AUTO-CONVERSION FROM 2D DRAWING TO 3D MODEL WITH DEEP LEARNING

A THESIS SUBMITTED TO
THE GRADUATE SCHOOL OF NATURAL AND APPLIED SCIENCES
OF
MIDDLE EAST TECHNICAL UNIVERSITY

BY

GIZEM YETIŞ

IN PARTIAL FULFILLMENT OF THE REQUIREMENTS
FOR
THE DEGREE OF MASTER OF SCIENCE
IN
BUILDING SCIENCE IN ARCHITECTURE

SEPTEMBER 2019

Approval of the thesis:

AUTO-CONVERSION FROM 2D DRAWING TO 3D MODEL WITH DEEP LEARNING

submitted by **GIZEM YETIŞ** in partial fulfillment of the requirements for the degree of **Master of Science in Building Science in Architecture Department, Middle East Technical University** by,

Prof. Dr. Halil Kalıpçılar
Dean, Graduate School of **Natural and Applied Sciences** _____

Prof. Dr. F. Cânâ Bilsel
Head of Department, **Architecture** _____

Prof. Dr. Arzu Gönenç Sorguç
Supervisor, **Architecture, METU** _____

Examining Committee Members:

Prof. Dr. Ali Murat Tanyer
Architecture, METU _____

Prof. Dr. Arzu Gönenç Sorguç
Architecture, METU _____

Prof. Dr. Birgül Çolakoğlu
Architecture, ITU _____

Assist. Prof. Dr. Mehmet Koray Pekeriçli
Architecture, METU _____

Prof. Dr. Soner Yıldırım
CEIT, METU _____

Date: 27.09.2019

I hereby declare that all information in this document has been obtained and presented in accordance with academic rules and ethical conduct. I also declare that, as required by these rules and conduct, I have fully cited and referenced all material and results that are not original to this work.

Name, Surname: Gizem Yetiş

Signature:

ABSTRACT

AUTO-CONVERSION FROM 2D DRAWING TO 3D MODEL WITH DEEP LEARNING

Yetiş, Gizem
Master of Science, Building Science in Architecture
Supervisor: Prof. Dr. Arzu Gönenç Sorguç

September 2019, 148 pages

Modeling has always been important as it transfers knowledge to end users. From the very first line on a computer screen to AR/VR applications have broaden the perception, communication and implementation of design-related industries, and each representation technique has become one another's base, information source and data supplier. Yet, transforming the information that one includes into another has still major problems. It requires precise data, qualified personnel and human intervention. This research aims to represent an automated reconstruction from low level data sources to higher level digital models in order to eliminate these problems. This auto-conversion process only examines the architectural usage and makes a sample of its usability in different fields.

2D floor plans and elevation drawings in raster format, which are collected and/or produced from scratch, are used as datasets. These drawings are semantically segmented with three different Convolutional Neural Networks to obtain relevant architectural information since Deep Learning shows promising success to solve a wide range of problem with its widespread use. Semantically segmented drawings are then transformed into 3D by using Digital Geometry Processing methods. Lastly, a web application is introduced to allow any user to obtain a 3D model with ease.

Semantic segmentation results in 2D and two case studies in 3D are evaluated and compared separately with different metrics to represent accuracy of the process.

To conclude, this research has proposed an automated process for reconstruction of 3D models with the state-of-the-art methods and made it ready for use even for a person without technical knowledge.

Keywords: Architectural Drawing Dataset, 3D Reconstruction, Semantic Segmentation, Convolutional Neural Networks, Digital Geometry Processing

ÖZ

DERİN ÖĞRENME İLE 2B ÇİZİMDEN 3B MODELE OTO-DÖNÜŞÜM

Yetiş, Gizem
Yüksek Lisans, Yapı Bilimleri
Tez Danışmanı: Prof. Dr. Arzu Gönenç Sorguç

Eylül 2019, 148 sayfa

Modelleme, bilgiyi kullanıcılara aktardığı için her zaman önemli olmuştur. Bilgisayar ekranındaki ilk çizgide, sanal gerçeklik uygulamalarına kadar bütün temsil teknikleri tasarım endüstrilerinin algılama, iletişim kurma ve uygulama biçimlerini genişletmiştir. Her bir temsil tekniği bir diğerinin temeli, bilgi kaynağı ve veri tedarikçisi haline gelmiştir. Ancak, bir temsilin içerdiği bilgiyi başka bir temsilin bilgisine dönüştürmek için doğru veriye, kalifiye personele ve insan müdahalesine ihtiyaç duyulmaktadır. Bu araştırma, bu sorunları gidermek için düşük seviyeli veri kaynaklarından yüksek seviyeli dijital modellere otomatik bir yeniden yapılanmayı temsil etmeyi amaçlamaktadır. Bu otomatik dönüştürme işlemi yalnızca mimari kullanımı inceler ve farklı alanlarda kullanılabilirliğinin bir örneğini oluşturur.

Bu tez kapsamında sıfırdan toplanan ve/veya üretilen 2B kat planları ve cephe çizimleri veri seti olarak kullanılmıştır. Bu veriler, ilgili mimari bilgiyi elde etmek için üç farklı Evrişimsel Sinir Ağı ile anlamsal olarak bölünmüştür, çünkü Derin Öğrenme yaygın kullanımıyla, geniş bir problem yelpazesini çözmede umut verici başarılar göstermektedir. Anlamsal olarak bölümlendirilmiş çizimler daha sonra Dijital Geometri İşleme yöntemleri kullanılarak 3B modele dönüştürülür. Son olarak, herhangi bir kullanıcının kolaylıkla 3B model elde etmesini ve kullanmasını sağlamak için bir web uygulaması tanıtılmıştır. 2B ortamdaki anlamsal bölümleme sonuçları ve

3B’deki iki vaka çalışması, sürecin doğruluğunu temsil edebilmek için farklı ölçümleme yöntemleriyle ayrı ayrı değerlendirilmiştir ve karşılaştırılmıştır.

Sonuç olarak, bu araştırma, en gelişmiş yöntemlerle 3B modellerin yeniden yapılandırılması için otomatik bir işlem önermiş ve teknik bilgisi olmayan bir kişi için bile kullanıma hazır hale getirmiştir.

Anahtar Kelimeler: Mimari Çizim Veri Seti, 3B Yeniden Üretim, Anlamsal Bölütleme, Evrişimsel Sinir Ağları, Dijital Geometri İşleme

To my loving family that has supported me from the very beginning

ACKNOWLEDGEMENTS

I would like to express my sincere thankfulness to my mentor and thesis supervisor Prof. Arzu Gönenç Sorguç for her support, advice and emboldening. This thesis would not be possible without her guidance that has started from the earliest time of my academic adventure.

I am more than grateful for the support of my colleagues Fatih Küçüksubaşı, Ozan Yetkin, Kubilay Şahinler, Selin Coşan, Fırat Özgenel and Müge Kruşa Yemişcioğlu. I would not be at this point without their patience, sophistication and endless fruitful discussions.

I am also very thankful for my gals and pals from METU Architecture who have suffered enough because of me during the whole time.

I would like to show my sincere appreciation to METU Design Factory Team.

I would like to thank Röyksopp for their tasteful music that has kept me mused to research, study and enjoy at the same time. My rocky road has become less painful with their art.

Last but not least, I am speechless about how my family has supported me from the day one. I must express my gratitude to my mother Nesrin, my father Turgut and my sister Gözde for their endless love, encouragement and support throughout my education. None of my accomplishments would be possible without them.

TABLE OF CONTENTS

LIST OF TABLES

TABLES

LIST OF FIGURES

FIGURES

LIST OF ABBREVIATIONS

ABBREVIATIONS

ANN	Artificial Neural Network
BOW	Bag-of-Words
CAD	Computer Aided Design
CNN	Convolutional Neural Network
CRF	Conditional Random Field
DBN	Deep Belief Network
DL	Deep Learning
DNN	Deep Neural Network
FAST	Features from Accelerated Segment Test
FCN	Fully Connected Convolutional Network
GPU	Graphical Processing Unit
HOG	Histogram of Oriented Gradients
IoU	Intersection over Union
LIDAR	Laser Imaging Detection and Ranging
MRF	Markov Random Field
PCA	Principal Component Analysis
RBM	Restricted Boltzmann Machine
R-CNN	Region-Based Convolutional Network
ReLu	Rectified Linear Unit
RNN	Recurrent Neural Network
SIFT	Scale-Invariant Feature Transform
SVM	Support Vector Machine
UAV	Unmanned Aerial Vehicles
WAG	Wall Adjacency Graph

CHAPTER 1

INTRODUCTION

1.1. Motivation

Modelling has always been important in all the disciplines since it is a way of representing the information. It includes both abstract and concrete knowledge, and conveys this knowledge to end users. Sharing information through different models is a vital feature for all design related industry stakeholders. Yet, transforming 2D information to 3D had some defects on perception even for a qualified human. Therefore, today a new level of information is required, which cannot be just seen as a representation but also can be perceived as data to allow this transformation.

Design-related disciplines such as architecture and engineering have continuously improved modes of modelling to broaden perception, collaboration, communication and implementation. Firstly, 2D and 3D virtual representation styles have been introduced into our lives. In the late 50s and early 60s, Pronto by Hanratty and Sketchpad by Sutherland pioneered the developments in Computer Aided Design (CAD). While Hanratty raised the first numerically controlled programming tool (Harris & Meyers, 2009), Sutherland made it possible to interact a human and a machine to design based on line drawings in a 2D virtual environment (Sutherland, 1964) as shown in Figure 1.1. Later, these major events enriched the developments in CAD, and solid modelling methods has emerged afterwards with the technological developments. Representation of edges, boundary surfaces, and primitive objects have broadened the horizon of CAD technologies and today, designers can work in an environment in which they can create, define and control a 3D digital model (Tornincasa & Di Monaco, 2010).

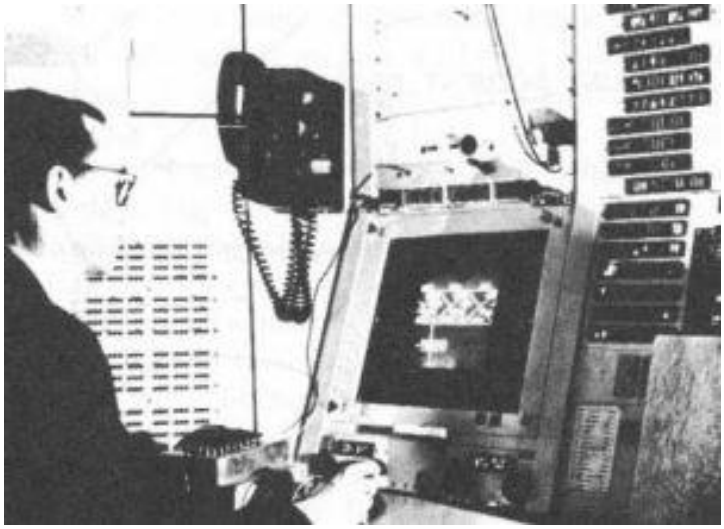


Figure 1.1. Sketchpad in use (Sutherland, 1964)

While CAD media have been developing, new tools for emergent needs have been launched simultaneously. Beside of producing digital models from scratch for a design process, novel 3D tools have been put forward to respond the needs of different usages such as simulation, manufacturing, reconstruction, interdisciplinary and collaborative working environments and so on. Thereby, it can clearly be said that CAD tools for all of the disciplines have been unceasingly evolving, and they become not just a source for representation, but also a source for perception shifter, information transmitter and different level of data supplier. Figure 1.2 shows some 3D model usages in different design related disciplines.

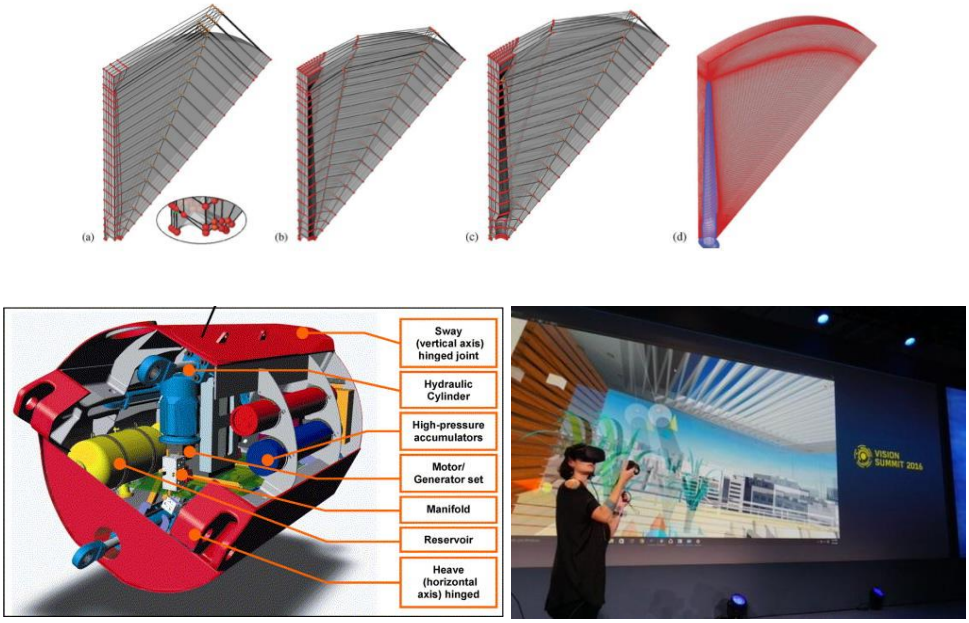


Figure 1.2. Usage areas with different representations (Upper: Simulation example (Bazilevs et.al., 2011), Bottom Left: 3D modelling for a product example (Henderson, 2006), Bottom Right: VR example (Horsey, 2016))

Emerging tools obviously offer different perception levels by allowing users to design new objects/products/buildings in 3D environment. Now, end users can imagine an object's appearance completely via 3D models. Different industry partners can work in the very same virtual environment simultaneously to simulate and integrate their ideas. Non-existing or existing objects can be generated to be analyzed, reconstructed or restored in a 3D virtual space. Therefore, conducting a design process by 3D modelling becomes inevitable to serve for a better perception and implementation environment.

Even though end users can easily gain perceptual experience with different representation modes via various tools, it is important to address complexity of perceiving and matching the information of an object in different dimensions. For instance, it is troublesome to imagine 3D version of 2D technical drawings. Likewise, it is hard to perceive 2D images as in real-life condition without predicting depth information. In these kind of cases, the information in one representation becomes a

data source that can be converted into another dimension. However, considerable expertise, time and precise data are required to transfer low level information to higher level. Therefore, motivation of this dissertation is searching for a new way that can be applicable for different industries to expose the information conveyed by various dimensions that make a difference in perception.

1.2. Problem Statement

There is a growing demand for higher level modelling in which the most applicable and available version is in 3D. Simulations and evaluation are important to simulate real-life conditions. Having a 3D model also allows contributors to understand and control design and management processes. Each stakeholder can state an opinion based on the 3D model. Manufacturing operations can be analyzed with these models. Planned expenditures for construction can be measured. Even real estate agencies now opt for showing digital models to clients for making them gain perspective on building's condition (Hartmann, Gao & Fischer, 2008). Conservation projects require these models to restore, renovate or remodel the artwork (Pieraccini, Guidi & Atzeni, 2001). Figure 1.3 shows survey results representing the current usage and awareness of different working principles, and 3D modelling usage turns out the highest in both sense.

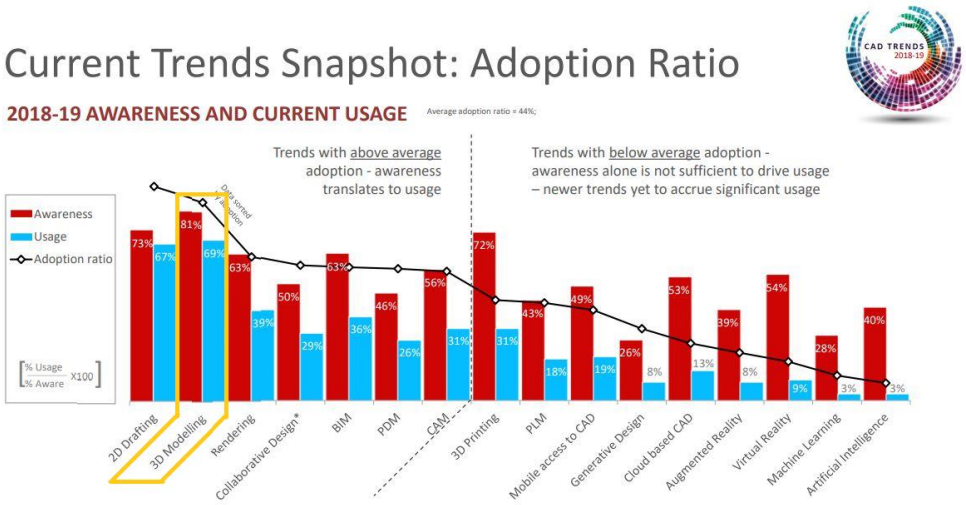


Figure 1.3. Current trends on different working environments (Business Advantage 2018/19 Survey)

Though developing technologies propose new ways of 3D modelling to create a better perception environment for end users, there are still major problems in benefiting low level representations as data sources to construct 3D digital models, which is referred as reconstruction/regeneration in this research. Since the data sources for regeneration processes can be different for one project to another, many drawbacks can be observed. For example, cases using aerial images suffer from excessive amount of data samples, time consumption and need for highly qualified personnel for conversion (Suveg & Vosselman, 2004). In Pu and Vosselman's paper (2009), the problem on manual reconstruction is pointed out. They coined that when working manually, the creation of city models is a slow and expensive process since city models include many buildings in which complexity differs immensely. In their work El-Hakim, Beraldin, Picard & Godin (2004) also mentioned that modelling from scratch with CAD software might result in low-precise 3D models.

As implied above, it is tricky to perceive discrete information even for a qualified human perception. Moreover, employing one data source with a possibility of missing information makes a reconstruction process even trickier. It is obvious that the need for precise data, qualified personnel or human intervention is inevitable in most cases.

Nonetheless, obtaining them is not always possible, and errors, precision, time consumption problems can emerge all together. In this context, an automated reconstruction process with the state-of-the-art methods can solve the problems indicated above. The following research questions should be answered to achieve this:

- Can deep learning and digital geometry processing techniques take place in the automation process of transforming low level to high(er) level?
- Which deep learning and digital geometry processing techniques are more suitable for the automation process?
- How much time will be spent for the process?
- Do accuracy and precision of the reconstructions meet the needs of industry?

1.3. Aim and Objectives

The aim of this study is to transform low level information to higher level for enabling a precise and automated reconstruction process with the state-of-the-art automation methods, which all the industries can exploit from. While low level information is extracted from 2D environment, high level information is constituted in 3D. In other words, 3D digital model reconstructions can be automatically obtained with 2D technical drawings by using deep learning and digital geometry processing techniques. Yet, only architectural usage is covered in the scope of this dissertation since different usage areas need distinct datasets.

Moreover, objectives of this study can be listed as follow:

- To search for an appropriate deep learning architecture to automatically extract relevant information from 2D architectural drawings.

- To generate a dataset in order to facilitate 3D reconstruction process.
- To explore convenient digital geometry processing algorithms for transforming low level information gathered from the 2D dataset into a higher level in 3D.
- To see the level of complexity of reconstructed 3D model.

1.4. Contributions

The endeavor of this dissertation is putting effort into integrating low and high level information to each other to achieve automatic 3D model reconstruction that can be benefitted from in all the disciplines for overcoming the problems noted before and enhancing perception of end users. Main contributions are to prepare new floor plan and elevation datasets since point of subject is architectural usage, and to automate 3D reconstruction process with state-of-the-art deep learning and digital geometry processing methods utilizing these datasets.

The proposed method takes account of the major building components available in the datasets. These components are extracted to be transformed into vectors, and subsequently, 3D model generation is performed with this information. As a result, a 3D model is automatically constructed with elevation and floor plan drawings with this proposed method.

1.5. Disposition

This dissertation consists of 6 main topics. Chapter II presents related work to understand current developments with respect to architectural digital model reconstruction. In Chapter III, implementation of this dissertation is explained. In Chapter IV, results of the implementations are discussed. Chapter V illustrates two case studies. Finally, Chapter VI draws attention on the outcomes, limitations and future work.

CHAPTER 2

RELATED WORK

There are many research topics with respect to data utilization for reconstruction. Employing different level of information as data sources for such model generation processes broadens the end users' perception since different levels imply various semantic and geometrical information. However, there are still major challenges with respect to transforming low level information to higher level. Therefore, reconstruction applications have become the subject of interest in many disciplines since the beginning of 70s (Barillot, Gibaud, Scarabin & Coatrieux, 1985; Tom, Medina, Garreau, Jugo & Carrasco, 1970; Ware & Lopresti, 1975). Yet, this dissertation only focuses on architectural usage, so all the related work above is reviewed under architectural 3D reconstructions.

History of automatic regeneration of architectural models goes back to 90s. Extracting lines or objects from an image was already in the literature during that time (Burns, Hanson & Riseman, 1986; Koutamanis & Mitossi, 1992), however 2D representations remained inadequate with the need of 3D databases in built-in areas for planning, analyzing and simulating purposes within all the stakeholders. During the late 90s, new studies have been emerged with the idea of using Laser Imaging Detection and Ranging (LiDAR), airborne imagery among other technologies in such a research area. While some researchers preferred fully automatic reconstruction by using images, some utilized laser-scanning data. Also, there are semi-automatic examples that used point clouds with partial human interaction (Brenner, 2005).

All of these contributions drew attention of the community, and the number of studies has increased year by year. In fact, in Figure 2.1, Gimenez, Hippolyte, Robert, Suard, and Zreik (2015) illustrates the cumulative number of peer reviewed publications in this research environment using the keywords “automatic 3D reconstruction”, “3D reconstruction” and “existing building”. It is obvious that reconstruction of buildings has gained practical importance.

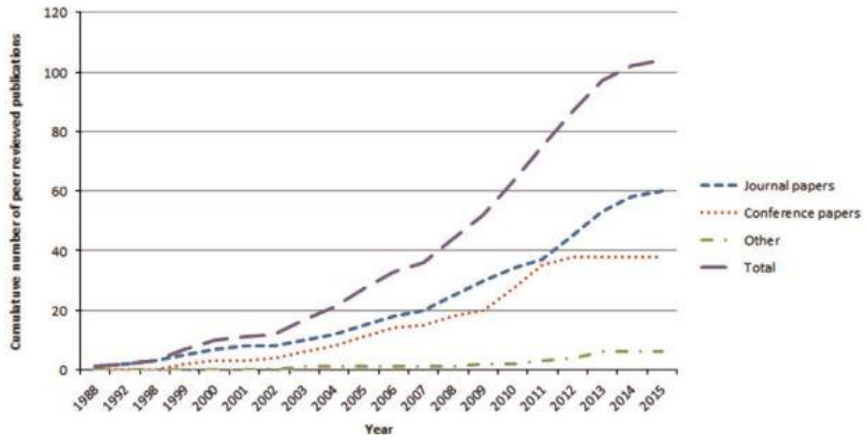


Figure 2.1. Interest in research of 3D reconstruction per year (Gimenez et. al., 2015)

When the literature is reviewed, it can be seen that reconstruction of building models is achieved by adapting data sources which are low level for computers but expose different level of semantic information for human perception. Hence, regeneration processes are investigated in two major topics. For the first topic, low level building representations as data sources should be observed to understand how they are taken advantage of. For the latter, conversion to high level techniques, which can be referred as model generation in a virtual environment, should be analyzed for utilizing the most convenient approach.

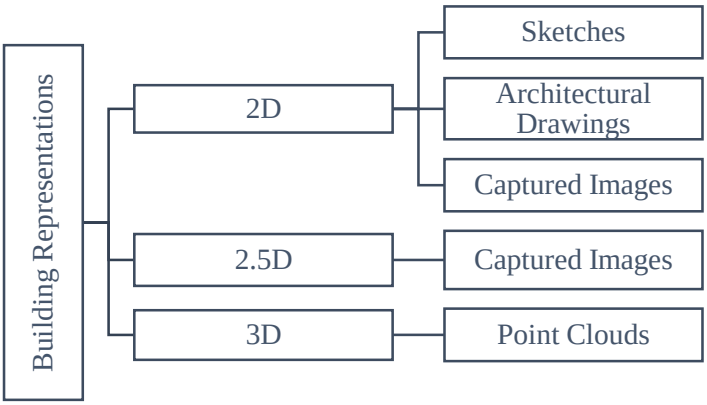
2.1. Low Level Information: Data Sources

Each designed object has its own unique representations that are based on mostly similar shape grammar. As indicated before, these representations are considered as

rich data sources for reconstruction processes since they include many information. Nonetheless, it is important to be aware of the possibility of missing information in the low level representations. As an instance, it is hard to fully perceive an object with only looking at images due to lack of 3D clues even for a qualified human.

As in all fields, there are different representation techniques in architecture. A building can be represented within 2D, 2.5D and 3D environments. Each representation has its own features with different meanings. For example, architectural drawings are represented in 2D pixel-wise environment and involves lines with several widths which are not arbitrarily chosen. Instead, they indicate vital figurations such as structural elements, openings and furnishing. As another example, 2D or 2.5D images consist of building representations with different textures and colors. These representations can be extended, and Table 2.1 shows building representation alternatives.

Table 2.1. *Building Representations*



Although representation sources can be classified according to dimensions, they are fundamentally low level information sources that consist of points, edges and corners in a computer screen. Therefore, each representation style is reviewed mainly according to the availability, processing time, processing complexity and level of

detail to comprehend the possibility of using them as a data source on a reconstruction process.

2.1.1. Sketches

Free-hand drawings are the initial step of a design process to communicate with ease. Either in pen-and-paper or online/digital format, from conceptual ideas to more concrete illustrations can be achieved with these drawings. Sketches naturally are more subjective than any other representation style. Although many information can be interpreted from sketches, subjectivity may cause misreading the architectural elements. Also, it may be hard to find proper architectural element information since lines are hand-drawn without considering any precision (Figure 2.2).

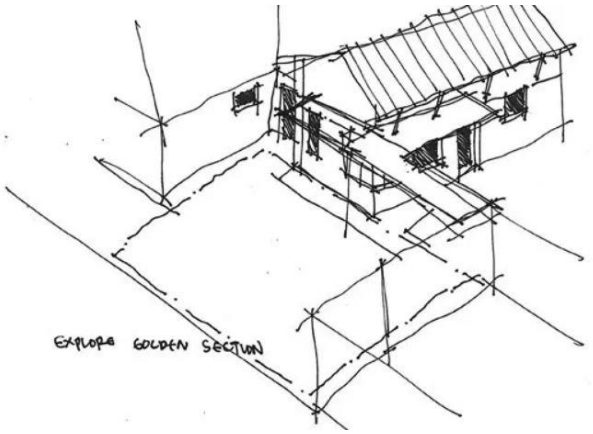


Figure 2.2. An architectural sketch (Borson, 2019)

The information in sketches has been considered as a valuable data source for many researchers. In the sketch-related works, some researchers directly use sketches as raster format (Camozzato, Dihl, Silveira, Marson & Musse, 2015; Juchmes, Leclercq & Azar, 2005; Rossa, Camozzato, Marson & Hocevar, 2016; Shio & Aoki, 2000) (Figure 2.3), and some others prefer using an interface or human interaction in a virtual environment (Eggli, Hsu, Bruederlin, & Elber, 1997; Zeleznik, Herndon & Hughes, 2007).

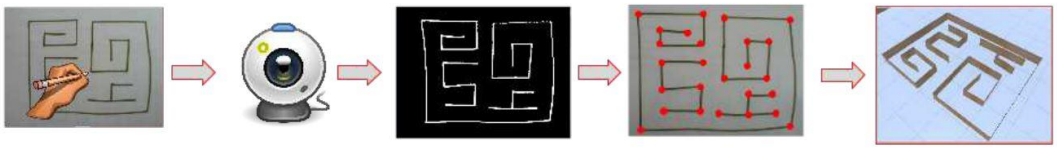


Figure 2.3. The original sketch, its synthesis and reconstructed 3D model out of it (Rossa et.al., 2016)

Most of the researchers that use sketches as a dataset for 3D reconstruction reveal the fact that it highly depends on human perception and intervention. Although there are improvements in sketch-based modelling, according to the broad survey conducted by Ding & Liu (2016), some challenges still can be encountered. Contours and lines in a sketch are essential, but they may not be enough to regenerate a complete model. Also, it is hard to find depth information from hand-drawings. Discontinuity and irregularity in lines may cause blobby and imprecise 3D models.

2.1.2. 2D Architectural Drawings

Architectural drawings are the main information source for a building design. All of the information from quantitative specifications to material properties can be derived. They are used by architects and many others to transfer design ideas into a proper proposal.

A set of architectural drawings may contain floor plan(s), elevation(s), section(s), oblique(s) and perspective(s) (Figure 2.4). Each drawing in this set includes a set of rules in terms of line thicknesses and shapes. While thick lines or hatched regions remark structure and walls, thinner lines indicate openings and annotations. These semantic and geometrical meanings in 2D are a powerful and adequate tool for 3D reconstruction processes. All the relevant information necessary for a 3D model can be deduced from them. Hereby, 2D architectural drawings become drastic and comprehensive data source, and do not need any complimentary source for a regeneration process.

Figure 2.4. A set of architectural drawings (Stephens, 2011)

Architectural drawing-based studies in the literature mainly focus on floor plans (Gimenez, Robert, Suard & Zreik, 2016; Lewis & Séquin, 1998; Lu, Tai, Su & Cai, 2005; Pandey & Sharma, 2016). Figure 2.5 shows example input and output. Either in vector or raster format, researchers who manage to reconstruct 3D models show that they need human interference to decide on building height. Eventually, all of the above applications become semi-automatic processes.

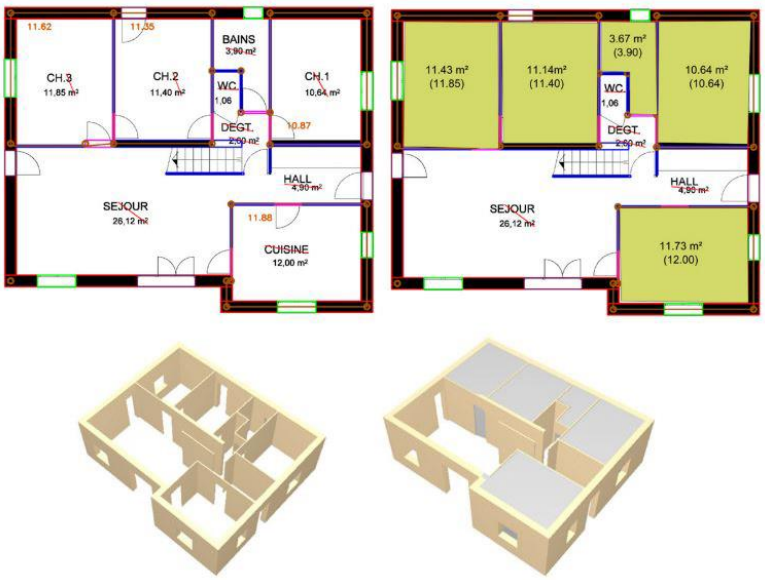


Figure 2.5. Recognition of building elements and 3D reconstruction (Gimenez et.al., 2016)

As an advantage, this type of data includes detailed geometric and semantic information much more than sketches. They can be present both for built and not-built-yet buildings. Therefore, a data source from 2D-drawing-documentation of such buildings can be obtained readily. As a disadvantage, as Huang, Lo, Zhi and Yuen (2008) pointed out that drafting mistakes may cause errors. Also, irrelevant information such as text and annotations can lead a noisy data. It may be hard to distinguish openings if architectural plans are the only data source.

2.1.3. Captured Images

Images are like sketches, and they can be captured with several tools. A fair amount of architectural elements can be extracted from this 2D medium similar to architectural drawings and sketches. However, it takes multiple steps to document a building inside out unlike architectural drawings since a drawing can show inner and outer information at the same time (i.e. floor plans). There is a wide range of image types such as monocular, stereo, aerial and street-level. The main distinguishing characteristic is that captured images involve wide range of color codes (e.g. trees are green, roads are gray). Although they are composed of edges and corners with colored

lines, they do not indicate generalized semantic information with varying thicknesses as in architectural drawings. They are specific to the scene they are in, so each type has its own characteristics (Figure 2.6).

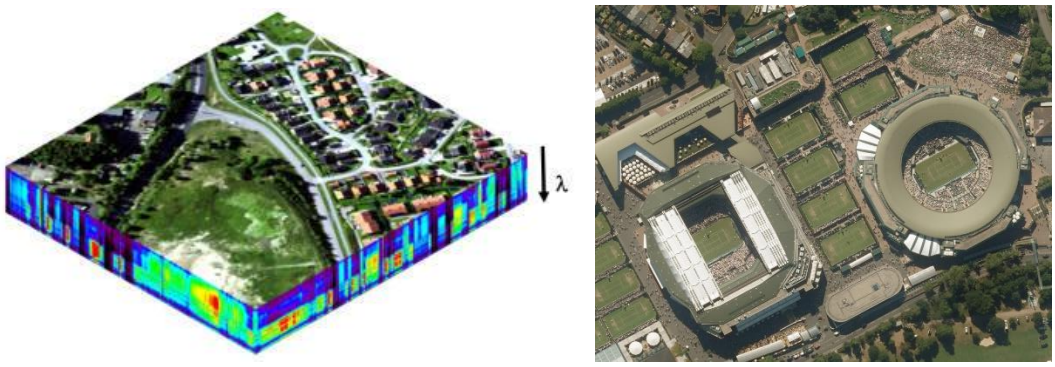


Figure 2.6. Different image types (Left: Hyperspectral imaging (HySpex, n.d.); Right: Aerial imaging (Getmapping, n.d.))

From façade reconstruction to mass generation, models with different level of details can be generated by benefiting from image features. Naturally, this depends on the detail level of the objects in an image and quality of the image itself. For instance, researchers that use large scale images or maps mostly end up with only mass generations (Haala, 1995; Huang & Kwoh, 2007; Willneff, Poon & Fraser, 2005). Street-view or façade images ensure more detailed 3D models since multi-view images are preferred (Müller, Zeng, Wonka & Van Gool, 2007; Wefelscheid, Hänsch & Hellwich, 2011; Xiao et.al., 2008). Imagery with depth reference simplifies the work because height of a building can be interpreted with it. The depth in an image is a tricky notion which is difficult to position in this context, and yet there are several implementations using depth information in image processing (De Reu et.al., 2014; Henry, Krainin, Herbst, Ren, & Fox, 2012; Yue, Chen, Wu & Liu, 2014).

According to Musialski et.al. survey (2013), the biggest perk of using images is the accessibility. Since every year many images are taken, the images as data source are

getting larger, and mostly exchangeable in online platforms. However, concerns about the quality and scalability of the end model are not in vain. Urban scale images generally cause only mass models rather than a complete building envelope. Also, taking high-quality images can be difficult considering the environmental factors such as cars, trees, and even sunny or cloudy days. All of these factors may impair the image quality, and accordingly, spoil reconstruction precision.

2.1.4. Point Clouds

Point clouds can be described as digital versions of existing objects. They contain a considerable amount of data points on the exterior surfaces of the objects. They show a similarity to captured images in terms of gathering inner and outer view of the object. Likewise, inner and outer surfaces are accumulated separately. Point clouds can be acquired on site from different optical sensors such as LiDAR, cameras and infrared scanners. While sketches, architectural drawings and images are digitized documentations in 2D, they are the direct digital versions of the objects in 3D format. The distinguishing feature of them is that they are positioned in a 3D environment which allow users to benefit from height. Even though they include 3D clues, they are still composed of low level information: points, so they need a processing for precise reconstruction. (Figure 2.7).

Figure 2.7. Laser scanning of a building under construction (Tang et.al, 2010)

There is a growing tendency to use point clouds as a data source because of the increasing ability of point cloud processing. Many technological media provide multi-view data. For example, LiDAR can serve for ground mapping as well as aerial mapping. Currently, Unmanned Aerial Vehicles (UAVs) are widely used, so gathering data is easier. As a result, it is obvious that this type of data source gives an opportunity to reconstruct indoor and outdoor scenes (Ochmann, Vock, Wessel, Tamke & Klein, 2014; Previtali, Barazzetti, Brumana& Scaioni, 2014; Tang et.al, 2010; Thomson & Boehm, 2015; Xiong, Adan, Akinci & Huber, 2013).

Nonetheless, point clouds include huge amount of data points (Figure 2.8) which make it difficult to reach semantic information, and tend have noise and inconsistency. Pre-processing and/or post-processing should be applied to prevent such drawbacks (Wang, 2013). There might be some problems to generate realistic architectural forms depending on the capturing technique. Trees and narrow streets can cause loss of information since they can overlap with the building. As a result, incomplete reconstructions become inevitable (Sun & Salvaggio, 2013).

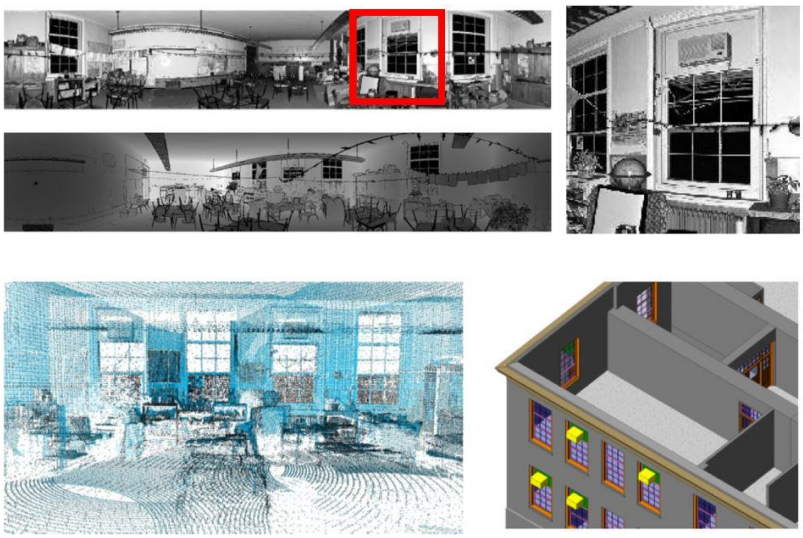


Figure 2.8. Reconstruction with point cloud data (Xiong et.al., 2013)

2.1.5. Data Source Comparison

It can be observed that building representations has their own unique characteristics. Being low level is the main common factor for utilizing them as data source for a precise high level transformation. Information that they include can be properly processed with many algorithms. Moreover, geometrical and semantic references in them make 3D model regeneration possible. Thereby, each building representation becomes a potential candidate for creating a dataset for such a generation process.

While sketches and architectural drawings can be counted as documentations prepared beforehand, captured images and point clouds can be considered as on-site acquisition. Advantages and disadvantages of each category should be regarded beforehand to arrange an appropriate dataset. Therefore, availability, level of detail, time of processing, complexity of processing and semantic information are compared to each other under the categories of “documentation prepared beforehand” and “on-site acquisition” in the scope of this research.

Documentations prepared beforehand is obtainable with ease in most cases. However, it is important to consider the fact that availability rate decreases from hard-copy drawings to CAD drawings, and to sketches. Since they are in pixel-wise environment, their processing time and complexity are lower than 2.5D and 3D representations like captured images and point clouds. Sketch-based studies prove that information with few details is the biggest drawback. Due to lack of detail, subjective representation techniques, possibility of irregular and overlapping lines and least semantic information, sketches become out of interest for this research. On the other hand, architectural drawings include all the semantic information. They are mostly represented in a shape-grammar that can be understood by all the stakeholders. They are present both for built and not-built-yet buildings, so information from any design can be revealed. Even though they might be corrupted while scanning or in time,

representation styles do not usually vary person to person as in sketches. Hence, architectural objects become easier to recognize and process.

For the on-site data sources, the biggest obstacle is the necessity of equipment. Gathering data with aforementioned tools, processing them either manually or automatically, and picking out the semantic information are complicating. Certainly, successful results are achieved in the literature because they include many reliable, detailed and up-to-date information more than building documentation; but, they are highly affected by the environmental conditions. For example, if any shadow is occurred, the photograph, and so input data, would be spoiled. Also, it is hard to find an on-site data source for a building that is built in much earlier times. They are only applicable for extant buildings.

A part of Gimenez et.al. (2015) excessive survey demonstrates the comparison of such data sources. In Table 2.2 and Table 2.3, each data is compared to others with respect to data acquisition, data characteristics, data processing and accuracy of the resulting model.

Table 2.2 *Comparison of building documentation (Gimenez et.al., 2015)*

Themes	Sketches	CAD plans	2D paper plans
Data acquisition			
1. Availability	Low	Medium	High
2. Quality	Medium	High	Variable
3. Up-to-date	Low	Medium	Medium
Data characteristics			
7. Data type	Image	Vector format	Image
8. Level Of Detail(LoD)	Low	High	High
9. Data volume	Low	Medium	Low
Data processing			
10. Time of processing	Medium	Medium	Medium
11. Degree of automation	Medium	High	Medium
12. Complexity of the processing	Medium	Medium	Medium
Accuracy of the resulting model			
13. Geometry	Medium	High	High
14. Topology	Low	High	High
15. Semantic	Low	High	High

Table 2.3. *Comparison of on-site data (Gimenez et.al., 2015)*

Themes	Aerial images	City images	Laser scanning
Data acquisition			
1. Cost	High	Medium	High
2. Time	Medium	Medium	High
3. Influence of size and complexity of the scene	High	High	High
4. Influence of environmental conditions	High	High	Medium
5. Equipment portability	Low	High	Medium
6. Equipment durability and robustness	High	High	High
Data characteristics			
7. Data type	Remotely sensed images	Images	3D cloud points
8. Level Of Detail(LoD)	Poor (0–2)	Medium (0–3)	High (0–4)
9. Data volume	High	Medium	High
Data processing			
10. Time of processing	Medium	Medium	High
11. Degree of automation	Medium	High	Low
12. Complexity of the processing	Medium	Medium	High
Accuracy of the resulting model			
13. Geometry	Medium	Medium	High
14. Topology	Medium	Medium	High
15. Semantic	Low	Medium	Medium

In the light of the literature, gathering data from building documentation is much easier than from on-site acquisition in terms of cost, time spend and need of equipment. Also, processing the data from building documentation rather than on-site data is less complex since data volume, level of detail and time of processing is fewer. Considering the comparison results, it is determined that building documentation is more applicable than on-site data in the scope of this research.

While discovering the building documentation, it is also observed that architectural drawings are more proper than sketches. They are more available than sketches, especially for built environment. They include more topological and semantic information. Therefore, end results are more promising according to the literature. As a result, architectural drawings are set as the data source for this research.

2.2. Transformation to Higher Level: 3D Model Generation

Many applications are present in the literature that transform low level information gathered from aforementioned data sources into 3D level. There are extensive surveys to investigate processing methods on these data types (Bourke, 2018; Chen, Lai & Hu, 2015; Chiabrando, Sammartano & Spanò, 2016; Olsen, Samavati, Sousa & Jorge, 2009; Özdemir & Remondino, 2018; Tang, Huber, Akinci, Lipman & Lytle, 2010). However, this topic covers only 2D architectural drawings since they include promising information for higher level transformation. According to the literature, it is seen that 3D reconstruction is only attainable with architectural information extraction which varies from conventional to state-of-the-art applications. Consequently, model generation with architectural drawings are examined in twofold: conventional computer vision and machine learning applications.

2.2.1. Conventional Computer Vision Techniques

Automated 3D model generation has always been an important part of computer vision. Many architectural element extraction and modeling applications with conventional computer vision techniques are present in the literature. Most of them are generally based on image processing for removing irrelevant information in architectural drawings, extracting architectural features and generating 3D building models.

Some researchers employ shape-based approach which is based on line & primitive shape and pattern detection. For example, Macé, Locteau, Valveny and Tabbone (2010) first use a preprocessing algorithm for removing the noise in architectural drawings to separate text and graphics. This preprocessing algorithm is based on Tombre, Tabbone, Pélissier, Lamiroy and Dosch's study (2002) which is an improved version of Fletcher and Kasturi's (1988) method. While Fletcher and Kasturi use Connected Component Generation that encloses strings with rectangles with respect to area/ratio filtering, Tombre *et.al.* improved study transforms the same methodology with some additions. Since architectural drawings include dashes, punctuation marks among other elongated shapes, they separate a drawing into three parts: components presumed to be text, components presumed to be graphics, and small components which serve for both by dashed lines detection and by character string extraction (Figure 2.9). After preprocessing, Mace et.al. extract the architectural elements by applying a combination of HT and image vectorization based on Bresenham algorithm (Bresenham, 1965).

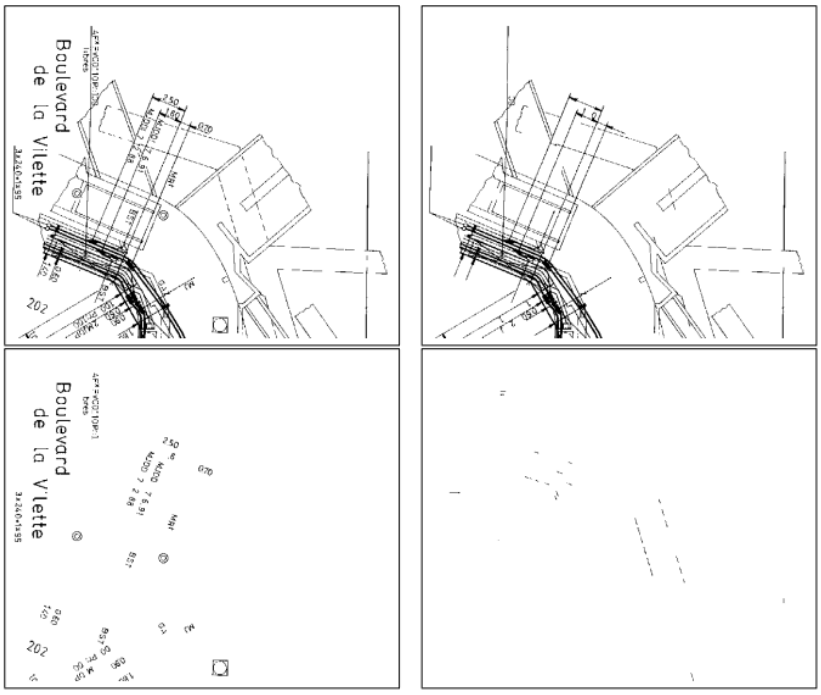


Figure 2.9. Text, dashed lines and graphics separation (Tombre et.al., 2002)

In another example, Gimenez *et.al.* (2016) use Tombre *et.al.* approach for text and graphics separation as preprocessing. Later, they divide relevant graphics into windows and openings according to a pattern recognition function in accordance with shape specifications such as line length and parallelism. By doing so, they achieve a building space identity with topological and geometrical information that they extract from previous steps and end up with a building model by extruding vectorized geometries. The necessary height value for extrusion step is given as default. Thereby, it becomes a semi-automatic process. Another approach (Or, Wong, Yu and Chang, 2005) uses the same preprocessing step for noise removal in architectural drawings and utilizes vectorized polylines of windows, walls and doors to generate block of polygons. Walls are represented with thick lines while windows are as rectangles and doors are as arcs. They prefer to create blocks or surfaces out of wall edges and extrude them. However, this example is also not fully precise since they predefine wall height and opening dimensions & locations as constant values (Figure 2.10).

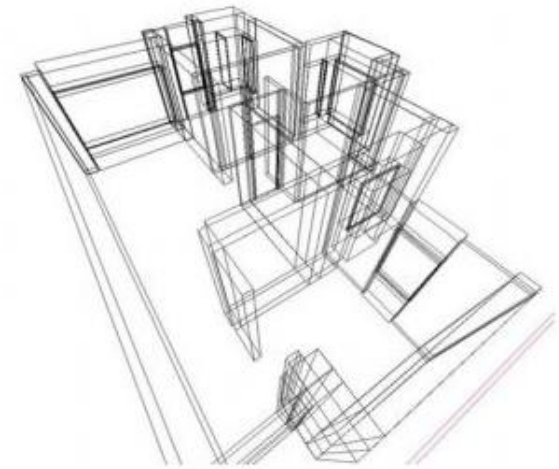


Figure 2.10. Model generation example by extruding walls as blocks (Or *et.al.*, 2005)

Relation-based algorithms are primarily based on adjacency and intersection of lines in a vectorized architectural floor plan. Most of the researchers apply a preprocessing step for removing the noise in drawings similar to previous applications. Domínguez, García and Feito (2012) propose a semi-automatic detection application that captures line segments, and use Wall Adjacency Graph (WAG) to extract walls and openings with a set of rules based on intersection and parallelism (Figure 2.11).

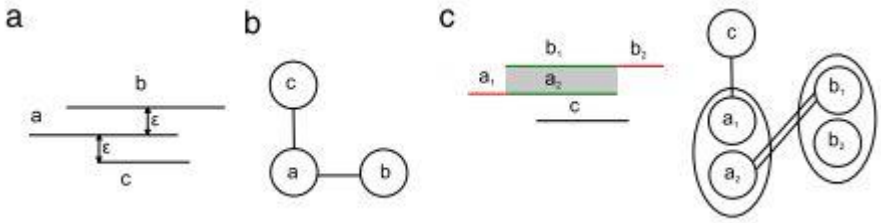


Figure 2.11. Wall detection process by Domínguez et.al. (2012) (a: Three parallel lines, b: Initial WAG, c: Wall detection with line a & line b, and representation of hierarchical relations)

Likewise, some other implementations can be observed that use adjacency information (Horna, Meneveaux, Damiand & Bertrand, 2009; Zhi, Lo & Fang, 2003). While Horna *et.al.* use consistency constraints to define walls and openings for 3D reconstruction, Zhi *et.al.* apply graph-based loop algorithm to rebuild enclosing spaces and topological relationships. Reconstruction process proposed by Horna *et.al.* relies

on leaving the openings as blank. Then, walls are extruded through boundaries of windows and doors. Finally, all the wall and opening mass are merged. In Zhu, Zhang and Wen's research (2014), structural components are recognized with a shape-opening graph that correlates wall and openings with a loop-searching algorithm. Wall edges are extruded to a default wall height. Later, opening information are extruded at the limit of default values. At last, openings are cut off from the wall extrusion (Figure 2.12).

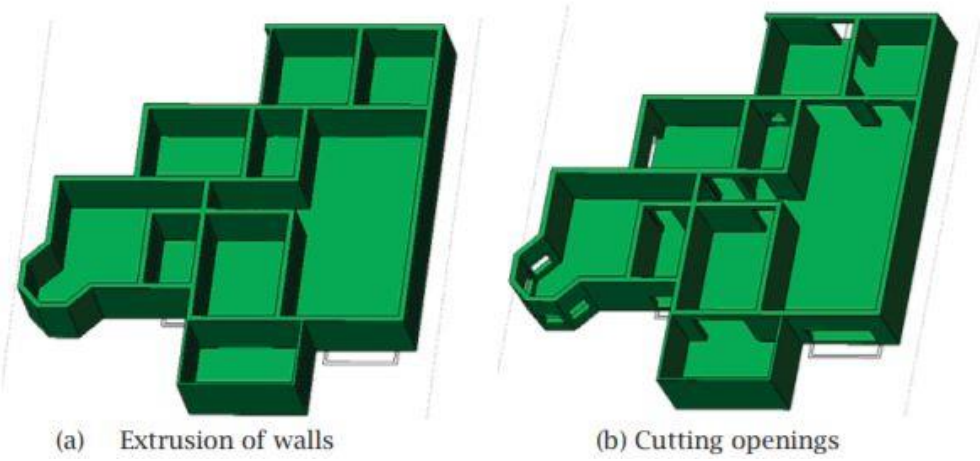


Figure 2.12. Model generation from edges (Zhu, Zhang and Wen, 2014)

2.2.2. Machine Learning Techniques

Development of computation power enables machine learning usage in automatic 3D reconstruction. From traditional machine learning to deep learning applications have started to be utilized in such a research area. According to the literature, there are multiple segmentation based studies by using 2D architectural drawings dataset. As the name implies, segmentation based approach mainly depends on dividing a drawing into meaningful parts such as walls and openings. Even though semantic segmentation can be achieved with a considerable amount of machine learning algorithms, there is a main difference between traditional machine learning techniques with deep learning.

While traditional ones need a preprocessing with feature descriptors situated on an image, deep learning does not require such a step.

A feature descriptor is a representation of visual features of an image. These features identify characteristics such as shape, texture and color. The main aim is to recognize useful information that can be later converted to vectors. As feature descriptors, there are Histogram of Oriented Gradients (HOGs) (Figure 2.13), Bag-of-words (BOW), Scale-Invariant Feature Transform (SIFT) among others (Mukherjee, Wu & Wang, 2015). Their working principle relies on filtering and thresholding, and each descriptor has its own eligibility. For example, while SIFT is successful at noisy data and uses Gaussian filtering, Features from Accelerated Segment Test (FAST) is prominent in corner detection with a decision tree vectorization method. These descriptors may be used solely in shape recognition tasks without needing any further algorithms. However, in more complex tasks such as feature extraction and image segmentation, they might be used as preprocessing step especially for traditional machine learning algorithms (Chen, Li, Ren & Qiao, 2015; Gao, Zhou, Ye & Wang, 2017; Goyal, Bhavsar, Patel, Chattopadhyay & Bhatnagar, 2018; Goyal, Chattopadhyay & Bhatnagar, 2018). On the other hand, deep learning applications do not require this step. They achieve the whole feature extraction step within their structure.

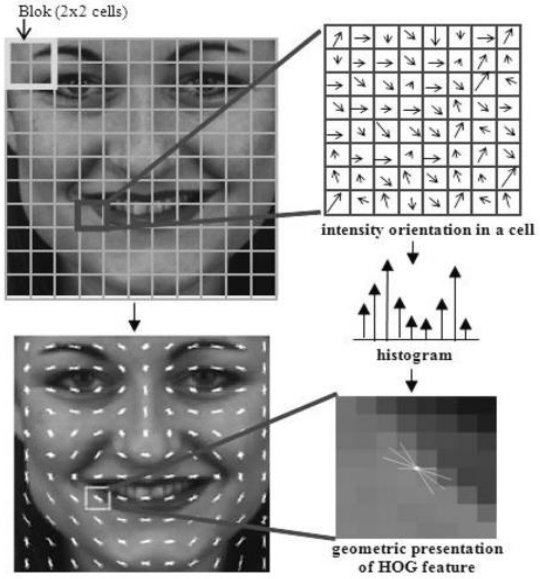


Figure 2.13. An example of HOG visualization (Greche & Es-Sbai 2016)

Traditional machine learning applications can be found in the architectural feature extraction and model generation applications. de las Heras, Mas and Valveny (2011) use overlapping square patches method. After labelling each patch as "wall" and "not wall", Principal Component Analysis (PCA) (Jolliffe, 2011) and K-Means (Elkan, 2003) are utilized for extracting and clustering those patches to generate a vocabulary, which is used for labelling the architectural components. Later, the same authors conducted a research based on unsupervised learning for wall patching (de las Heras, Fernández, Valveny, Lladós & Sánchez, 2013).

Other than traditional machine learning, some deep learning approaches can be noticed as well. Yang, Jang, Kim and Kim (2018) apply a Convolutional Neural Network based on Ronneberger, Fischer and Brox's implementation (2015) to analyze architectural floor plans. Dodge, Xu and Stenger (2017) parse the floor plan images with Fully Connected Network (FCN) described in Long, Shelhamer and Darrell's research (2015) to gather semantically segmented floor plans. They extrude the wall

to a default value and leave the openings as complete blanks, which causes the generated model not to be fully precise (Figure 2.14).

Figure 2.14. Model generation example (Dodge, Xu & Stenger, 2017)

Similarly, Liu, Wu, Kohli and Furukawa (2017) vectorize raster architectural floor plans by junction and per-pixel classification. They combine heat map regression (Bulat & Tzimiropoulos, 2016) and deep learning (He, Zhang, Ren & Sun, 2016) to predict heat maps at pixel-level (Figure 2.15). After pixel-wise conversion, 3D pop-up models are generated by extruding the vectors to a predefined value that again hinders the whole procedure from a fully automated process.

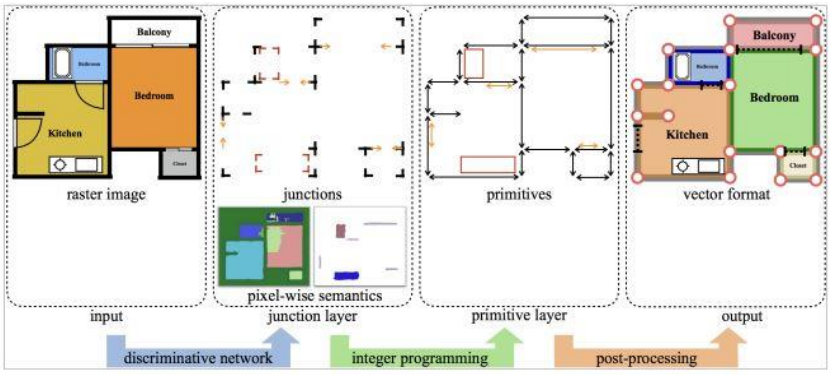


Figure 2.15. Raster to vector implementation process (Liu et.al., 2017)

2.3. Remarks

Choosing an appropriate data source as well as technique mainly depends on the research context. If documentation prepared beforehand is not available, on-site data acquisition is mostly used for reconstruction processes. Likewise, if on-site acquisition is barely possible to achieve, documentations prepared beforehand is utilized by most of the researchers. For instance, if a building's environment is enclosed, or includes many trees or narrow streets around it, building documentation is opted. Dataset is also highly related with the detail level of a reconstructed model. Dataset features as input effect the end model, so data sources should be considered beforehand according to the literature findings. The other important concern is budget of the project. The budget is not always enough to cover workload and equipment such as cameras and laser scanners. Understanding the connection between building condition, required detail level and budget guides researchers to specify the most convenient method for obtaining a dataset.

It is observed that 2D drawings are more available than on-site data in accordance with the literature. On-site data provides more details but semantic information is not proportional to it. Since it is difficult to scan a building inside & outside, and manipulate the merged result, it is proper to employ architectural drawings to process everything at the same time. By doing so, complexity of the data and time-spent are mostly eliminated. Also, it is seen that using only architectural floor plans is inadequate, especially in the model generation phase. It is not possible to generate a realistic 3D model without the knowledge of building height, and specific locations/dimensions of openings. Therefore, architectural floor plan dataset needs complementary resources. As a result, 2D floor plans and elevations are determined as data source to eliminate the possibility of missing information in the scope of this research.

All the methods discussed above show that there is no specific methodological solution for how to regenerate a 3D model with 2D drawings. Yet, a comprehensive solution can be obtained for a high automation process by choosing proper methods with a convenient dataset. According to the observations, conventional computer vision methods highly depend on the image quality and principally work with thresholding, so human intervention is needed and automation of such a process fails. Also, conventional computer vision and traditional machine learning algorithms need a preprocessing step due to the need for noise removal or extracting feature descriptors. Although traditional machine learning algorithms show more promising results than conventional computer vision applications, the demand for preprocessing is a big obstacle. At this point, deep learning approaches attract the attention. Instead of using one algorithm specific to one dataset containing similar characteristics, deep learning serves for a bigger problem space with a wide range of dataset characteristics since it is more adaptive.

Lastly, it is noticed that model generation process and level of details differ according to input, geometrical complexity and desired semantic richness. If extracted features are not accurate enough, then 3D models and semantic richness become imprecise. Also, geometrical complexity of 3D models increases depending on the extrusion style. Therefore, extrusion of all the wall segments and subtraction of openings are adapted for this dissertation to reduce geometrical complexity.

This dissertation aims to benefit from state-of-the-art methods that are widely used in current research fields due to valuable potential and compatibility on automation of 3D regeneration. Although, there are many research examples in the literature, these methods are slightly observed in architectural usage. To sum up, 3D reconstruction with 2D dataset via deep learning and digital geometry processing is utilized for this research, and it can be extended to any other discipline with the relevant dataset.

CHAPTER 3

DEVELOPMENT OF THE IMPLEMENTATION

Digital models are at high interest in the realm of architecture and engineering industry. 3D modeling of objects/products/buildings in a virtual environment is becoming easier with the emerging design tools, and yet some obstacles can be encountered while compounding different level of information. Obtaining precise data, spending considerable time and having qualified personnel can be counted as some of the problems for such a process.

As discussed in the previous chapter, automated 3D architectural reconstruction methods mostly rely on conventional computer vision, traditional machine learning and deep learning algorithms. They are developed on a specific architectural dataset which can be gathered from either on-site or documentations prepared beforehand. According to observations, building documentation is more accessible, processable and includes more semantic information than on-site acquisition. This is why architectural drawings are employed as dataset.

When focused on 3D reconstruction with architectural drawing dataset, removing irrelevant information is noticed most particularly in conventional computer vision methods as preprocessing. Traditional machine learning algorithms apply preprocessing with feature description algorithms such as HOGs and SIFT. However, deep learning algorithms do not need this step since feature extraction can be accomplished within its structure. Figure 3.1 demonstrates a comparison between traditional machine learning and deep learning algorithms in terms of working principle.

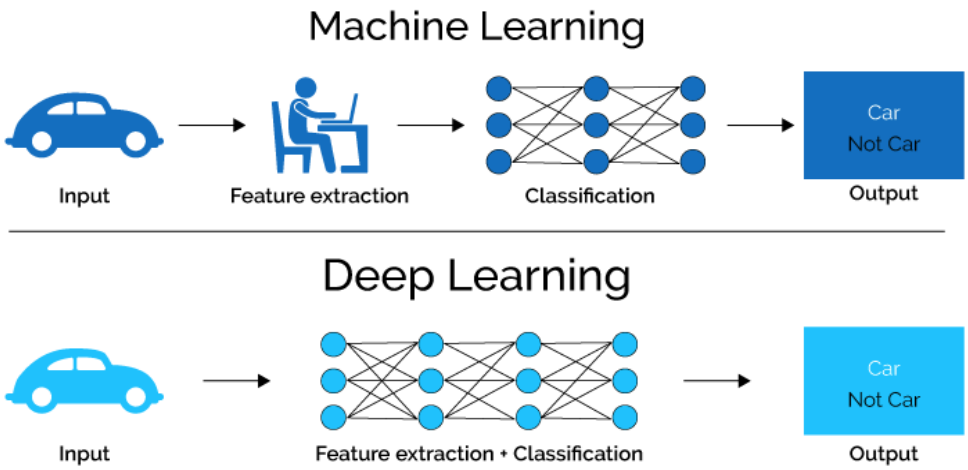


Figure 3.1. Traditional machine learning and deep learning comparison (Mahapatra, 2018)

For architectural element extraction step, using conventional computer vision techniques may suffer from a scarcely generalized solution due to required noise removal, high-quality-image dataset and special filters to specific problems. This situation causes human intervention, so desired automation level decreases. On the other hand, machine learning serves for a bigger and more generic problem space as discussed in the previous chapter. Nonetheless, deep learning algorithms reduce the number of parameters when compared to traditional machine learning. They also do not need preprocessing and use much faster activation functions while adding non-linearity to the system. More computation cost for training than traditional methods and big amount of data demands are main drawbacks of deep learning so far. However, Graphical Processing Units (GPUs), data augmentation techniques and regularizations generally overcome these problems.

There are two major approaches for model generation after vectorising the extracted information from the dataset. First one is based on extrusion of wall segments and openings separately, and then merge them. Second one extrudes a complete wall mass and subtracts openings. Second one is utilized to minimize the geometrical complexity due to the objectives of this research.

To sum up, the hypothesis of this research is put as *automatically transforming low level information to higher level is possible with deep learning and digital geometry processing techniques*. The proposed implementation is established on preparing architectural drawing datasets, learning features from them by using deep learning algorithms and vectorising the features by contouring with morphological transformations. Later, vectorized elements are transformed into 3D environment to obtain architectural model. A web-based application is then implemented to allow users to generate their own models. Figure 3.2 shows the overview of the implementation.

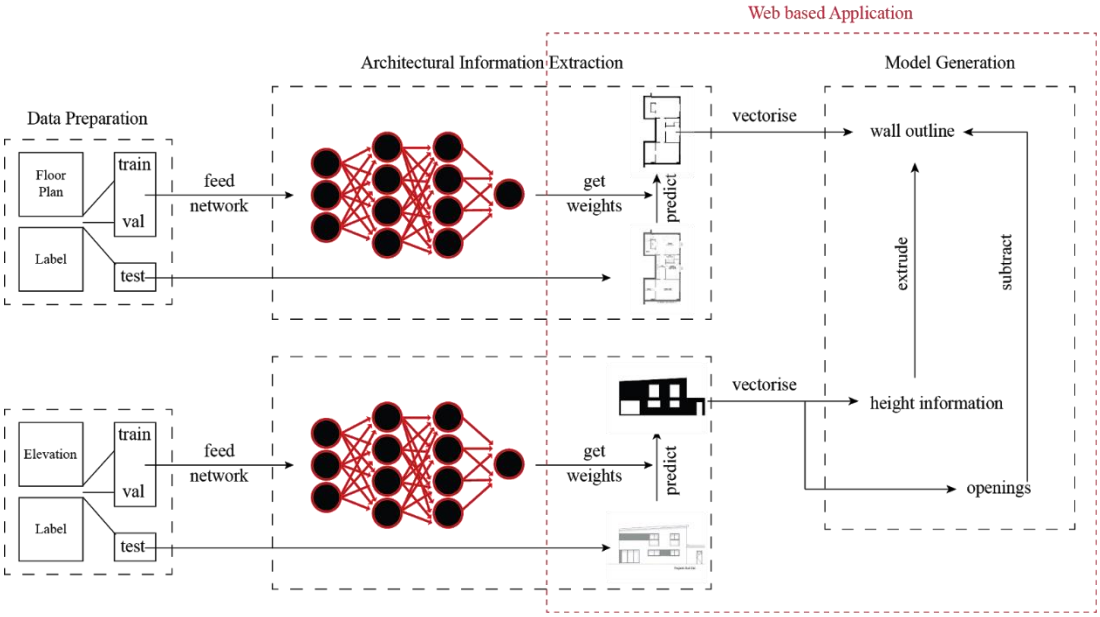


Figure 3.2. Overview of the implementation of the proposed method (produced by the author)

3.1. Data Preparation

There are different types of architectural drawings such as floor plans, sections and elevations which are semantically rich and enough to be converted to a 3D model. Each of them can readily be obtained from a 3D model while it is hard, yet not impossible, to regenerate a model using these drawings. However, it is important to consider the types beforehand. Using only floor plans do not satisfy a complete 3D

building model reconstruction because height information and openings location cannot be interpreted from them. A complementary data source should be exploited for a convenient and automated model generation. Among other types, elevation drawings exclusively reflect what is on floor plans since they are referenced to each other in a systematic way. On the other hand, sections and perspectives are tricky to be linked directly to floor plans even for a qualified person. A floor plan does not necessarily imply section details. Likewise, a section may not be able to present all the components in a floor plan. Hence, floor plans and elevation drawings are converted from potential data sources to datasets to regenerate such models. This conversion will be explained in detail in the following topics.

Reconstructing a 3D model without any human intervention requires automation in every step, so it is obvious that there should be a dataset to learn from to recognize the components automatically. However, a comprehensive dataset is scarcely available in the literature. Even though there are few raster floor plan drawing datasets (de las Heras, Terrades, Robles, & Sánchez, 2015; Rakuten, Inc., NII, & Alagin, 2017; Sharma, Gupta, Chattopadhyay, & Mehta, 2017), neither raster elevation drawing datasets nor labels for such images are present. To this respect, floor plan and elevation datasets with corresponding labels are prepared from scratch for this research. 2D architectural housing drawing images are taken from existing datasets or online platforms (ArchDaily, n.d., de las Heras et.al., 2015; Hanley Wood LLC, n.d.; Houseplans LLC, n.d; Le forum pour faire construire sa maison, n.d.; The Garlinghouse Company, n.d.; Vision One Homes, n.d.), but all of the labels are prepared manually. Datasets are then divided into train, validation and test sets. Train set is used for learning features of data samples as the name implies. Validation set is applied for checking if an algorithm tends to fail during learning the features. Lastly, test set is employed for examining if the algorithm learns the features appropriately.

3.1.1. Floor Plan Dataset

A floor plan is a scaled drawing showing a spatial layout. Details of a floor plan change according to the needs of a project. For example, a conceptual floor plan may include only wall outline while an as-built drawing might contain projections of upper levels, structural details, electrical features and landscape items (Figure 3.3). At most cases, a floor plan represents room layouts including functions and dimensions, hallways, openings, interior features and furnishing such as fireplaces and sinks.

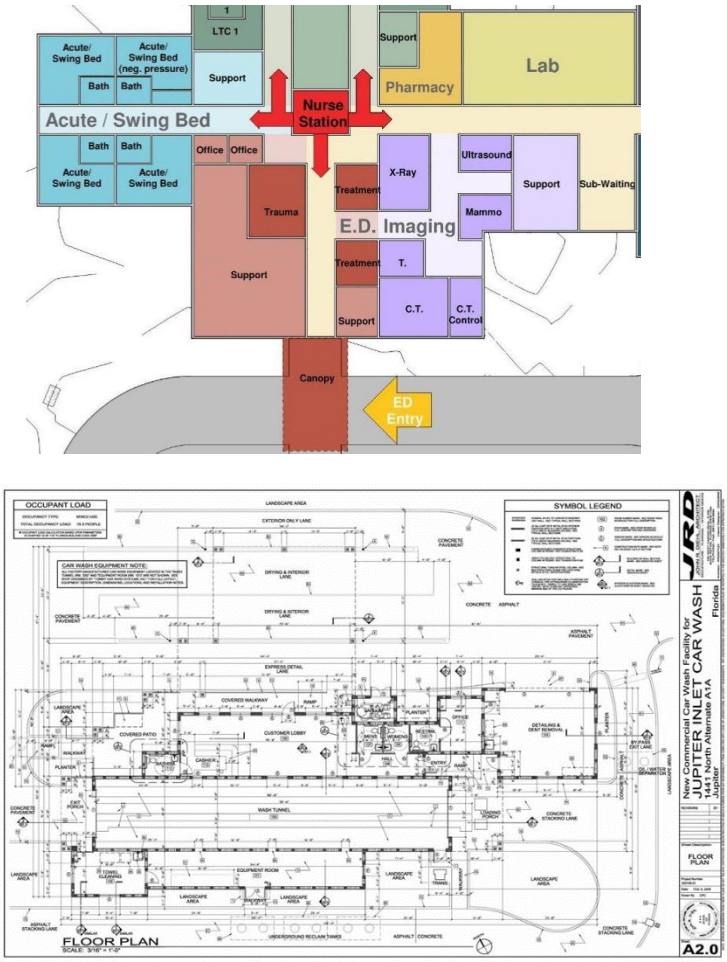


Figure 3.3. Conceptual and as-built floor plan drawing examples

The more a floor plan includes details, the more it contains juxtaposition of architectural elements and gets complex in terms of unraveling the relevant features. Selection of building type and details of its floor plan gain importance for preparing a dataset. When the scale of a building increases, the number of architectural elements and so details of its floor plan also increase regardless of being conceptual or as-built drawing. In this context, detailed conceptual floor plan drawings of one-storey residential houses become the subject of interest of this research (Figure 3.4).

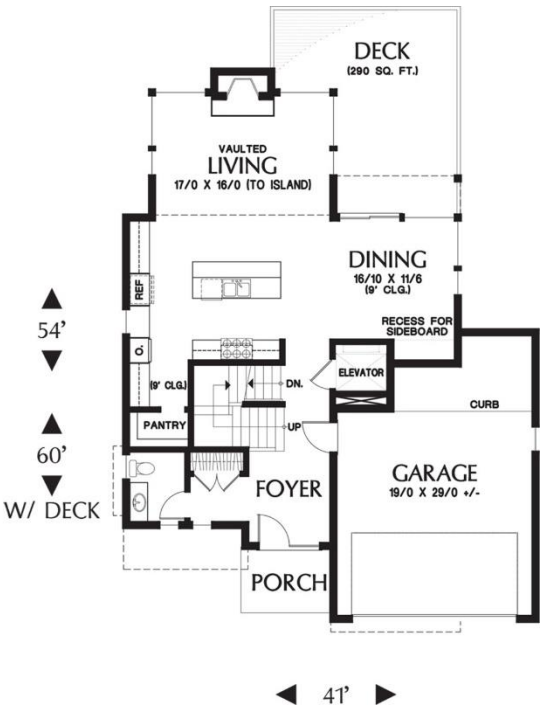


Figure 3.4. A sample from floor plan dataset

The floor plan dataset is generated from raster floor plan images representing walls, annotations, windows, doors and furnishing as shown in Figure 3.4. Although the whole graphical information above is important to represent a house in 2D environment, all the annotations, projections, furnishing and stairs are disregarded to preserve what is important for 3D model generation: *structural elements* and *openings*.

The integrity of structural elements is only apprehensible in floor plans, and that is the only source that can be utilized for generating structural elements in 3D environment. Indication of them can only be achieved by marking them geometrically. Inner and outer openings can be extracted from a floor plan as well, however they need human interference for 3D transformation since exact locations and dimensions cannot be unraveled. Considering this situation, a marking system is developed to specify essential architectural elements in a floor plan. This marking system is based on the representation of expected outcome of a training process. Therefore, it is prepared by introducing new images that contain and reflect the exact preserved geometrical information of the drawings. These images are termed as *label set* which is employed to segment walls and openings in a raster floor plan image. The floor plan label set includes binary images in which walls are marked as continuous bold lines passing through outer openings while inner openings are spaced out since inner openings can only be obtained from floor plans and outer opening specifications can easily be derived from elevation drawings with reference to floor plans (Figure 3.5).

Figure 3.5. Floor plan drawing and corresponding label image

Floor plan dataset encapsulates 250 raster floor plan drawings, and 250 corresponding labels in binary format. Drawings and label image sizes are the same and equal to 512 (height) x 256 (width) pixels. The number of data is split into train, validation and test sets as the ratio of 70%, 20% and 10%, respectively.

3.1.2. Elevation Dataset

An elevation is a flat representation of the view seen from one side of the building. As it is not common for a building to have simple rectangular shaped plan, an elevation drawing represents projection of all the building parts seen from a specific direction with the perspective flattened. The main characteristics of an elevation is that it does not include any depth, however employing both floor plans and elevations at the same time can overcome this problem. In general, elevations are generated for four directional views such as east, west, north and south to give an inclusive information about a building's appearance on the outside. Elevations can contain different level of details depending on the usage area as in floor plans, but simple elevation drawings show annotations such as level datum and dimensions, exterior walls, openings, roofs and projections (Figure 3.6).

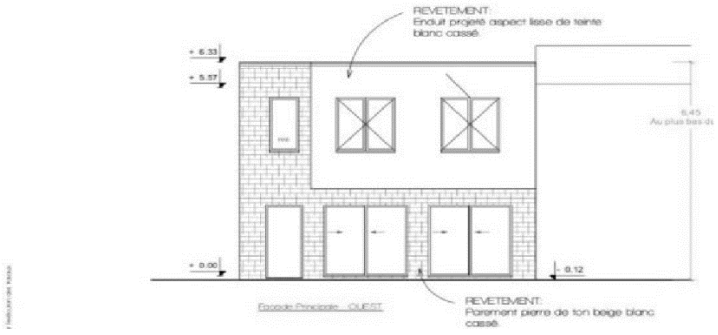


Figure 3.6. A simple elevation drawing

There are two major challenges in elevations. First challenge is the same of floor plan drawings. Content of an elevation drawing may increase when the building scale grows. Simple one-storey and two-storey height residential building drawings are

compiled as elevation dataset to be coherent with floor plan dataset and to minimize this challenge. Second is that elevations may consist of different representation styles unlike plans. Usage of shades, shadows, landscape items, railings, materials, window and door types can vary designer to designer. As a result, it becomes hard to identify a pattern. This situation necessitates having a bigger elevation dataset than the floor plan to learn the features appropriately.

The elevation dataset is created from raster simple elevation drawings with walls, openings, annotations, roofs and projections. It is already discussed that walls and openings are the foremost notion for 3D model reconstruction, and the most accurate outer openings are only available in elevation dataset. Similar to indication of the floor plan dataset, elevations are marked accordingly to create a corresponding label set. Again, the main idea of an elevation label set is to represent expected outcome of a learning process. The elevation label set is prepared by denoting the vital geometrical information in the drawings respectively. Only walls and openings are labelled because elevations are the supplementary data source for floor plan dataset. Consequently, walls are marked as solid black while openings are marked as solid white for labelling the elevation dataset as shown in Figure 3.7. Similar to floor plan labels, elevation labels are binary images as well.

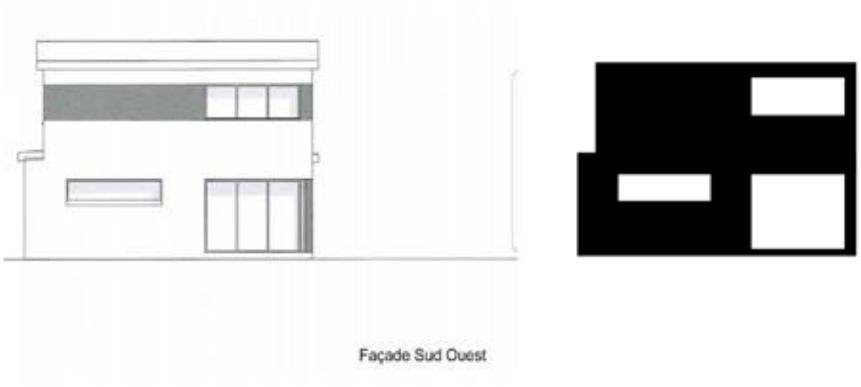


Figure 3.7. Elevation drawing and corresponding label image

Elevation dataset includes 450 raster floor plan drawings, and 450 labels in binary format. Drawings and label image sizes are the same and equal to 256 (height) x 512 (width) pixels. The number of data is split into train, validation and test sets as the ratio of 70%, 20% and 10%, respectively.

3.1.3. Data Augmentation

It is not always possible to generate a dataset including thousands of images as in ImageNet (Deng, et.al., 2009) or MSCOCO (Lin, et.al, 2014) datasets. It takes much computation, time and labor. When there is no chance for gathering more instances, data can be augmented to enhance the available dataset. Data augmentation increases image data samples by using basic transformations such as rotation and scaling. The idea behind of data augmentation is obviously to have an enlarged dataset to be able to learn a more generalized pattern over the images.

When there is no proper dataset in terms of size and homogeneity, learning the patterns and features may fail that results in either overfitting and underfitting. While overfitting implies that dataset is learned completely, underfitting is a resultant of learning the dataset inadequately. In cases of overfitting, there is actually no *learning*, but instead there is *memorizing*. Therefore, the system does not perform well on any unseen sample. Likewise, an unseen sample is not recognized in underfitting since the

system cannot learn even the dataset itself. (Figure 3.8). The potential of data augmentation mostly prevents these problems by increasing data samples, and provides a more robust recognition process.

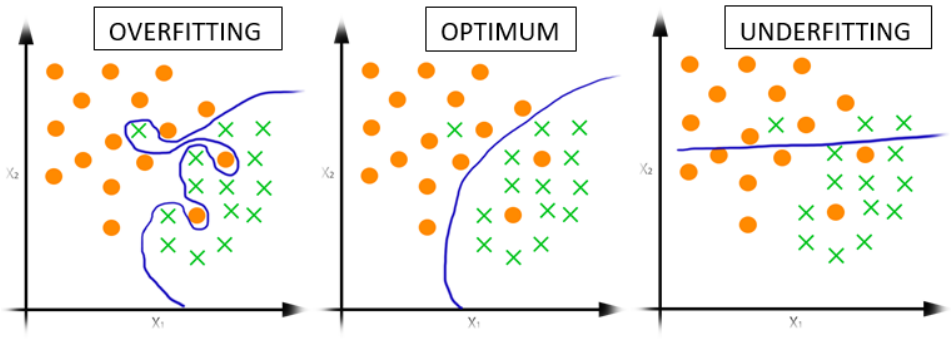


Figure 3.8. Overfitting, optimum fit and underfitting representations (Joglekar, 2018)

While augmenting a dataset, it is important to consider the side effects of the translations. For instance, if every image in a dataset centered in the frame, it would become a distinguishing feature of the dataset. Therefore, an object in an image is recognized according to location instead of corners and edges. This is why corner-aligned transformations should be applied to add variety to dataset. As another example, horizontal or vertical mirroring should be contemplated beforehand, because it might be irrelevant for an image to be flipped as shown in Figure 3.9.

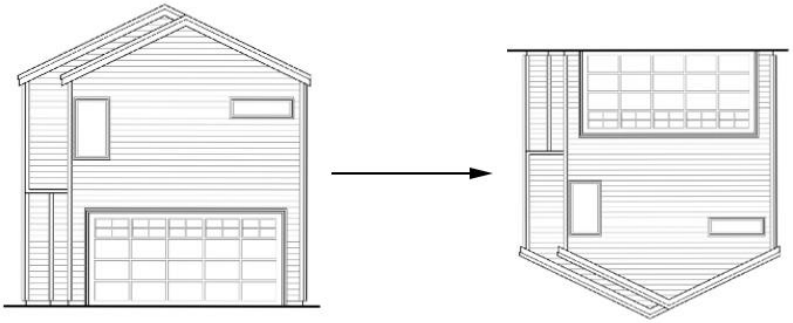


Figure 3.9. Vertical flip of an elevation drawing (produced by the author)

In this research, corner alignment, rotation, width and height shift, shearing and horizontal flip are applied for both of the datasets. Besides, vertical flip is additionally applied to floor plan dataset. Figure 3.10 illustrates the augmented datasets.

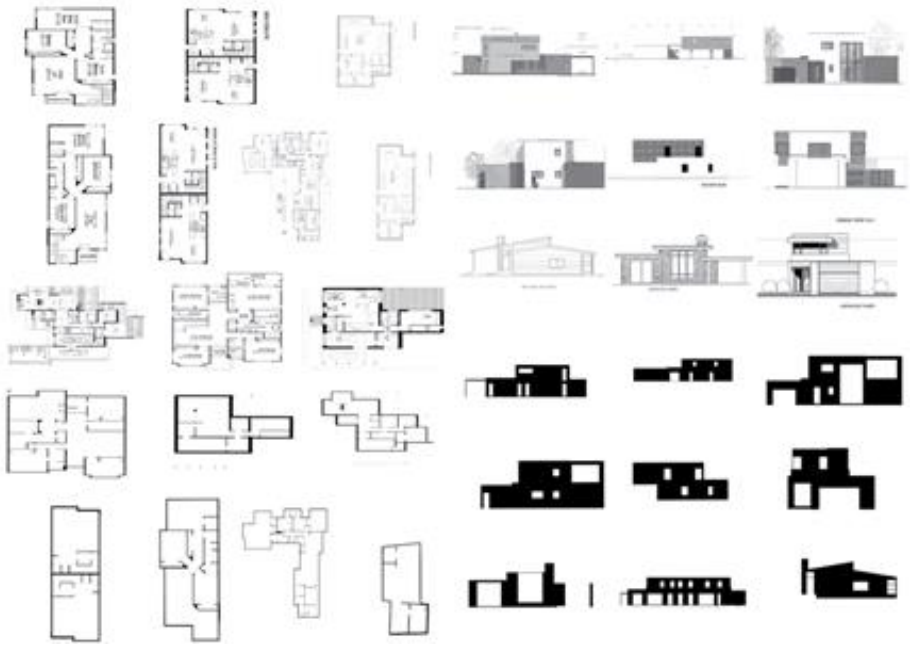


Figure 3.10. Data augmentation examples (produced by the author)

3.2. Architectural Element Extraction

Among other feature extraction methods in the literature, semantic segmentation applications are more convenient considering the objectives of this dissertation. Architectural drawings include excessive amount of information. It is hard to follow a pattern and construct a generalized rule-based approach for a custom dataset as in shape-based and relation-based approaches. Unlike these algorithms, semantic segmentation does not require a predefined rule set. It instead perceives an image at pixel level and assign each pixel to an object class according to a pattern it creates by iterating the dataset.

Apart from entitling objects in an image such as ‘wall’ and ‘window’, these objects are also geometrically represented in semantic segmentation algorithms (Figure 3.11). These representations can either belong to two categories (“wall” or “not wall”) or multi-class (“wall”, “window”, “door”), which should be decided beforehand. Two categories are utilized in this research because 3D model regeneration is based on structural elements and openings. Since using only the categorical information of architectural elements are not sufficient enough to create a 3D model, there should be geometries (e.g. rectangular, circular, triangular, non-orthogonal shapes) to be transformed into 3D environment. Thereby, semantically distinguished architectural elements will be the cornerstone for 3D model generation of this dissertation.

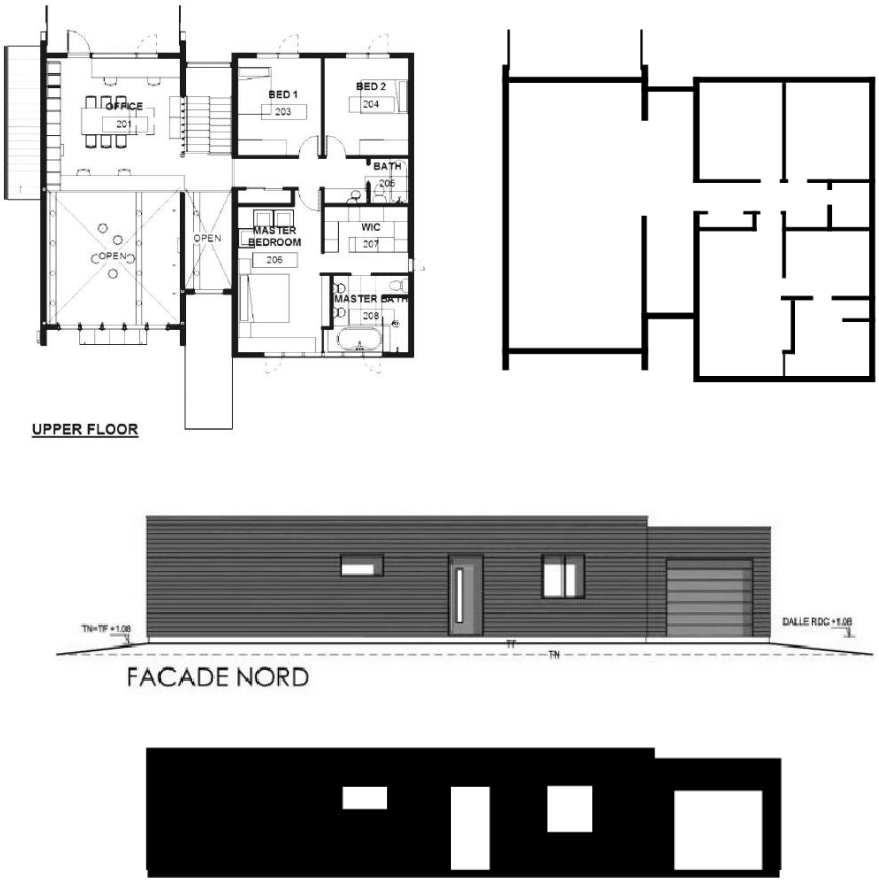


Figure 3.11. Input images and architectural elements (produced by the author)

Semantic segmentation has always been one of the important concepts in 3D model reconstruction. Many applications need accurate, efficient and robust segmentation methods for evaluating the visuals and making real-time decisions. These methods can be categorized as traditional machine learning approaches and neural network applications. Traditional methods require a preprocessing step to gather features to predict. On the other hand, Artificial Neural Networks (ANNs) take raw input and process it within a hierarchical biological-like neuron structure to learn parameters of the given input (Thoma, 2016).

Traditional applications require a method starting with the choice of local or global features. There is a wide range of feature descriptors used for semantic segmentation such as HOGs (Xu, Hancock & Zhou, 2019), SIFT (Jiang, Wu & Lu, 2018) and poselets (Zhu et.al., 2016). Appropriate features are then utilized in traditional methods such as K-means and Support Vector Machines (SVMs). For example, SVMs serve basically for binary classification tasks. Given a feature of the training set is labelled as 0 or 1 (Liu, Schwing, Kundu, Urtasun & Fidler, 2015; Kim, Oh & Sohn, 2016). SVMs highly depend on dataset linearity. It becomes harder when nonlinearity increases. As another approach, Markov Random Fields/Conditional Random Fields (MRFs/CRFs) construct a variety of links between classes according to the adjacent pixels. (Thøgersen, Escalera, Gonzàlez & Moeslund, 2016; Zheng, Zhang & Wang, 2017). While MRFs learn the distribution, CRFs uses conditional probabilities. Their complex structure comes with a burden of computation cost. Also, it is not possible for them to predict an unknown sample if it does not appear in the dataset. Overall, the main challenge for traditional methods is deciding feature representations for performance improvement (Yu *et.al.*, 2018; Zaitoun & Aqel, 2015; Zhu *et.al.*, 2016). This consequence leads this research into learning features without any prior treatment which can be accomplished with ANNs.

ANNs were brought forward as an analogy of human brain's neural circuits. Briefly, they are meant to mimic the way humans learn. They are formed of *'neurons'* that are input(s), hidden units and output(s) connected with coefficients (weights) in most cases. Patterns that are hard for humans to recognize are actually ordinary tasks for NN's complex structure. Connections between neurons have a substantial effect on progressing of ANNs. Artificial neurons take the output from previous input and feeds the learnt outcome through the final output with a linear function. If the final output is not equal to the desired one, then a backpropagation starts to minimize errors occurred because of the gap between final and desired outputs (Agatonovic-Kustrin & Beresford, 2000). Figure 3.12 represents an ANN scheme.

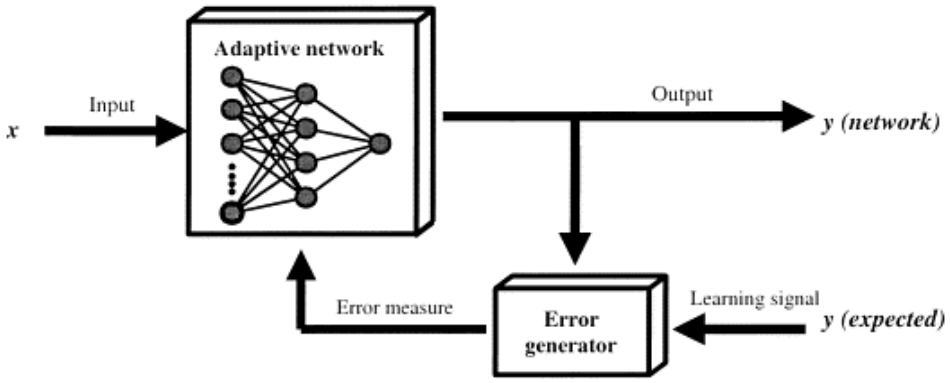


Figure 3.12. ANN with backpropagation (Agatonovic-Kustrin & Beresford, 2000)

Though successful results can be achieved with feeding forward, backpropagation is still a major issue for ANNs. Since backpropagation depends on local gradient information, the algorithm is mostly stuck with local optima. Moreover, an overfitting problem can be confronted especially if the size of an input dataset is not sufficiently large (Liu et.al, 2017). Deep Learning (DL) draws attention of the computer vision society at this point to handle such obstacles. Originated from ANNs, many architectures with multiple non-linear information processing units can be found under DL methods such as Convolutional Neural Networks (CNNs) and Restricted Boltzmann Machine (RBM). DL consists of a hierarchy of layers that convert the input

into more abstract representations. While a regular NN includes approximately three layers (input, hidden layer and output), a Deep Neural Network (DNN) contains more than three (Figure 3.13).

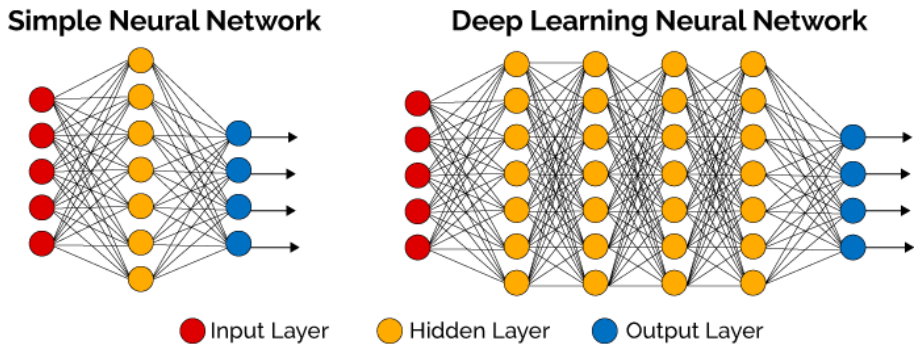


Figure 3.13. ANN and DNN comparison (Gupta, 2018)

These hidden layers directly define features of the input data samples and generate new feature series out of them. The output layer compounds these series and makes a prediction (LeCun, Bengio & Hinton, 2015). As a network contains more layers, it learns more features and predicts more accurately (Figure 3.14). Therefore, it can learn more complex patterns in a dataset than a regular NN. Also, DNN usage becomes easier with the increasing computation power and dataset diversity. Applications with neural networks for semantic segmentation are more applicable than traditional methods due to the reasons emphasized before, yet Deep Neural Networks show more practicable characteristics than ANNs.

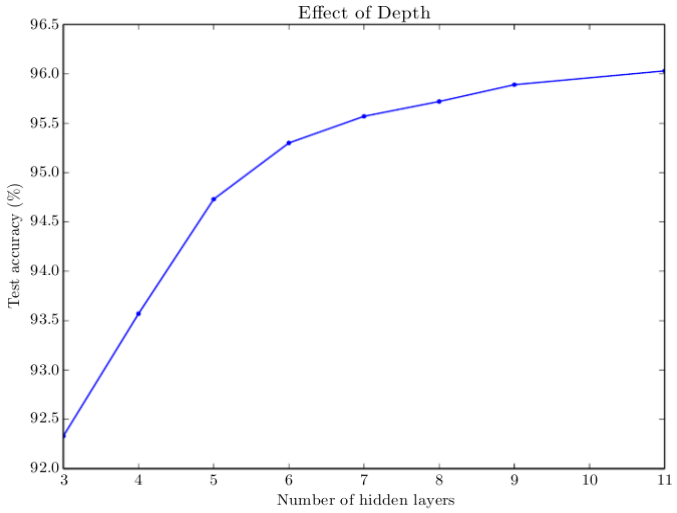


Figure 3.14. Graph showing effect of hidden layers (Goodfellow, Bengio & Courville, 2016)

DL applications for semantic segmentation vary significantly. There are excessive surveys about DL methods and architectures (Garcia-Garcia et.al., 2017; Lateef & Ruichek, 2019; Liu et.al., 2018). However, it would be appropriate to declare that most of the architectures based on CNNs when images take place. Established for image classification tasks, CNNs can be adapted to semantic segmentation problems, too. The main difference of a CNN from a DNN is that while a DNN connects all the neurons to each other, a CNN shares the weights and uses local connectivity. For being the case, CNNs reduce the number of parameters and computation. They can be counted as the first prospering architecture because of successfully trained hierarchical layers in the scope of semantic segmentation. Reflections of CNN usage in architecture and construction area can be seen in the literature such as research of Chang et.al. (2017) and Wang, Savva, Chang and Ritchie (2018).

Fully Connected Convolutional Networks (FCNs) (Long, Shelhamer & Darrell, 2015), Region-Based Convolutional Neural Networks (R-CNNs) (Girshick, Donahue, Darrell & Malik, 2014) including Fast R-CNN (Girshick, 2015) and Mask R-CNN (He, Gkioxari, Dollár & Girshick, 2017) and Dilated CNNs (Yu & Koltun, 2015) are

all extensions of CNNs to optimize the given architecture for a spatial dataset. Apart from these extensions, there are other DL applications such as Recurrent Neural Networks (RNNs) and Deep Belief Networks (DBNs). However, they mostly perform better on tasks other than computer vision problems such as speech and text recognition (Liu et.al, 2017).

According to the observations, CNNs are the most applicable and concordant models due to the direct compatibility with semantic image segmentation. Weight sharing, capability of extracting both low and high level features in an image, using less number of parameters, customizability for various datasets, capability of transformation to other models, ability of using pre-trained model weights are the substantial reasons for applying CNNs for the architectural drawing dataset.

3.2.1. CNN for Semantic Segmentation

CNN is basically a neural network that have learnable weights and biases. Each neuron in the system takes some input volume, encodes its features and transforms into another neuron through a differentiable function. Unlike regular neural networks that take 1D vector and use fully-connected layers, CNNs work with 3D vectors of size *h* x *w* x *d*, where *h* is image height, *w* is image width and *d* is the color channel. CNNs can serve both for classification and semantic segmentation tasks. If the task is classification, encoding with a fully connected layer which is basically a regular Neural Networks is implemented. If semantic segmentation is needed, decoding following encoding is applied (Figure 3.15 & Figure 3.16).

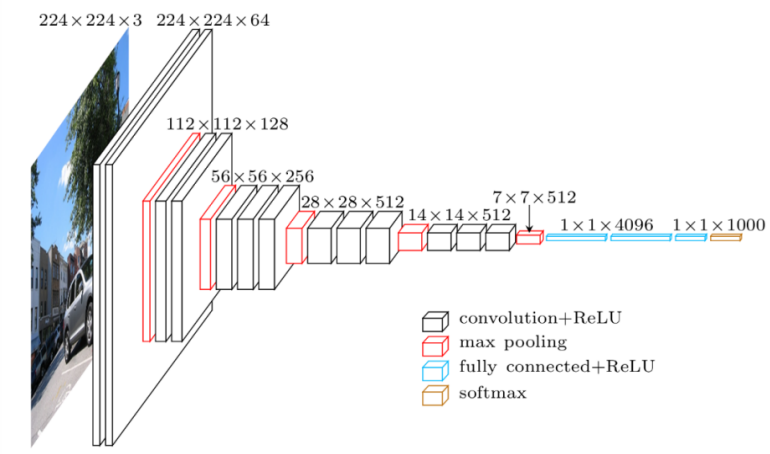


Figure 3.15. CNN for classification (ul Hassan, 2018)

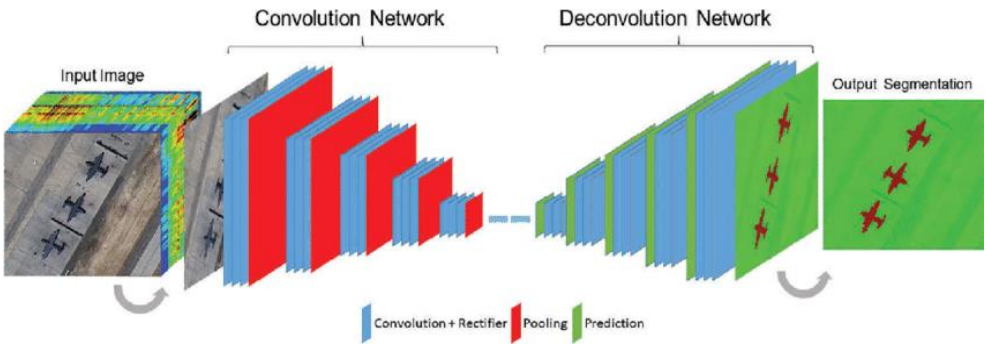


Figure 3.16. CNN for semantic segmentation (Chen, Weng, Hay & He, 2018)

CNNs for semantic segmentation consists of series of layers which can be called 'Convolution Layer', 'Activation Layer', 'Downsampling Layer', and 'Upsampling Layer'. These layers can be rearranged depending on the task and the dataset. Convolution layer includes a filter/kernel to slide around the input image until all the pixel values are covered. Filter and image pixel values are multiplied during convolution. After this linear operation, activation layer is introduced which generally contains different functions such as Rectified Linear Units (ReLu) (Nair & Hinton, 2010) and sigmoid function (Han & Moraga, 1995) to add non-linearity to the system. Downsampling layer mainly helps eliminating computational cost by reducing the activation map dimensionality. Upsampling layer makes the previous activation layer

output improve from low resolution to higher resolution. By doing so, prediction image size will be as same as the input size. Figure 3.17 illustrates the main layers for CNNs.

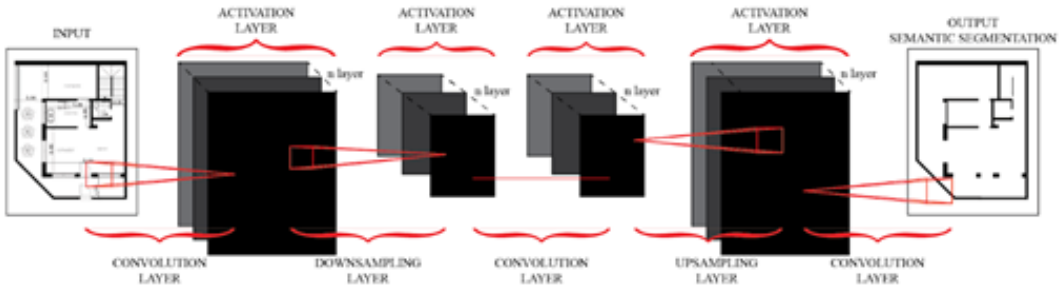


Figure 3.17. Illustration of a CNN pipeline for semantic segmentation (produced by the author)

All of these steps are configured accordingly to achieve a well-performed semantic segmentation of which performance is optimized by a cost function. The main aim is to minimize the cost function as much as possible by optimizing the dataset and CNN structure. The cost function evaluates the label class predictions for each pixel in the image, and averages over the whole pixels. Binary cross entropy is employed as the cost function of this research since the task is binary semantic segmentation. Binary cross entropy can be calculated as follows:

Let probability of opening $\mathbf{P}(Y = 0) = p$ and probability of wall $\mathbf{P}(Y = 1) = 1\text{-}p$

$$\boldsymbol{P}\left(\widehat{Y} = 0\right) = \frac{1}{1 - e^{-x}} = \hat{p}$$

$$\boldsymbol{P}\left(\widehat{Y} = 1\right) = 1 - \frac{1}{1 - e^{-x}} = 1 - \hat{p}$$

$$Binary\ Cross\ Entropy = BCE(p, \hat{p}) = -(p\log(\hat{p}) + (1 - p)\log(1 - \hat{p}))$$

Common state-of-the-art CNN architectures and datasets are overviewed to employ an appropriate one for this dissertation. Every CNN model in the literature has its own layer configuration and relevant cost function. Since CNN architectures are mostly constructed for specific datasets and purposes, it may be tricky to customize the models according to a custom dataset. Depending on the desired number of low or high level features in an image, any layer and/or cost function should be analyzed and integrated to each other accordingly.

3.2.2. CNN Architectures and Datasets

Many practical applications have been recently achieved by employing CNNs. They are widely well-received in real-time segmentations such as autonomous driving (Treml *et.al.*, 2016), agricultural robotics (Milioto, Lottes & Stachniss, 2018), video-based studies (Shelhamer, Rakelly, Hoffman & Darrell, 2016) and medical interpretations (Roth *et.al.*, 2017). All of them prove that datasets and architectures are inseparable from each other. Accuracy and performance enhancement of the models can be accomplished as the number of relevant data samples increases. However, gathering or generating adequate dataset for a semantic segmentation task is one of the most difficult challenges. Therefore, it is a common approach to employ an existing dataset depending on the problem domain.

Although there are 2D, 2.5D and 3D datasets currently available in generic, urban, indoor, medical and object themes (Alhaija *et.al.*, 2018; Armeni, Sax, Zamir & Savarese, 2017; Brostow, Fauqueur & Cipolla, 2009; Cardona *et.al.*, 2010; Cordts *et.al.*, 2016; Everingham *et.al.*, 2015; Hoover & Goldbaum, 2003; Lin *et.al.*, 2014; Marcus *et.al.*, 2007; Schneider & Gavrila, 2013; Silberman, Hoiem, Kohli & Fergus, 2012; Zhou *et.al.*, 2017), none of them is precisely applicable for recognition of elements in an architectural drawing.

For being the case, architectural drawing datasets are prepared for this research to achieve accurate 3D model generation process as mentioned before. These datasets contain 2D binary format drawing images. Floor plan set includes 250 detailed conceptual drawings with different level of complexity and scale. Elevation set includes 450 simple drawings of one or two storey height buildings with various details such as shades and shadows. Their corresponding image labels are prepared to highlight walls and openings in binary format. Figure 3.18 represents a comparison between architectural drawing datasets and available ones.

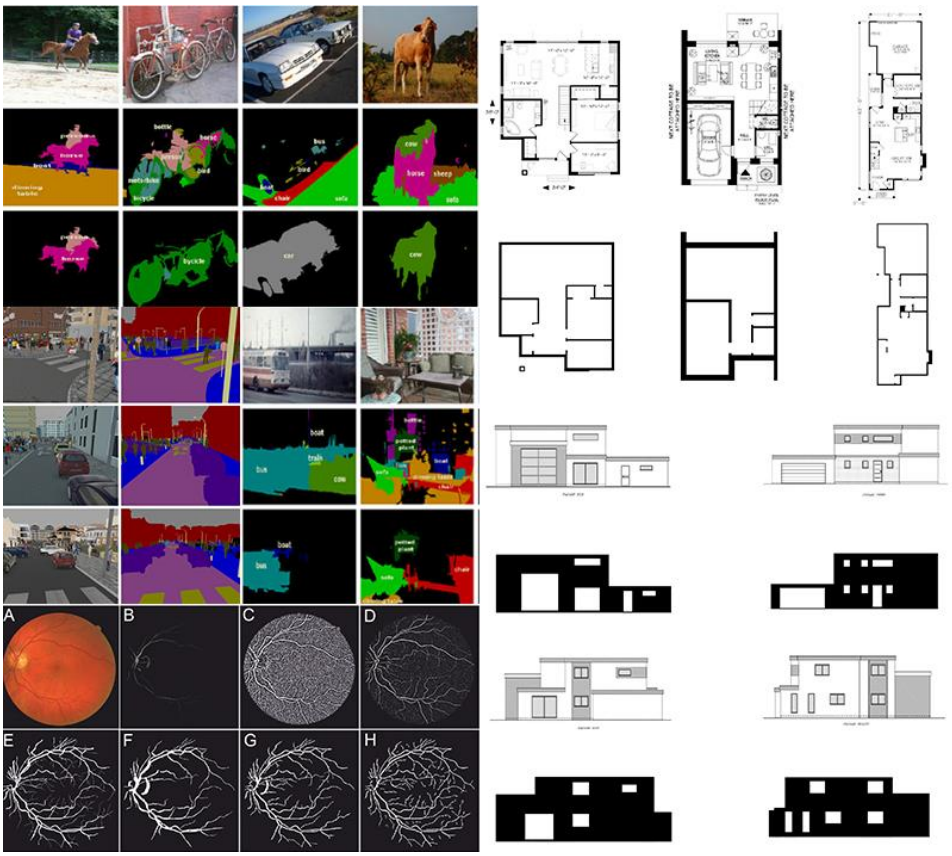


Figure 3.18. Datasets with different themes and architectural drawing datasets

(produced by the author)

All the images above are in 2D pixel environment and compose of edges and corners, and yet the texture and pattern they contain differ very much. Even though the

information on the existing datasets can be useful, customized architectural datasets are introduced to emphasize relevant patterns of architectural components. Also, a custom CNN architecture should be launched since datasets and architectures are extremely related to each other. For example, U-Net (Ronneberger, Fischer & Brox, 2015) is highly popular among medical binary semantic segmentation applications while FCN (Long *et.al.*, 2015) is preferable in generic themed datasets with multiple classes. Searching for appropriate models for architectural datasets will be a part of this dissertation.

One issue is that CNN models accuracy, efficiency and training rate may differ from one dataset theme to another. Moreover, if the theme and feature-space distribution change, they have to be rearranged from scratch, which is hard to optimize and fine-tune. This is why it is widespread to pretrain a CNN and use it for a new domain and task, especially in semantic segmentation applications. Table 3.1 shows some of the contemporary models, and they are mostly constructed upon CNN architectures that are constitutively prepared for classification tasks such as VGG (Simonyan & Zisserman, 2014) and ResNet (He, Zhang, Ren & Sun, 2016). While using CNNs that are built upon classification ensures a steady semantic segmentation process, employing CNNs with pre-trained weights, which is called '*Transfer Learning*', can provide better performance and robustness.

Table 3.1. *CNN architectures for semantic segmentation (Garcia-Garcia et.al., 2018)*

Name and reference	Architecture	Targets							Source code	Contribution(s)
		Accuracy	Efficiency	Training	Instance	Sequences	Multi-modal	3D		
Fully Convolutional Network [68]	VGG-16(FCN)	⋆	⋆	⋆	×	×	×	×	√	Forerunner
U-Net [69]	Fully CNN, 4 downsampling/upsampling steps	⋆⋆	⋆⋆	⋆	×	×	×	×	√	Data augmentation, Skip-layer, Patch wise training/inference
SegNet [70]	VGG-16 + Decoder	⋆⋆⋆	⋆⋆	⋆	×	×	×	×	√	Encoder-decoder
Bayesian SegNet [71]	SegNet	⋆⋆⋆	⋆	⋆	×	×	×	×	√	Uncertainty modeling
DeepLab [72,73]	VGG-16/ResNet-101	⋆⋆⋆	⋆	⋆	×	×	×	×	√	Standalone CRF, atrous convolutions
MINC-CNN [46]	GoogLeNet(FCN)	⋆	⋆	⋆	×	×	×	×	√	Patchwise CNN, Standalone CRF
CRFasRNN [74]	FCN-8s	⋆	⋆⋆	⋆⋆⋆	×	×	×	×	√	CRF reformulated as RNN
Dilation [75]	VGG-16	⋆⋆⋆	⋆	⋆	×	×	×	×	√	Dilated convolutions
ENet [76]	ENet bottleneck	⋆⋆	⋆⋆⋆	⋆	×	×	×	×	√	Bottleneck module for efficiency
Multi-scale-CNN-Raj [77]	VGG-16(FCN)	⋆⋆⋆	⋆	⋆	×	×	×	×	×	Multi-scale architecture
Multi-scale-CNN-Eigen [78]	Custom	⋆⋆⋆	⋆	⋆	×	×	×	×	√	Multi-scale sequential refinement
Multi-scale-CNN-Roy [79]	Multi-scale-CNN-Eigen	⋆⋆⋆	⋆	⋆	×	×	⋆⋆	×	×	Multi-scale coarse-to-fine refinement
Multi-scale-CNN-Bian [80]	FCN	⋆⋆	⋆	⋆⋆	×	×	×	×	×	Independently trained multi-scale FCNs
ParseNet [81]	VGG-16	⋆⋆⋆	⋆	⋆	×	×	×	×	√	Global context feature fusion
PSPNet [82]	ResNet50	⋆⋆⋆	⋆	⋆⋆	×	×	×	×	√	Image context modeling, training optimization strategy for ResNet
ReSeg [83]	VGG-16 + ReNet	⋆⋆	⋆	⋆	×	×	×	×	√	Extension of ReNet to semantic segmentation
LSTM-CF [84]	Fast R-CNN + DeepMask	⋆⋆⋆	⋆	⋆	×	×	×	×	√	Fusion of contextual information from multiple sources
2D-LSTM [85]	MDRNN	⋆⋆	⋆⋆	⋆	×	×	×	×	×	Image context modeling
rCNN [86]	MDRNN	⋆⋆⋆	⋆⋆	⋆	×	×	×	×	√	Different input sizes, image context
DAG-RNN [87]	Elman network	⋆⋆⋆	⋆	⋆	×	×	×	×	√	Graph image structure for context modeling
SDS [10]	R-CNN + Box CNN	⋆⋆⋆	⋆	⋆	⋆⋆	×	×	×	√	Simultaneous detection and segmentation
DeepMask [88]	VGG-A	⋆⋆⋆	⋆	⋆	⋆⋆	×	×	×	√	Proposals generation for segmentation
SharpMask [89]	DeepMask	⋆⋆⋆	⋆	⋆	⋆⋆⋆	×	×	×	√	Top-down refinement module
MultiPathNet [90]	Fast R-CNN + DeepMask	⋆⋆⋆	⋆	⋆	⋆⋆⋆	×	×	×	√	Multi path information flow through network
Huang-3DCNN [91]	Own 3DCNN	⋆	⋆	⋆	×	×	×	⋆⋆⋆	×	3DCNN for voxelized point clouds
PointNet [92]	Own MLP-based	⋆⋆	⋆	⋆	×	×	×	⋆⋆⋆	√	Segmentation of unordered point sets
PointNet++ [93]	Own PointNet-based	⋆⋆	⋆	⋆	×	×	×	⋆⋆⋆	√	Improve PointNet by capturing local information
Dynamic Graph CNN (DGCNN) [94]	Own EdgeConv	⋆⋆	⋆	⋆	×	×	×	⋆⋆⋆	×	EdgeConvolution module for point clouds as graphs
Clockwork Convnet [95]	FCN	⋆⋆	⋆⋆	⋆	×	⋆⋆⋆	×	×	√	Clockwork scheduling for sequences
3DCNN-Zhang	Own 3DCNN	⋆⋆	⋆	⋆	×	⋆⋆⋆	×	×	√	3D convolutions and graph cut for sequences
End2End Vox2Vox [96]	C3D	⋆⋆	⋆	⋆	×	⋆⋆⋆	×	×	×	3D convolutions/deconvolutions for sequences
SegmPred [97]	Own multi-scale net	⋆⋆	⋆	⋆	×	⋆⋆⋆	×	×	√	Predicting future frames in the space of semantic segmentation

Transferring knowledge from one domain to another is called transfer learning in machine learning field (Figure 3.19). The need for transfer learning arises when there is a relatively small training dataset. For example, ImageNet consists of more than one million images with 1000 categories. It is not always possible to have such a dataset for each domain. At this point, transfer learning endeavors to solve new problems by using a pre-trained model with vast amount of images like ImageNet dataset. It overcomes specialization of an architecture to a particular domain or task, and increases the performance (Pan & Yang, 2010).

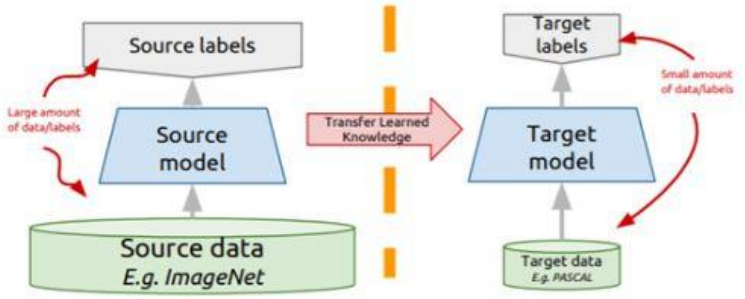
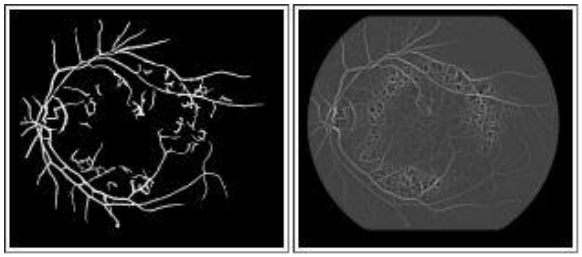
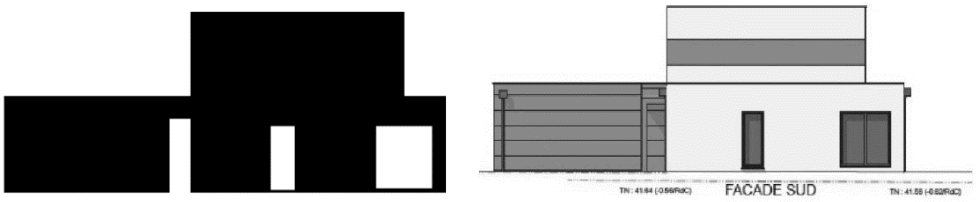


Figure 3.19. Example of transfer learning (Sarkar, 2018)

Considering the observations, there are two major concerns to regard. First one is to seek a resemblance between architectural datasets and current ones in the literature since semantic segmentation applications are engaged with CNNs and datasets at once. It is noticed that medical datasets are generally presented as grayscale as in architectural datasets. This creates an immediate link between them (Figure 3.20). This resemblance puts forward U-Net directly since it is built for medical segmentation. Its structure is highly compatible with fewer data samples and binary image format. Likewise, TernausNet (Iglovikov & Shvets, 2018) attracts notice. It is also developed for medical semantic segmentation tasks. Also, it employs transfer learning to overcome aforementioned problems, and it shows promising results on binary images.



(a)



(b)

Figure 3.20. Dataset comparison (a: STARE dataset; b: Architectural drawing dataset)

Second is the stability of a CNN model on semantic segmentation tasks. There are multiple algorithms that have a proven track record of accomplishment: FCN (Long et.al., 2015), SegNet (Badrinarayanan, Kendall & Cipolla, 2017), DeepLab (Chen, Papandreou, Kokkinos, Murphy & Yuille, 2017). Among others, SegNet is opted for architectural drawing datasets. It is one of the earliest semantic segmentation algorithm which uses decoding phase. It is not composed of fully connected layers as in FCN, so it has fewer parameters. Even though it is slower than FCN or DeepLab while training, it needs lower memory requirements and shows better accuracy and efficiency in most cases (Badrinarayanan, Kendall & Cipolla, 2017).

To sum up, U-Net, SegNet and TernausNet are adapted for a binary semantic segmentation in the scope of this research. Each of them will be analyzed based on different configurations and optimization approaches in the following chapter.

U-Net

U-Net is developed for biomedical image segmentation by Olaf Ronneberger, Philipp Fischer, and Thomas Brox (2015) with an inspiration from the research of Long *et.al.* (2015). Its architecture is modified to work with few training images and yet get

precise segmentations. The symmetric-like shape of the model enabling precise localization (Figure 3.21) is a resultant of using many feature channels in upsampling part which provides information transfer to higher resolution layers.

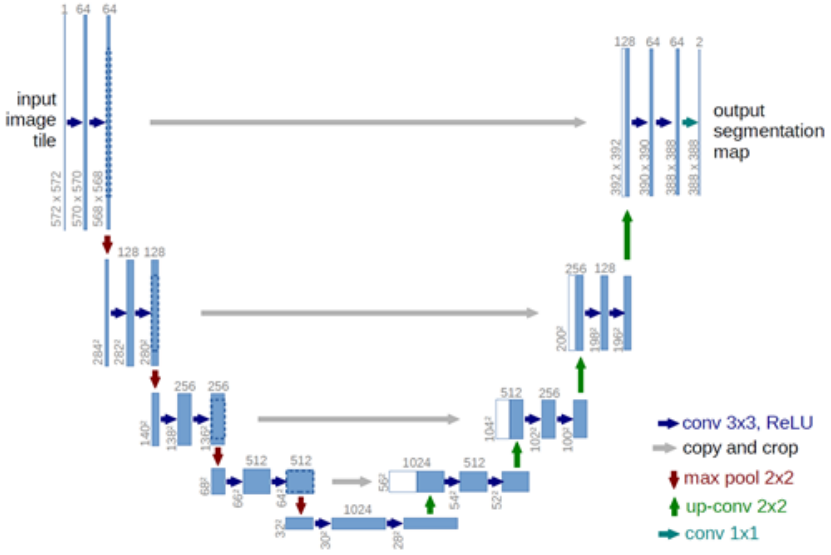


Figure 3.21. U-Net Architecture (Ronneberger et.al., 2015)

Even though researchers use evenly sized images to empower results, U-Net is adaptive to varying sizes. Encoding part includes 3x3 convolutions followed by a ReLu and 2x2 max pooling layers. At each downsampling step, the number of feature channels are doubled. Decoding part comprises upsampling of feature channels with 2x2 convolutions, concatenation with cropped feature channels to avoid information loss and 3x3 convolution layers followed by a ReLu. Finally, 1x1 convolution is used to extract segmentation map. Within its 23 convolutional layers, U-Net creates its own weights with random initialization.

In most cases, image segmentation datasets involve thousands of images to impart diversity. However, preparation of non-existing train and label sets is a challenging and costly process. This is why U-Net's capability of training relatively small dataset

is an important feature in the scope of this dissertation since size of the architectural datasets provided is not very large as well. Data augmentation is preferred as in this research to enhance power of the network and obtain invariance and robustness properties.

SegNet

SegNet (Badrinarayanan, Kendall & Cipolla, 2017) is primarily designed for multi-class segmentation with an inspiration from road scene parsing to understand shape, appearance and context. Encoding part of this architecture takes its convolutional layers from VGG16 structure. Fully connected layer of VGG16 is removed, so encoding part of SegNet gets smaller and easy to train. Similar to U-Net, it has symmetric-like shape with a hierarchical network of decoders and encoders corresponding to each other (Figure 3.22).

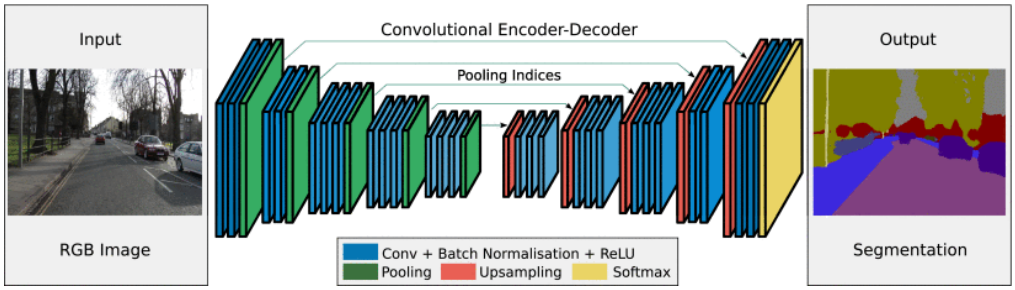


Figure 3.22. SegNet Architecture (Badrinarayanan, Kendall & Cipolla, 2017)

SegNet consists of 13 encoding and 13 decoding convolutional layers. Since fully connected layers of VGG16 architecture is discarded for keeping the higher resolution feature maps, the number of parameters is reduced significantly. What is special in this architecture is that max-pooling indices of encoder layers are used in decoder layers. This process does not need to learn to upsample and ensures a non-linear upsampling. Figure 3.23 illustrates max-pooling indices in SegNet.

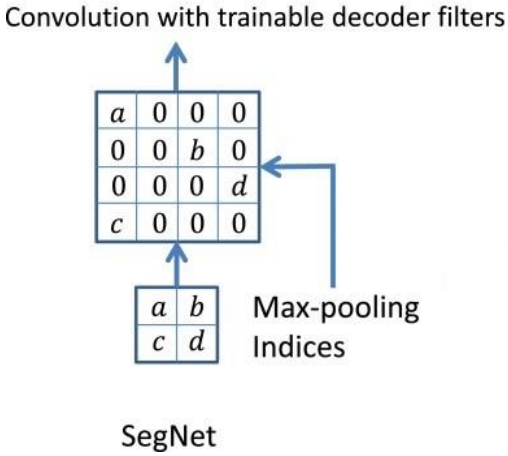


Figure 3.23. Illustration of max-pooling indices in SegNet (Badrinarayanan, Kendall & Cipolla, 2017)

TernausNet

TernausNet (Iglovikov & Shvets, 2018) is built upon U-Net to increase its performance with transfer learning (Simonyan & Zisserman, 2014). The only difference of TernausNet from the main pipeline of U-Net is to use 11 convolutional layers of VGG16 without fully connected layers as encoders. It also adapts the ImageNet weights. 11 layers of VGG includes 7 convolutional layer with ReLu activation function and 5 max pooling operations. Decoding layers consist of transposed convolutional layers, and the output of them are concatenated with the output of corresponding encoding layers. 5 times of upsampling are applied to pair up with 5 max pooling layers of VGG11. Figure 3.24 represents TernausNet architecture.

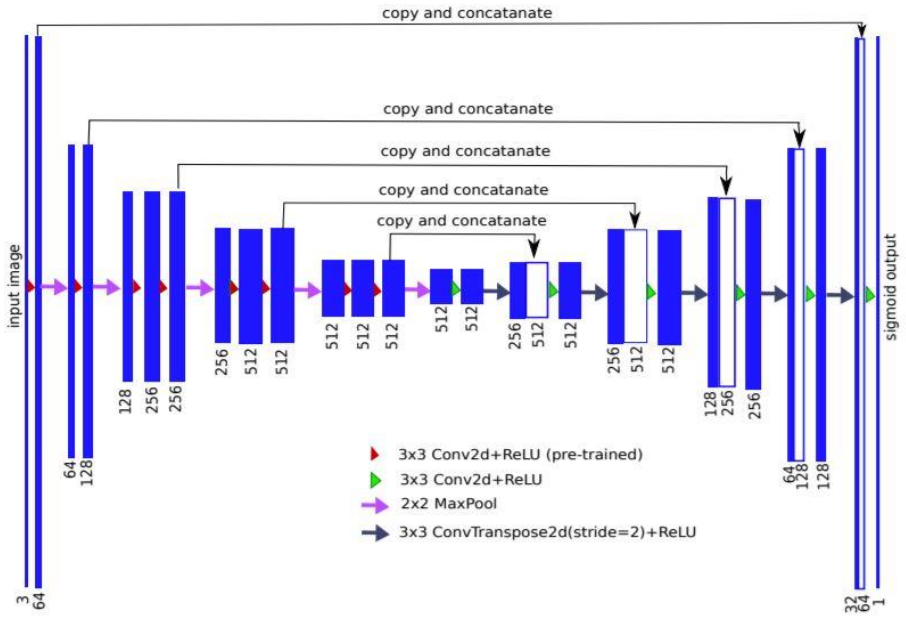


Figure 3.24. TernausNet Architecture (Iglovikov & Shvets, 2018)

TernausNet includes less convolutional layers than both U-Net and SegNet, so it is relatively a lightweight architecture. Also, pre-trained weights decreases training time and avoids overfitting. Even though it is based on U-Net which takes input in any size, current TernausNet is only compatible with images having a size that can be divided by 32 since 5 max-pooling layers downsample an image two times (2^5).

3.3. 3D Model Generation

After architectural drawings are semantically segmented with CNN architectures in a 2D pixel environment, 3D conversion becomes possible to obtain 3D reconstructed models. Before vectorising the architectural components in a semantically segmented image, there is a post-processing step for removing noise in the images. There are different approaches for removing obvious errors of a found segmentation (Brox, Bourdev, Maji & Malik, 2011; Chen, Luo & Parker, 1998; Farabet, Couprie, Najman & LeCun, 2013; Hariharan, Arbeláez, Girshick & Malik, 2014). In this research, vectorisation of a predicted image firstly relies on morphological transformations for image enhancement. Later, enhanced images are contoured. This approach is

primarily employed due to simple mathematical transformations and applicability with ease.

3.3.1. Morphological Transformations

Morphological transformations are generally used for edge detection, noise removal, image refinements and image segmentation in image processing applications. A transformation that is applied to binary or grayscale images includes a morphological operator based on a structuring element. The structuring element is basically a kernel which contains a pattern specified beforehand relative to an origin point and slides around the image (Figure 3.25). If the structuring element and underlying image ensure the requirements defined by the operator, pixel of the image underneath origin of the structuring element is set to a predefined value (Fisher, Perkins, Walker & Wolfart, 2000).

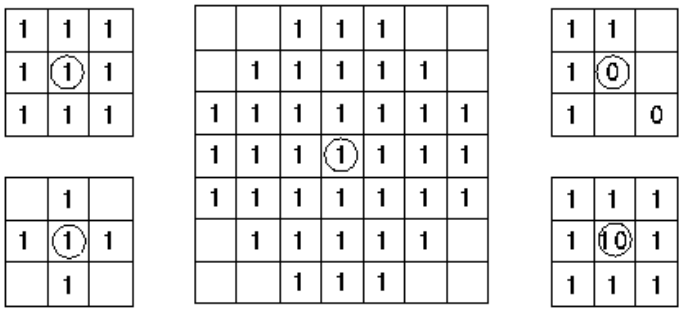


Figure 3.25. Structuring element examples and their origin points inside a circle (Fisher *et.al.*, 2000)

Morphological operators enable pixels to intersect, unite and/or complete. Erosion and dilation are basic operators, yet there are variant forms such as opening, closing and thinning. The erosion operator shrinks the boundaries of foreground pixels, so foreground regions become smaller and holes within those regions become larger. Oppositely, dilation operator expands the foreground pixels and they become bigger while holes get smaller (Fisher *et.al.*, 2000). Figure 3.26 represents operations with 3x3 sized structuring element used for both erosion and dilation.

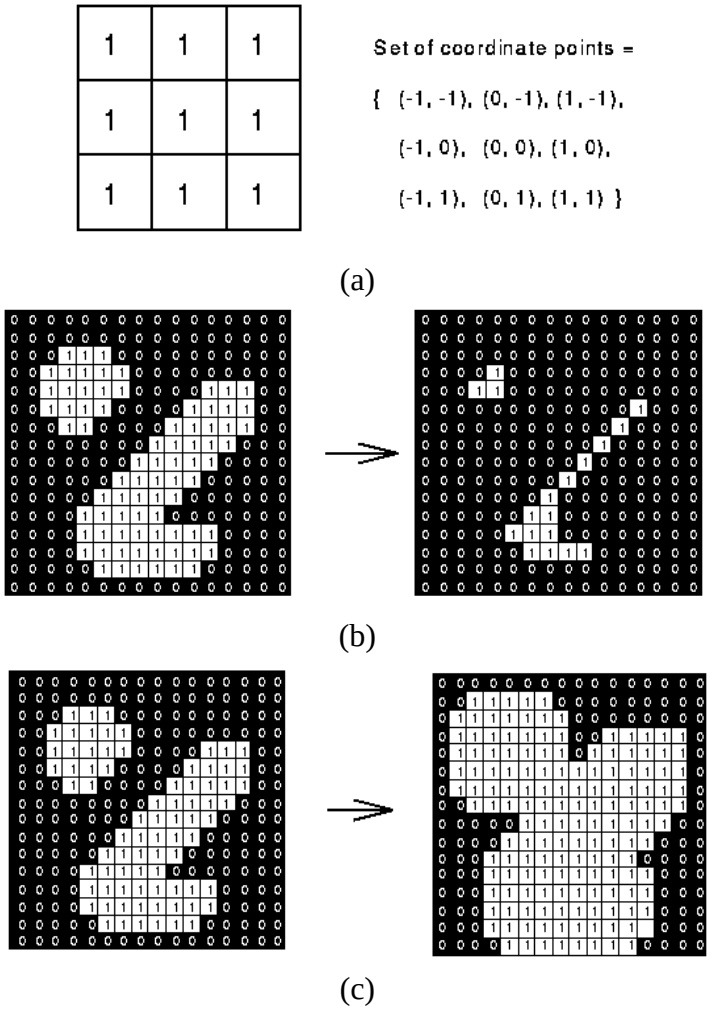


Figure 3.26. Basic morphological operators (a: 3x3 Structuring Element, b: Erosion, c: Dilation) (Fisher *et.al.*, 2000)

In this research, closing operator which is a dilation followed by an erosion is utilized both for floor plan and elevation predictions. A 3x3 sized structuring element with rectangular-shape is used for floor plan predictions. Any other special structuring element for openings is not used because there are only walls in floor plan segmentations. For elevation segmentations, the structuring element is constituted with rectangular-shape based on wall height and width sized grid to designate the overall wall. However, a 3x3 sized structuring element with rectangular-shape is

sufficient for opening extraction. Figure 3.27 shows morphological transformations in floor plan and elevation predictions.

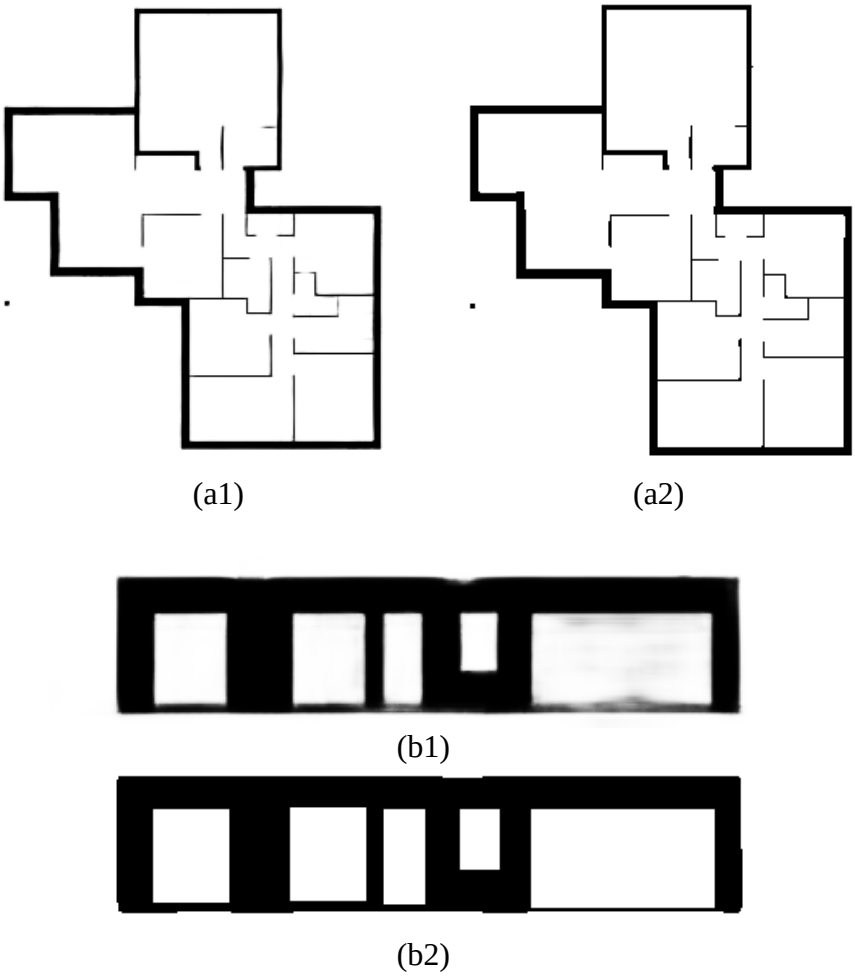


Figure 3.27. Morphological transformations (a1: prediction of floor plan, a2: transformation result; b1: prediction of elevation, b2: transformation result)

3.3.2. Contouring

According to Suzuki and Abe (1983), any black-and-white image can be computed with the help of ‘0’s and ‘1’s representing color intensity. These 0s and 1s are corresponded to each other so that a relation can be built. Firstly, the image size in pixels is accepted as origin point on the upper left corner of the image. The algorithm continues step by step first from top to bottom, then from left to right. While

processing each pixel, if it encounters with a ‘1’, it counts that ‘1’ as a child border point and gives a new value. Process continues with the same steps and finally points with same values are recognized and connected to create a contour (Figure 3.28).

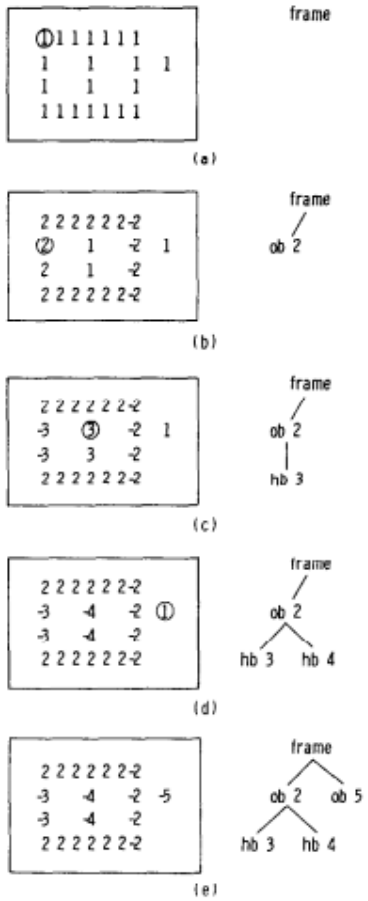


Figure 3.28. Contouring illustration (Suzuki & Abe, 1983)

It is straightforward to contour morphologically transformed floor plan and elevation predictions since they are binary images. Contoured predictions of floor plan and elevation can be seen in Figure 3.29.

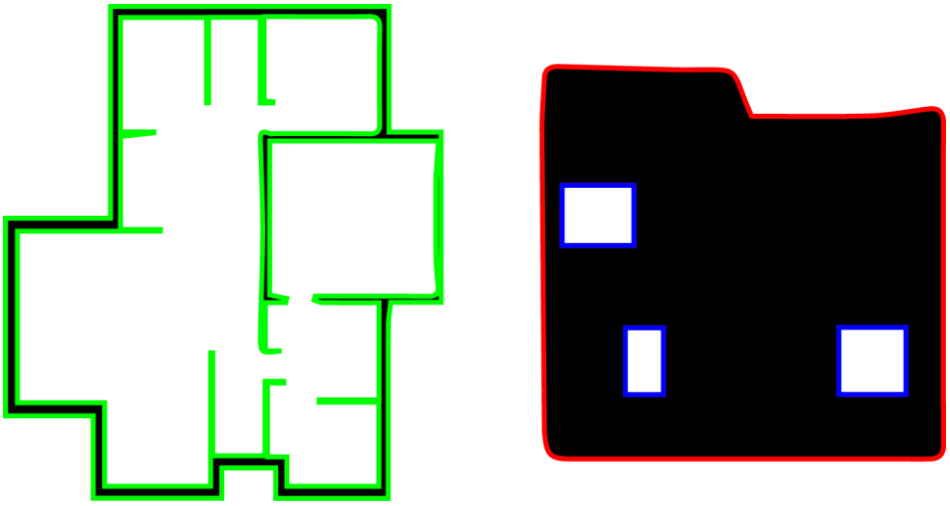


Figure 3.29. Predicted floor plan and elevation contours (produced by the author)

3.3.3. Conversion to 3D

Contours include 2D point information in an image. Thus, these points provide an environment from 2D to 3D conversion. The conversion process is based on six steps. Firstly, each of the elevation contours is aligned to the related floor plan as north, east, south and west. There is no solution for predicting the directions from architectural drawings, so a user should specify the elevations beforehand. After alignment, elevations should be scaled with reference to floor plan edge lengths since elevations may not be predicted as equal to each other in image size.

Alignment also provides height information. This is an important step for an automated flow because height is mostly determined as a default value in most of the previous studies. Then, height information is used to extrude floor plan contours through z-axis. Finally, aligned elevations and extruded wall mass are merged. Contours of openings on the elevations are extruded and intersected with the mass. Thus, intersected areas are subtracted from 3D envelope. The end result of the model includes only floor, outer walls with openings as void, inner walls and columns if applicable. Figure 3.30 shows the main pipeline of the process, and Figure 3.31 illustrates the representative model generation.

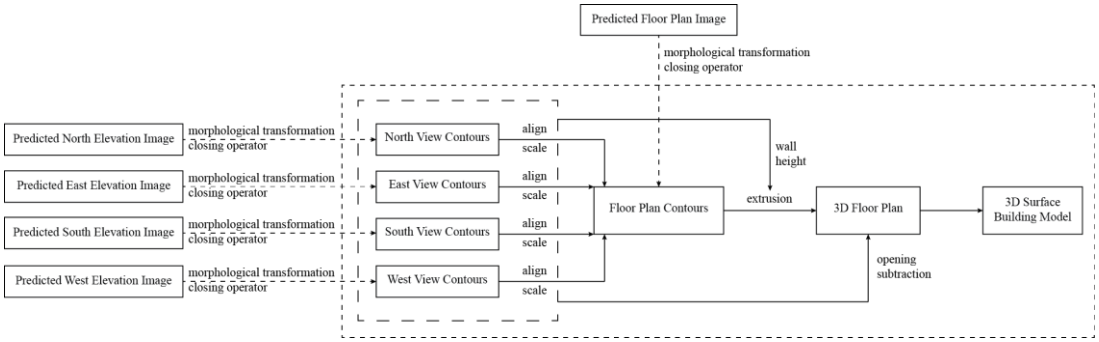


Figure 3.30. Floor plan and elevation alignment illustration (produced by the author)

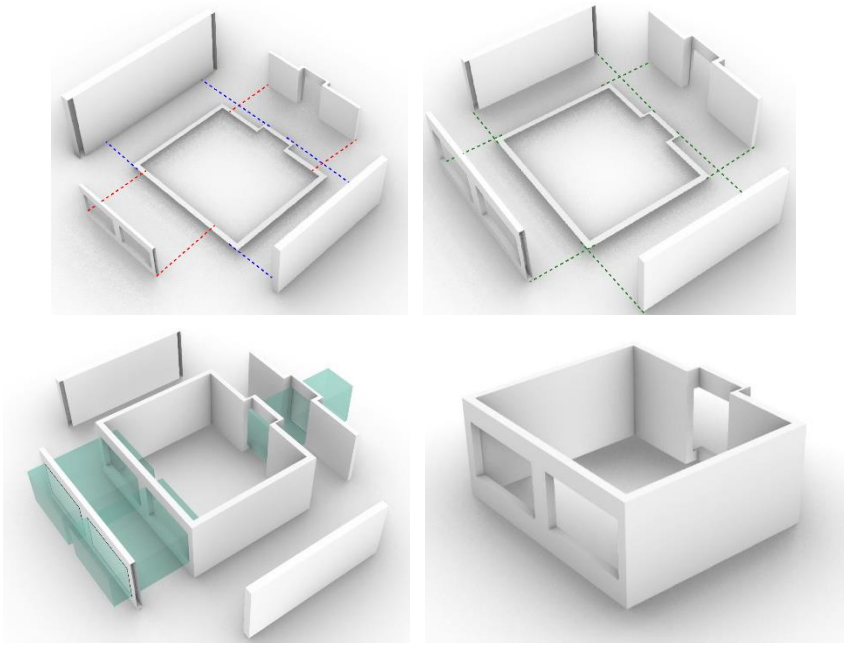


Figure 3.31. Representative surface model generation (produced by the author)

3.4. Web Application

3D model generation implementation in the scope of this research is also presented in a web application. Weights taken from the trained semantic segmentation algorithm is the back-bone of this application. Users who even have no prior knowledge on 3D modelling can utilize it with ease. Usage of the web application steps as follows and interface of the web application can be seen in Figure 3.32.

- A user specifies one floor plan and its four elevation drawings with directions.
- Drawings are directly dragged & dropped to the web server by the user.
    - Size of the drawings are re-scaled according to the train dataset samples.
    - Resized floor plan drawing is predicted with floor plan segmentation weights
    - Resized elevation drawings are predicted with elevation segmentation weights.
    - Predicted results are transformed, contoured and converted to 3D.
- 3D model is automatically downloaded to the user device as STL format.

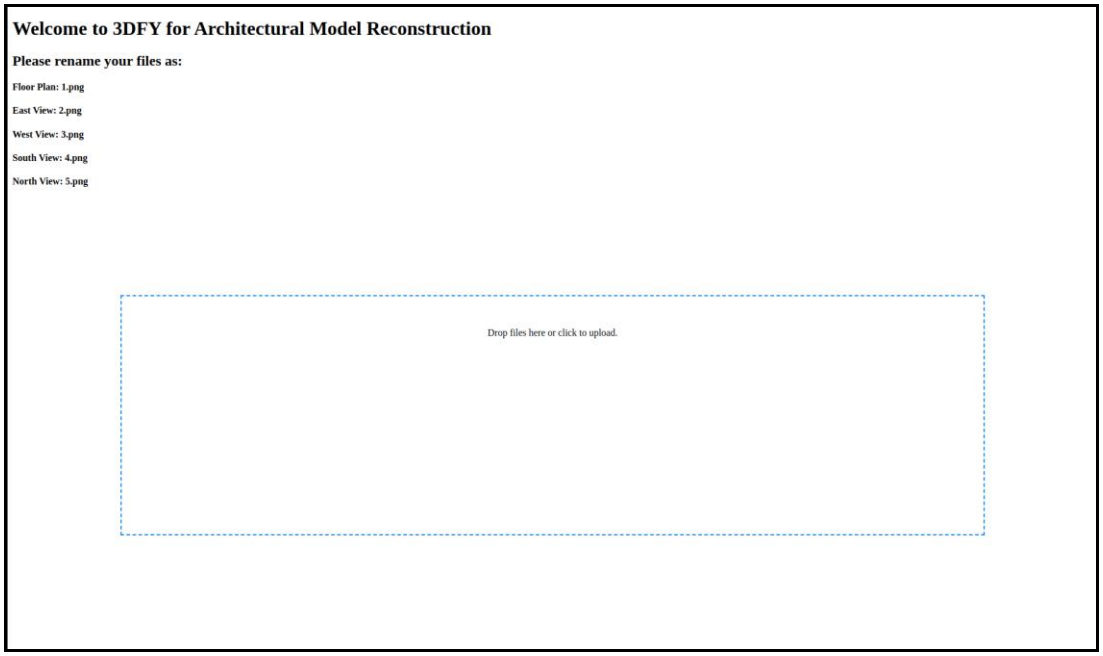


Figure 3.32. Web application interface (produced by the author)

CHAPTER 4

RESULTS

Automated 3D architectural model reconstruction of this research is based on several steps. Floor plan and elevation drawings with corresponding labels are prepared as two separate datasets to highlight walls and openings. Later, these datasets are trained with various CNN architectures to gain proper geometrical patterns through architectural drawings. While these patterns are being recognized by convolving around raster drawing images, the low level information that they include is stored as "weights" to be used for segmenting the new samples that are not in the dataset. Segmented drawing images are vectorized in 2D environment after a morphological transformation process and converted into higher level representation: 3D models.

Precision of the 3D models is highly related to the outcomes of semantic segmentation phase, and should be evaluated. This evaluation is achieved by fundamental evaluation metrics such as Pixel Accuracy and Intersection over Union. These metrics are established to compare ground-truth labels with predicted results to assess the pixel-wise accuracy in 2D environment. Then, 3D models that are generated with predicted drawings are compared with their ground-truth 3D models to verify and ensure the calculated accuracy in 2D pixel environment, which will be held in the following chapter.

The results will be compared according to their evaluation metrics and cost function performance. The possible outcomes are evaluated by training both augmented and non-augmented datasets. Accuracy of these trainings are measured and compared to each other by utilizing original and reconfigured structures of U-Net, SegNet and

TernausNet to find out the best in performance for an appropriate 3D architectural model generation process.

All of the training and evaluation processes are conducted in a workstation having 8 core Intel Xeon CPU at 3.5GHz, NVIDIA Quadro M4000 GPU and 32GB memory. It operates with Linux 16.04 LTS. All the algorithms are developed under Python 3.7, and Keras with Tensorflow backend is the main library for creating CNN models. Implementation of this research is publicly available.[1]

4.1. Evaluation Metrics

The performances of the methods are evaluated according to four semantic segmentation metric sets on both floor plan and elevation datasets. These metrics are calculated according to the ground-truth and predicted pixels.

Mean Intersection over Union (Mean IoU) is a method based on the pixel area that remains in the intersection of predicted image with ground-truth image. Formula is as below:

$$Mean\ IoU = \frac{ground\ truth\ \cap prediction}{ground\ truth\ \cup prediction}$$

$$= \frac{1}{n_{cl}} * \frac{\sum_i n_{ii}}{t_i + \sum_j n_{ji} - n_{ii}}$$

Pixel Accuracy is the percentage of correctly labelled pixels. Its calculation relies on pixels that are correctly predicted to a given class (true positive) and pixels that are correctly identified but not belonging to the class (true negative). Pixel accuracy calculation is as follows:

$$Pixel\ Accuracy = \frac{TP + TN}{TP + TN + FP + FN}$$

[1] https://github.com/gyetis

$$= \frac{\sum_i n_{ii}}{\sum_i t_i}$$

Mean Accuracy calculates the accuracy of per class. It averages the percentages of pixels that are correctly classified per pixel. Its calculation is as follows:

$$Mean\ Accuracy = \frac{TP}{TP + FN}$$

$$= \frac{1}{n_{cl}} * \sum_i \frac{n_{ii}}{t_i}$$

Frequency Weighted IoU (FWIoU) is similar to mean IoU in terms of accuracy calculation. The underrepresented class pixels gain smaller weights than presented ones, and intersection based calculation is achieved by these weights.

$$Frequency\ Weighted\ IoU = \sum_k t_k * \frac{\sum_i t_i n_{ii}}{t_i + \sum_j n_{ji} - n_{ii}}$$

4.2. CNN Results

A CNN architecture is a complex network to perform directly on a customized dataset. Some optimization and configuration steps might be needed to obtain an optimum CNN model since several problems can be encountered while training such as overfitting and underfitting. All these problems may occur if there is any incompatibility between dataset and CNN architecture.

If a model learns a training dataset completely (overfitting) or deficiently (underfitting), a generalized recognition cannot be achieved. To be more clear, if a CNN model identifies an image as it is or perceives it with a narrow perspective, instead of recognizing its features (e.g. corners, lines, geometries and so on), it will be hard for that model to classify and/or segment a new image that is not in the training dataset accordingly.

The success of the training process can be interpreted by feeding CNNs with training and validation sets simultaneously. While training, loss on both train and validation sets that should be minimized as much as possible starts to decrease over each epoch. CNN model achieves its optimum capacity when both loss rates reach their global minimums. Optimum loss values designate the underfitting and overfitting zones as can be seen in Figure 4.1. The optimum training process should end at the global minimum of train and validation losses.

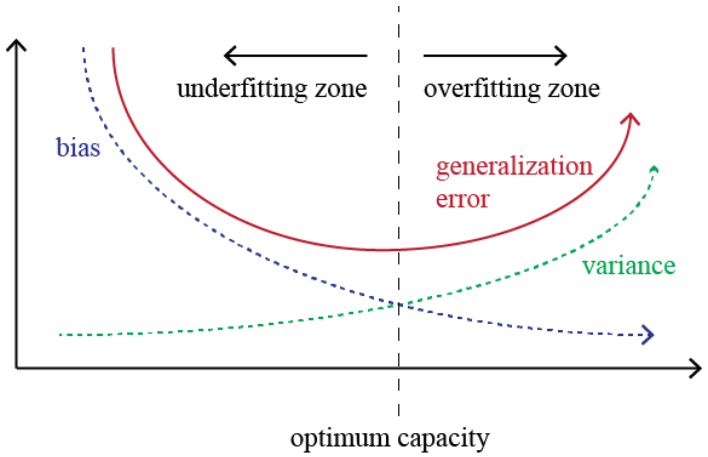


Figure 4.1. Underfitting and overfitting (produced by the author)

There are multiple ways to make a CNN model optimized and correlated with a custom dataset. The first common thing to do is increasing data samples. It can be achieved either adding new data and corresponding labels into the existing dataset or augmenting the dataset. Most of the implementations in the literature are employing data augmentation since adding new data to an existing dataset can be difficult and time consuming. Data augmentation is also utilized in this research as mentioned before, and data is augmented in each epoch differently as training progresses (Figure 4.2). Dataset with and without augmentation are trained separately to comprehend the effect of data augmentation.

Figure 4.2. Data augmentation (produced by the author)

One another approach is choosing a CNN model that generalizes well and/or configuring the model. Some models may be complex for some problems. It is important to analyze the characteristics of CNN models and adjust respectively. Changing the complexity of a model by using fewer number of convolutional layers can help optimization. Likewise, arranging network parameters (e.g. number of image channels, filter size, and so on) can overcome problems on optimization since smaller parameters ensure a lightweight structure. For example, if the filter size that convolves through the input data gets bigger, higher-level and more global information can be obtained (Figure 4.3). Yet, bigger parameters cause more complex models and creates a sensitivity to the statistical fluctuations in the dataset. In this research, only different number of convolutional layers are implemented and compared with original structures. Changing network parameters is not obtainable due to hardware limitations, and consequently, the smallest parameter values are utilized.

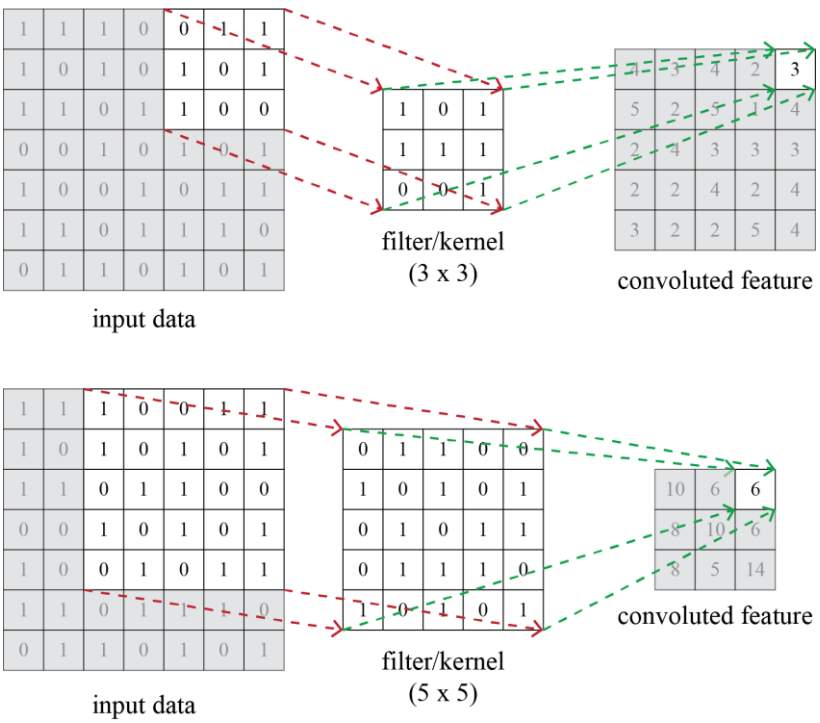


Figure 4.3. Filter/kernel size comparison (produced by the author)

Using regularizations in CNNs is also a helpful approach for optimization. Filter regularizations, early stopping and/or dropout can be integrated in a model so that the model can learn a dataset more properly. Filter regularizations called “L1” and “L2” regularize the model by penalizing the weights. These functions are added to the cost function of the model, and decrease the values of weight matrices. Thereby, underfitting and overfitting can be reduced to an extent. Early stopping is another method for an appropriate training process. It basically terminates the training when the performance rates get worse (Figure 4.4).

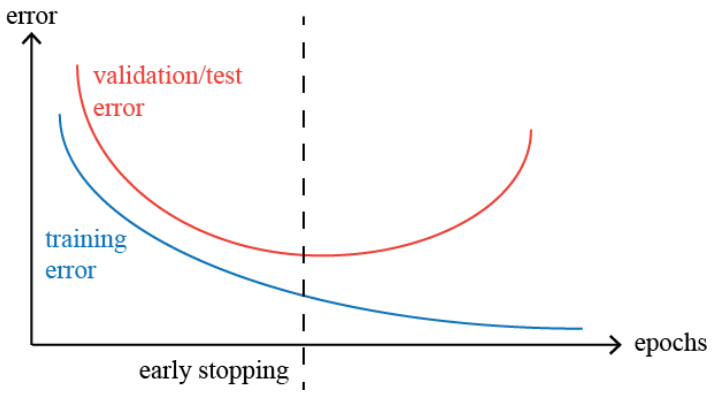


Figure 4.4. Early stopping (produced by the author)

The most common and more stable approach as regularization is dropout which is basically randomly excluding some neurons in the network with a probability. There are complex links between inputs and outputs in a CNN architecture, and may create noise in a limited dataset. By applying dropout to any layer in an architecture, output of the neuron does not overly depend on each of the hidden units. Therefore, co-adaptation in between features is prevented, and predictions of all possibilities in the parameters are averaged (Srivastava, Hinton, Krizhevsky, Sutskever & Salakhutdinov, 2014). Figure 4.5 gives a schematic explanation on dropout.

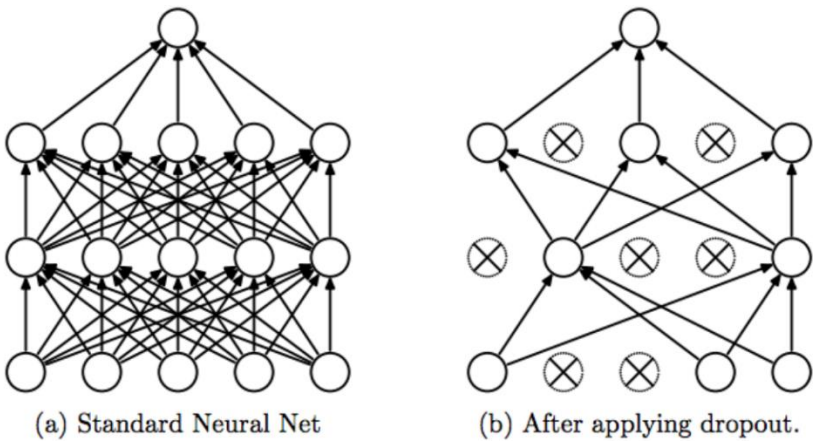


Figure 4.5. Dropout (Srivastava et.al., 2014)

To sum up, it is difficult to attain a good-fitted CNN model. Challenges in data, hardware and training processes should be considered beforehand. For this research, augmented and not augmented dataset are trained both with original structures and reconfigured structures of U-Net, SegNet and TernausNet. It is desired to underscore the dataset's role and effects of CNN structures in semantic segmentation applications.

4.2.1. Original vs. Augmented Datasets with Original CNN Structures

Drawing datasets of this research are not quite large in terms of data samples as in common datasets that are presented in the literature. Current CNN applications have been conducted with large datasets, and they show that the bigger a dataset is, the better results are. Therefore, U-Net, SegNet and TernausNet are trained both with augmented and non-augmented datasets to grasp the difference.

Originally, there are 250 raster drawings in floor plan dataset and 450 raster drawings exist in elevation dataset. After training with the original state of these drawings, data is augmented randomly at each epoch by flipping and/or distorting the images to enrich the variety of the samples in both datasets and trained accordingly.

Original CNN architectures of U-Net, SegNet and TernausNet are utilized with dropout layers and early stopping to avoid underfitting/overfitting. As introduced in the previous chapter, U-Net is configured with 23 convolutional, 4 downsampling and 4 upsampling layers. Its symmetric-like shape enables to copy and crop the features of a downsampled image into corresponding upsampled image. SegNet is constructed upon 26 convolutional, 5 downsampling and 5 upsampling layers. Its encoding part is exactly taken from VGG16 structure and decoding part is designed respectively for semantic segmentation. TernausNet is composed of 19 convolutional, 5 downsampling and 6 upsampling layers. It is an example of transfer learning in which the encoding part is directly taken from VGG16 structure and weights. The decoding

part is similar to the idea of U-Net which is concatenation of the related layers. Figure 4.6 illustrates the models.

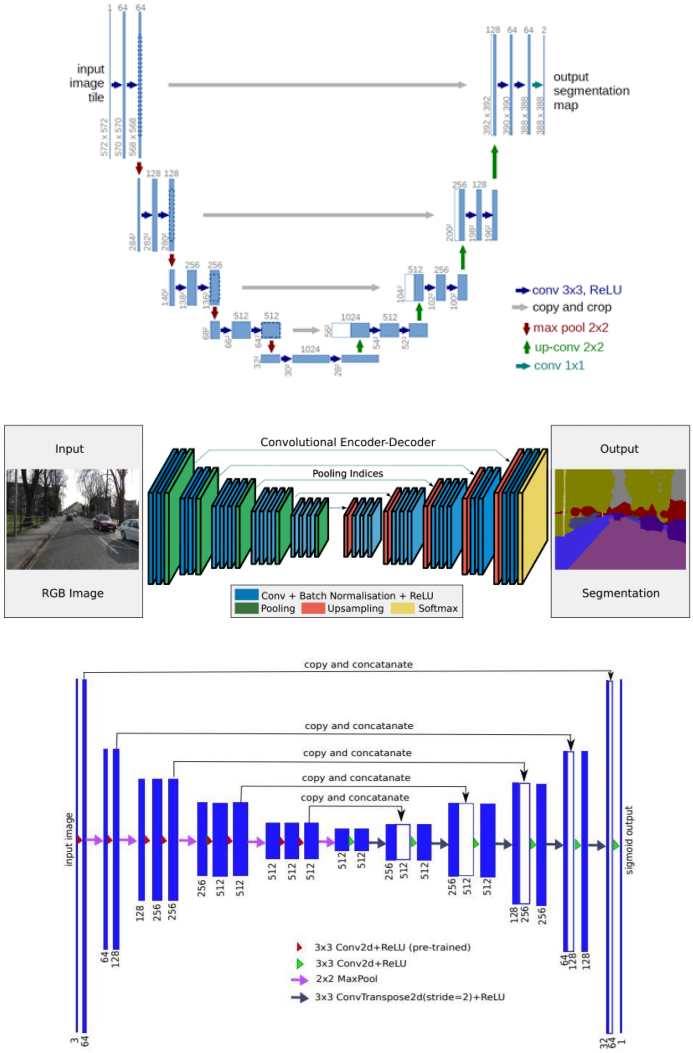
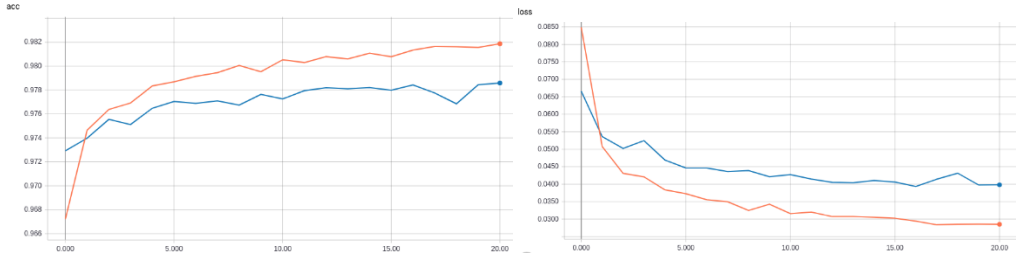


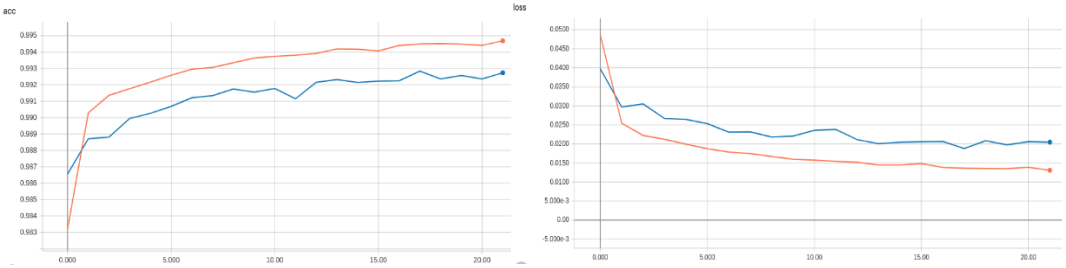
Figure 4.6. U-Net (Ronneberger et.al., 2015), SegNet (Badrinarayanan, Kendall & Cipolla, 2017) and TernausNet (Iglovikov & Shvets, 2018)

As discussed earlier, layers of a CNN architecture may cause undesired results since each epoch reveals the higher level features of an image. Architectural drawings do not generally include high level features as in human face or urban scenes. Thereby, floor plan and elevation datasets are trained with the CNN models shown above separately to evaluate the compatibleness of the original configurations with these

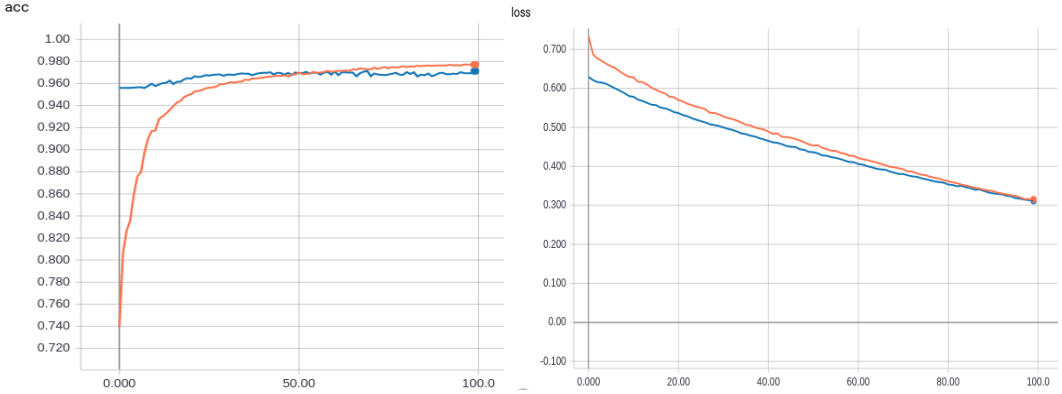
datasets. Figure 4.7 and Figure 4.8 show the learning curves of U-Net, SegNet and TernausNet for both augmented and non-augmented datasets. Detailed versions of the curves can be found in Appendices A.



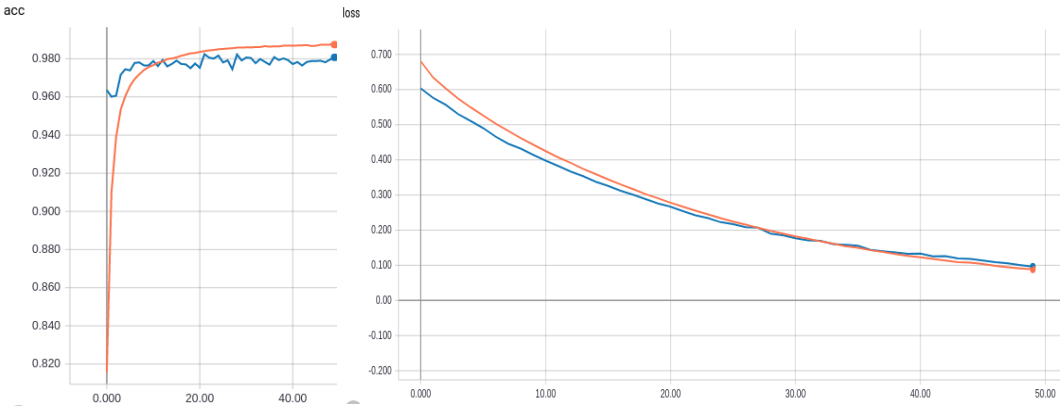
a) U-Net accuracy and loss graphics on *not augmented floor plan dataset*



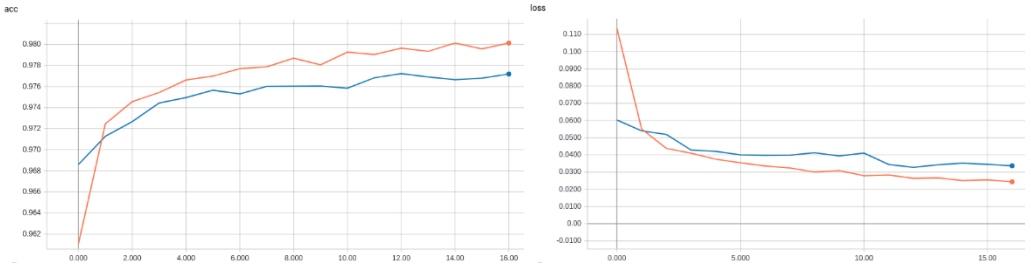
b) U-Net accuracy and loss graphics on *augmented floor plan dataset*



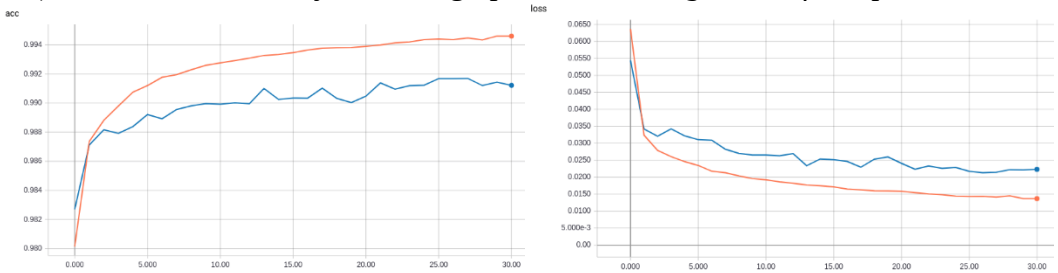
c) SegNet accuracy and loss graphics on *not augmented floor plan dataset*



d) SegNet accuracy and loss graphics on *augmented floor plan dataset*

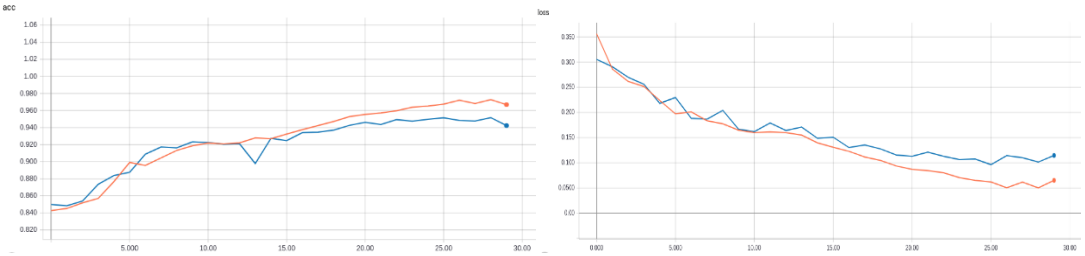


e) TernausNet accuracy and loss graphics on *not augmented floor plan dataset*

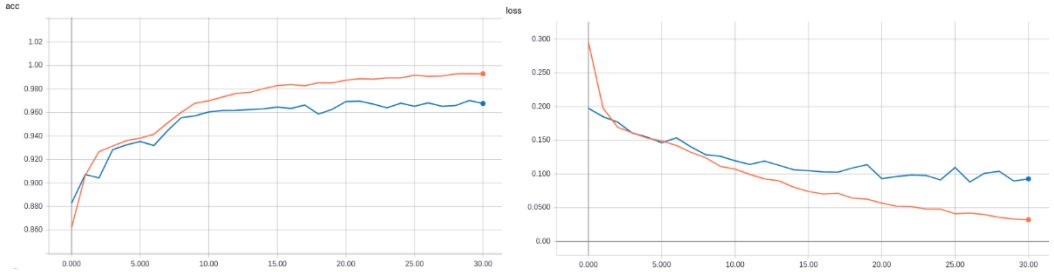


f) TernausNet accuracy and loss graphics on *augmented floor plan dataset*

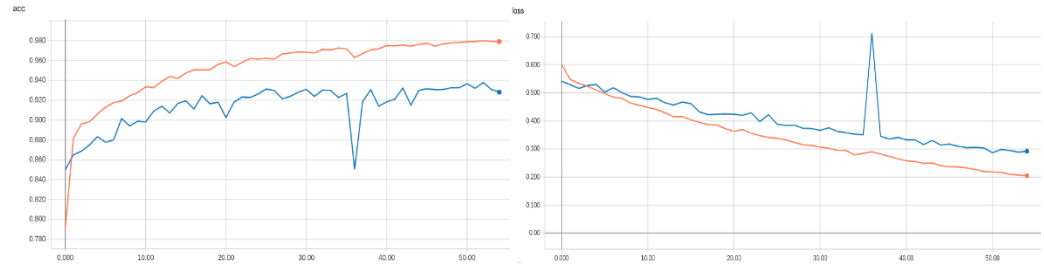
Figure 4.7. CNNs learning curves on augmented and non-augmented floor plan dataset (produced by the author)



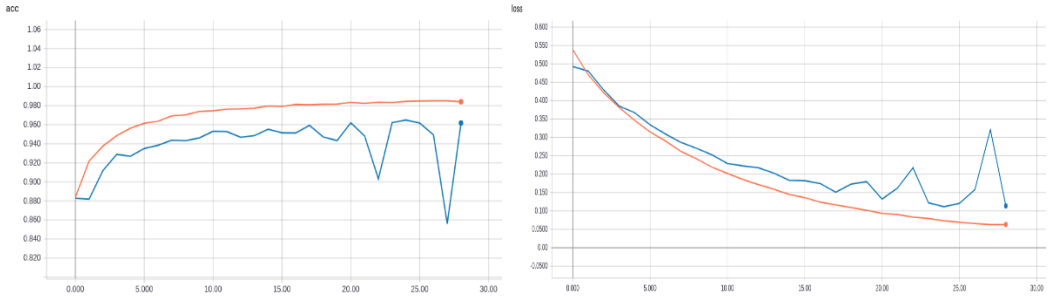
a) U-Net accuracy and loss graphics on *not augmented elevation dataset*



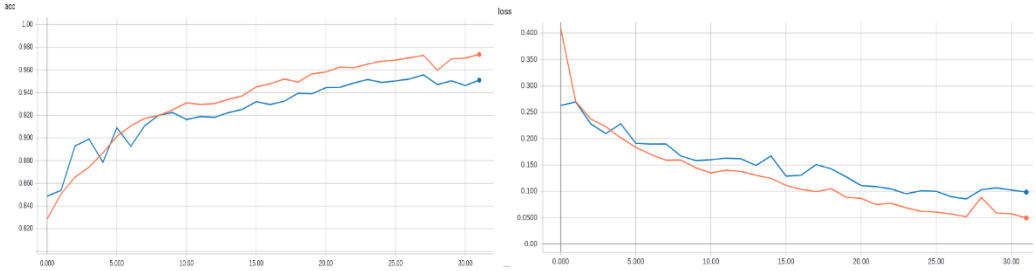
b) U-Net accuracy and loss graphics on *augmented elevation dataset*



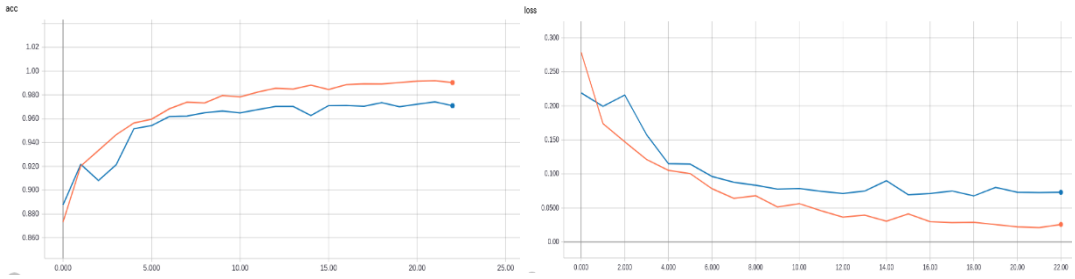
c) SegNet accuracy and loss graphics on *not augmented elevation dataset*



d) SegNet accuracy and loss graphics on *augmented elevation dataset*



e) TernausNet accuracy and loss graphics on *not augmented elevation dataset*



f) TernausNet accuracy and loss graphics on *augmented elevation dataset*

Figure 4.8. CNNs learning curves on augmented and non-augmented elevation dataset (produced by the author)

Floor plan and elevation test sets are predicted according to the weights that are gained from these CNN processes. Even though predictions seem black and white, they are in grayscale and RGB values of desired black and white pixels may vary in between grayscale range. Hence, these predictions are first converted to black and white and evaluated with the metrics mentioned before to overcome grayscale and binary format complication. Some of the predictions can be seen in Figure 4.9. Furthermore, Table 4.1 and Table 4.2 show the relevant evaluation results on floor plan and elevation test sets separately.

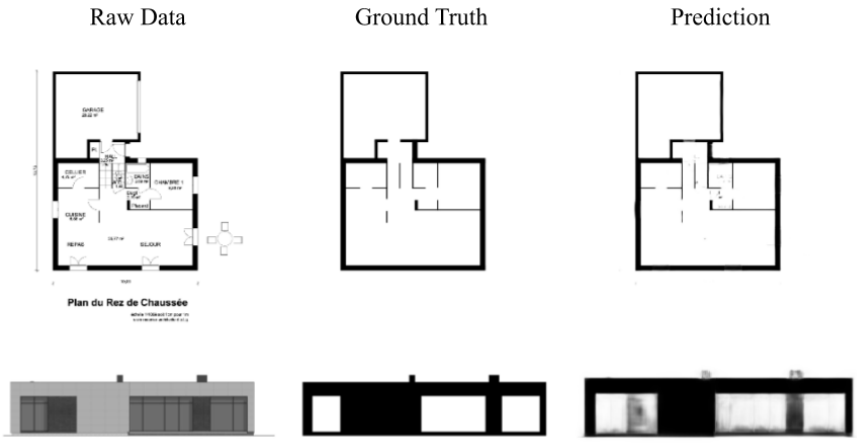


Figure 4.9. Prediction of architectural drawing examples with original structures (produced by the author)

Table 4.1. *Evaluation results on floor plan test set*

	Accuracy	Loss	Pixel Accuracy	Mean Accuracy	Mean IoU	FWIoU
Original dataset with U-Net	0.981	0.029	0.941	**0.646**	0.574	0.914
Augmented dataset with U-Net	**0.994**	**0.013**	0.949	0.641	**0.603**	0.924
Original dataset with SegNet	0.977	0.315	0.944	0.554	0.523	0.913
Augmented dataset with SegNet	0.987	0.088	**0.955**	0.581	0.552	**0.926**
Original dataset with TernausNet	0.979	0.026	0.941	0.644	0.567	0.913
Augmented dataset with TernausNet	**0.994**	0.014	0.947	0.645	0.596	0.921

Table 4.2. *Evaluation results on elevation test set*

	Accuracy	Loss	Pixel Accuracy	Mean Accuracy	Mean IoU	FWIoU
Original dataset with U-Net	0.967	0.061	0.851	**0.770**	0.618	0.771
Augmented dataset with U-Net	0.990	0.042	0.856	0.759	0.626	0.777
Original dataset with SegNet	0.978	0.218	0.856	0.752	0.616	0.775
Augmented dataset with SegNet	0.984	0.072	**0.860**	0.743	0.622	**0.780**
Original dataset with TernausNet	0.972	0.051	0.848	0.767	0.619	0.769
Augmented dataset with TernausNet	**0.992**	**0.021**	0.858	0.754	**0.628**	0.779

4.2.2. Original vs. Augmented Datasets with Reconfigured CNN Structures

CNN configuration with several approaches prevent many problems that are pointed out previously. Changing the network parameters is infeasible in the realm of this research due to hardware limitations. Thus, only network layer reconfiguration is applied for all the CNN models along with dropout and early stopping.

Similar to original CNN structure evaluation, reconfigured CNNs are also trained with augmented and non-augmented datasets to understand the effects of dataset and reconfiguration approaches in a nutshell. All of the CNN models' last encoding

convolutional layer and first decoding layer are deducted. The reason behind not increasing the convolutional layers is that more convolutional layers reveal higher features of the dataset which are not necessary in architectural datasets since the original structure results show no underfitting. CNNs with less layers become more lightweight, consequently training time is reduced. Figure 4.10 shows the reconfigured CNNs.

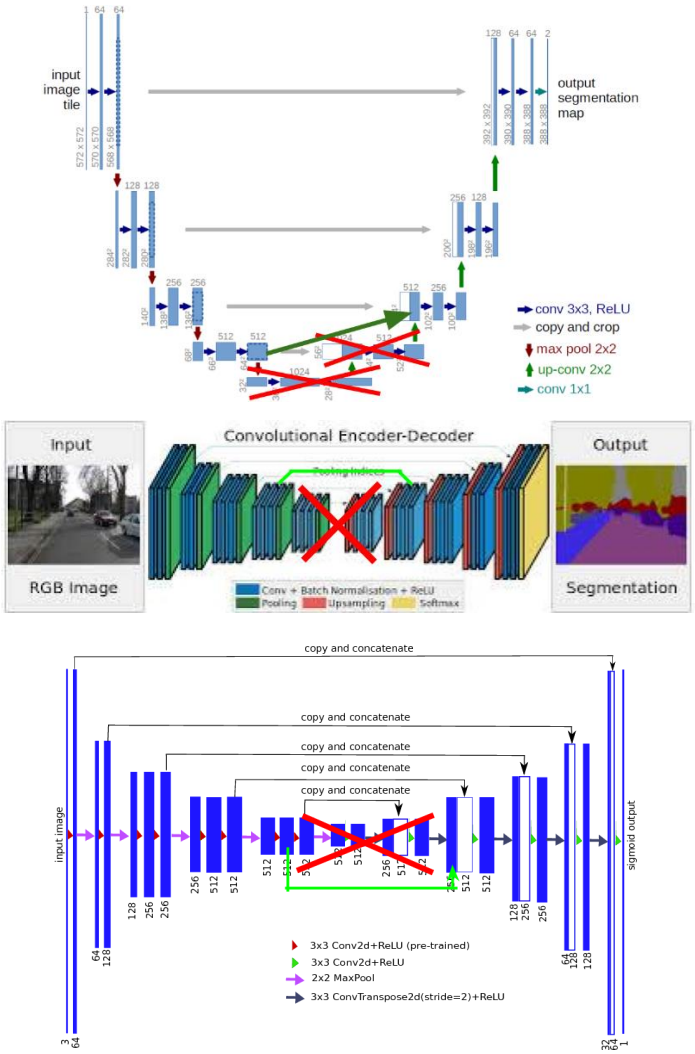
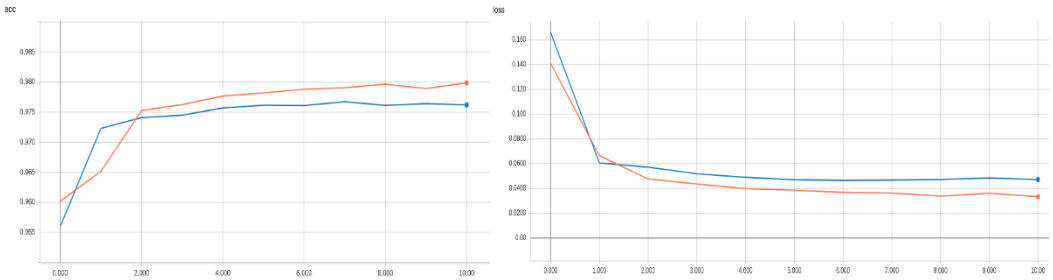
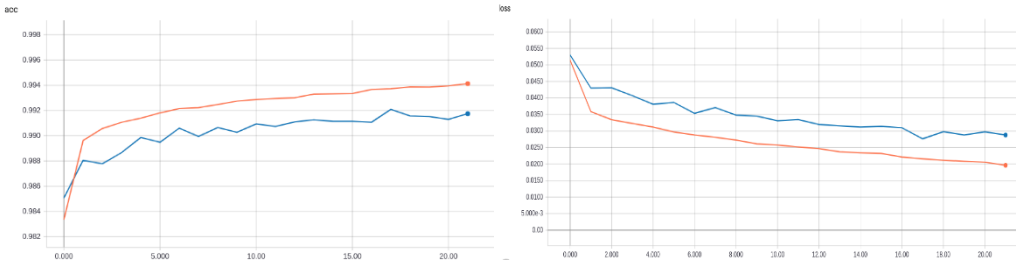


Figure 4.10. Reconfigured U-Net, SegNet and TernausNet (adapted from Ronneberger et.al., 2015; Badrinarayanan, Kendall & Cipolla, 2017; Iglovikov & Shvets, 2018)

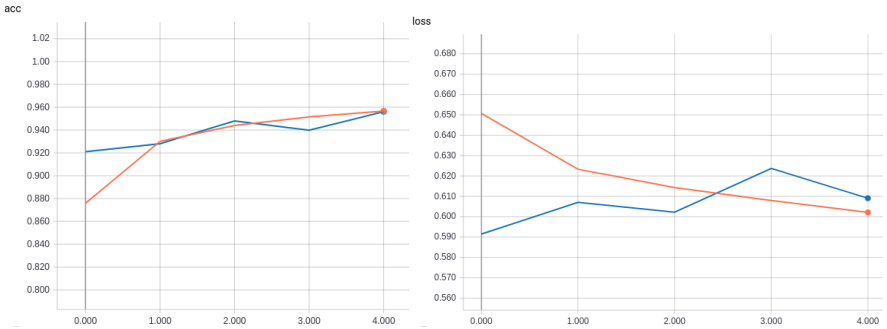
CNNs with less convolutional layers with dropout and early stopping approaches are performed on both of the datasets. Figure 4.11 and Figure 4.12 represent the learning curves of the reconfigured architectures for both augmented and non-augmented datasets. These graphics can be found under Appendices B.



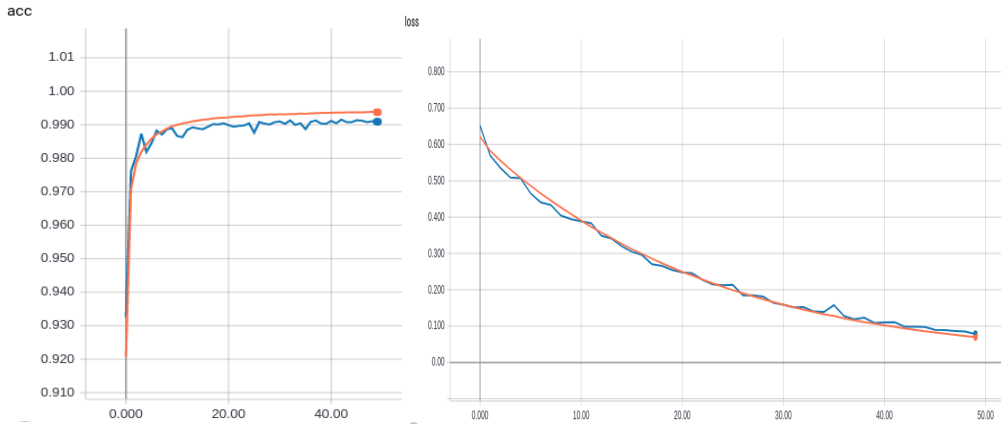
a) *Reconfigured* U-Net accuracy and loss graphics on *not augmented floor plan dataset*



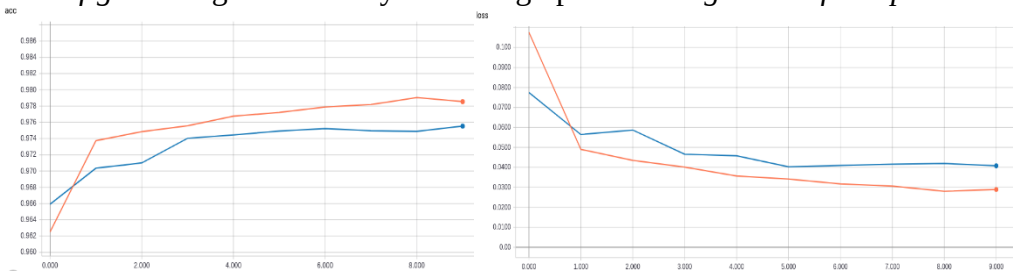
b) *Reconfigured* U-Net accuracy and loss graphics on *augmented floor plan dataset*



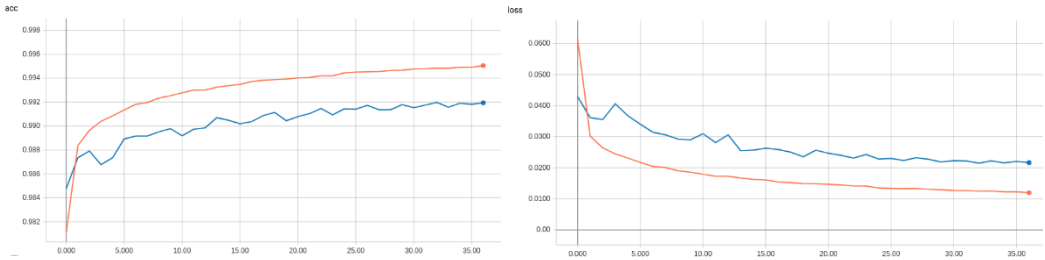
c) *Reconfigured* SegNet accuracy and loss graphics on *not augmented floor plan dataset*



d) *Reconfigured* SegNet accuracy and loss graphics on *augmented floor plan dataset*

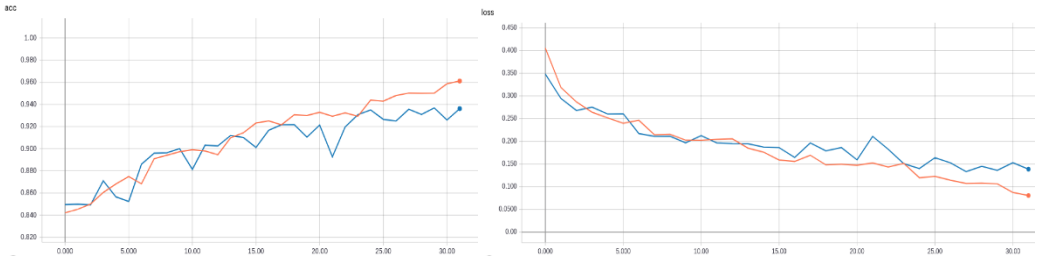


e) *Reconfigured* TernausNet accuracy and loss graphics on *not augmented floor plan dataset*

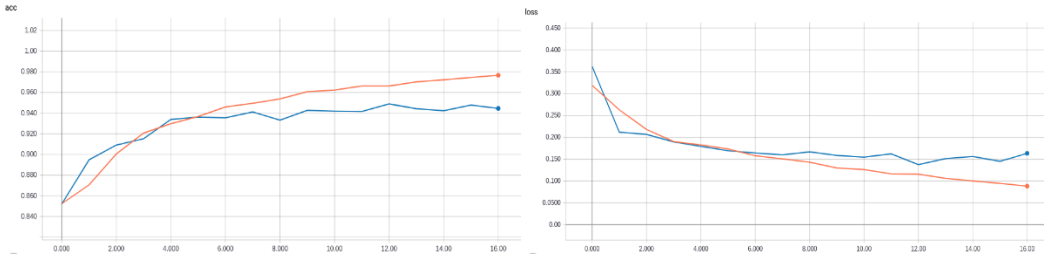


f) *Reconfigured* TernausNet accuracy and loss graphics on *augmented floor plan dataset*

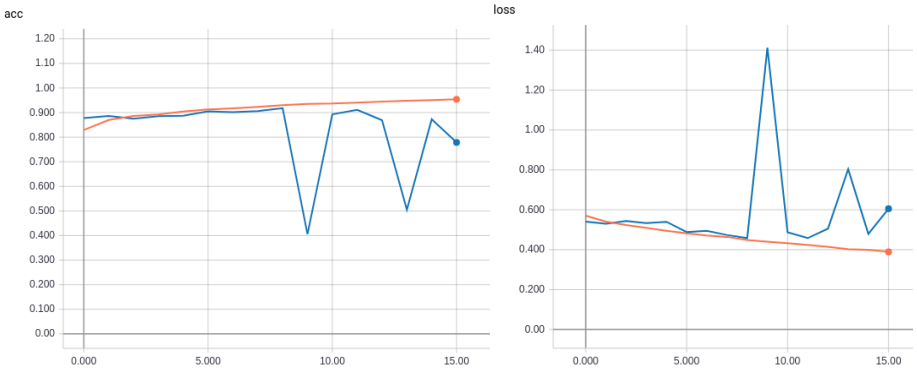
Figure 4.11. Reconfigured CNNs learning curves on augmented and non-augmented floor plan dataset (produced by the author)



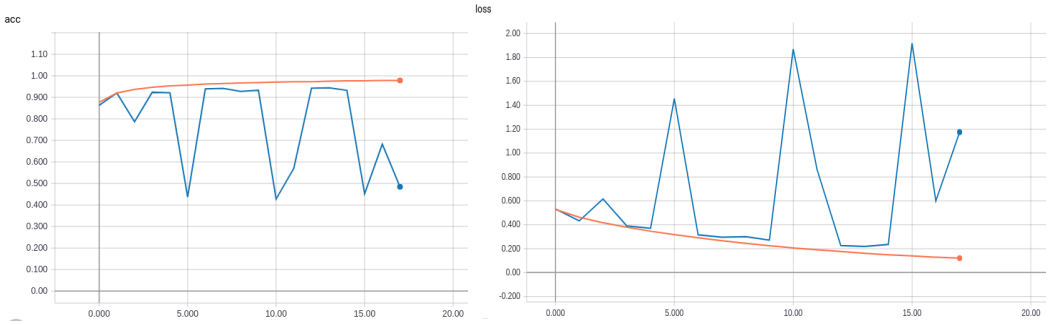
a) *Reconfigured* U-Net accuracy and loss graphics on *not augmented elevation dataset*



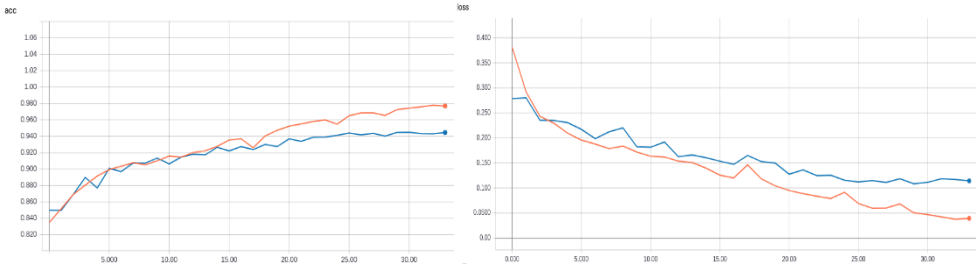
b) *Reconfigured* U-Net accuracy and loss graphics on *augmented elevation dataset*



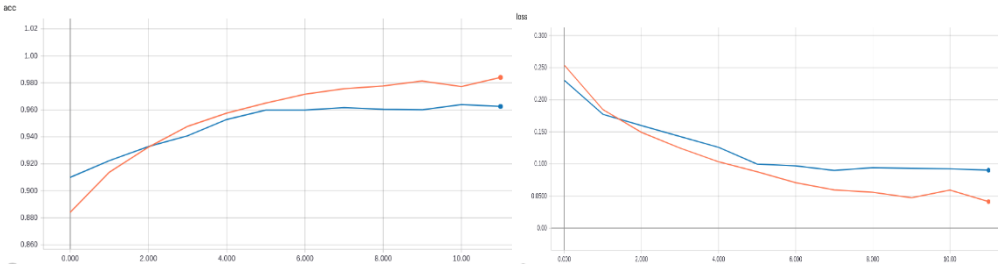
c) *Reconfigured* SegNet accuracy and loss graphics on *not augmented elevation dataset*



d) *Reconfigured* SegNet accuracy and loss graphics on *augmented elevation dataset*



e) *Reconfigured* TernausNet accuracy and loss graphics on *not augmented elevation dataset*



f) *Reconfigured* TernausNet accuracy and loss graphics on *augmented elevation dataset*

Figure 4.12. Reconfigured CNNs learning curves on augmented and non-augmented elevation dataset (produced by the author)

Test sets of floor plan and elevation drawings are predicted with the weights of above training procedures. Similar to original CNN configuration comparison, predictions are evaluated and compared to each other with reconfigured CNN architectures. Some of the predictions are represented in Figure 4.12. Also, Table 7 and Table 8 display the evaluations on floor plan and elevation test sets separately.

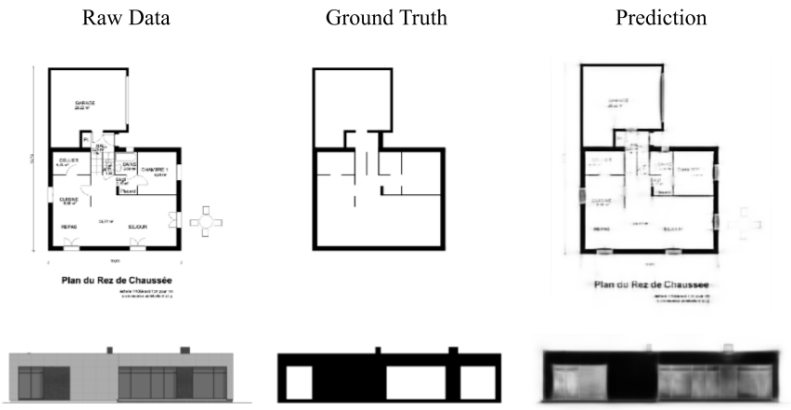


Figure 4.13. Prediction of architectural drawing examples with reconfigured structures (produced by the author)

Table 4.3. *Evaluation results with reconfigured structures on floor plan test set*

	Accuracy	Loss	Pixel Accuracy	Mean Accuracy	Mean IoU	FWIoU
Original dataset with reconfigured U-Net	0.978	0.036	0.941	0.643	0.570	0.914
Augmented dataset with reconfigured U-Net	0.993	0.021	0.947	0.641	0.594	0.921
Original dataset with reconfigured SegNet	0.875	0.650	0.942	0.606	0.551	0.914
Augmented dataset with reconfigured SegNet	0.993	0.070	**0.950**	0.634	0.593	**0.924**
Original dataset with reconfigured TernausNet	0.977	0.034	0.943	0.638	0.575	0.916
Augmented dataset with reconfigured TernausNet	**0.994**	**0.012**	0.948	**0.642**	**0.597**	0.922

Table 4.4. *Evaluation results with reconfigured structures on elevation test set*

	Accuracy	Loss	Pixel Accuracy	Mean Accuracy	Mean IoU	FWIoU
Original dataset with reconfigured U-Net	0.950	0.106	0.848	0.768	0.614	0.767
Augmented dataset with reconfigured U-Net	0.966	0.115	0.854	0.764	**0.627**	**0.776**
Original dataset with reconfigured SegNet	0.940	0.424	0.852	0.735	0.601	0.769
Augmented dataset with reconfigured SegNet	0.975	0.162	**0.856**	0.727	0.605	0.773
Original dataset with reconfigured TernausNet	0.972	**0.050**	0.846	**0.771**	0.611	0.765
Augmented dataset with reconfigured TernausNet	**0.975**	0.059	0.853	0.747	0.618	0.773

4.2.3. Result Comparison

According to the training graphics and evaluation metrics on both datasets in different conditions, it is observed that dataset characteristics are really important for training processes. While floor plan dataset involves similar shape grammar, elevation dataset does not contain that generalized grammar since elevation drawings can have shadows, railings, different window types and so on. This is why elevation data samples almost two times of floor plan samples to overcome the foreseen problems. Yet, evaluation metric results on elevation dataset are not good as floor plan. It is

deduced that more data samples should added to the elevation dataset to increase prediction performance.

Another observation is that data augmentation increases the performance of the CNN models. For example, TernausNet on augmented floor plan data has an accuracy of 0.994 and a loss of 0.014 while having an accuracy of 0.979 and a loss of 0.026 on not augmented floor plan dataset. This situation is also valid for reconfigured CNN structure training processes. It is proved that data augmentation is important for deep learning applications, especially when there are few data samples in a dataset.

It is detected that even though fewer convolutional layers in a model provide a more lightweight structure and less training time, convergence problems may occur. The number of training steps (the number of epochs) decreases, but the reconfigured model tends to overfit easier than original one according to the observations of this research. Floor plan training shows that original U-Net and TernausNet are strong enough to learn floor plan features. However, reconfigured SegNet structure performance is better than the original architecture on floor plan dataset. In the meantime, none of the reconfigured models are coherent enough to learn elevation features, and this proves once again that elevation features are more complex to learn than floor plan. Eventually, original configurations of the CNN models perform better than reconfigured ones. Although differences on the evaluation rates are not much, original configurations are good enough to learn architectural features on both of the datasets.

CHAPTER 5

CASE STUDIES

Transforming low level information to higher level is represented with two case studies in this chapter. Low level information obtained from 2D architectural datasets by training with different CNN models become readily available for conversion to 3D models. The previous chapter shows that original configurations of U-Net, SegNet and TernausNet perform better than reconfigured ones. Therefore, their weights are opted for the transformation process in the scope of this research.

Walls and openings of an architectural drawing are unraveled with semantic segmentation implementation. These architectural elements are transformed morphologically and vectorized to construct 3D architectural models. Model generation is based on floor plan, aligning elevations to the related floor plan to extrude walls and subtract openings. Generated model can be downloaded directly from a web application that is designed for this dissertation.

The last step of evaluation on 3D model regeneration based on 2D drawings with deep learning and digital geometry processing relies on calculating the time spent while reconstruction process, and assessing the accuracy of generated model on 3D environment. This accuracy is similar to Mean IoU. Since any modelling tool is actually based on Boolean operations, the calculation of reconstructions is established with these operations as well. The logic behind the assessment is basically using intersection and union of the ground truth and generated 3D model volumes, and calculated as follows:

$$Volumetric\ IoU = \frac{ground\ truth\ volume\ \cap generated\ volume}{ground\ truth\ volume\ \cup generated\ volume}$$

Two case studies are demonstrated to analyze the implementation perks, drawbacks and limitations. Ground truth of the buildings are prepared as mesh models in which openings are subtracted from the walls and left as voids since the generated models have the same characteristics. 3D model generations are executed with predictions based on the weights that are extracted by training augmented datasets with original U-Net, SegNet and TernausNet configurations.

5.1. First Case

The first case is a flat-roof, small-scale and one-storey height housing. Figure 5.1 shows floor plan and elevation drawings, and illustrates the ground truth 3D model.

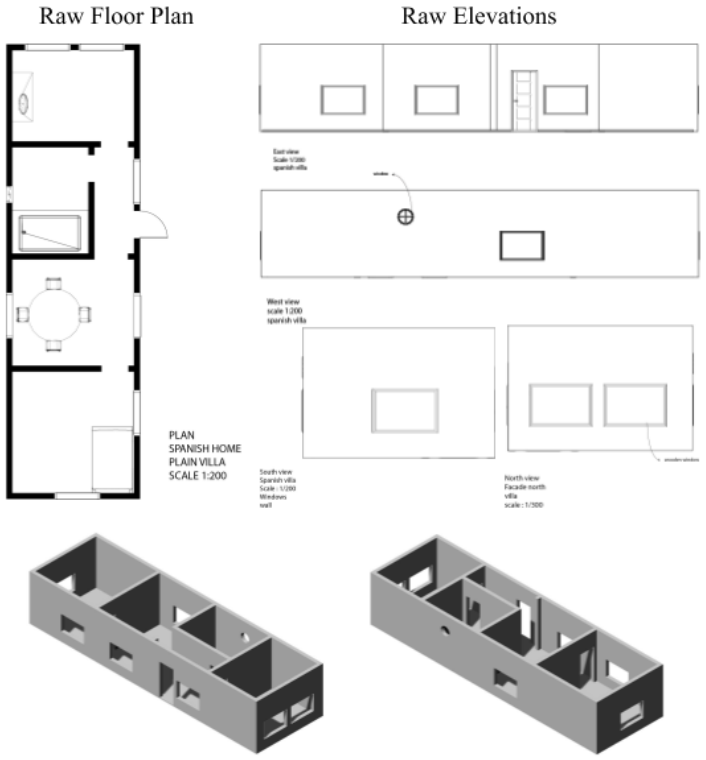


Figure 5.1. Drawings and 3D ground-truth model for First Case (produced by the author)

The raster architectural drawings are predicted with the introduced CNNs. Predictions are resized according to the required CNN input size, morphologically transformed

and vectorized which are then converted to 3D environment with relevant geometry processing algorithms. The comparative volumetric evaluation and required time for the generation process are presented in Table 5.1.

Table 5.1. *3D model generation results on First Case*

	U-Net	SegNet	TernausNet
Volumetric IoU	0.685	0.519	0.719
Time	20.57 sec.	29.51 sec.	18.41 sec.

Results can be visualized better with architectural drawings predictions and isometric views of generated models (Figure 5.2). They all are processed with U-Net, SegNet and TernausNet weights that are explained in the previous chapter.

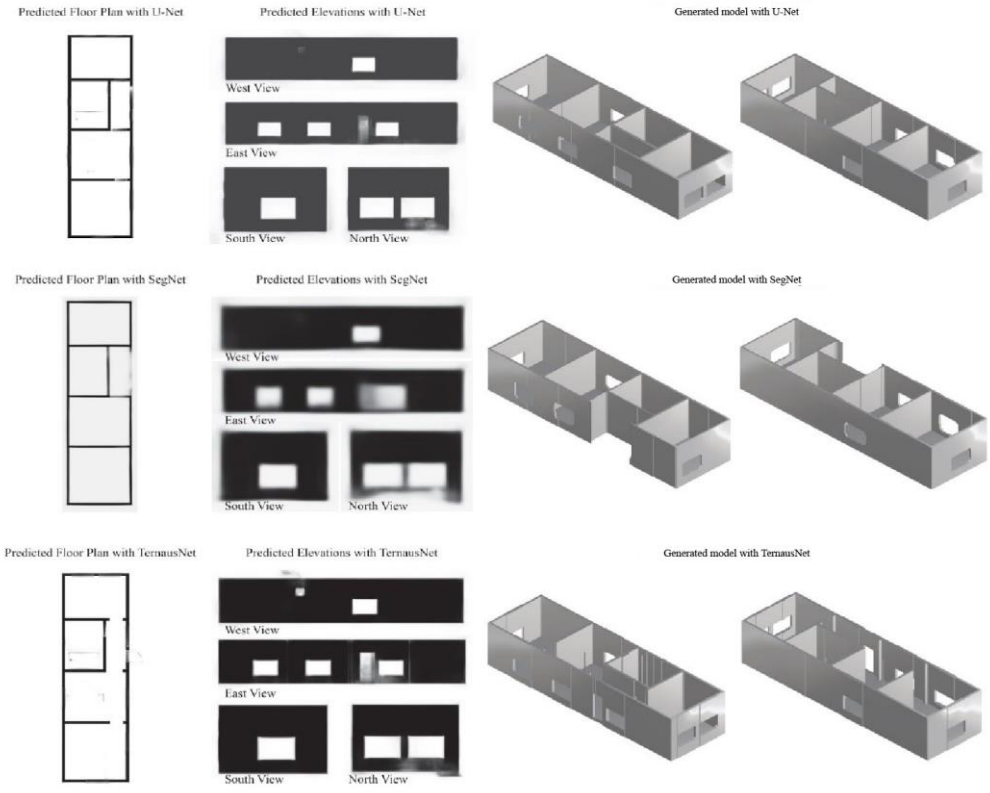


Figure 5.2. Predicted architectural drawings and isometric views of 3D model on First Case (produced by the author)

5.2. Second Case

The second case is a low pitched roof, and bigger scale than previous housing which includes different storey levels. Relevant architectural drawings are present in Figure 5.3.

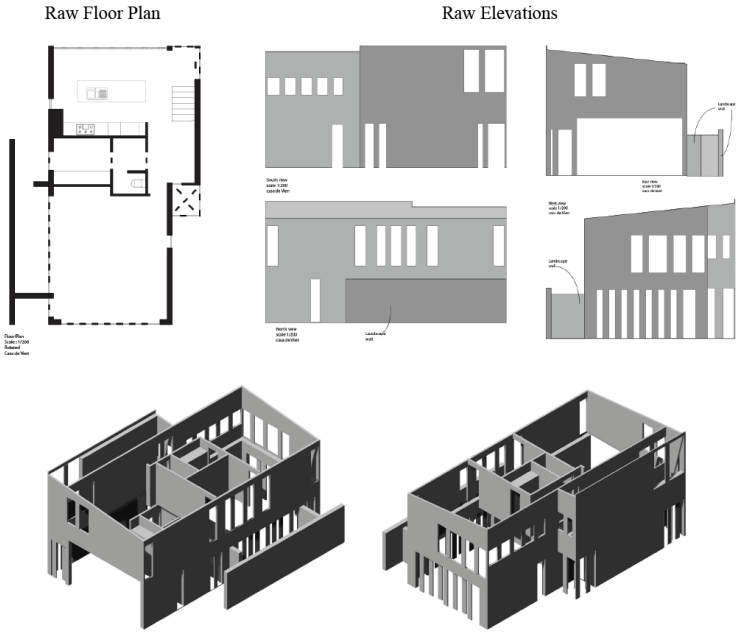


Figure 5.3. Drawings and 3D ground-truth model for Second Case (produced by the author)

Similar to first case, second case raster drawings are test with CNNs introduced in this research. Predicted images are resized, transformed and vectorized for a conversion process. The volumetric comparison and required time for 3D generation are shown in Table 5.2.

Table 5.2. *3D model generation results on Second Case*

	U-Net	SegNet	TernausNet
Volmetric IoU	0.668	0.468	0.238
Time	24.05 sec.	31.11 sec.	19.69 sec.

Second case predictions with generated models are visualized in Figure 5.4. Predictions are achieved with U-Net, SegNet and TernausNet, respectively.

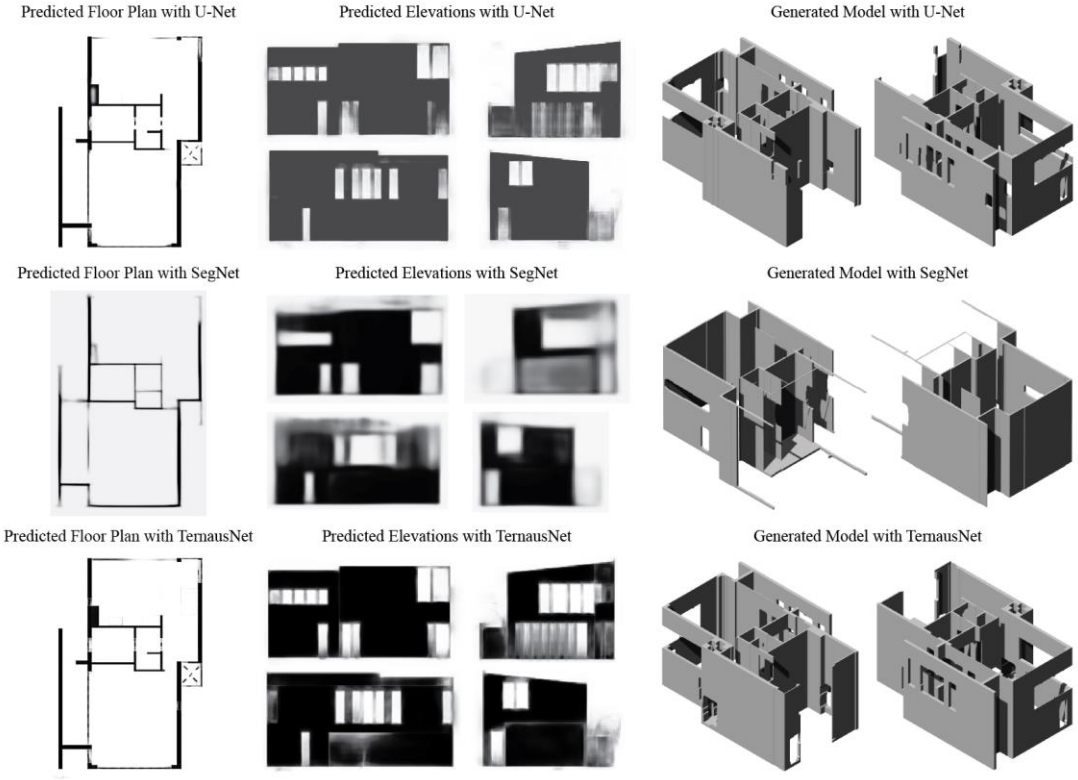


Figure 5.4. Predicted architectural drawings and isometric views of 3D model on Second Case (produced by the author)

5.3. Findings

The main output of the case studies obviously is that 3D architectural regeneration is attainable within a short period of time without any human intervention. Two case studies are achieved in approximately 20 seconds, and it is obvious that automated process is much shorter than a manual 3D reconstruction.

When considering the CNN performance, TernausNet perform better than U-Net and SegNet in terms of volumetric calculation and reconstruction speed, and this gives a

solid clue about the success of Transfer Learning on semantic segmentation applications. Also, SegNet gives blurrier predictions than U-Net and TernausNet which is not appropriate for such a reconstruction research.

Floor plan and elevation datasets only include one/two-storey height building drawings. Although data samples of non-orthogonal and multifaceted buildings are included in the datasets, it is troublesome to regenerate buildings that have level differences and different floor plans on each level. First case is a one-storey height building envelope in which architectural elements are extracted successfully with few missing opening information via semantic segmentation. On the other hand, second case is a multifaceted two-storey height building, and architectural information is not accomplished properly due to having different floor plan on each level and not being able to match openings with multifaceted floor plan. Consequently, first case is more successful than second case, and building types affect reconstruction process drastically.

Reconstructed 3D models show that detail level does not completely satisfy industry needs due to lack of roof, stairs, beams and any further details. Using multi-class segmentation instead of binary can detect different architectural elements, but there should be wider datasets to achieve it.

CHAPTER 6

CONCLUSION

Modelling has always been an important feature in design-related disciplines. The need for planning and analyzing in a digital environment to simulate real-life conditions has gained importance in from city and regional planning to restoration projects. 3D models have been widely required in expenditure planning, inspection, and search and rescue operations. 3D modelling has become essential to accelerate a design process, yet usage of outdated tools is still in the industry, which poses problems on integration and implementation of the ideas on a virtual environment. All of these areas require qualified personnel, precise data and many processing time to generate 3D models. Therefore, this research aims to represent a solid implementation by automatically transforming low level information into a higher level within less than a minute, which can be potentially used in above usage areas. While this transformation is applicable for all the current disciplines, only architectural usage is covered due to dataset limitations.

It is known that 2D technical drawings are still the backbone for a design process. Semantic information that they include is valuable for the end users, but their geometrical representations become low level information in digital medium due to the ability of processing the information. Therefore, they can be counted as valuable data sources for transformation purposes. In this research, this 2D-3D transformation is achieved with deep learning and geometry processing techniques by utilizing 2D architectural drawings. An implementation is rendered to enable end users to gain such models easily within a short period of time, and to provide a 3D model to the ones who need it to accelerate a design process as mentioned earlier. Most of the studies in

this area use conventional computer vision techniques and only recognize floor plan(s) to reconstruct such models. However, this study puts an effort to prepare a brand-new architectural dataset including a novel labelling system, to employ deep learning, to process floor plan(s) and elevation(s) at the same time for increasing the level of automation, and to present an open-source web application for any user who needs such reconstructions.

Structure of the dissertation is mainly based on reviewing literature, and performing an implementation accordingly. Related work in the literature is reviewed to reveal background of 3D reconstruction processes. The implementation workflow is explained deeply. Firstly, floor plan and elevation datasets which are prepared from scratch are introduced. Latter, different CNNs are trained with these datasets to semantically segment architectural elements and gain convenient weights. Training processes are evaluated and compared to each other under different conditions. Lastly, two case studies are conducted to illustrate the results of semantic segmentation in 3D environment. Floor plan and four elevation drawings of the cases are predicted based on these weights. Predicted results are converted to 3D models and compared to their 3D ground truths.

As a result, this study contributes to an automated 3D architectural model generation with preparing new datasets, and utilizing deep learning and geometry processing methods. It acquaints a web application to enable all users to gain these models automatically.

6.1. General Discussion

Machine learning and computer vision have started to diffuse in many disciplines gradually for manufacturing, modelling, simulation and so on. They also have a big part in this research in terms of automated architectural model reconstruction. All the

reconstruction process relies on semantic segmentation of low level data, and converting it to a high level information with geometry processing techniques.

The implementation starts with dataset preparation since there is no proper dataset in the literature for architectural reconstruction purposes. Though floor plan data samples exist in online platforms, there is no accurate elevation dataset. These datasets are trained with three different CNN architectures to perform semantic segmentation and to uncover the potentials of different architectures in an automated reconstruction. However, CNNs are complex structures to handle for customized problems. It is important to be aware of the dataset characteristics as well as reconfiguration of CNNs accordingly. It is almost impossible to say that there is one specific CNN to solve one specific problem. Instead, there are different CNN options to perform such tasks, and it is important to decide what is best in performance by trying different optimization steps and structures.

The evaluation metrics and learning curves show that U-Net, SegNet and TernausNet are accurate enough to segment 2D architectural drawings semantically. Yet, assessment on different conditions such as ‘augmented vs. not augmented’ and ‘original structure vs. reconfigured structure’ demonstrates that data augmentation is a vital step for such a research area. Furthermore, original configurations of the CNN models have better rates than reconfigured ones.

Another notice is that SegNet predictions are fuzzier than of U-Net and TernausNet. Considering the fact that U-Net and TernausNet are constructed for biomedical images of which features and dataset size are really alike architectural drawings, they are more likely to predict better. Moreover, their structure is based on concatenation of features from encoding part to decoding part, which is missing in SegNet. Also, evaluation on

3D environment infers that TernausNet is best in performance, which can be a resultant of being a transfer learning implementation.

Even though training time of CNNs can be long depending on the hardware qualifications, prediction and model generation last barely half of one minute that is much faster than manually constructing a 3D model. Volumetric evaluation shows that predictions are accurate enough according to the ground truth volume, however right now, the end 3D models have only walls as mass, and opening as voids. Therefore, automatically generated 3D models are only on conceptual level, and barely satisfy the industry's needs, which can be enhanced by overcoming dataset, CNN and computer vision problems.

6.2. Limitations

The biggest limitation of this research is to find an optimum way to standardize raw data and corresponding labels. Standardization process is a challenging issue since the image quality and semantic information of raster drawings differ significantly. As referred before, the raw data is taken from online sources and corresponding labels are produced by hand from scratch. Although floor plans are more reachable, elevation drawings are difficult to obtain. Consequently, these datasets are only composed of simple residence drawings with medium level details to achieve optimum datasets for training.

Even though floor plan data samples include a few non-orthogonal geometries, elevation dataset only include orthogonal geometries. Therefore, the application becomes only limited to orthogonal model generation. Yet, freeform generations can be accomplished with further data samples and training.

6.3. Future Work

Different needs in current disciplines regarding 3D models show that reconstruction processes are inevitable. Hence, this research endeavors to reveal the potential for use in different areas including architecture. However, datasets and CNN structures should be rearranged for other fields since each discipline has its own specifications with different semantic information. This implementation can be extended to various areas from manufacturing mechanical parts to augmented reality walkthroughs for real estate agencies. If more details could be integrated to the architectural reconstructions, these models would be used in construction technologies.

First thing to do should be enlarging the variety of dataset to achieve more accurate results and to perform multi-storey architectural reconstructions. The data samples should encapsulate not only residential architecture but also public, healthcare, education and so on. Gathering not only floor plan(s) and elevation(s) but also section(s) and/or detail drawing(s) for different architectural types would increase the desired level of detail in a 3D model.

Although 2D evaluation metrics used in this research are very common in semantic segmentation applications, 3D evaluation metric that is introduced in the previous chapter can be reconsidered. Its working principle is based on simple Boolean operations, yet vertex, corner or even edge information for evaluation can be implemented from scratch or integrated to the introduced one. Therefore, new metrics to compare the results and accuracy would be beneficial to improve reconstruction procedures.

At last but not least, new parameters can be integrated to the workflow so that further usage areas such as simulations and as-built modelling can be accomplished. For

example, recognition of other elements than walls and openings will increase the level of detail in a model. Also, generating non-orthogonal architectural elements can be executed to cover different types of elements. As another instance, integration of materials or structural information will provide a better environment for simulation purposes.

REFERENCES

Agatonovic-Kustrin, S., & Beresford, R. (2000). Basic concepts of artificial neural network (ANN) modeling and its application in pharmaceutical research. *Journal of pharmaceutical and biomedical analysis*, *22*(5), 717-727.

Agatonovic-Kustrin, S., & Beresford, R. (2000). Basic concepts of artificial neural network (ANN) modeling and its application in pharmaceutical research. *Journal of pharmaceutical and biomedical analysis*, *22*(5), 717-727.

Alhaija, H. A., Mustikovela, S. K., Mescheder, L., Geiger, A., & Rother, C. (2018). Augmented reality meets computer vision: Efficient data generation for urban driving scenes. *International Journal of Computer Vision*, *126*(9), 961-972.

ArchDaily. (n.d.). Houses architecture and design. Retrieved from https://www.archdaily.com/search/projects/categories/houses

Armeni, I., Sax, S., Zamir, A. R., & Savarese, S. (2017). Joint 2d-3d-semantic data for indoor scene understanding. *arXiv preprint arXiv:1702.01105*.

Badrinarayanan, V., Kendall, A., & Cipolla, R. (2017). Segnet: A deep convolutional encoder-decoder architecture for image segmentation. *IEEE transactions on pattern analysis and machine intelligence*, *39*(12), 2481-2495.

Barillot, C., Gibaud, B., Scarabin, J. M., & Coatrieux, J. L. (1985). 3 D reconstruction of cerebral blood vessels. *IEEE computer graphics and applications*, *5*(12), 13-19.

Bazilevs, Y., Hsu, M. C., Akkerman, I., Wright, S., Takizawa, K., Henicke, B., ... & Tezduyar, T. E. (2011). 3D simulation of wind turbine rotors at full scale. Part I: Geometry modeling and aerodynamics. *International journal for numerical methods in fluids*, *65*(1-3), 207-235.

Borson, B. (2019, January 09). Your sketches speak for themselves. Retrieved from https://www.lifeofanarchitect.com/what-your-sketches-say-about-you-and-how-you-think/

Bourke, P. (2018). Automatic 3D reconstruction: An exploration of the state of the art. *GSTF Journal on Computing (JoC), 2*(3).

Brenner, C. (2005). Building reconstruction from images and laser scanning. *International Journal of Applied Earth Observation and Geoinformation,* 6(3-4), 187-198.

Bresenham, J. E. (1965). Algorithm for computer control of a digital plotter. *IBM Systems journal*, *4*(1), 25-30.

Brostow, G. J., Fauqueur, J., & Cipolla, R. (2009). Semantic object classes in video: A high-definition ground truth database. *Pattern Recognition Letters*, *30*(2), 88-97.

Brox, T., Bourdev, L., Maji, S., & Malik, J. (2011, June). Object segmentation by alignment of poselet activations to image contours. In *CVPR 2011* (pp. 2225-2232). IEEE.

Bulat, A., & Tzimiropoulos, G. (2016, October). Human pose estimation via convolutional part heatmap regression. In *European Conference on Computer Vision* (pp. 717-732). Springer, Cham.

Burns, J. B., Hanson, A. R., & Riseman, E. M. (1986). Extracting straight lines. *IEEE transactions on pattern analysis and machine intelligence*, (4), 425-455.

Business Advantage. (n.d.). Business Advantage Annual CAD Trends Survey 2018-19 Webinar. Retrieved from https://www.business-advantage.com/landing_page_CAD_Trends_2017_Webinar.php

Camozzato, D., Dihl, L., Silveira, I., Marson, F., & Musse, S. R. (2015). Procedural floor plan generation from building sketches. *The Visual Computer*, *31*(6-8), 753-763.

Cardona, A., Saalfeld, S., Preibisch, S., Schmid, B., Cheng, A., Pulokas, J., ... & Hartenstein, V. (2010). An integrated micro-and macroarchitectural analysis of the Drosophila brain by computer-assisted serial section electron microscopy. *PLoS biology*, *8*(10), e1000502.

Chang, A., Dai, A., Funkhouser, T., Halber, M., Niessner, M., Savva, M., ... & Zhang, Y. (2017). Matterport3d: Learning from rgb-d data in indoor environments. *arXiv preprint arXiv:1709.06158*.

Chen, C. W., Luo, J., & Parker, K. J. (1998). Image segmentation via adaptive K-mean clustering and knowledge-based morphological operations with biomedical applications. *IEEE transactions on image processing*, *7*(12), 1673-1683.

Chen, G., Weng, Q., Hay, G. J., & He, Y. (2018). Geographic Object-based Image Analysis (GEOBIA): Emerging trends and future opportunities. *GIScience & remote sensing*, *55*(2), 159-182.

Chen, K., Lai, Y. K., & Hu, S. M. (2015). 3D indoor scene modeling from RGB-D data: a survey. *Computational Visual Media*, *1*(4), 267-278.

Chen, L. C., Papandreou, G., Kokkinos, I., Murphy, K., & Yuille, A. L. (2017). Deeplab: Semantic image segmentation with deep convolutional nets, atrous convolution, and fully connected crfs. *IEEE transactions on pattern analysis and machine intelligence*, *40*(4), 834-848.

Chen, S., Li, M., Ren, K., & Qiao, C. (2015, June). Crowd map: Accurate reconstruction of indoor floor plans from crowdsourced sensor-rich videos. In *2015 IEEE 35th International conference on distributed computing systems* (pp. 1-10). IEEE.

Chiabrando, F., Sammartano, G., & Spanò, A. (2016). Historical buildings models and their handling via 3D survey: from points clouds to user-oriented HBIM. *International Archives of the Photogrammetry, Remote Sensing & Spatial Information Sciences*, *41*.

Cordts, M., Omran, M., Ramos, S., Rehfeld, T., Enzweiler, M., Benenson, R., ... & Schiele, B. (2016). The cityscapes dataset for semantic urban scene understanding. In *Proceedings of the IEEE conference on computer vision and pattern recognition* (pp. 3213-3223).

de las Heras, L. P., Fernández, D., Valveny, E., Lladós, J., & Sánchez, G. (2013, August). Unsupervised wall detector in architectural floor plans. In *Document Analysis and Recognition (ICDAR), 2013 12th International Conference on* (pp. 1245-1249). IEEE.

de las Heras, L. P., Mas, J., & Valveny, E. (2011, September). Wall patch-based segmentation in architectural floorplans. In *2011 International Conference on Document Analysis and Recognition* (pp. 1270-1274). IEEE.

de las Heras, L. P., Terrades, O. R., Robles, S., & Sánchez, G. (2015). CVC-FP and SGT: a new database for structural floor plan analysis and its groundtruthing tool. *International Journal on Document Analysis and Recognition (IJDAR)*, *18*(1), 15-30.

Deng, J., Dong, W., Socher, R., Li, L. J., Li, K., & Fei-Fei, L. (2009). Imagenet: A large-scale hierarchical image database.

De Reu, J., De Smedt, P., Herremans, D., Van Meirvenne, M., Laloo, P., & De Clercq, W. (2014). On introducing an image-based 3D reconstruction method in archaeological excavation practice. *Journal of Archaeological Science*, 41, 251-262.

Ding, C., & Liu, L. (2016). A survey of sketch based modeling systems. *Frontiers of Computer Science*, *10*(6), 985-999.

Dodge, S., Xu, J., & Stenger, B. (2017, May). Parsing floor plan images. In *Machine Vision Applications (MVA), 2017 Fifteenth IAPR International Conference on* (pp. 358-361). IEEE.

Domínguez, B., García, Á. L., & Feito, F. R. (2012). Semiautomatic detection of floor topology from CAD architectural drawings. *Computer-Aided Design*, *44*(5), 367-378.

Eggli, L., Hsu, C. Y., Bruederlin, B. D., & Elber, G. (1997). Inferring 3D models from freehand sketches and constraints. *Computer-Aided Design*, 29(2), 101-112.

El-Hakim, S. F., Beraldin, J. A., Picard, M., & Godin, G. (2004). Detailed 3D reconstruction of large-scale heritage sites with integrated techniques. *IEEE Computer Graphics and Applications*, 24(3), 21-29.

Elkan, C. (2003). Using the triangle inequality to accelerate k-means. In *Proceedings of the 20th International Conference on Machine Learning (ICML-03)* (pp. 147-153).

Everingham, M., Eslami, S. A., Van Gool, L., Williams, C. K., Winn, J., & Zisserman, A. (2015). The pascal visual object classes challenge: A retrospective. *International journal of computer vision*, *111*(1), 98-136.

Farabet, C., Couprie, C., Najman, L., & LeCun, Y. (2013). Learning hierarchical features for scene labeling. *IEEE transactions on pattern analysis and machine intelligence*, *35*(8), 1915-1929.

Fisher, R., Perkins, S., Walker, A., & Wolfart, E. (2000). HYPERMEDIA IMAGE PROCESSING REFERENCE - Morphology. Retrieved from http://homepages.inf.ed.ac.uk/rbf/HIPR2/morops.htm

Fletcher, L. A., & Kasturi, R. (1988). A robust algorithm for text string separation from mixed text/graphics images. *IEEE Transactions on Pattern Analysis & Machine Intelligence*, (6), 910-918.

Gao, R., Zhou, B., Ye, F., & Wang, Y. (2017, May). Knitter: Fast, resilient single-user indoor floor plan construction. In *IEEE INFOCOM 2017-IEEE Conference on Computer Communications* (pp. 1-9). IEEE.

Garcia-Garcia, A., Orts-Escolano, S., Oprea, S., Villena-Martinez, V., & Garcia-Rodriguez, J. (2017). A review on deep learning techniques applied to semantic segmentation. *arXiv preprint arXiv:1704.06857*.

Garcia-Garcia, A., Orts-Escolano, S., Oprea, S., Villena-Martinez, V., Martinez-Gonzalez, P., & Garcia-Rodriguez, J. (2018). A survey on deep learning techniques for image and video semantic segmentation. *Applied Soft Computing*, *70*, 41-65.

Getmapping. (n.d.). Aerial Data - GB Imagery. Retrieved from http://www.getmapping.com/products/aerial-data-high-resolution-imagery/aerial-data-gb-imagery

Gimenez, L., Hippolyte, J. L., Robert, S., Suard, F., & Zreik, K. (2015). Review: reconstruction of 3D building information models from 2D scanned plans. *Journal of Building Engineering*, 2, 24-35.

Gimenez, L., Robert, S., Suard, F., & Zreik, K. (2016). Automatic reconstruction of 3D building models from scanned 2D floor plans. *Automation in Construction*, 63, 48-56.

Girshick, R., Donahue, J., Darrell, T., & 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).

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

Goodfellow, I., Bengio, Y., & Courville, A. (2016). *Deep learning*. MIT press.

Goyal, S., Bhavsar, S., Patel, S., Chattopadhyay, C., & Bhatnagar, G. (2018). SUGAMAN: Describing Floor Plans for Visually Impaired by Annotation Learning and Proximity based Grammar. *arXiv preprint arXiv:1812.00874*.

Goyal, S., Chattopadhyay, C., & Bhatnagar, G. (2018, October). ASYSST: A Framework for Synopsis Synthesis Empowering Visually Impaired. In *Proceedings of the 2018 Workshop on Multimedia for Accessible Human Computer Interface* (pp. 17-24). ACM.

Gupta, R. (2018, May 09). MNIST vs MNIST - how I was able to speed up my Deep Learning. Retrieved from https://towardsdatascience.com/mnist-vs-mnist-how-i-was-able-to-speed-up-my-deep-learning-11c0787e6935

Haala, N. (1995). 3D building reconstruction using linear edge segments. *In Photogrammetric Week* (Vol. 95, pp. 19-28). Wichmann, Karlsruhe.

Han, J., & Moraga, C. (1995, June). The influence of the sigmoid function parameters on the speed of backpropagation learning. In *International Workshop on Artificial Neural Networks* (pp. 195-201). Springer, Berlin, Heidelberg.

Hanley Wood LLC. (n.d.). Architectural House Plan Designs and Home Floor Plans. Retrieved from https://www.homeplans.com/

Hanley Wood LLC (n.d.). Official House Plan & Blueprint Site of Builder Magazine. Retrieved from https://www.builderhouseplans.com/

Hariharan, B., Arbeláez, P., Girshick, R., & Malik, J. (2014, September). Simultaneous detection and segmentation. In *European Conference on Computer Vision* (pp. 297-312). Springer, Cham.

Harris, L. V. A., & Meyers, F. (2009). Engineering design graphics: Into the 21st century. *Engineering Design Graphics Journal*, 71(3).

Hartmann, T., Gao, J., & Fischer, M. (2008). Areas of application for 3D and 4D models on construction projects. *Journal of Construction Engineering and management*, 134(10), 776-785.

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

He, K., Zhang, X., Ren, S., & Sun, J. (2016). Deep residual learning for image recognition. In *Proceedings of the IEEE conference on computer vision and pattern recognition* (pp. 770-778).

Henderson, R. (2006). Design, simulation, and testing of a novel hydraulic power take-off system for the Pelamis wave energy converter. *Renewable energy*, *31*(2), 271-283.

Henry, P., Krainin, M., Herbst, E., Ren, X., & Fox, D. (2012). RGB-D mapping: Using Kinect-style depth cameras for dense 3D modeling of indoor environments. *The International Journal of Robotics Research*, 31(5), 647-663.

Hoover, A., & Goldbaum, M. (2003). Locating the optic nerve in a retinal image using the fuzzy convergence of the blood vessels. *IEEE transactions on medical imaging*, *22*(8), 951-958.

Horna, S., Meneveaux, D., Damiand, G., & Bertrand, Y. (2009). Consistency constraints and 3D building reconstruction. *Computer-Aided Design*, *41*(1), 13-27.

Horsey, J. (2016, March 24). Unity VR Editor Application Demonstrated At GDC 2016 (video). Retrieved from https://www.geeky-gadgets.com/unity-vr-editor-application-demonstrated-at-gdc-2016-24-03-2016/

Houseplans LLC (n.d.). Houseplans. Retrieved from https://www.houseplans.com/

Huang, H. C., Lo, S. M., Zhi, G. S., & Yuen, R. K. (2008). Graph theory-based approach for automatic recognition of CAD data. *Engineering Applications of Artificial Intelligence*, 21(7), 1073-1079.

Huang, X., & Kwoh, L. K. (2007, July). 3D building reconstruction and visualization for single high resolution satellite image. In *Geoscience and Remote Sensing Symposium, 2007. IGARSS 2007. IEEE International* (pp. 5009-5012). IEEE.

HySpex. (n.d.). What is Hyperspectral Imaging? Retrieved from https://www.hyspex.no/hyperspectral_imaging/

Iglovikov, V., & Shvets, A. (2018). Ternausnet: U-net with vgg11 encoder pre-trained on imagenet for image segmentation. *arXiv preprint arXiv:1801.05746*.

Jiang, M., Wu, Y., & Lu, C. (2018). Pointsift: A sift-like network module for 3d point cloud semantic segmentation. arXiv preprint arXiv:1807.00652.

Joglekar, S. (2018, January 24). Overfitting and Human Behavior. Retrieved from https://medium.com/@srjoglekar246/overfitting-and-human-behavior-5186df1e7d19

Jolliffe, I. (2011). Principal component analysis. In *International encyclopedia of statistical science* (pp. 1094-1096). Springer, Berlin, Heidelberg.

Juchmes, R., Leclercq, P., & Azar, S. (2005). A freehand-sketch environment for architectural design supported by a multi-agent system. *Computers & Graphics*, 29(6), 905-915.

Kim, K., Oh, C., & Sohn, K. (2016). Non-parametric human segmentation using support vector machine. *IEEE Transactions on Consumer Electronics*, *62*(2), 150-158.

Koutamanis, A., & Mitossi, V. (1992, September). Automated recognition of architectural drawings. *In Pattern Recognition, 1992. Vol. I. Conference A: Computer Vision and Applications, Proceedings., 11th IAPR International Conference* on (pp. 660-663). IEEE.

Lateef, F., & Ruichek, Y. (2019). Survey on Semantic Segmentation using Deep Learning Techniques. *Neurocomputing*.

LeCun, Y., Bengio, Y., & Hinton, G. (2015). Deep learning. *nature*, *521*(7553), 436.

Le forum pour faire construire sa maison. (n.d.). Retrieved from https://www.forumconstruire.com/

Lewis, R., & Séquin, C. (1998). Generation of 3D building models from 2D architectural plans. *Computer-Aided Design*, 30(10), 765-779.

Lin, T. Y., Maire, M., Belongie, S., Hays, J., Perona, P., Ramanan, D., ... & Zitnick, C. L. (2014, September). Microsoft coco: Common objects in context. In *European conference on computer vision* (pp. 740-755). Springer, Cham.

Liu, C., Wu, J., Kohli, P., & Furukawa, Y. (2017). Raster-to-vector: Revisiting floorplan transformation. In *Proceedings of the IEEE International Conference on Computer Vision* (pp. 2195-2203).

Liu, L., Ouyang, W., Wang, X., Fieguth, P., Chen, J., Liu, X., & Pietikäinen, M. (2018). Deep learning for generic object detection: A survey. *arXiv preprint arXiv:1809.02165*.

Liu, W., Wang, Z., Liu, X., Zeng, N., Liu, Y., & Alsaadi, F. E. (2017). A survey of deep neural network architectures and their applications. *Neurocomputing*, *234*, 11-26.

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

Lu, T., Tai, C. L., Su, F., & Cai, S. (2005). A new recognition model for electronic architectural drawings. *Computer-Aided Design*, 37(10), 1053-1069.

Macé, S., Locteau, H., Valveny, E., & Tabbone, S. (2010, June). A system to detect rooms in architectural floor plan images. In *Proceedings of the 9th IAPR International Workshop on Document Analysis Systems* (pp. 167-174). ACM.

Mahapatra, S. (2018, March 21). Why Deep Learning over Traditional Machine Learning? Retrieved from https://towardsdatascience.com/why-deep-learning-is-needed-over-traditional-machine-learning-1b6a99177063

Marcus, D. S., Wang, T. H., Parker, J., Csernansky, J. G., Morris, J. C., & Buckner, R. L. (2007). Open Access Series of Imaging Studies (OASIS): cross-sectional MRI data in young, middle aged, nondemented, and demented older adults. *Journal of cognitive neuroscience*, *19*(9), 1498-1507.

Milioto, A., Lottes, P., & Stachniss, C. (2018, May). Real-time semantic segmentation of crop and weed for precision agriculture robots leveraging background knowledge in cnns. In *2018 IEEE International Conference on Robotics and Automation (ICRA)* (pp. 2229-2235). IEEE.

Mukherjee, D., Wu, Q. J., & Wang, G. (2015). A comparative experimental study of image feature detectors and descriptors. *Machine Vision and Applications*, *26*(4), 443-466.

Müller, P., Zeng, G., Wonka, P., & Van Gool, L. (2007, August). Image-based procedural modeling of facades. In *ACM Transactions on Graphics (TOG)* (Vol. 26, No. 3, p. 85). ACM.

Musialski, P., Wonka, P., Aliaga, D. G., Wimmer, M., Van Gool, L., & Purgathofer, W. (2013, September). A survey of urban reconstruction. *In Computer graphics forum* (Vol. 32, No. 6, pp. 146-177).

Nair, V., & Hinton, G. E. (2010). Rectified linear units improve restricted boltzmann machines. In *Proceedings of the 27th international conference on machine learning (ICML-10)* (pp. 807-814).

Ochmann, S., Vock, R., Wessel, R., Tamke, M., & Klein, R. (2014, January). Automatic generation of structural building descriptions from 3d point cloud scans. I*n Computer Graphics Theory and Applications (GRAPP), 2014 International Conference on* (pp. 1-8). IEEE.

Olsen, L., Samavati, F. F., Sousa, M. C., & Jorge, J. A. (2009). Sketch-based modeling: A survey. *Computers & Graphics*, 33(1), 85-103.

Or, S. H., Wong, K. H., Yu, Y. K., Chang, M. M. Y., & Kong, H. (2005). Highly automatic approach to architectural floorplan image understanding & model generation. *Pattern Recognition*, 25-32.

Özdemir, E., & Remondino, F. (2018). SEGMENTATION OF 3D PHOTOGRAMMETRIC POINT CLOUD FOR 3D BUILDING MODELING. *International Archives of the Photogrammetry, Remote Sensing & Spatial Information Sciences*.

Pandey, J., & Sharma, O. (2016). Fast and Robust Construction of 3D Architectural Models from 2D Plans. *International Conference in Central Europe on Computer Graphics, Visualization and Computer Vision (WSCG)*

Pieraccini, M., Guidi, G., & Atzeni, C. (2001). 3D digitizing of cultural heritage. *Journal of Cultural Heritage*, 2(1), 63-70.

Previtali, M., Barazzetti, L., Brumana, R., & Scaioni, M. (2014). Towards automatic indoor reconstruction of cluttered building rooms from point clouds. *ISPRS Annals of Photogrammetry, Remote Sensing & Spatial Information Sciences*, 2(5).

Pu, S., & Vosselman, G. (2009). Knowledge based reconstruction of building models from terrestrial laser scanning data. *ISPRS Journal of Photogrammetry and Remote Sensing*, 64(6), 575-584.

Rakuten, Inc., NII, and Alagin (2017). Floor Plan from Rakuten Real Estate (powered by LIFULL Co., Ltd.) and Pixel-wise Wall Label [Data File]. Available from Rakuten Institute of Technology Website: https://rit.rakuten.co.jp/data_release/

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

Rossa, P., Camozzato, D., Marson, F., & Hocevar, R. (2016). 3D Model Generation from Freehand Drawings. In *Proceedings of XV Brazilian Symposium on Computer Games and Digital Entertainment* (pp. 226-229).

Roth, H. R., Oda, H., Hayashi, Y., Oda, M., Shimizu, N., Fujiwara, M., ... & Mori, K. (2017). Hierarchical 3D fully convolutional networks for multi-organ segmentation. *arXiv preprint arXiv:1704.06382*.

Sarkar, D. (2018, November 14). A Comprehensive Hands-on Guide to Transfer Learning with Real-World Applications in Deep Learning. Retrieved from https://towardsdatascience.com/a-comprehensive-hands-on-guide-to-transfer-learning-with-real-world-applications-in-deep-learning-212bf3b2f27a

Schneider, N., & Gavrila, D. M. (2013, September). Pedestrian path prediction with recursive bayesian filters: A comparative study. In *German Conference on Pattern Recognition* (pp. 174-183). Springer, Berlin, Heidelberg.

Sharma, D., Gupta, N., Chattopadhyay, C., & Mehta, S. (2017, November). DANIEL: A deep architecture for automatic analysis and retrieval of building floor plans. In *Document Analysis and Recognition (ICDAR), 2017 14th IAPR International Conference on* (Vol. 1, pp. 420-425). IEEE.

Shelhamer, E., Rakelly, K., Hoffman, J., & Darrell, T. (2016, October). Clockwork convnets for video semantic segmentation. In *European Conference on Computer Vision* (pp. 852-868). Springer, Cham.

Shio, A., & Aoki, Y. (2000). Sketch Plan: A prototype system for interpreting hand-sketched floor plans. *Systems and Computers in Japan,* 31(6), 10-18.

Silberman, N., Hoiem, D., Kohli, P., & Fergus, R. (2012, October). Indoor segmentation and support inference from rgbd images. In *European Conference on Computer Vision* (pp. 746-760). Springer, Berlin, Heidelberg.

Simonyan, K., & Zisserman, A. (2014). Very deep convolutional networks for large-scale image recognition. *arXiv preprint arXiv:1409.1556*.

Srivastava, N., Hinton, G., Krizhevsky, A., Sutskever, I., & Salakhutdinov, R. (2014). Dropout: a simple way to prevent neural networks from overfitting. *The Journal of Machine Learning Research*, *15*(1), 1929-1958.

Stephens, M. (2011, February 17). The difference between planning and construction drawings (Part 2). Retrieved from http://www.markstephensarchitects.com/the-difference-between-planning-and-construction-drawings-part-2/

Sun, S., & Salvaggio, C. (2013). Aerial 3D building detection and modeling from airborne LiDAR point clouds. *IEEE Journal of Selected Topics in Applied Earth Observations and Remote Sensing*, 6(3), 1440-1449.

Sutherland, I. E. (1964, January). Sketch pad a man-machine graphical communication system. *In Proceedings of the SHARE design automation workshop* (pp. 6-329). ACM.

Suveg, I., & Vosselman, G. (2004). Reconstruction of 3D building models from aerial images and maps. *ISPRS Journal of Photogrammetry and remote sensing*, 58(3-4), 202-224.

Suzuki, S & Abe, K. (1985). Topological structural analysis of digitized binary images by border following. *Computer vision, graphics, and image processing*, 30(1), 32-46.

Tang, P., Huber, D., Akinci, B., Lipman, R., & Lytle, A. (2010). Automatic reconstruction of as-built building information models from laser-scanned point clouds: A review of related techniques. *Automation in construction*, 19(7), 829-843.

The Garlinghouse Company. (n.d.). Search House Plans. Retrieved from https://www.familyhomeplans.com/

Thoma, M. (2016). A survey of semantic segmentation. *arXiv preprint arXiv:1602.06541*.

Thomson, C., & Boehm, J. (2015). Automatic geometry generation from point clouds for BIM. *Remote Sensing*, 7(9), 11753-11775.

Thøgersen, M., Escalera, S., Gonzàlez, J., & Moeslund, T. B. (2016). Segmentation of RGB-D indoor scenes by stacking random forests and conditional random fields. *Pattern Recognition Letters*, *80*, 208-215.

Tom, J., Medina, R., Garreau, M., Jugo, D., & Carrasco, H. (1970). Left ventricle 3D reconstruction using gibbs random fields and simulated annealing. *WIT Transactions on Biomedicine and Health*, *3*.

Tombre, K., Tabbone, S., Pélissier, L., Lamiroy, B., & Dosch, P. (2002, August). Text/graphics separation revisited. In *International Workshop on Document Analysis Systems* (pp. 200-211). Springer, Berlin, Heidelberg.

Tornincasa, S., & Di Monaco, F. (2010, September). The future and the evolution of CAD. *In Proceedings of the 14th international research/expert conference: trends in the development of machinery and associated technology* (pp. 11-18).

Treml, M., Arjona-Medina, J., Unterthiner, T., Durgesh, R., Friedmann, F., Schuberth, P., ... & Nessler, B. (2016, December). Speeding up semantic segmentation for autonomous driving. In *MLITS, NIPS Workshop*.

ul Hassan, M. (2018, November 21). VGG16 - Convolutional Network for Classification and Detection. Retrieved from https://neurohive.io/en/popular-networks/vgg16/

Vision One Homes (n.d.). Vision One Homes. Retrieved from https://www.visiononehomes.com.au/

Wang, K., Savva, M., Chang, A. X., & Ritchie, D. (2018). Deep convolutional priors for indoor scene synthesis. *ACM Transactions on Graphics (TOG)*, *37*(4), 70.

Wang, R. (2013). 3D building modeling using images and LiDAR: A review. *International Journal of Image and Data Fusion*, 4(4), 273-292.

Ware, R. W., & Lopresti, V. (1975). Three-dimensional reconstruction from serial sections. In *International review of cytology* (Vol. 40, pp. 325-440). Academic Press.

Wefelscheid, C., Hänsch, R., & Hellwich, O. (2011). Three-dimensional building reconstruction using images obtained by unmanned aerial vehicles. *International Archives of the Photogrammetry, Remote Sensing and Spatial Information Sciences*, 38(1).

Willneff, J., Poon, J., & Fraser, C. (2005). Single-image high-resolution satellite data for 3D information extraction. *International Archives of Photogrammetry, Remote Sensing and Spatial Information Sciences*, 36(1/W3), 1-6.

Xiao, J., Fang, T., Tan, P., Zhao, P., Ofek, E., & Quan, L. (2008, December). Image-based façade modeling. In *ACM transactions on graphics (TOG)* (Vol. 27, No. 5, p. 161). ACM.

Xiong, X., Adan, A., Akinci, B., & Huber, D. (2013). Automatic creation of semantically rich 3D building models from laser scanner data. *Automation in Construction*, 31, 325-337.

Xu, H., Hancock, E. R., & Zhou, W. (2019). The low-rank decomposition of correlation-enhanced superpixels for video segmentation. *Soft Computing*, 1-11.

Yang, J., Jang, H., Kim, J., & Kim, J. (2018, October). Semantic Segmentation in Architectural Floor Plans for Detecting Walls and Doors. In *2018 11th International Congress on Image and Signal Processing, BioMedical Engineering and Informatics (CISP-BMEI)* (pp. 1-9). IEEE.

Yu, F., & Koltun, V. (2015). Multi-scale context aggregation by dilated convolutions. *arXiv preprint arXiv:1511.07122*.

Yu, H., Yang, Z., Tan, L., Wang, Y., Sun, W., Sun, M., & Tang, Y. (2018). Methods and datasets on semantic segmentation: A review. *Neurocomputing*, *304*, 82-103.

Yue, H., Chen, W., Wu, X., & Liu, J. (2014). Fast 3D modeling in complex environments using a single Kinect sensor. *Optics and Lasers in Engineering*, 53, 104-111.

Zaitoun, N. M., & Aqel, M. J. (2015). Survey on image segmentation techniques. *Procedia Computer Science*, *65*, 797-806.

Zeleznik, R. C., Herndon, K. P., & Hughes, J. F. (2007, August). SKETCH: An interface for sketching 3D scenes. *In ACM SIGGRAPH 2007 courses* (p. 19). ACM.

Zheng, C., Zhang, Y., & Wang, L. (2017). Semantic segmentation of remote sensing imagery using an object-based Markov random field model with auxiliary label fields. *IEEE Transactions on geoscience and remote sensing*, *55*(5), 3015-3028.

Zhi, G. S., Lo, S. M., & Fang, Z. (2003). A graph-based algorithm for extracting units and loops from architectural floor plans for a building evacuation model. *Computer-Aided Design*, *35*(1), 1-14.

Zhou, B., Zhao, H., Puig, X., Fidler, S., Barriuso, A., & Torralba, A. (2017). Scene parsing through ade20k dataset. In *Proceedings of the IEEE conference on computer vision and pattern recognition* (pp. 633-641).

Zhu, H., Lu, J., Cai, J., Zheng, J., Lu, S., & Thalmann, N. M. (2016). Multiple human identification and cosegmentation: A human-oriented crf approach with poselets. *IEEE Transactions on Multimedia*, *18*(8), 1516-1530.

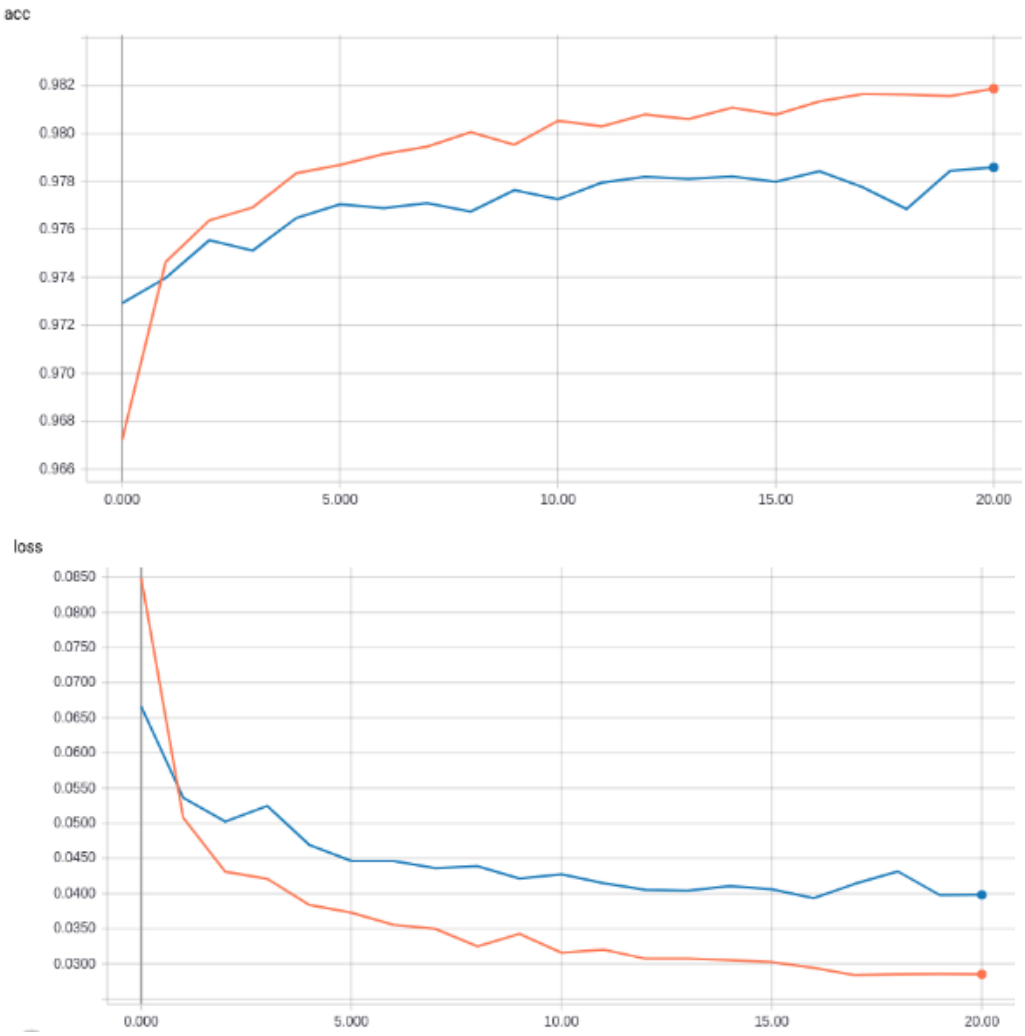
Zhu, H., Meng, F., Cai, J., & Lu, S. (2016). Beyond pixels: A comprehensive survey from bottom-up to semantic image segmentation and cosegmentation. *Journal of Visual Communication and Image Representation*, *34*, 12-27.

Zhu, J., Zhang, H., & Wen, Y. (2014). A new reconstruction method for 3D buildings from 2D vector floor plan. *Computer-Aided Design and Applications*, *11*(6), 704-714.

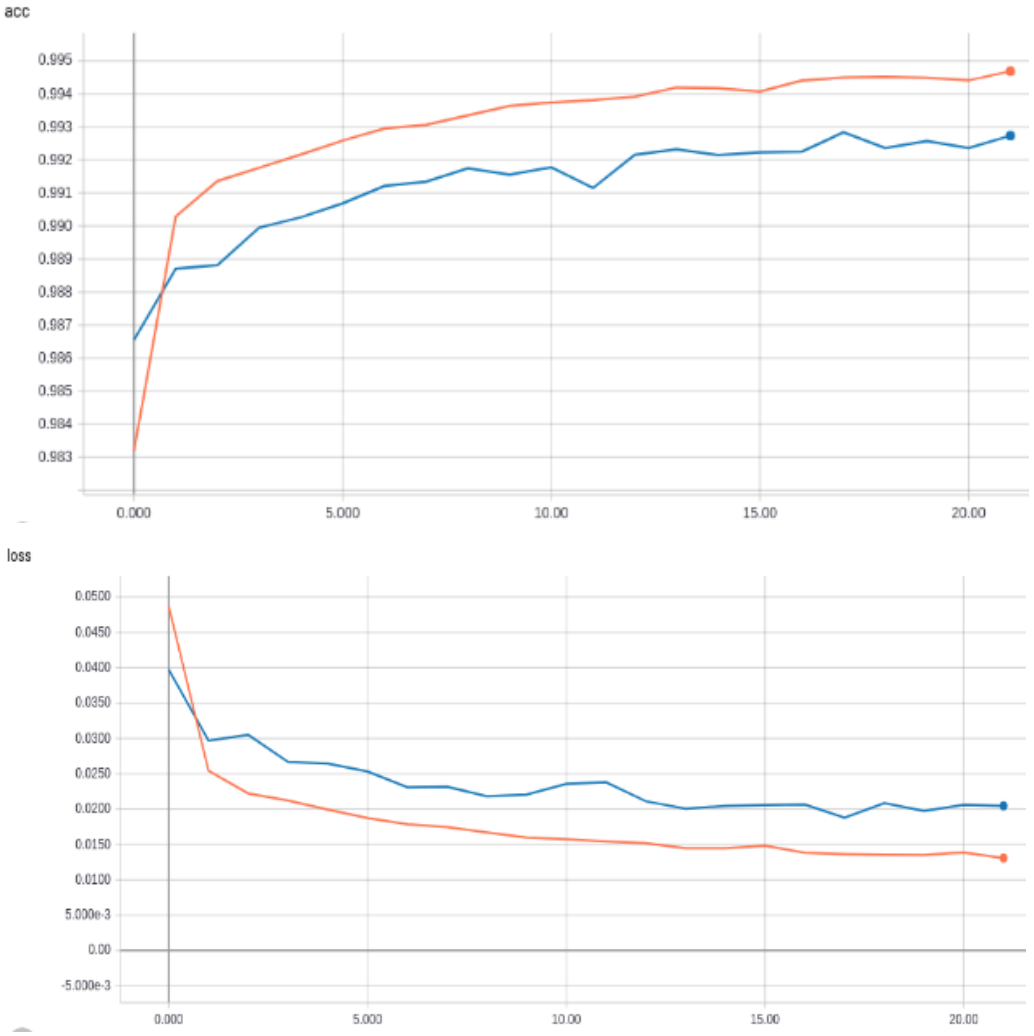
APPENDICES

A. CNN Results with Original Architecture

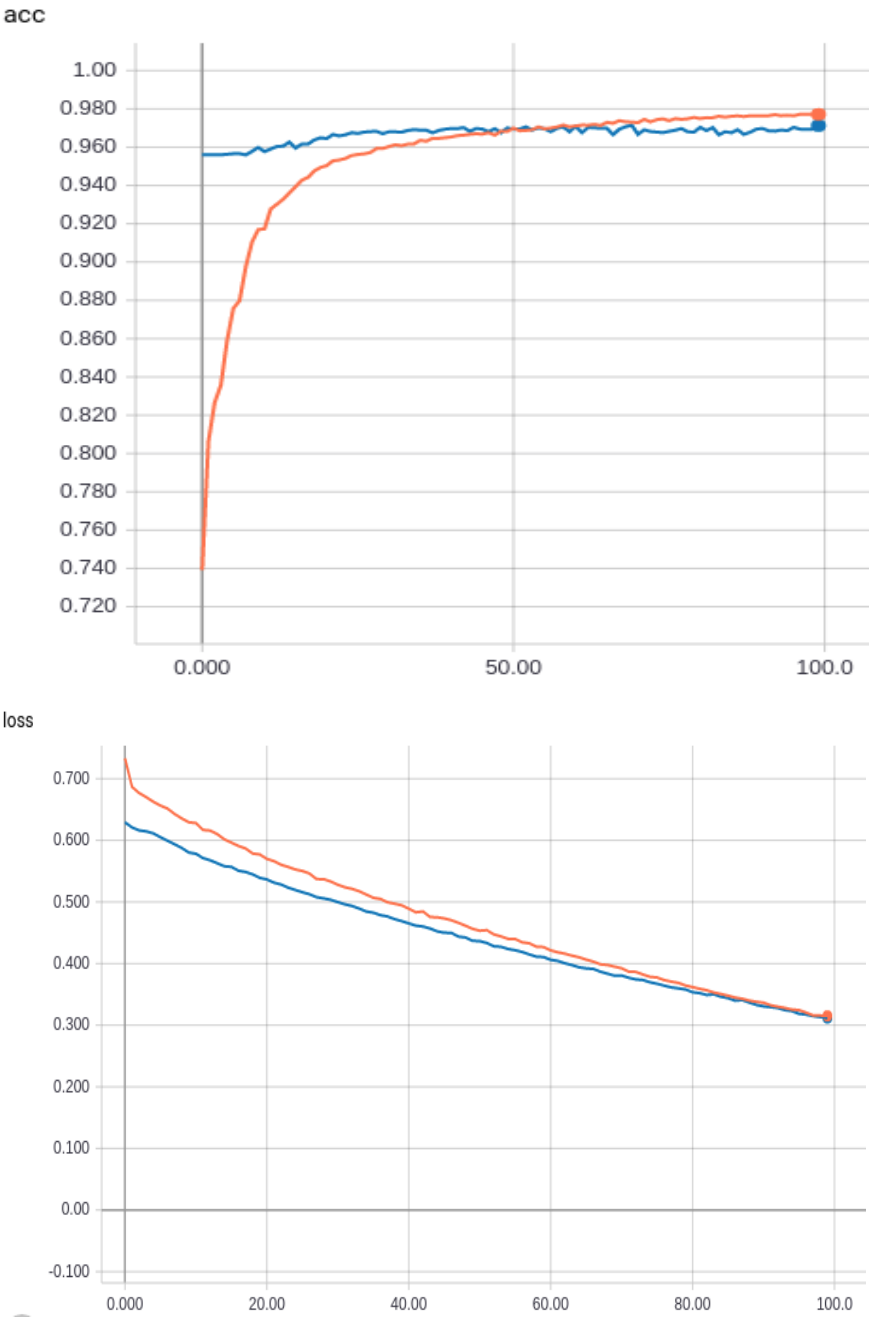
The following graphs are CNNs learning curves on augmented and non-augmented floor plan dataset with original architectures. While *orange curve* demonstrates the rate on training set, *blue curve* indicates validation set.



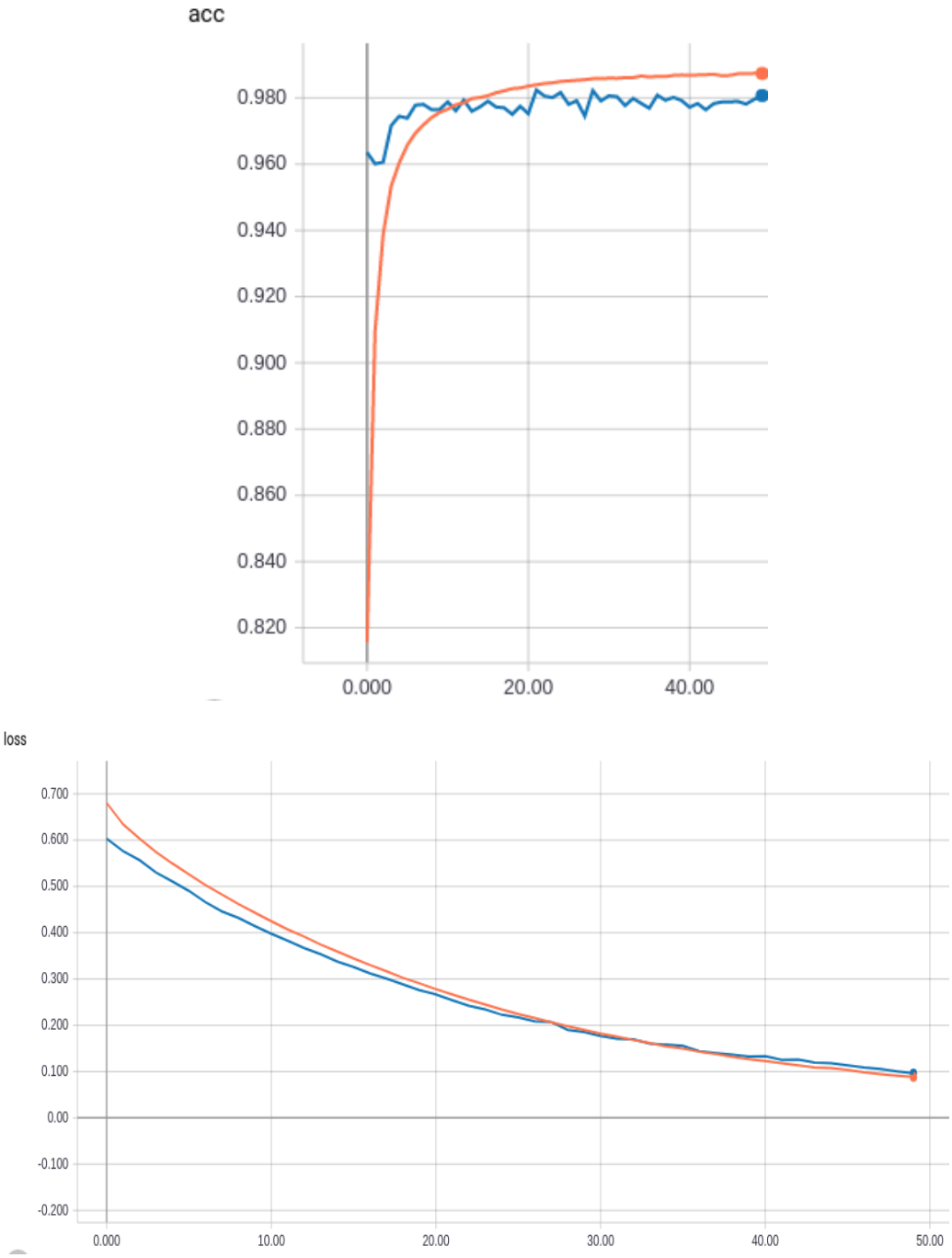
a) U-Net accuracy and loss graphics on *not augmented floor plan dataset*



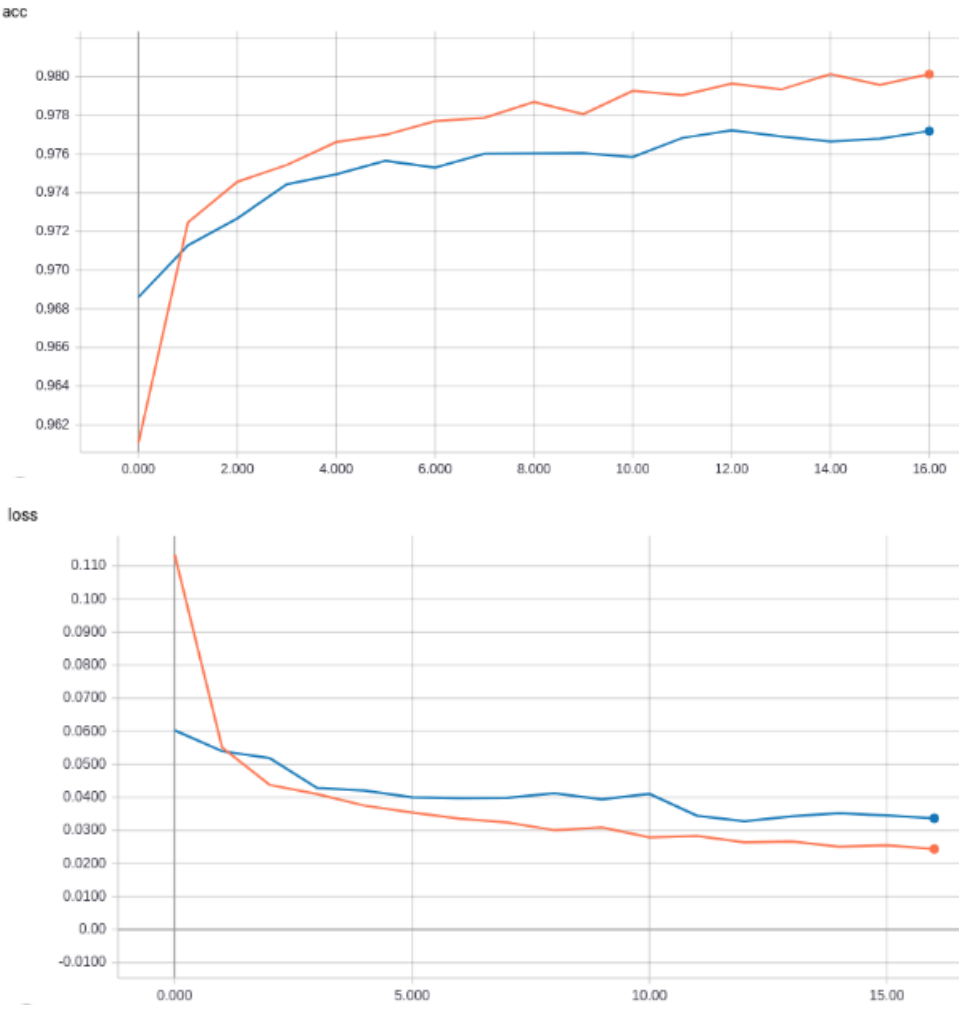
b) U-Net accuracy and loss graphics on *augmented floor plan dataset*



c) SegNet accuracy and loss graphics on *not augmented floor plan dataset*



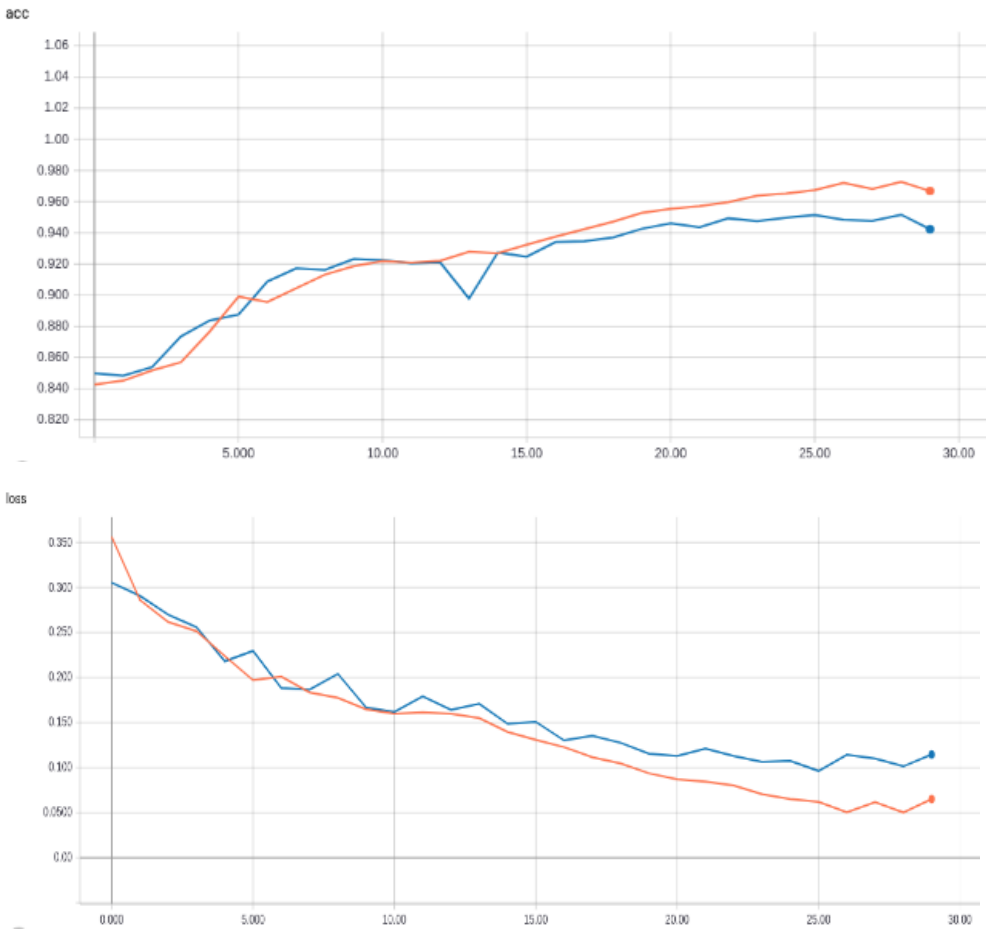
d) SegNet accuracy and loss graphics on *augmented floor plan dataset*



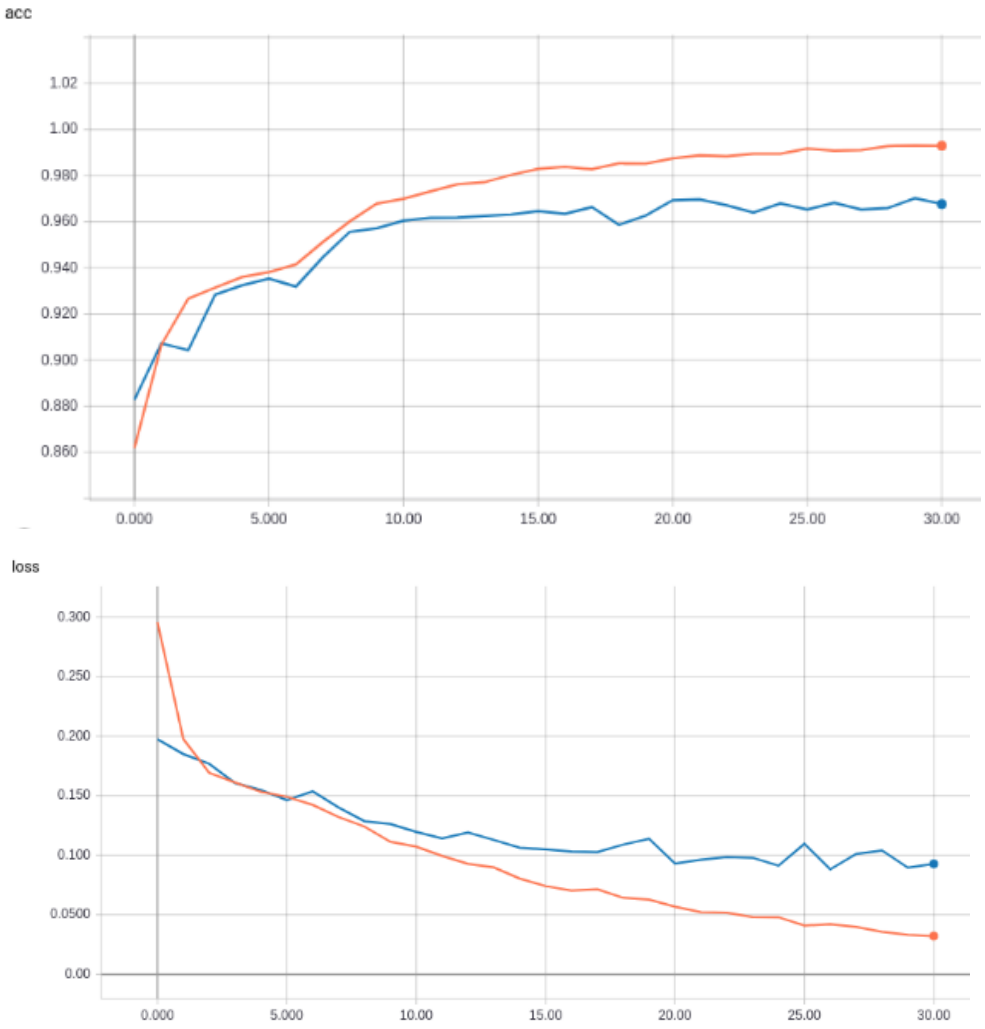
e) TernausNet accuracy and loss graphics on *not augmented floor plan dataset*

f) TernausNet accuracy and loss graphics on *augmented floor plan dataset*

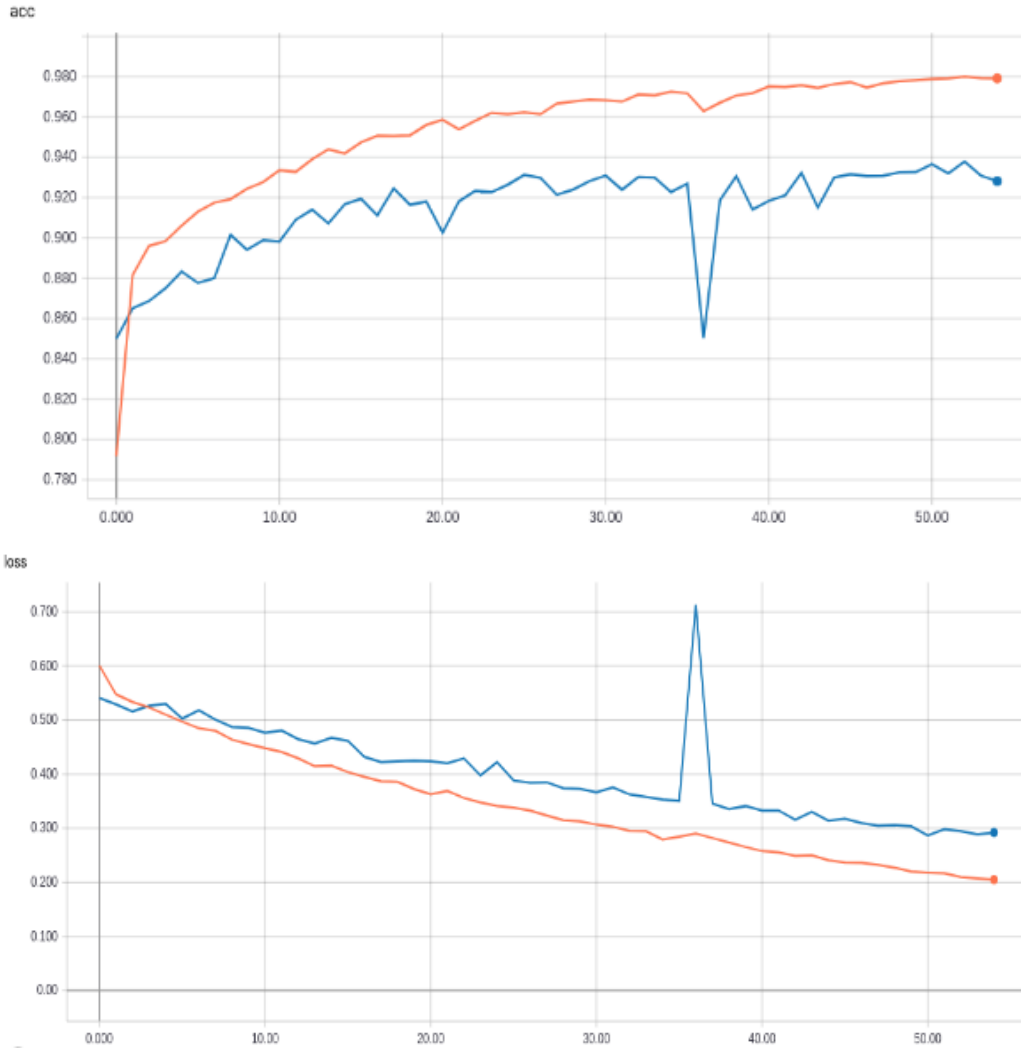
The graphs below show CNNs learning curves on augmented and non-augmented elevation dataset with original architectures. *Orange line* represents training set, and *blue line* shows validation set.



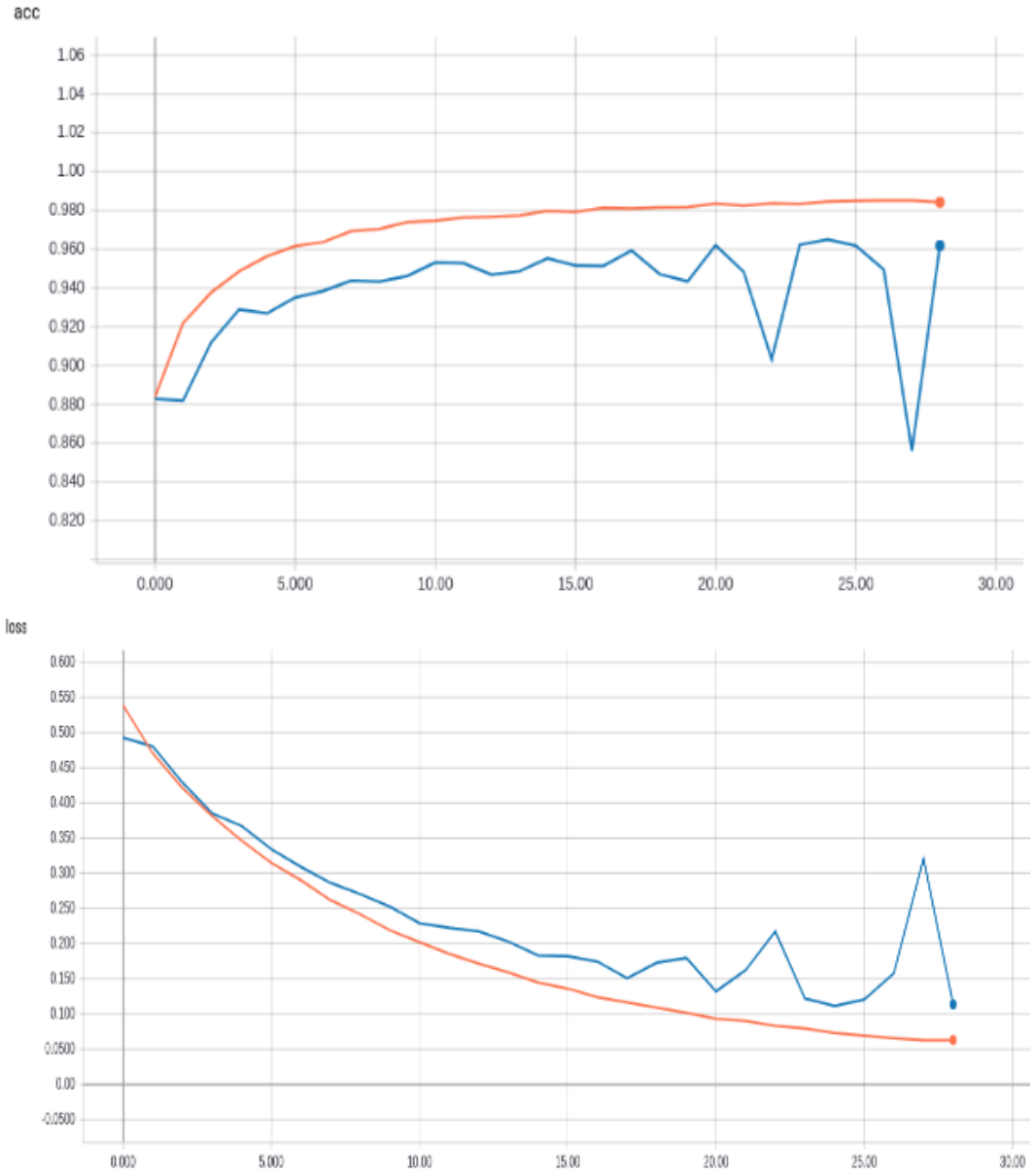
a) U-Net accuracy and loss graphics on *not augmented elevation dataset*



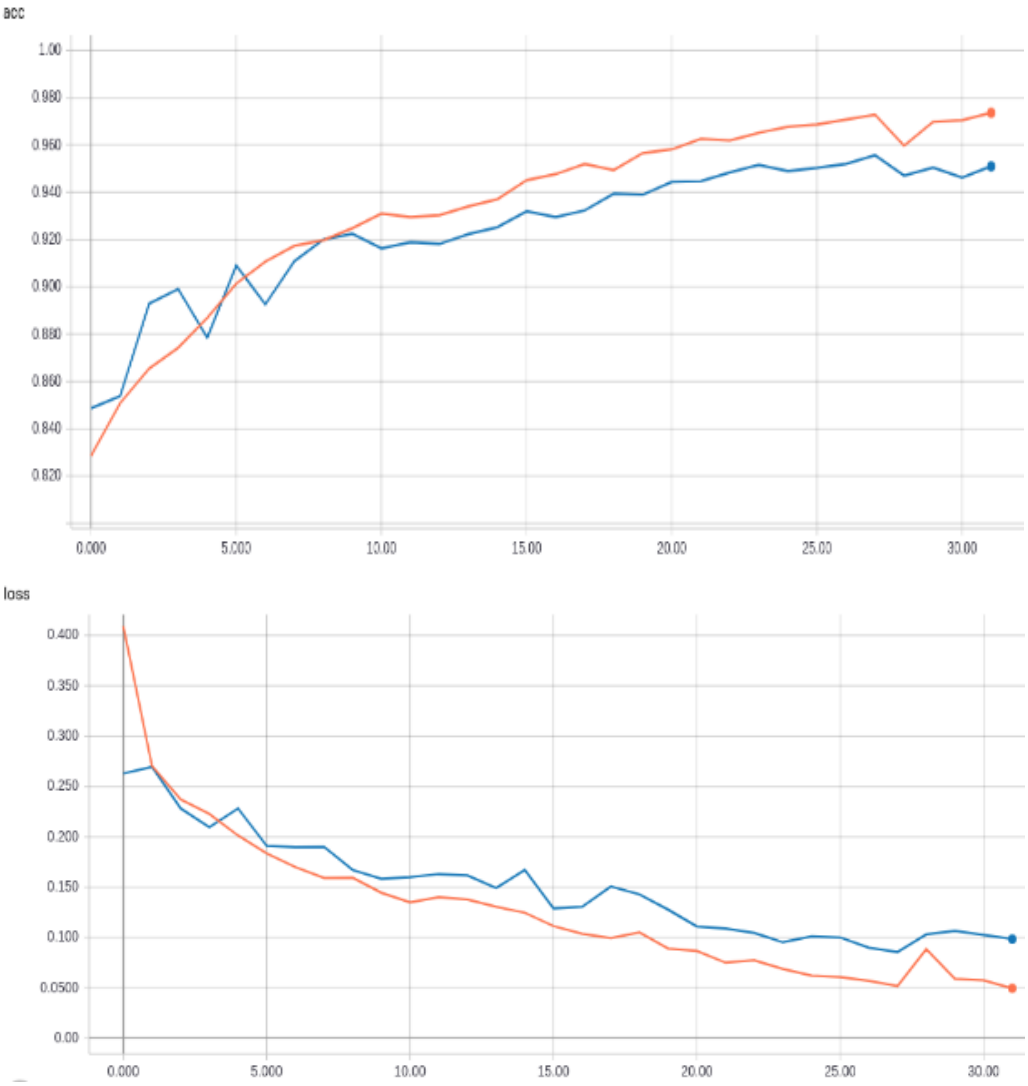
b) U-Net accuracy and loss graphics on *augmented elevation dataset*



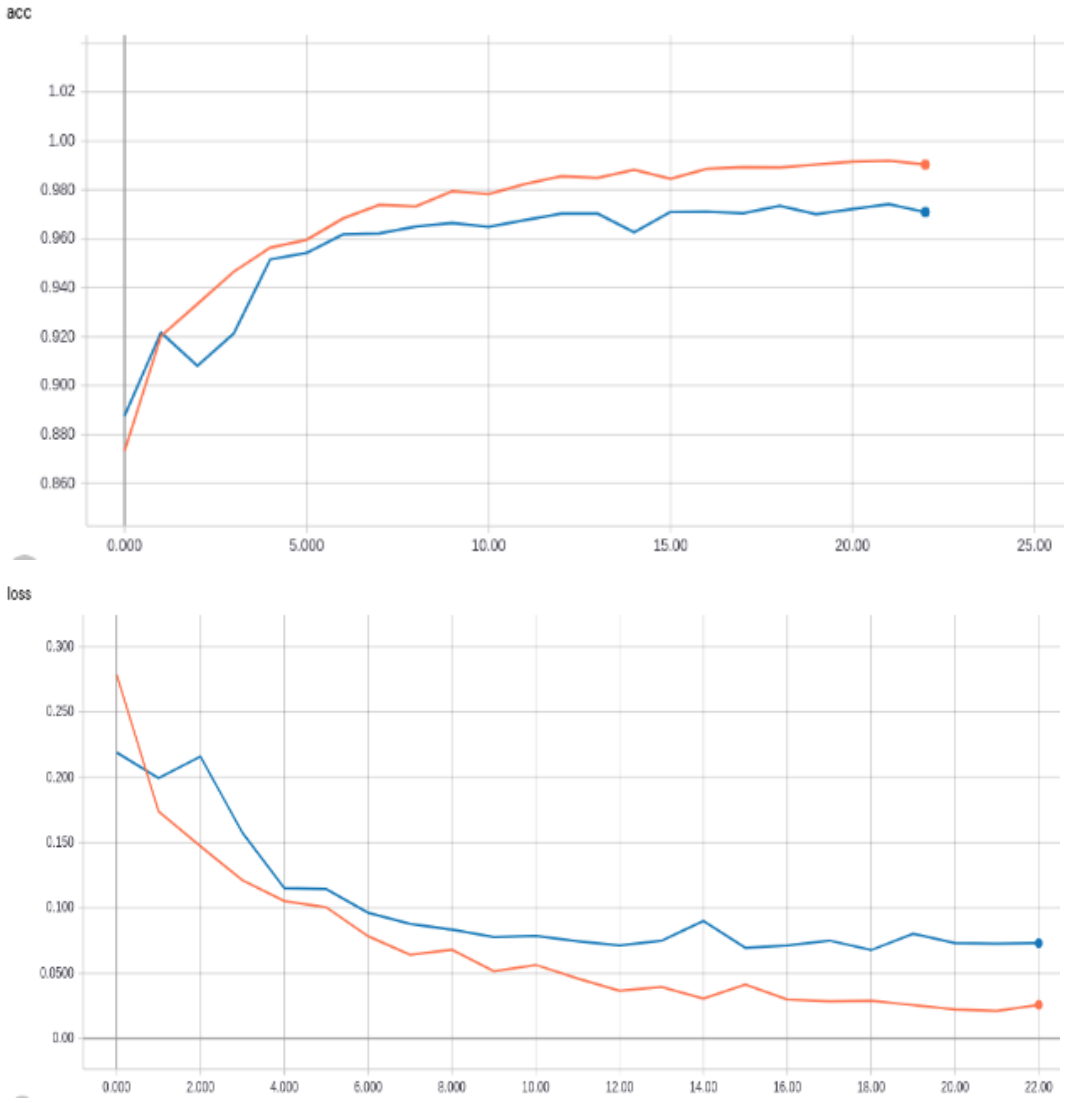
c) SegNet accuracy and loss graphics on *not augmented elevation dataset*



d) SegNet accuracy and loss graphics on *augmented elevation dataset*



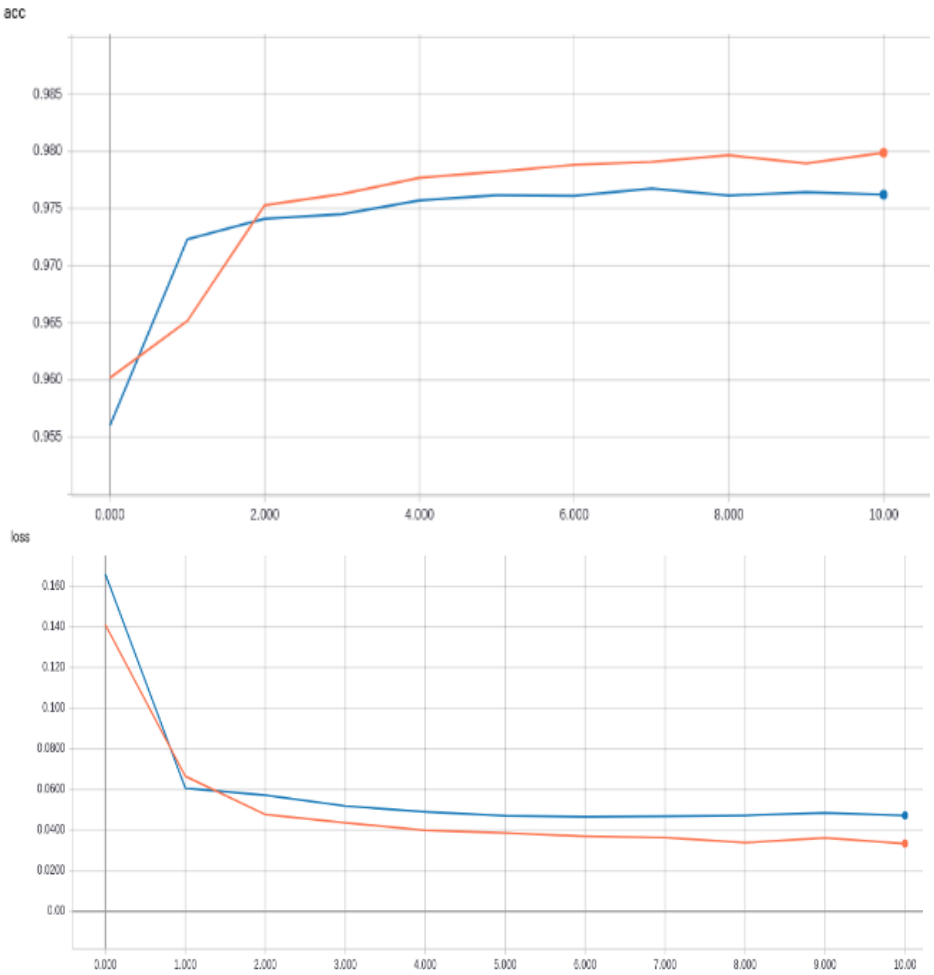
e) TernausNet accuracy and loss graphics on *not augmented elevation dataset*



f) TernausNet accuracy and loss graphics on *augmented elevation dataset*

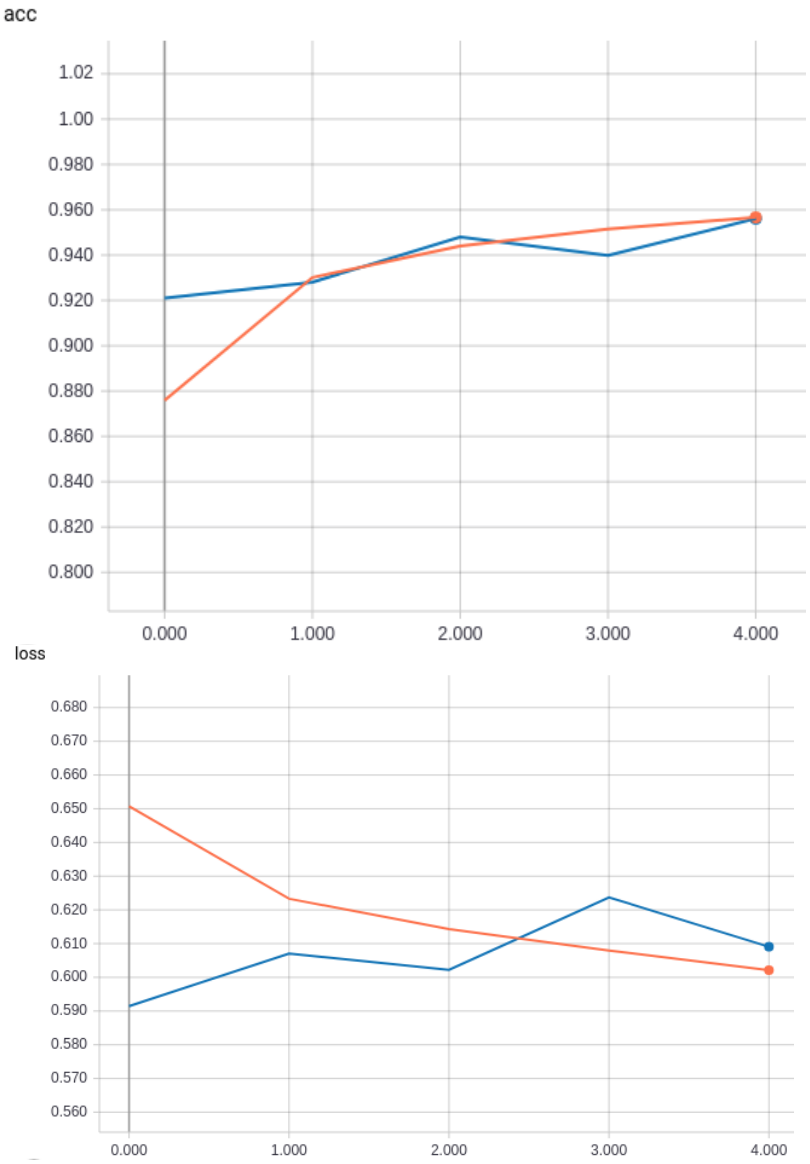
B. CNN Results with Reconfigured Architecture

CNNs learning curves on augmented and non-augmented floor plan dataset with *reconfigured* architectures are shown above. *Orange curve* highlights the training set rate, and validation set rate is represented with *blue curve*

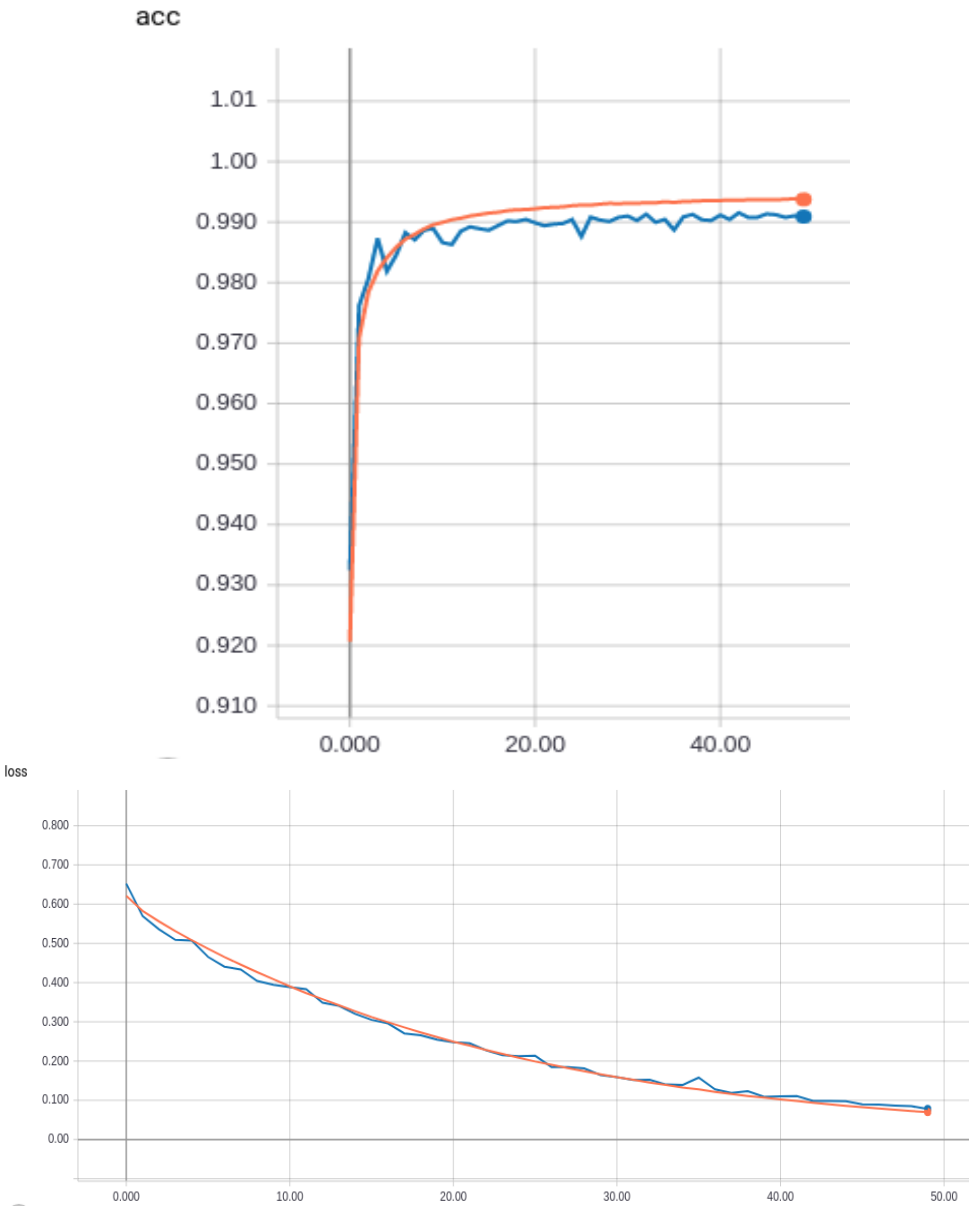


a) *Reconfigured* U-Net accuracy and loss graphics on *not augmented floor plan dataset*

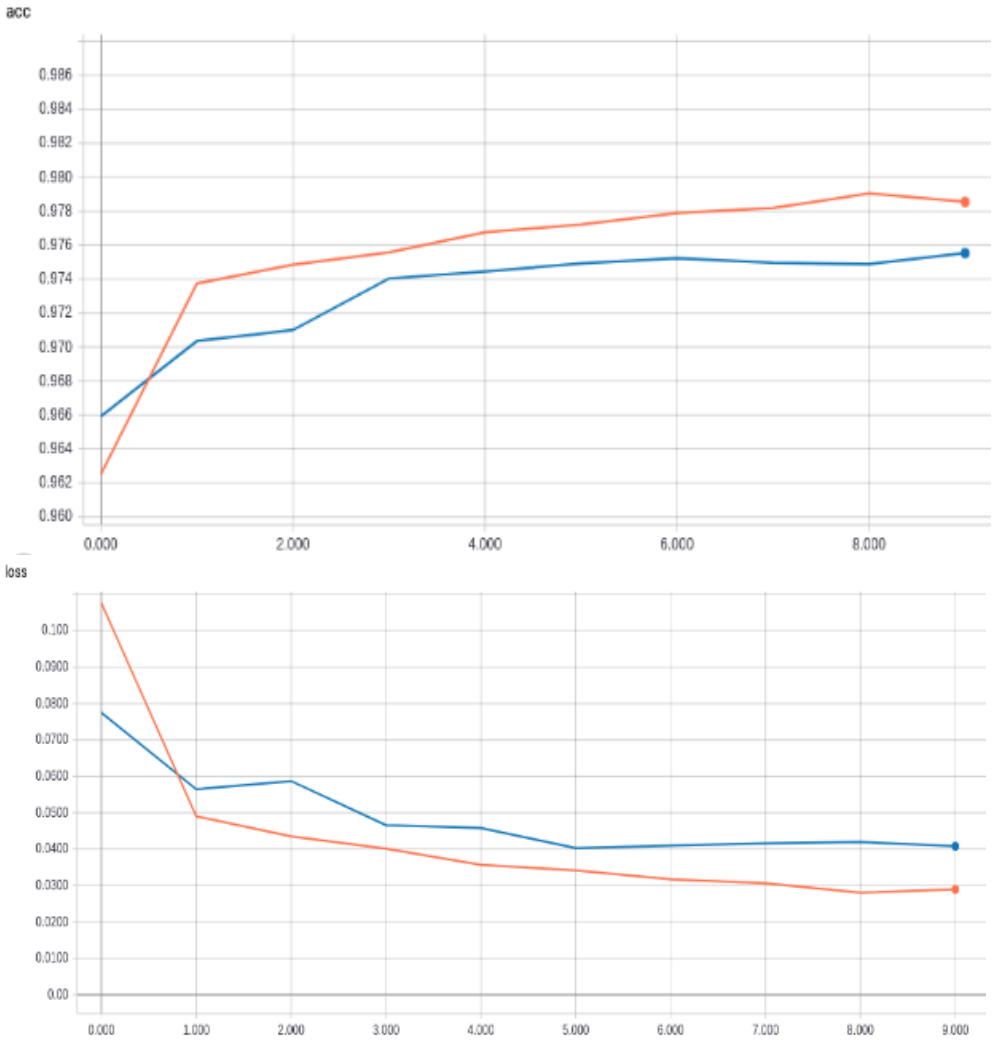
b) *Reconfigured* U-Net accuracy and loss graphics on *augmented floor plan dataset*



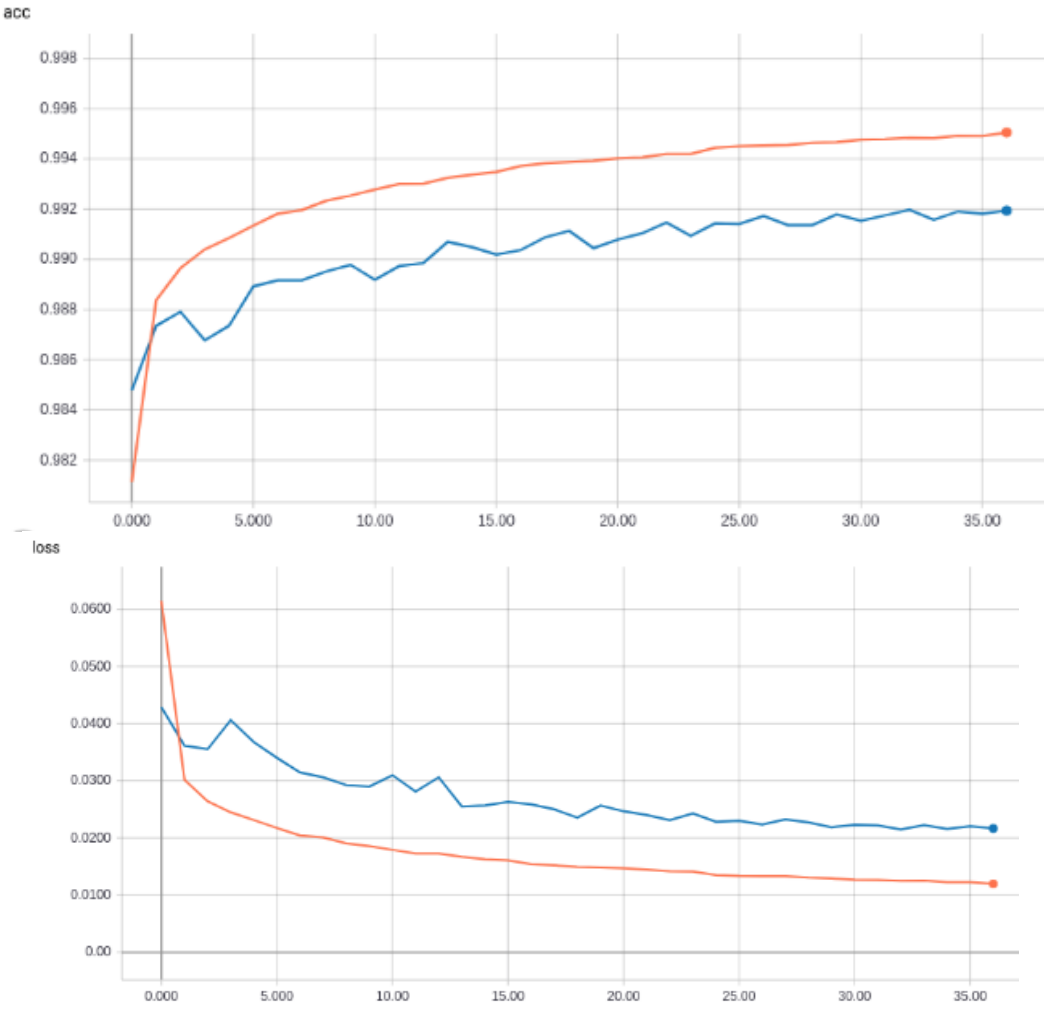
c) *Reconfigured* SegNet accuracy and loss graphics on *not augmented floor plan dataset*



d) *Reconfigured* SegNet accuracy and loss graphics on *augmented floor plan dataset*

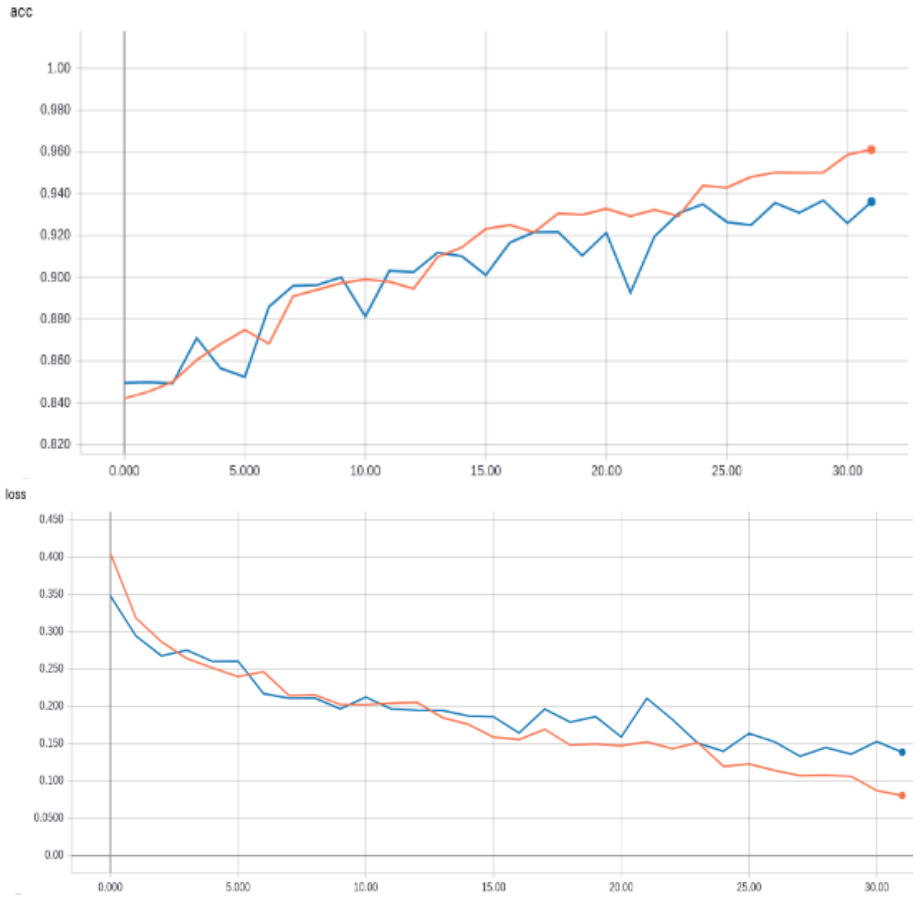


e) *Reconfigured* TernausNet accuracy and loss graphics on *not augmented floor plan dataset*

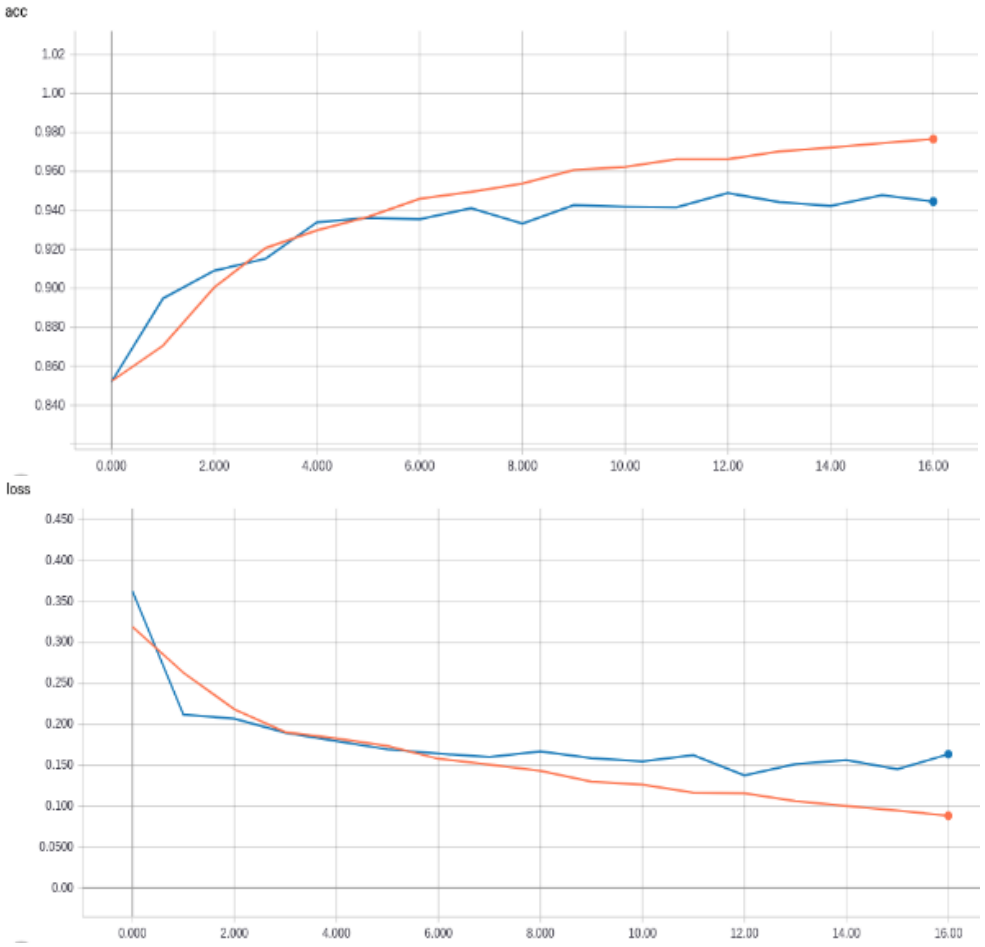


f) *Reconfigured* TernausNet accuracy and loss graphics on *augmented floor plan dataset*

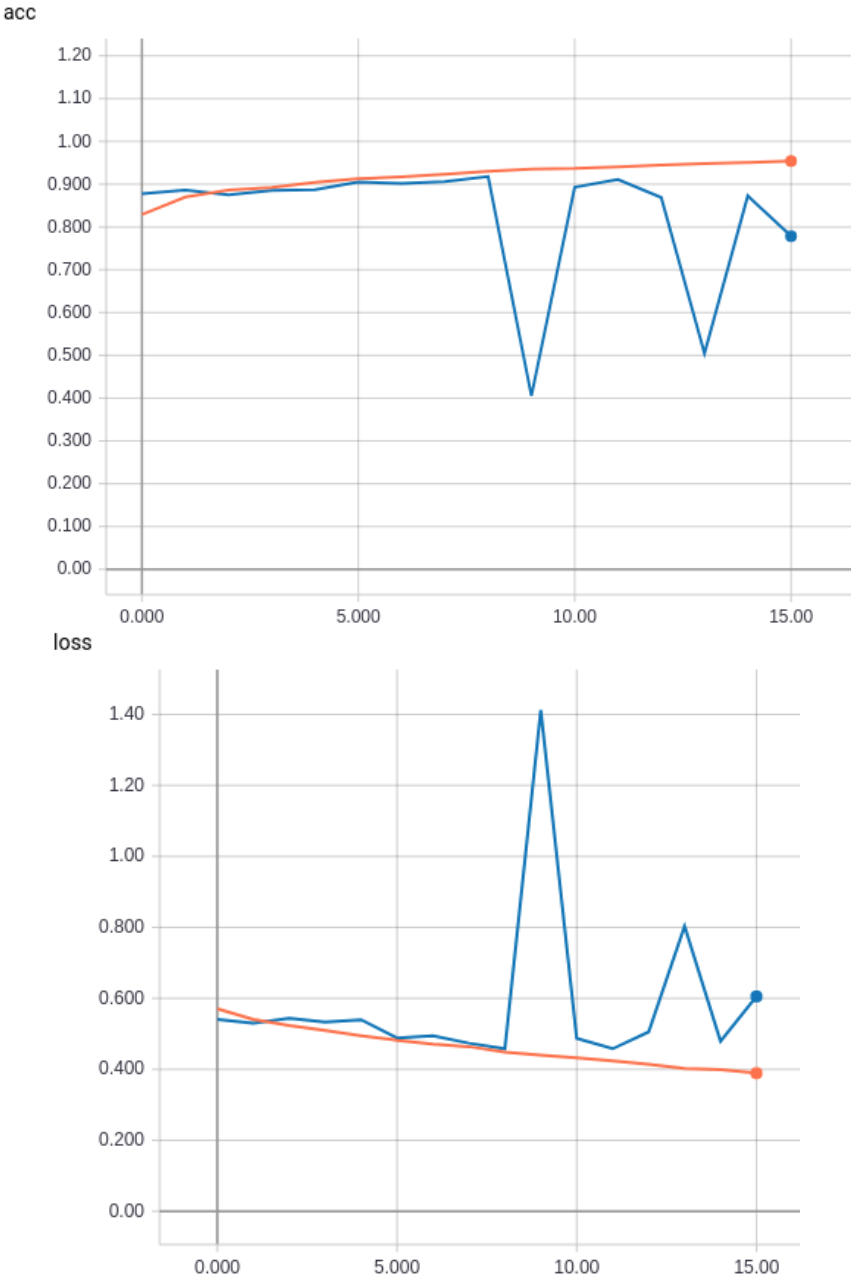
CNNs learning curves on augmented and non-augmented elevation dataset with *reconfigured* architectures as follows. Training set and validation set rates are shown as *orange and blue curves*, respectively.



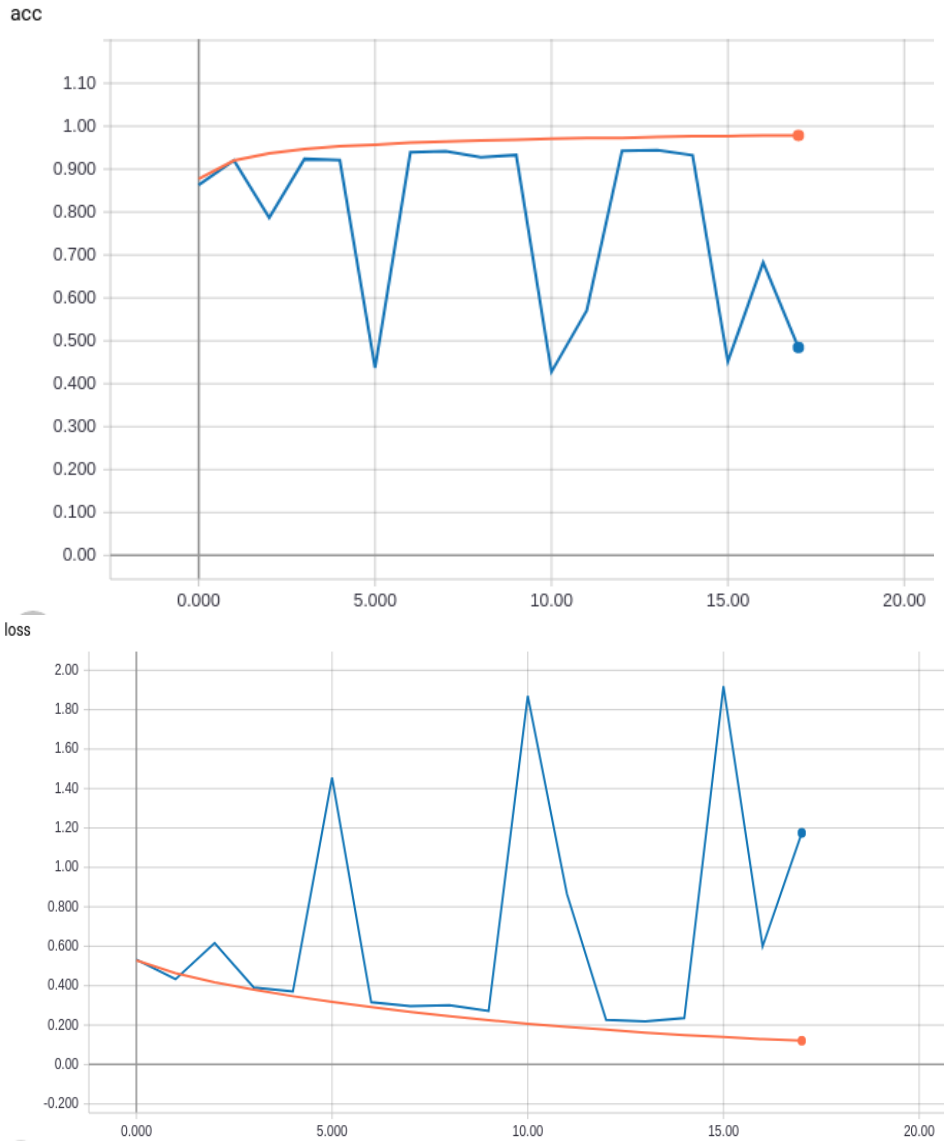
a) *Reconfigured* U-Net accuracy and loss graphics on *not augmented elevation dataset*



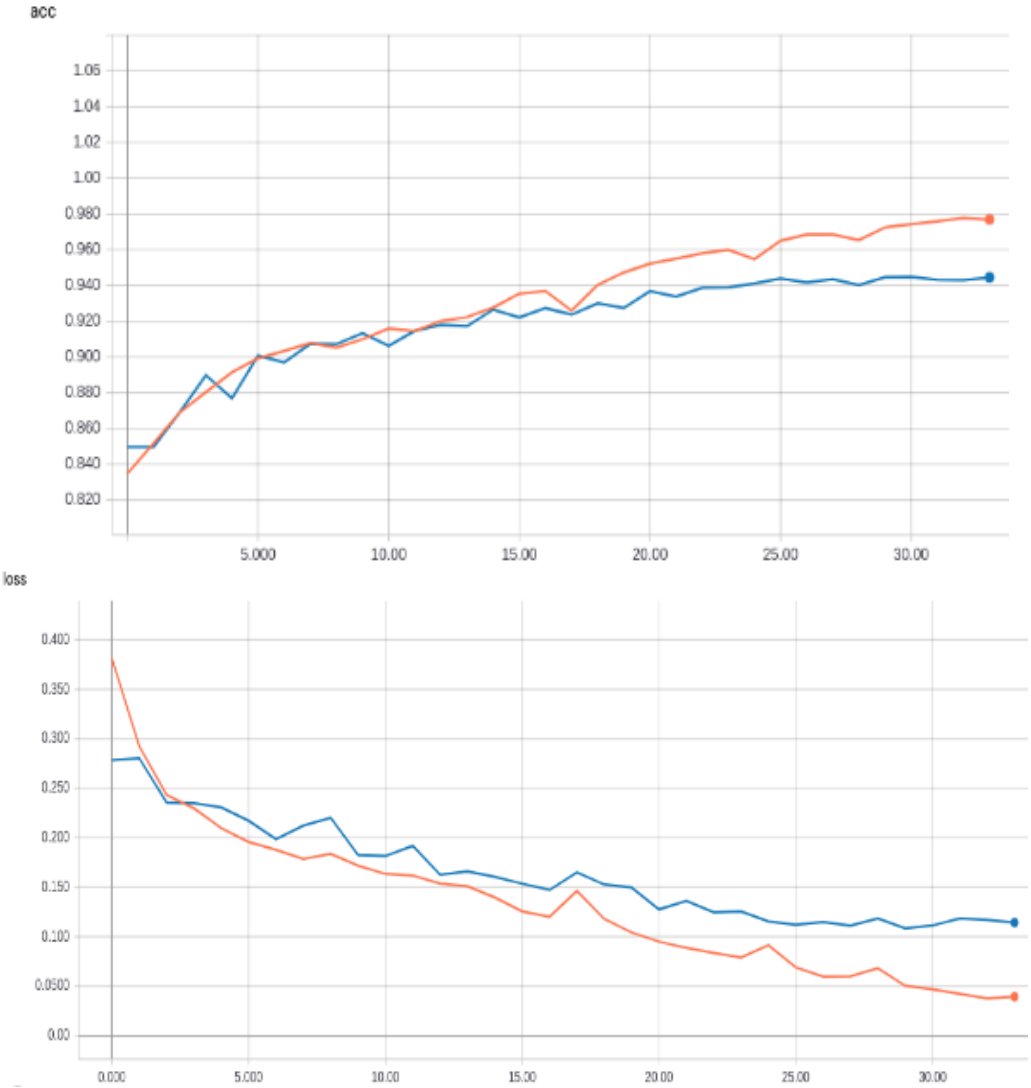
b) *Reconfigured* U-Net accuracy and loss graphics on *augmented elevation dataset*



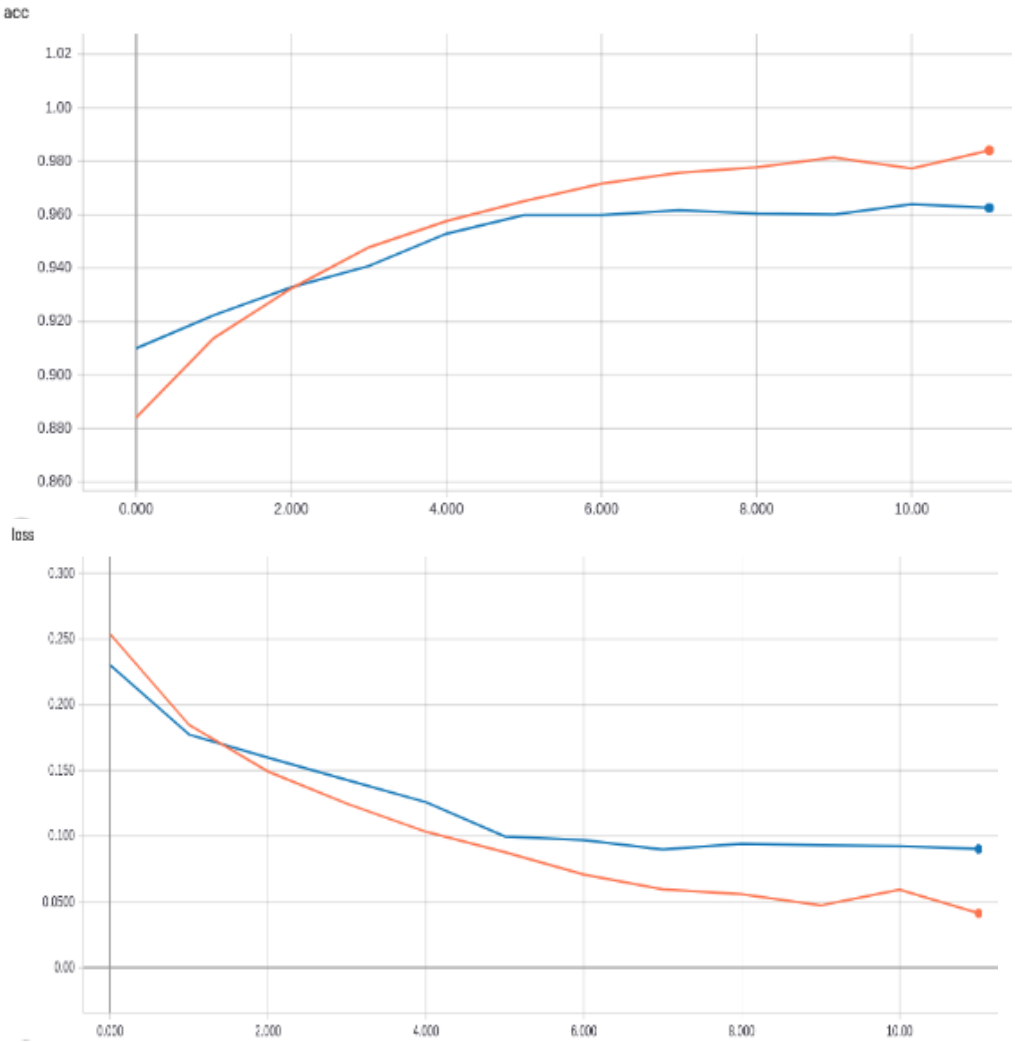
c) *Reconfigured* SegNet accuracy and loss graphics on *not augmented elevation dataset*



d) *Reconfigured* SegNet accuracy and loss graphics on *augmented elevation dataset*



e) *Reconfigured* TernausNet accuracy and loss graphics on *not augmented elevation dataset*



f) *Reconfigured* TernausNet accuracy and loss graphics on *augmented elevation dataset*