)} \tag{2.6}$$

In addition to the invariance of input order, another important consideration that PointNet makes is the transformation invariance, which is addressed through a transformation network (T-Net). As shown in Fig 2.7 (b), T-Net also uses MLPs and FCs to learn a set of variables for the location transformation. The output variables can be adopted as a matrix multiplier (e.g., 3x3, 64x64) to the input raw points. Through these architectures, point clouds can be directly use as input data without any formation transformation. In this study, the task of PointNet is a binary classification, where the output is labeling the point clouds with either 0 (non-soft-story) or 1 (soft-story). The implementation of PointNet in this study is conducted on a TensorFlow [1] platform using a self-assembled computer with a single GPU (CPU:Intel(R) Core i7-8700 @ 3.20 GHz, RAN:16.0 GB, and GPU:Nvidia RTX 2080).

A real-world implementation of PointNet done by Qi et al.[7] is shown in Fig 2.8, where indoor objects are identified into multiple categories. In their implementation, raw point clouds are scanned by a unit block with size 1m by 1m and a 1m stride. The idea behind this is similar to the kernel filter (e.g., 3x3, 5x5) that widely used in the 2D image classification task. The block size, stride, and the number of points in each block are further investigated in this study as hyper-parameters. It will be discussed in the next chapter. An example of the identification of soft-story points is shown in Fig 2.9



(a)

(b)

Figure 2.7. PointNet Architecture: (a) full structure; (b) T-Net (adapted from [28]).

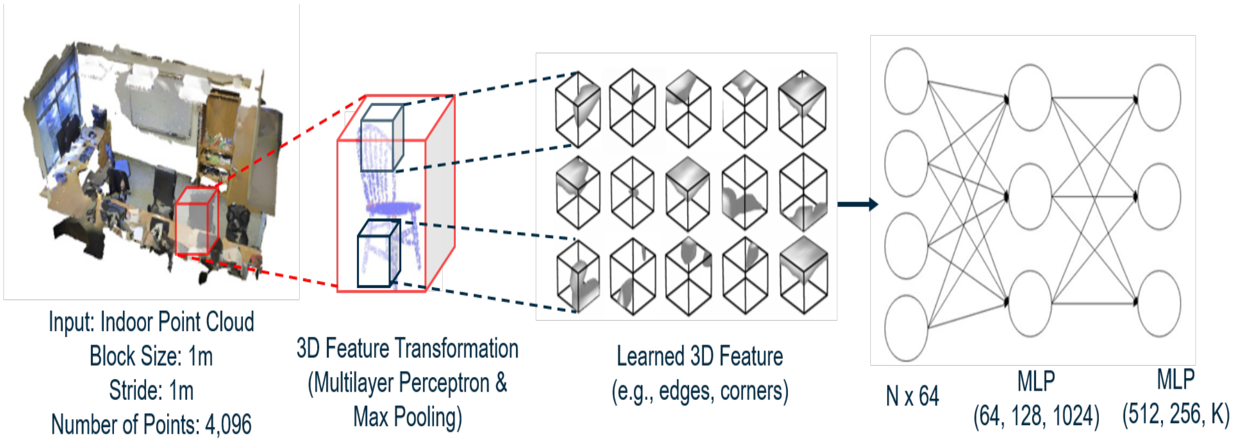


Figure 2.8. Implementation of PointNet structure.

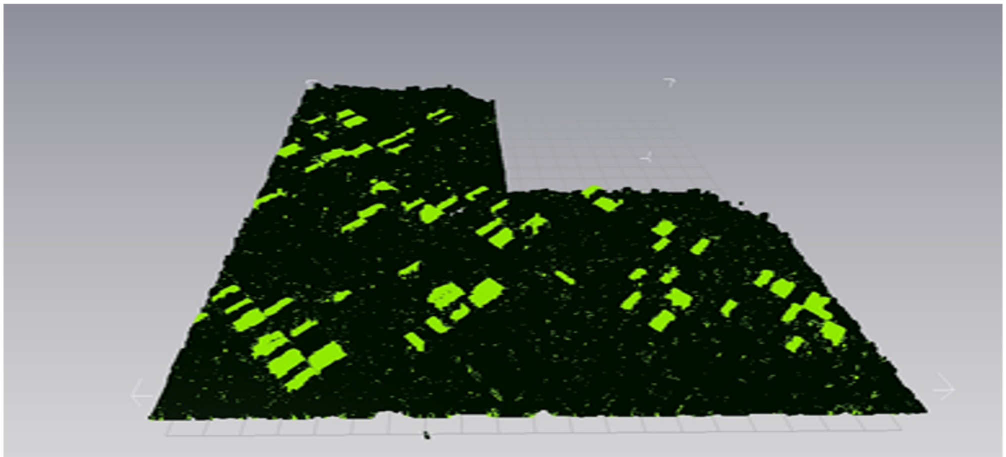


Figure 2.9. PointNet prediction: soft-story points.

2.3 Locations Identification: Clustering Methods

As shown in Fig 2.9, sparse green dots (i.e., classified as soft-story points) can be observed. These points should not belong to building groups (i.e., obvious green clusters), and hence need to be removed before the real-world addresses are generated from the prediction. Specifically, two widely used clustering methods (i.e., K-means [23], Gaussian mixture model [32]) are implemented to conclude the real building points (i.e., clusters), and the centers of the clusters are further mapped into the real-world addresses through Google Map API (`https://pypi.org/project/googlemaps/1.0.2/`). This postprocessing is implemented with the programming language Python and well-developed machine learning package scikit-learn[27].

For the Gaussian mixture model (GMM), k independent Gaussian distributions are used to model k separate clusters with mean μ_k and covariance matrix Σ_k. The model has the form expressed in Eq.2.7, and the goal is to maximize the posterior probability that point i belongs to cluster k as shown in Eq.2.8

$$p(x_i|\theta) = \sum_{k=1}^{K} \pi \mathcal{N}(x_i|\mu_k, \Sigma_k) \tag{2.7}$$

$$p(y_i = k|x_i, \theta) = \frac{p(yi = k|\theta)p(x_i|y_i = k, \theta)}{\sum_{k'=1}^{K} p(y_i = k'|\theta)p(x_i|z_i = k', \theta)} \tag{2.8}$$

The algorithm starts with the initialization of μ, Σ, and the probability of point i belongs to cluster k ($p(y_i = k) = \pi_k$). Through maximizing the posterior, each point can be labeled and new μ, Σ, and π can be obtained. The posterior can be further computed and used to update variables like the above mentioned. The time to reach the convergence depends on the size of data and the number of clusters.

While GMM provides a probability description of clusters, K-means is a non-probabilistic clustering algorithm. In K-means, $\Sigma_k = \sigma^2 1$, and $\pi_k = 1/K$. Only the cluster centers μ_k have to be estimated. The most probable cluster for x_i can be computed by finding the nearest prototype expressed in Eq.2.9.

$$y_i^* = argmin_k ||x_i - \mu_k||_2^2 \tag{2.9}$$

The algorithm starts with the initialization of the cluster center μ_k and the assigning each point i to the closest cluster center k. Once all points have been assigned, new central points can be recalculated and used for the new assignment. These two methods are implemented through the scikit-learn package in this study and their performance will be discussed in the next chapter.

2.4 Chapter Summary

In this chapter, components in the proposed workflow are introduced. It starts with the data preparation where the photogrammetry is implemented to generate city-scale point clouds at Santa Monica. A heuristic algorithm is then introduced to help label data. The algorithm uses a pretrained classifier using Semantic3D out-door data set to identify building points. Ground-truth polygons are utilized to further label soft-story-building points and non-soft-story-building points.

A DNN model that is able to train on raw point clouds is implemented in the workflow, namely, PointNet. In PointNet, multi-layer perceptrons and invariant functions are the main components that are able to address the sparsity and the unordered characteristic of point clouds. Finally, two classic clustering methods (i.e., K-means method, Gaussian mixture method) are included into the workflow for concluding the center of each building-point set, which is further transformed into the geographic location and real-world address.

CHAPTER 3

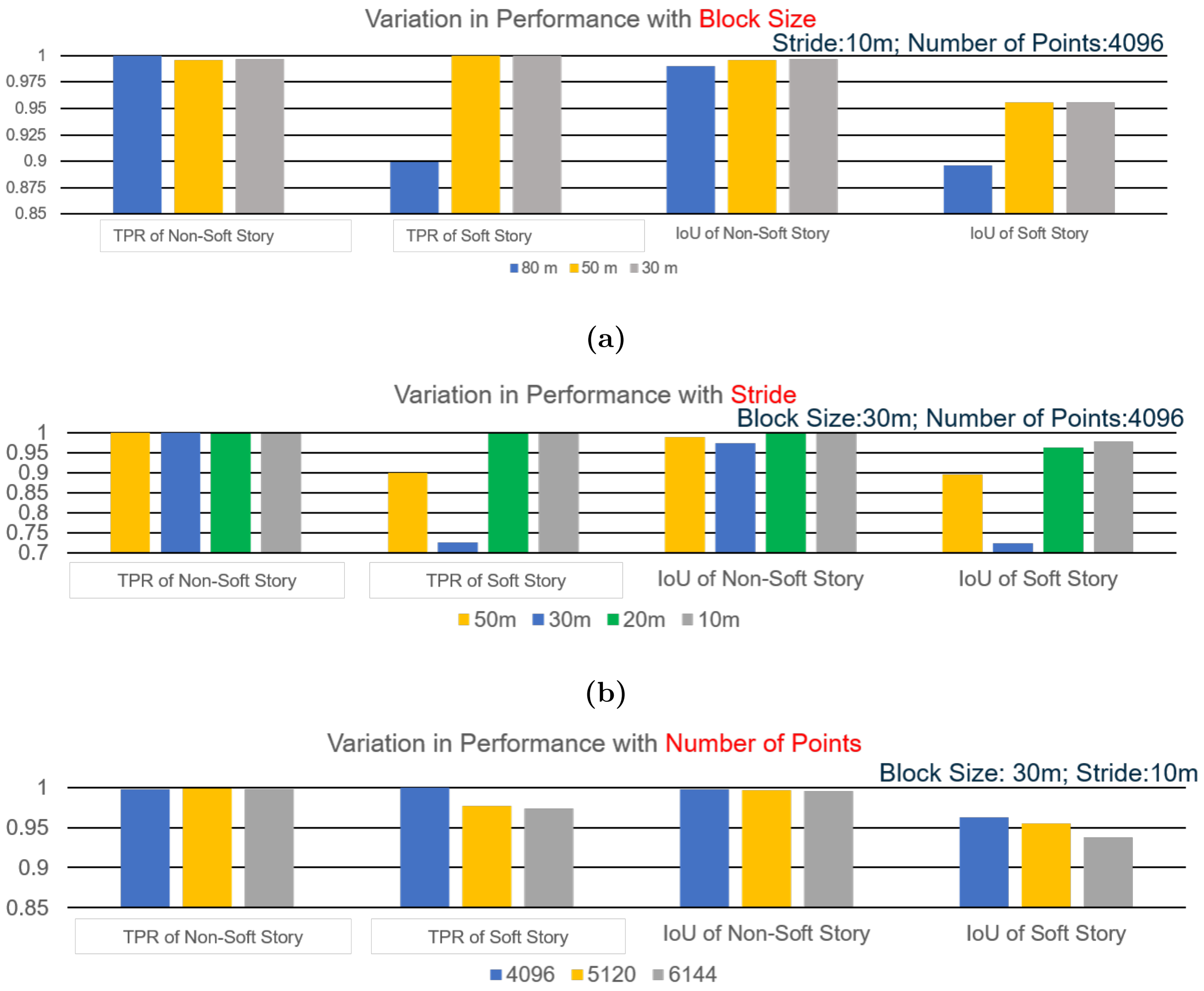
Results Analysis and Discussion

In this chapter, the performance of PointNet with multiple hyperparameters will be discussed first. The purpose is to identify the optimal hyperparameters for the proposed workflow. Meanwhile, the performance of clustering methods are also evaluated through the comparison with the ground-truth addresses.

3.1 Performance of PointNet and Hyperparameter Tuning

The labeled points shown in Table 2.1 are randomly split into training (80%), validation (10%), and test (10%) sets. 10-fold cross-validation is implemented to train PointNet and investigate the optimal hyperparameters (i.e., block size, stride, number of point in a block). The ranges of hyperparameters are limited due to the computational resource used in this study. The block size includes [30m x 30m, 50m x 50m, 80m x 80m], the stride includes [10m, 20m, 30m, 50m], and the number of points in a block includes [4096, 5120, 6144]. If too small block, too short stride, or too many points in a block is used, the memory will be crashed. It must be noted that, the recommended hyperparameters are only useful for the specific region studied here. Further investigation may be necessary when applied this workflow to another region or point clouds collected through different methodologies.

Fig 3.1 shows the variation of performance with different hyperparameters. The performance metrics used are expressed in Eq.3.1 and Eq.3.2, all of which follow the metrics defined in PointNet paper[28]. The results indicate that a 30m by 30m block with 10m stride and 4096 points in it has the optimal performance. With these hyperparameters, all training and validation sets are combined to train the final model, and the performance on the test set is shown in Table.3.1



(a)

(b)

(c)

Figure 3.1. Variation in performance with (a) block size, (b) stride, and (c) number of points in a block.

$$True\ Positive\ Rate\ (TPR) = \frac{TruePositive}{PredictedPositive} \tag{3.1}$$

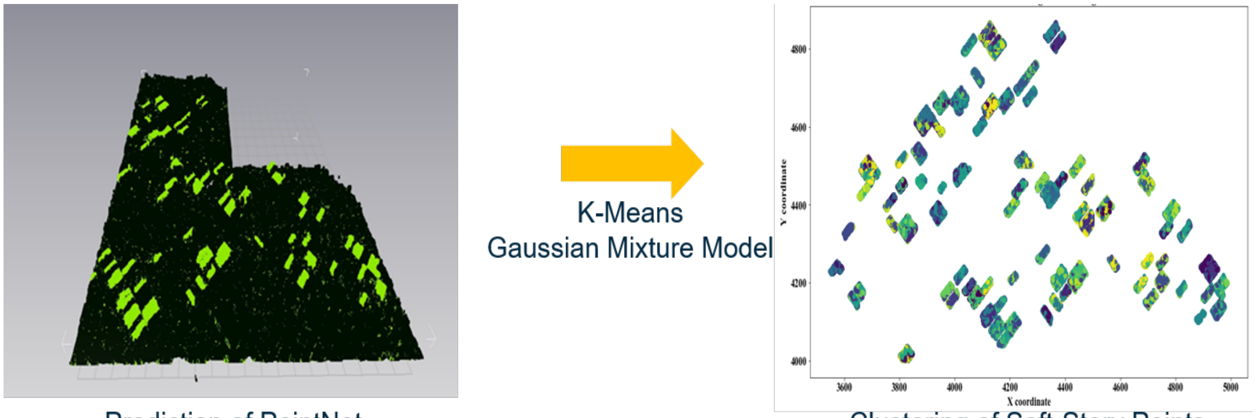
$$
\begin{aligned}
&Intersection\ over\ Union\ (IoU)\\
&= \frac{TruePositive}{GroundTruth + PredictedPositive - TruePositive}
\end{aligned}
\tag{3.2}
$$

Table 3.1. Performance of PointNet on test set.

Testing	Non-Soft-Story	Soft-story
TPR	0.997	0.899
IoU	0.989	0.896

3.2 Evaluation of Clustering Methods

The predicted soft-story points shown in Fig 2.9 are filtered through K-means and GMM to remove points that do not belong to a building and compute the center of a building to obtain the real-world address. The procedure described in section 2.3 is followed to train both algorithms. The optimal number of clusters k (i.e.,the number of buildings) are obtained through cross-validation implemented in scikit-learn package.



Prediction of PointNet

Clustering of Soft-Story Points

Figure 3.2. Clustering points to buildings.

Once the predicted centers are obtained, their longitudes and latitudes are converted to real-world addresses through Google map API. A spatial query is then implemented to evaluate the number of addresses that are located within the ground-true building polygons. The results are shown in Fig 3.3 and Table 3.2, and GMM is recommended with much stable prediction.

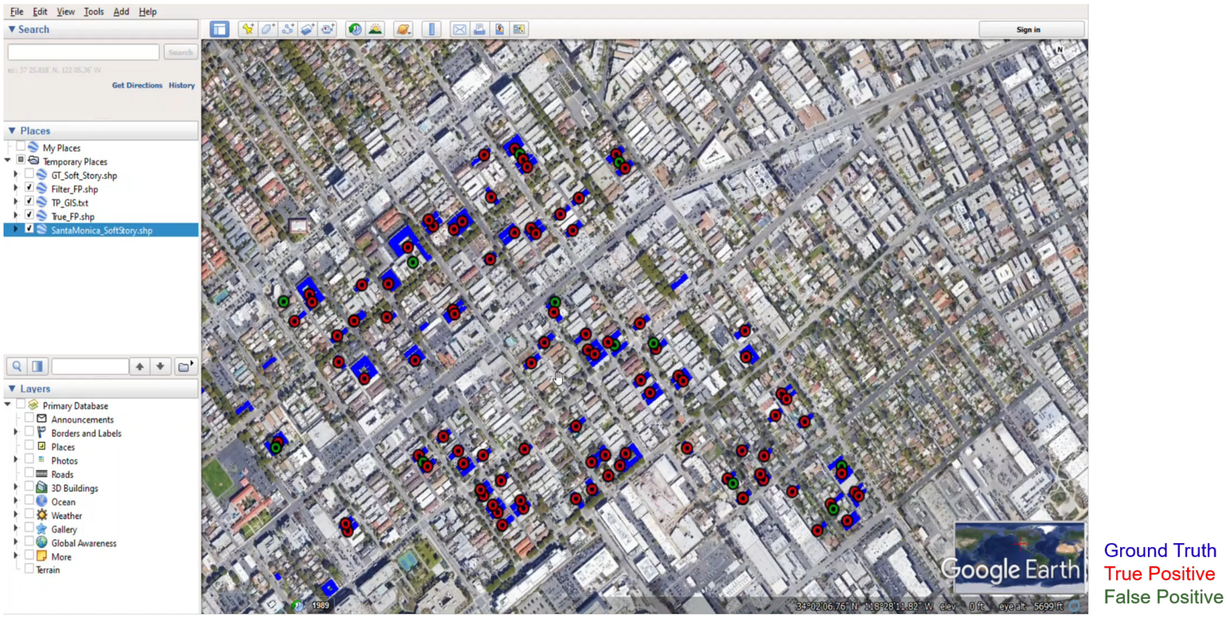


Figure 3.3. Evaluation of the predicted addresses.

Table 3.2. Performance of clustering on test set.

Testing	K-means	GMM
TPR	0.905	0.883
IoU	0.796	0.82

3.3 Chapter Summary

In this chapter, PointNet classifier is evaluated and fine-tuned through numerical experiments. Hyperparameters in PointNet including the block size, stride, and number of points in each block. The results show that the model trained with 30m blocks, 10m stride, and 4096 points in each block dominates the performance of classification accuracy and intersection-over-union. Meanwhile, Gaussian mixture method provides a stable prediction of centers of building clusters, which hence leads to more reliable real-world addresses.

CHAPTER 4

Conclusions and Suggestions of Future Work

4.1 Conclusions

This study developed a workflow that directly uses 3D point clouds to identify soft-story buildings at a city scale. The proposed workflow greatly improve the limitation of prior studies where 2D deep learning models are used to identify seismically vulnerable buildings in street-view images. Buildings in street-view images may some time be blocked and hence result in a false prediction. On the contrary, point clouds storing the position data of objects can elaborate the relative distance between objects so that buildings can be segmented.

In order to directly use the advantage of point clouds, a recently developed 3D deep learning model, namely, PointNet, is implemented. PointNet is developed to train a classifier with raw point clouds, which is invariant to the input order and direction transformation. The city of Santa Monica in California is selected as the target region in this study because it has a well-investigated database where ground-true soft-story buildings are available. Around 1.1 billions of points are obtained through the photogrametry method and labeled into non-soft-story and soft-story points by a proposed heuristic algorithm.

Point-cloud data is split into training, validation, and testing sets for conducting cross-validation. The block size, stride, and number of points in a block are the hyperparameters in PointNet. The optimal values of them are identified in this study for this specific region (i.e., Santa Monica). A 30m by 30m block with 10m stride and 4096 points in the block has the highest TPR and IoU. However, it must be noted that the optimal values may be different when applying the proposed workflow to other regions or if point clouds are collected through different methods.

To make the proposed much benefit to the engineering committee, a postprocessing that converts the point clouds to real-world addresses is also developed. K-means and Gaussian mixture models are selected to filter points that belong to buildings. The centers of predicted clusters (i.e., building clusters) are then transformed into addresses using Google map API and their geolocations (i.e., latitude, longitude). The evaluation shows that GMM is more reliable with much stable TPR and IoU.

4.2 Potential Impacts

This study tries to use advanced types of data and mining techniques to provide a potential solution for a labor-intensive civil engineering issue. Computer-aided technologies are actively being studied and implemented in the life-cycle of an infrastructure. For example, building information modeling, which stores different tiers of information of building components including structures and electrical equipment. This study can be used to identify/classify these components for detailed modeling in the preliminary design, construction, or maintenance stage.

4.3 Suggestions of Future Work

Because of the investigation done by prior studies, many soft-story buildings have been retrofitted with reinforcing structures (e.g., a garage of a single-story residential house). These buildings should not be labeled as soft stories anymore. In other words, using the proposed method for retrofitted buildings could result in a false prediction. To address this issue, advanced methods (e.g., LiDAR scanning) can be used to obtain much refined point-cloud data, which ideally can reflect the existence of components designed for reinforcement and improve the identification accuracy.

Due to the computational costs, PointNet is selected herein. However, it is not state-of-the-art 3D deep learning models that can deal with point clouds. PointNet++[29], and PointCNN[22] are more advanced models that can also use the texture and intensity information stored in point-cloud data to provide a better prediction. The application of these models can be the future work of this study.

REFERENCES

[1] Martín Abadi, Paul Barham, Jianmin Chen, Zhifeng Chen, Andy Davis, Jeffrey Dean, Matthieu Devin, Sanjay Ghemawat, Geoffrey Irving, Michael Isard, et al. Tensorflow: A system for large-scale machine learning. In *12th {USENIX} symposium on operating systems design and implementation ({OSDI} 16)*, pages 265–283, 2016.

[2] Mohamad Alipour, Devin K Harris, and Gregory R Miller. Robust pixel-level crack detection using deep fully convolutional neural networks. *Journal of Computing in Civil Engineering*, 33(6):04019040, 2019.

[3] Michel Bruneau, Stephanie E Chang, Ronald T Eguchi, George C Lee, Thomas D O'Rourke, Andrei M Reinhorn, Masanobu Shinozuka, Kathleen Tierney, William A Wallace, and Detlof Von Winterfeldt. A framework to quantitatively assess and enhance the seismic resilience of communities. *Earthquake spectra*, 19(4):733–752, 2003.

[4] Henry Burton, Aryan Rezaei Rad, Zhengxiang Yi, Damian Gutierrez, and Koyejo Ojuri. Seismic collapse performance of los angeles soft, weak, and open-front wall line woodframe structures retrofitted using different procedures. *Bulletin of earthquake engineering*, 17(4):2059–2091, 2019.

[5] Young-Jin Cha, Wooram Choi, and Oral Büyüköztürk. Deep learning-based crack damage detection using convolutional neural networks. *Computer-Aided Civil and Infrastructure Engineering*, 32(5):361–378, 2017.

[6] Jingdao Chen, Zsolt Kira, and Yong K Cho. Deep learning approach to point cloud scene understanding for automated scan to 3d reconstruction. *Journal of Computing in Civil Engineering*, 33(4):04019027, 2019.

[7] Xiaozhi Chen, Kaustav Kundu, Ziyu Zhang, Huimin Ma, Sanja Fidler, and Raquel Urtasun. Monocular 3d object detection for autonomous driving. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 2147–2156, 2016.

[8] Yun-Jian Cheng, Wenge Qiu, and Jin Lei. Automatic extraction of tunnel lining cross-sections from terrestrial laser scanning point clouds. *Sensors*, 16(10):1648, 2016.

[9] Jia Deng, Wei Dong, Richard Socher, Li-Jia Li, Kai Li, and Li Fei-Fei. Imagenet: A large-scale hierarchical image database. In *2009 IEEE conference on computer vision and pattern recognition*, pages 248–255. Ieee, 2009.

[10] Piotr Dollár and C Lawrence Zitnick. Structured forests for fast edge detection. In *Proceedings of the IEEE international conference on computer vision*, pages 1841–1848, 2013.

[11] Martin Engelcke, Dushyant Rao, Dominic Zeng Wang, Chi Hay Tong, and Ingmar Posner. Vote3deep: Fast object detection in 3d point clouds using efficient convolutional neural networks. In *2017 IEEE International Conference on Robotics and Automation (ICRA)*, pages 1355–1361. IEEE, 2017.

[12] Yuqing Gao and Khalid M Mosalam. Deep transfer learning for image-based structural damage recognition. *Computer-Aided Civil and Infrastructure Engineering*, 33(9):748–768, 2018.

[13] Timo Hackel, Nikolay Savinov, Lubor Ladicky, Jan D Wegner, Konrad Schindler, and Marc Pollefeys. Semantic3d. net: A new large-scale point cloud classification benchmark. *arXiv preprint arXiv:1704.03847*, 2017.

[14] Kaiming He, Georgia Gkioxari, Piotr Dollár, and Ross Girshick. Mask r-cnn. In *Proceedings of the IEEE international conference on computer vision*, pages 2961–2969, 2017.

[15] Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. Deep residual learning for image recognition. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 770–778, 2016.

[16] Rony Kalfarisi, Zheng Yi Wu, and Ken Soh. Crack detection and segmentation using deep learning with 3d reality mesh model for quantitative assessment and integrated visualization. *Journal of Computing in Civil Engineering*, 34(3):04020010, 2020.

[17] Hua Kang, Zhengxiang Yi, and Henry Burton. Effect of the los angeles soft-story ordinance on the post-earthquake housing recovery of impacted residential communities. *Natural Hazards*, 99(1):161–188, 2019.

[18] Jian Kang, Marco Körner, Yuanyuan Wang, Hannes Taubenböck, and Xiao Xiang Zhu. Building instance classification using street view images. *ISPRS journal of photogrammetry and remote sensing*, 145:44–59, 2018.

[19] Alireza G Kashani, Patrick S Crawford, Sufal K Biswas, Andrew J Graettinger, and David Grau. Automated tornado damage assessment and wind speed estimation based on terrestrial laser scanning. *Journal of Computing in Civil Engineering*, 29(3):04014051, 2015.

[20] Hongjo Kim, Hyoungkwan Kim, Yong Won Hong, and Hyeran Byun. Detecting construction equipment using a region-based fully convolutional network and transfer learning. *Journal of computing in Civil Engineering*, 32(2):04017082, 2018.

[21] Srinath Shiv Kumar, Mingzhu Wang, Dulcy M Abraham, Mohammad R Jahanshahi, Tom Iseley, and Jack CP Cheng. Deep learning–based automated detection of sewer defects in cctv videos. *Journal of Computing in Civil Engineering*, 34(1):04019047, 2020.

[22] Yangyan Li, Rui Bu, Mingchao Sun, Wei Wu, Xinhan Di, and Baoquan Chen. Pointcnn: Convolution on x-transformed points. *Advances in neural information processing systems*, 31:820–830, 2018.

[23] Aristidis Likas, Nikos Vlassis, and Jakob J Verbeek. The global k-means clustering algorithm. *Pattern recognition*, 36(2):451–461, 2003.

[24] Wei Liu, Dragomir Anguelov, Dumitru Erhan, Christian Szegedy, Scott Reed, Cheng-Yang Fu, and Alexander C Berg. Ssd: Single shot multibox detector. In *European conference on computer vision*, pages 21–37. Springer, 2016.

[25] Daniel Maturana and Sebastian Scherer. Voxnet: A 3d convolutional neural network for real-time object recognition. In *2015 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, pages 922–928. IEEE, 2015.

[26] Sinno Jialin Pan and Qiang Yang. A survey on transfer learning. *IEEE Transactions on knowledge and data engineering*, 22(10):1345–1359, 2009.

[27] Fabian Pedregosa, Gaël Varoquaux, Alexandre Gramfort, Vincent Michel, Bertrand Thirion, Olivier Grisel, Mathieu Blondel, Peter Prettenhofer, Ron Weiss, Vincent Dubourg, et al. Scikit-learn: Machine learning in python. *the Journal of machine Learning research*, 12:2825–2830, 2011.

[28] Charles R Qi, Hao Su, Kaichun Mo, and Leonidas J Guibas. Pointnet: Deep learning on point sets for 3d classification and segmentation. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 652–660, 2017.

[29] Charles R Qi, Li Yi, Hao Su, and Leonidas J Guibas. Pointnet++: Deep hierarchical feature learning on point sets in a metric space. *arXiv preprint arXiv:1706.02413*, 2017.

[30] Joseph Redmon, Santosh Divvala, Ross Girshick, and Ali Farhadi. You only look once: Unified, real-time object detection. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 779–788, 2016.

[31] Shaoqing Ren, Kaiming He, Ross Girshick, and Jian Sun. Faster r-cnn: Towards real-time object detection with region proposal networks. In *Advances in neural information processing systems*, pages 91–99, 2015.

[32] Douglas A Reynolds. Gaussian mixture models. *Encyclopedia of biometrics*, 741:659–663, 2009.

[33] Martin Simony, Stefan Milzy, Karl Amendey, and Horst-Michael Gross. Complex-yolo: An euler-region-proposal for real-time 3d object detection on point clouds. In *Proceedings of the European Conference on Computer Vision (ECCV) Workshops*, pages 0–0, 2018.

[34] Karen Simonyan and Andrew Zisserman. Very deep convolutional networks for large-scale image recognition. *arXiv preprint arXiv:1409.1556*, 2014.

[35] Shivangi Srivastava, John E Vargas-Muñoz, and Devis Tuia. Understanding urban landuse from the above and ground perspectives: A deep learning, multimodal solution. *Remote sensing of environment*, 228:129–143, 2019.

[36] Christian Szegedy, Sergey Ioffe, Vincent Vanhoucke, and Alexander A Alemi. Inception-v4, inception-resnet and the impact of residual connections on learning. In *Thirty-first AAAI conference on artificial intelligence*, 2017.

[37] Christian Szegedy, Wei Liu, Yangqing Jia, Pierre Sermanet, Scott Reed, Dragomir Anguelov, Dumitru Erhan, Vincent Vanhoucke, and Andrew Rabinovich. Going deeper with convolutions. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 1–9, 2015.

[38] Ruisheng Wang. 3d building modeling using images and lidar: A review. *International Journal of Image and Data Fusion*, 4(4):273–292, 2013.

[39] Bichen Wu, Alvin Wan, Xiangyu Yue, and Kurt Keutzer. Squeezeseg: Convolutional neural nets with recurrent crf for real-time road-object segmentation from 3d lidar point cloud. In *2018 IEEE International Conference on Robotics and Automation (ICRA)*, pages 1887–1893. IEEE, 2018.

[40] Z. Y. Wu and M. Hmosze. Soft-story detection by deep learning. Technical report, Bentley Systems, Incorporated, Watertown, CT: Bentley Systems, 2018.

[41] Z. Y. Wu, M. Hmosze, and R. Kalfarisi. Soft-story building detection using deep convolutional neural network for complying building resilience ordinance. In *ASCE Engineering Mechanics Institute Conference*, Caltech, Pasadena, CA., 2019.

[42] Qian Yu, Chaofeng Wang, Frank McKenna, X Yu Stella, Ertugrul Taciroglu, Barbaros Cetiner, and Kincho H Law. Rapid visual screening of soft-story buildings from street view images using deep learning classification. *Earthquake Engineering and Engineering Vibration*, 19(4):827–838, 2020.

[43] Allen Zhang, Kelvin CP Wang, Yue Fei, Yang Liu, Siyu Tao, Cheng Chen, Joshua Q Li, and Baoxian Li. Deep learning–based fully automated pavement crack detection on 3d asphalt surfaces with an improved cracknet. *Journal of Computing in Civil Engineering*, 32(5):04018041, 2018.

[44] Allen Zhang, Kelvin CP Wang, Baoxian Li, Enhui Yang, Xianxing Dai, Yi Peng, Yue Fei, Yang Liu, Joshua Q Li, and Cheng Chen. Automated pixel-level pavement crack detection on 3d asphalt surfaces using a deep-learning network. *Computer-Aided Civil and Infrastructure Engineering*, 32(10):805–819, 2017.

[45] Zixiang Zhou, Jie Gong, and Mengyang Guo. Image-based 3d reconstruction for posthurricane residential building damage assessment. *Journal of Computing in Civil Engineering*, 30(2):04015015, 2016.