UNIVERSIDAD DE CHILE
FACULTAD DE CIENCIAS FÍSICAS Y MATEMÁTICAS
DEPARTAMENTO DE CIENCIAS DE LA COMPUTACIÓN

WALL POLYGON RETRIEVAL FROM ARCHITECTURAL FLOOR PLAN IMAGES USING VECTORIZATION AND DEEP LEARNING METHODS

TESIS PARA OPTAR AL GRADO DE MAGÍSTER EN CIENCIAS, MENCIÓN COMPUTACIÓN

PABLO NICOLÁS PIZARRO RIFFO

PROFESORA GUÍA:
NANCY HITSCHFELD KAHLER

PROFESOR CO-GUÍA:
IVÁN SIPIRAN MENDOZA

MIEMBROS DE LA COMISIÓN:
AIDAN HOGAN
EDUARDO GRAELLS GARRIDO
DOMINGO MERY QUIROZ

Este trabajo ha sido parcialmente financiado por Proyecto FONDECYT Nº 1211484

SANTIAGO DE CHILE
2023

RESUMEN DE LA MEMORIA PARA OPTAR
AL TÍTULO DE MAGÍSTER EN CIENCIAS
MENCIÓN COMPUTACIÓN
POR: PABLO NICOLÁS PIZARRO RIFFO
FECHA: 2023
PROFESORA GUÍA: NANCY HITSCHFELD KAHLER
PROFESOR CO-GUÍA: IVÁN SIPIRAN MENDOZA

RECUPERACIÓN DE POLÍGONOS DE MUROS DESDE IMÁGENES DE PLANOS ARQUITECTÓNICOS MEDIANTE MÉTODOS DE VECTORIZACIÓN Y DEEP LEARNING

El análisis automático de planos es un área dentro de la visión por computadora que ha sostenido un importante crecimiento en los últimos cinco años debido a un creciente interés de la industria por el desarrollo de software en el sector de la construcción y el diseño. Pese a que éstos se crean usando herramientas CAD, la distribución a clientes suele ser a través de imágenes rasterizadas que pierden toda información geométrica y topológica de las intrincadas configuraciones de muros, vigas, losas, cotas o decoraciones. Aunque exista un modelo digital, no hay certezas de que la información y metadatos estén correctas; es posible que tanto muros como vigas estén dibujados de la misma manera, en la misma capa, y con las mismas etiquetas.

Si bien se han diseñado múltiples algoritmos de procesamiento, la recuperación de objetos es particularmente compleja, ya que no existe un estándar de diseño en la industria. Los planos pueden tener cualquier estilo, forma y anotaciones, que dependen de cada oficina de arquitectura e ingeniería. Por tanto, las metodologías de recuperación que dependen de un estilo particular no poseen buena capacidad de generalización, siendo poco adaptables; así, aquellas basadas en datos son los que han alcanzado mejores resultados ya que emplean las imágenes de planos para inferir las intrincadas reglas de reconocimiento y recuperación para tareas como segmentación, vectorización, clasificación, entre otros.

Debido a estos motivos, esta tesis presenta una revisión de la evolución en las metodologías que analizan este particular documento, desde las definidas por reglas manuales, a aquellas basadas en *deep learning*, desglozando sus tareas, técnicas y desafíos. Como objeto de estudio, se desarrolló un modelo segmentativo U-Net que permite recuperar los polígonos de muros desde planos complejos de edificios residenciales chilenos, contribuyendo tanto con una nueva base de datos como con una metodología de procesamiento de imágenes, así como un *baseline* para futuras comparaciones. La salida segmentada se vectorizó escogiendo un método deep learning recuperado desde la revisión del estado del arte, permitiendo así obtener los polígonos de muros de manera automática desde un plano rasterizado.

Nuestro trabajo es completamente de código abierto, disponible públicamente a la comunidad https://github.com/MLSTRUCT/MLSTRUCT-FP. Creemos que éste beneficiará a investigadores y desarrolladores dentro de las industrias de la construcción y el diseño, las que han experimentado un complejo escenario mundial de productividad y crecimiento, haciendo evidente la necesidad de nuevas herramientas capaces de acortar la brecha tecnológica, mitigar las pérdidas y reducir costes.

RESUMEN DE LA MEMORIA PARA OPTAR
AL TÍTULO DE MAGÍSTER EN CIENCIAS
MENCIÓN COMPUTACIÓN
POR: PABLO NICOLÁS PIZARRO RIFFO
FECHA: 2023
PROFESORA GUÍA: NANCY HITSCHFELD KAHLER
PROFESOR CO-GUÍA: IVÁN SIPIRAN MENDOZA

WALL POLYGON RETRIEVAL FROM ARCHITECTURAL FLOOR PLAN IMAGES USING VECTORIZATION AND DEEP LEARNING METHODS

Automatic plan analysis is an area within computer vision that has sustained significant growth in the last five years due to increasing industry interest in software development for the construction and design sector. Although these are created using CAD tools, the distribution to clients is usually through raster images that discard all geometric and topological information of the intricate configurations of walls, beams, slabs, elevations, or furniture. Conversely, even if a digital model exists, there is no certainty that the information and metadata are correct; both walls and beams may be drawn in the same way, in the same layer, and with the same labels.

While multiple processing algorithms have been designed, object retrieval is particularly complex, as no industry design standard exists. Additionally, floor plans come in diverse styles, shapes, and with annotations that are specific to each architectural and engineering office. Consequently, retrieval methods reliant on a particular style lack broad applicability and adaptability. Instead, data-driven approaches have demonstrated superior performance by leveraging plan images to infer intricate rules for tasks like segmentation, vectorization, and classification, among others.

For these reasons, this thesis examines the progression of methodologies employed in the analysis of such documents, transitioning from rule-based approaches to those bolstered by deep learning technology, distilling the typical tasks, techniques, and challenges. As an object of study, we developed a U-Net segmentation model to retrieve wall polygons from complex plans of Chilean residential buildings, contributing both a new dataset and an image processing method, as well as a comparison baseline for future work. The segmented output was vectorized by choosing a deep learning-based method retrieved from the state-of-the-art review, allowing us to obtain the wall polygons automatically from a raster plan.

Our work is entirely open-source and publicly available to the community https://github.com/MLSTRUCT/MLSTRUCT-FP. We believe that it will benefit researchers and developers within the construction and design industries, which have experienced a complex global scenario of productivity and growth, making evident the need for new tools to bridge the technology gap, mitigate losses, and reduce costs.

To my family.

Acknowledgments

I want to thank my family, Patricio, Rossana, Francisca, and Bárbara. Thanks also to my advisor Nancy, who has supported me throughout the thesis process. Now, after this being my second M.Sc., the only thing that remains is the Ph.D. in computer science, which I hope to do very soon. See you, 🇨🇭.

Table of Content

List of Tables

List of Figures

Chapter 1

Introduction

1.1 Motivation

Architectural floor plans are documents that result from an iterative design process to define a structure's layout, distribution, and usage, playing a crucial role while designing, understanding, or remodeling indoor spaces [1]. Plans are created from the knowledge and experience of designers and engineers, who use different annotations to integrate each site's layout, style, use, scale, and external properties, like the environment and regulation. Usually, these documents convey three components to be a valid and complete 2D drawing description of a 3D scene: (1) *geometry*, which defines the shape and dimension of its elements, (2) *topology*, which accounts for the connectivity between building components, and (3) *semantics*, which describes additional characteristics, such as the room function [2, 3]. Moreover, floor plans might include outer and inner walls, windows, furniture, dimension lines, grids, text, or icons, alongside the constraints and relationships between them, making automatic analysis and information recovery a challenging and open task [1].

Plans have been actively studied in the last 40 years as they are involved in large industries, such as construction, design, property rentals, interior remodeling, or indoor positioning and navigation. Among those, the construction industry, unlike others, has experienced a low growth rate since the late 1960s in major OECD economies, such as the US and UK, or even yielded a negative one (Japan, Germany); therefore, the declining output per hour worked and per person employed became the focus of extensive research [4, 5]. A productivity decrease, particularly for construction, has negative repercussions on the economy, being even one of the key barometers for the 2009 global financial crisis [6]. For these reasons, the computer science community has studied several applications to enhance the design and construction pipelines, simplify the processes, and mitigate losses, eventually reducing costs and improving productivity.

Although plans are designed and built using advanced software, these are frequently stored as raster images in the application process [3]. Similarly, for projects designed before the introduction of computer-aided design (CAD) tools, the architectural documents exist in a paper format that has been manually drawn and scanned to achieve their digital version [7]. Rasterized plans allow non-experts and clients (e.g., home buyers) to understand and acquire information handily. However, these discard semantic and topological metadata like layer or object information, as it is generally considered that only humans will review them [8].

Analyzing these rasterized floor plan images and recognizing their components through an automatic procedure is a long-standing open problem within computer vision, which currently poses four fundamental challenges. First, there is no standard notation among architectural and engineering firms, where colors, line thickness, and symbols usually differ [9]. Second, plans stored as raster images are commonly characterized by complex, fuzzy architectural drawings [10]. Third, the plan structure must satisfy high-level geometric, topologic, and semantic constraints; for example, doors are embedded within walls, generally composed of parallel lines, and walls define the perimeter of rooms, in which their label, furniture, and layout can define its usage. Finally, the floor layout might vary across examples (e.g., houses or apartments can have different room arrangements) [1].

From a technical standpoint, floor plan analysis research aims to generate the building model by automatically extracting meaningful information from diverse sources, such as architectural plans or in-scene photographs [10]. This process regularly involves different tasks like recognizing walls and non-structural elements (e.g., windows, furniture), detecting and classifying rooms, and building 2D/3D reconstruction. Typically, these procedures cover different disciplines within computer science, like image processing, pattern and symbol recognition, object vectorization, and graph modeling.

Among plan analysis tasks, wall identification is one of the most common because these objects define the main layout of the building and convey essential information to detect other elements, such as doors or beams [11]. Recognizing walls is also helpful across the spectrum of architecture, engineering, and construction as it provides data for design, analysis, and cost estimation, among others [12]. On the other hand, recovery of the room shape and classification has also played an essential role since it allows for understanding the scene and its layout. Both walls and room information, along with other objects studied, have led to many applications within the industry, for example, in Building Information Modeling (BIM) reconstruction [13–15], 3D modeling from 2D plans [2, 16, 17], architectural optimization [18, 19], structural design [20–23], plan synthetic description [24], Virtual Reality (VR) exploration [25], indoor navigation and modeling [26–28], 3D reconstruction from in-scene photographs [29–31] and volumetric points [32, 33], floor plan generation [34–36], building search and retrieval [12, 37, 38], architectural symbol spoofing [39, 40], plan sketch interpretation [41–43], apartment price estimation [44], the generation of accessible plans for visually impaired people [45] or the automatic analysis of ancient and historical buildings [46–48].

Within related work, wall and room recovery has been traditionally solved using rule-based image processing methods, which exploit heuristics to locate the object notations in floor plans using shape recognition, text filtering, line scanning, and pixel classification [49]. Nevertheless, relying on hand-crafted features is insufficient, as it lacks sufficient generality to handle diverse conditions [50]. Extensive effort is required to choose proper low-level processing operations, tune parameters, and craft rules and grammar based on drawing styles or architectural regularity [16], rules that are still highly dependent on the plan format [10]. In other words, it is difficult to generalize the conventional pipelines to deal with complex annotations and high diversity [27]. For such reasons, several learning-based methods have been recently proposed to retrieve and model building objects, mainly by the application of Convolutional Neural Networks (CNNs), Graph Neural Networks (GNNs), and Generative Adversarial Networks (GANs), improving accuracy while keeping a general approach for handling different input styles [27].

Given the latest research in deep learning (DL) and image vectorization techniques, can the wall polygons be automatically obtained from Chilean architectural floor plan images of residential buildings? That is a model which considers as input an image of the floor plan (from a given drawing style of the national reality) and as output the wall polygon data.

In order to answer that question, the present work implements the U-Net semantic segmentation model [51], which automatically retrieves the walls from a rasterized floor plan. These walls are subsequently vectorized using a DL model [52] that predicts the primitives that constitute the segmented output. For such a goal, the first step is the review of different methods considered throughout floor plan analysis to determine which models are best suited for the problem. Although some works introduced a brief literature revision [10, 50, 53, 54], to the best of our knowledge, a comprehensive methodology still needs to be developed in this area. A review of the methods that solve different problems within floor plan analysis can guide future development in the construction, design, and engineering industries, for instance, in BIM and 3D reconstruction [15, 17] or the retrieval of similar plans from large databases [37], because it provides a quick guide into which dataset and algorithm must be used to solve a specific task.

In particular, this thesis conceptualizes the research problem, describes the used and available datasets, details the methodologies and their evolution through decades, and presents the challenges & opportunities for new work, providing insights into which area future developers must cover. Also, we propose a novel multi-unit floor plan dataset comprising 954 high-resolution images with annotated walls and slabs as polygons. These plans were recovered from 165 Chilean residential buildings designed by 52 architecture offices, offering various drawing styles and research opportunities for a field that has sustained a growing interest in the industrial sector that looks to automate and enhance their processes by creating new and better software.

1.2 Problem statement

In this investigation, the problem to be solved is obtaining the wall polygons from a rasterized architectural floor plan image of a Chilean residential building in an automatic procedure, without the need for human input, in such a way that it can handle several input styles and the resulting polygon adequately represents the semantics underlying the plan drawing. In previous work, researchers have used low-level image processing methods that exploited manual heuristics to find the objects; however, as these methods lack generality to handle diverse conditions imposed from the highly variable input plans, a DL approach will be used.

1.3 Research questions

From the proposed problem, the following questions arose:

- What datasets exist within the area of plan analysis; what are their properties?
- What methods exist within the rule-based and learning approaches? What are the common tasks among them?

- How has the area of study evolved over the years, considering the rapid development of artificial intelligence (AI)?
- What are the challenges and opportunities within research?
- What are the main applications of these algorithms?
- Which DL model allows for segmenting the wall objects considering a dataset of Chilean architectural floor plans?
- Can the proposed model, assembled from the selected segmentation and vectorization algorithms, obtain the wall polygons directly from the floor plan image?

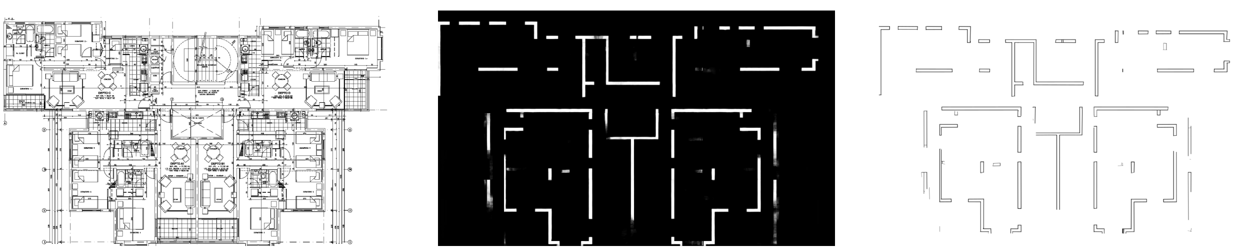
1.4 Hypothesis

The proposed model, assembled with the best DL and raster-to-vector segmentation models that emerge from comparing related work, will allow for obtaining the wall polygons from Chilean architectural plan images, with results similar to the state-of-the-art approaches that solve the problem for less complex and detailed plans.

1.5 Objectives

Main aims

In this investigation, we will develop a model (Figure 1.1) that considers as input an architectural floor plan image (Figure 1.1.a) and returns the primitives that constitute the segmented walls as output (Figure 1.1.c).



(a) Input – Architectural floor plan image from a Chilean residential building

(b) Segmented wall objects from the plan input using the proposed model in this thesis

(c) Output – Vectorized wall polygon primitives

Figure 1.1: Inputs and outputs of the proposed model.

Specific aims

O1) Compare discriminative and generative-based DL models for wall segmentation, which have been proven to perform better against low-level image processing methods that rely on manual heuristics. This aim considers an exhaustive literature review to determine which of the alternatives proposed by the scientific community meet the challenges and requirements of the thesis.

O2) Build a novel Chilean architectural floor plan dataset (954 floor plans from 165 different residential buildings) to find the best data structure to handle the wall segmentation and vectorization considering semantic and memory restraints.

O3) Implement the proposed model that automatically obtains the wall polygon from the floor plan image, consisting of DL segmentation and vectorization algorithms that emerged from the reviewed work. Results will be compared regarding the intersection over union (IoU) between the ground truth and the output images.

1.6 Methodology

1.6.1 Research

The first step of the research is to explore the state-of-the-art methods that solve obtaining the wall objects from an architectural floor plan image, considering both classical low-level image processing and the latest DL approaches (**O1**).

After the related work has been reviewed, in this thesis, we will implement an automatic wall vectorization method that employs the learning discriminative-based semantic segmentation U-Net model [51] to retrieve the wall objects from Chilean floor plans, which are later vectorized using a DL model [52] that predicts the primitives that constitute the segmented output (**O3**). The novel plan dataset was explicitly designed to develop segmentation/vectorization models of wall segments, providing a method for handling subsampling issues (**O2**).

1.6.2 Experimentation

Experiments will be performed throughout the steps to check the DL wall retrieval models. The aim is to compare the output polygons alongside the real solution (ground truth), which is already included in the 954 Chilean residential building floor plans dataset.

1.6.3 Technologies

The model will be implemented in Python, using Keras-TensorFlow [55, 56] as the machine learning backend. For image processing, OpenCV [57] is considered.

1.7 Structure of this work

The thesis structure is organized as follows:

1. In Chapter 2, we describe an extensive literature review of floor plan analysis research that retrieves several plan objects, considering both rule-based and learning-based models.

2. Chapter 3 describes our novel rasterized floor plan dataset of Chilean buildings, alongside the DL model that performs automated segmentation and vectorization of the walls.

3. Finally, in Chapter 4, we present the study conclusions, discuss its contributions, and outline areas for future work.

Chapter 2

Floor plan analysis review

Due to recent advances in machine learning, there has been an explosive development of multiple methods that automatically extract information from architectural floor plans. Nevertheless, the lack of a standard notation and the high variability in style and composition make it urgent to devise reliable and effective approaches to analyze and recognize objects like walls, doors, and rooms from rasterized images. For this reason, and with the aim of bringing some significant contribution to the state-of-the-art, this chapter critically reviews the methodologies and tools from rule-based and learning-based approaches. We discuss the datasets, scopes, and algorithms for guiding future developers to improve productivity and reduce costs in the construction and design industries. This chapter, which has been published in the *Automation in Construction* journal [58], concludes that most research relies on a particular plan style, facing problems regarding generalization and comparison due to the lack of a standard metric and limited public datasets. However, combining existing solutions can be employed in various and increasing applications.

2.1 Review method

The present study used content analysis [59] to select the reviewed literature. Content analysis is commonly employed to objectively make valid inferences according to collected data for disclosing central aspects of previous studies, further allowing for qualitative and quantitative operations [60]. In order to direct the review, the following research questions were proposed, which motivated the selection of the related work:

Q1. What datasets exist within the area of plan analysis; what are their properties?

Q2. What methodologies exist within the rule-based and learning approaches?

Q3. What are the common tasks among these methods?

Q4. How has the area of study evolved over the years, considering the rapid development of AI?

Q5. What are the challenges and opportunities within research?

Q6. What are the main applications of these algorithms?

Sample collection was performed in this study by searching and selecting peer-reviewed articles related to the research questions. Articles were collected from academic databases and cited works within them, considering their impact, contributions, and relationship with the review guidelines. The procedure of literature search and selection can be summarized as follows:

- The academic databases Web of Science, Scopus, IEEE/IET Xplore, Science Direct, ACM Digital Library, ASCE Library, ProQuest, and Springer were used for article search and selection. Also, Semantic Scholar and Connected Papers were employed to retrieve similar articles powered by AI and interactive graphs.

- Keywords such as "floor plan analysis", "floor plan recognition and interpretation", "floor plan segmentation", "floor plan image", "apartment structure", "architectural plan vectorization", "room and wall retrieval", "apartment graph", "object detection in floor plans", "multi-unit plans", and "parsing floor plan images" were used to search the databases. The search date period ranged from 1995 to December 1st, 2021. For each article, its cross-references and similar works were also considered for revision.

- The inclusion criteria correspond to English-only and peer-reviewed articles that used rasterized architectural floor plans of houses or apartments to perform the analysis. The recognized objects were walls or other non-structural elements (e.g., window, door) and rooms alongside their shape and classification, accounting for rule-based and learning-based techniques. Articles that vectorized or modeled a graph of the structure were also included.

- Works within floor plan analysis that recognized objects from sketches, volumetric points, CAD/XML-vector files, in-scene photographs, or examined other structures such as archaeologic or industrial complexes were excluded. Articles that did not consider the building semantics in the recognition, spotting, or vectorization of objects were also discarded; however, those that applied their algorithms to architectural plans were mentioned without further detail. Finally, articles that were only abstracts, minor revisions of previous authors' work, or that did not contemplate evaluating or validating their methods were also discarded. In total, 118 candidates were selected for further revision.

- Following the inclusion/exclusion criteria, a two-round selection technique was employed. In the first round, the titles, abstracts, and keywords of the noted articles were checked to ascertain if they met the criteria. The second round consisted of reading and analyzing the entire document, thus ensuring that all papers were closely related to the aforementioned objectives. Finally, 61 articles were selected and analyzed for the present review.

The analysis of each selected article considers the classification of its tasks, recognized objects, implemented models, used datasets, and a summary of the overall procedure. These features allow reviewed work to be represented in aggregated form within tables and figures, detailed in the following section, to quickly examine the methodologies, leading future developers to choose the appropriate one for their purposes.

2.2 Architectural floor plan analysis and recognition

Architectural floor plan analysis combines sequential processes that generate building models by automatically extracting meaningful information from rasterized floor plans [10, 13].

As these documents contain a large quantity of heterogeneous information, along with their constraints and interactions, most processes involve different tasks to clean the images and extract valuable data [61]. For example, the pipelines usually pre-process the image to remove distortions, grids, decorations, or titles through binarization. Text extraction and classification [62, 63], or line detection [9], are also common. Typically, pattern recognition, line scanning, or segmentation approaches are used to retrieve objects such as walls and doors, some of which are also vectorized to convert the recognized objects into a vector representation to be editable, scale-independent, and compact [52]. Room space is detected through geometry and semantic information, including textual data [64]. Symbol recognition is also an important part of building plan processing, which extracts labels to identify dimensions, room usages, and objects such as doors or windows [13, 65–67].

Although floor plan analysis considers many tasks and processes, they can be classified into four broad categories: (1) *Graphics separation*, a pre-processing technique for object recognition, which removes graphical elements from floor plans such as furniture or grids that do not bring new semantic information to the analysis, (2) *Object recognition*, a process which recognizes building elements like walls, openings, and rooms, being the core of the floor plan research, (3) *Vectorization*, a stage in which the structural elements are transformed into a vector form for their 2D/3D representation and analysis, and (4) *Structural modeling*, a process that aims to create a mathematical model of the floor topology, generally as a connected graph, by constructing an adjacency matrix based on the relationship among plan objects.

Rule-based methods, such as text filtering and line scanning, were initially proposed to recognize and vectorize elements like walls and rooms [49]. Traditionally, a pre-processing pipeline was carried out as the first step to separate graphical elements, for example, by distinguishing between lines of different thicknesses [10]. Nevertheless, relying on hand-crafted features is insufficient, as it lacks generality to handle diverse conditions [50]. Moreover, rule-based algorithms depend heavily on notation and empirical parameters, performing well in specific formats but having limitations in copying others. Extensive effort is required to choose proper low-level processing operations, tune parameters, and craft rules and grammar based on drawing styles or architectural regularity [16].

By contrast, learning-based approaches have garnered significant attention in recent years because they allow retrieving and assembling building models with better results, while being able to handle different input styles [27]. In the early learning approach, graphical separation and specific segmentation rules were needed. However, as DL was introduced, the applications have undergone rapid development or were simplified to a few steps. For example, many used the floor plan images directly to train the models without the need for complex image pre-processing pipelines, increasing the analysis versatility [10]. Compared to rule-based works, the research community has extensively focused on learning-based methods in the last five years, mainly due to the advances in machine learning models and the accessibility to richer and more extensive datasets. This trend is illustrated in Figure 2.1, which compares the number of published articles per year from 1995 to December 1st, 2021.

Although there has been a significant improvement in processing algorithms over the last years, floor plan analysis and recognition is still considered an open and challenging task

[1, 9]. Rule-based algorithms rely on particular plan styles that are hard to generalize or require expert knowledge to readjust for other formats. Learning-based models trained on various input floor plan datasets may have great adaptability. Still, their outputs may be blurry as they perform pixel-level segmentation, creating problems as some entities might have unconnected lines [68]. General-purpose object detection algorithms, such as Faster R-CNN [69] and YOLO [70], as well as other anchor-based frameworks, cannot retrieve curved or sloped walls or have problems recognizing objects in different conditions, as there is no suitable annotation to describe the complex geometrical characteristic of these architectural primitives [71]. Moreover, room detection and recognition depend heavily on structural elements in the floor plan, such as walls, doors, or windows. Thus, if a particular plan misses an element or some object polygons are not closed, it will considerably affect the room formation process [68]. Despite these difficulties and challenges, current works within the area have tackled many problems, from recognition to vectorization, with several applications for the construction and design industries, while improving accuracy and generalization to process diverse and complex floor plans.

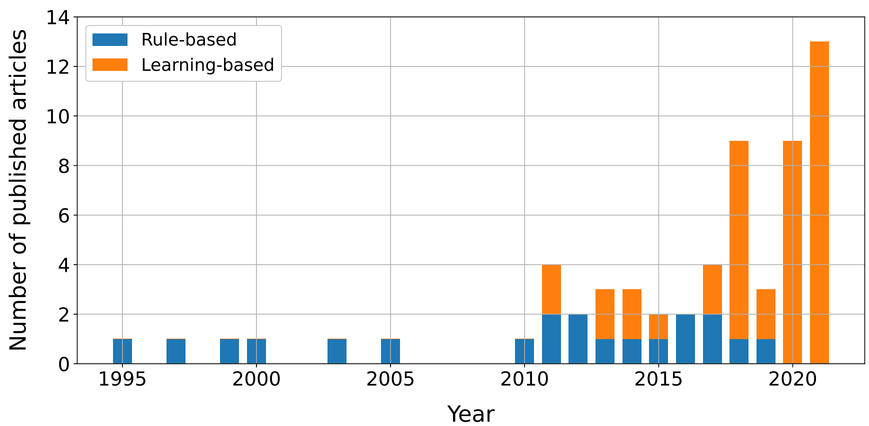


Figure 2.1: Articles published per year within reviewed works regarding rule-based (20 in total) and learning-based (41) approaches.

The following subsections describe the public and private datasets, as well as the rule-based and learning-based methodologies. In both cases, the reviewed works were cataloged according to the categories they satisfy (graphics separation, object recognition, vectorization, structural modeling), the objects they recognize (wall, door/window, rooms, OCR/dimension), and the model or algorithm implemented.

2.2.1 Datasets

Datasets have played an essential role within floor plan analysis because there is no standard notation for their composition; therefore, designed models must incorporate specific rules for each particular style. Typically, implementations face a high variability in their design due to three main reasons:

1. The plan representation, where, in best cases, only 70% of the graphical information is compliant with a standard rule [72].

2. The nature of these documents, where the total possible configurations and relationships between plan elements are extremely vast to handle manually.

3. The way information is visually represented, for example, in different styles, formats, or symbols [54].

Moreover, each floor plan dataset has limitations regarding quantity or complexity. Thus, researchers opt to utilize the one suitable for their purposes, including specific processing steps that could not be generalized to others [10].

For such datasets to be helpful in floor plan analysis, there must be annotations for objects such as walls and rooms. Annotating floor plans, despite other document types, is a complex and expensive task, as it requires high-level expertise to recognize the different elements due to ambiguity in notation [9, 54]. For example, in some plans, windows can be overlapped with beams, or the slab can contain paths, shafts, or custom symbols defined by architectural and structural firms. Even though several practical tools have been developed to annotate them conveniently [73–75], it is difficult to do so because there is no way to guarantee the same annotations from different experts, especially for complicated plans [10].

(a) **FPLAN-POLY** [76] (b) **SESYD** [77] (c) **CVC-FP** [75] (d) **R3D – Rent3D** [78] (e) **SydneyHouse** [79]

(f) **R-FP – Rakuten** [80] (g) **ROBIN** [12] (h) **R2V** [1] / **LIFULL** [81] (i) **CubiCasa5K** [82] (j) **RPLAN** [34]

(k) **BTI** [83] (l) **EAIS** [28, 84] (m) **ZSCVFP** [71] (n) **RFP** [85] (o) **RuralHomeData** [86]

Figure 2.2: Floor plan image examples from current datasets.

The reviewed datasets are summarized in Table 2.1, considering their source article, public availability, annotation, and the number of plans. Figure 2.2 illustrates a selection of images from the datasets considered within the review. It can be noted that there are distinct drawing styles among the apartment and house plans; some have color and textures (Cases *f, h, m, n*), room type labels (Cases *c-f, h, i*), icons (Cases *d, f*), dimension lines (Cases *c, l-n*), furniture (Cases *a-i, l, o*), and walls with several styles, angles, and complex arrangements. These diverse settings were exploited by rule-based methods, described in section 2.2.2, which recognize walls, doors, windows, furniture, and rooms by defining algorithms that considered different approaches specific to each style; or by learning-based ones (section 2.2.3), that trained models to automatically recognize the objects.

Table 2.1: Reviewed datasets used by floor plan analysis research.

Dataset **Name, reference (year)**	**Public** **access**	**Annotation**	**Number of** **plans**
FPLAN-POLY, [76] (2010)	✓ [87]	Walls, doors, windows, and furniture in vectorized format	42
SESYD, [77] (2010)	✓ [88]	Walls, doors, windows, and 6 different furniture types; 10 different synthetic apartment configurations, designed to study symbol recognition. Res 1,837–6,775	1,000
CVC-FP, [9, 75] (2010–2015)	✓ [89]	Walls, doors, windows, and rooms without type; 4 different subsets. Res 905–7,383	122
R3D – Rent3D, [78] (2015)	✓ [90]	Walls, doors, windows, and room types	215
SydneyHouse, [79] (2016)	✓ [91]	Walls, doors, and windows of multi-unit house floor plans; several styles. Res 404–4,678	174
R-FP – Rakuten, [80] (2017)	✓ [92]	Walls. Res 156–1,427	500
ROBIN, [12] (2017)	✓ [93]	Synthetic 3–5 room apartments; designed to study plan retrieval. Res 1,837–6,775	510
R2V, [1] (2017)	✓ [94]	Walls, openings, and room types. Res 96–1,920	815
CubiCasa5K, [82] (2019)	✓ [95]	80 object categories such as doors, windows, and walls. Res 50–8,000	5,000
RPLAN, [34] (2019)	✓ [96]	Wall, room, boundary, and inside masks; designed to study plan generation	80,788
Korea LH, [97] (2019)	✓ [98]	None. Res 230–5,092	343
BRIDGE, [99] (2019)	✗	Windows, doors, along with other 14 object types. Include region and paragraph descriptions	13,000
HouseExpo, [100] (2020)	✓ [101]	Binary house wall masks; designed to study indoor-layout learning. Res 110–10,086	35,126
BTI, [83] (2020)	✗	None	2,000
EAIS, [28, 84] (2020)	✗	Walls, doors	450
ZSCVFP, [71] (2021)	✗	Walls, rooms, entry, door, window, and balcony objects	10,800
RFP, [85] (2021)	✗	Walls, doors, windows, doorways, and 7 rooms types. Res 180–3,615	7,000
RuralHomeData, [86] (2021)	✗	Walls, doors, windows, stairs, slopes, text, and 21 room types. Res 1,600–2,560	800
RUB, [102] (2021)	✓ [103]	Segment nodes classified as door or non-door, both in image and CAD format. Res 500–18,000	74
LIFULL, [81] (–)	✗	None	5,300,000+

Note: Res – Resolution in pixels (px).

2.2.2 Rule-based methods

Early research within floor plan analysis studied the object recognition and modeling from CAD files, as these vector documents already contain the exact and accurate geometry of their elements in separate layers; however, the topological and semantic properties are usually not present or exist as icons or text. An early study, by Cherneff *et al.* [104], proposed an interpretation method to extract the plan structure, i.e., walls, doors, windows, rooms, and its associated spatial relations considering a limited drawing grammar. Shape Grammar

(SG) was a popular rule-based approach within automatic floor plan analysis, comprising a set of rules that can be applied consecutively to generate a geometrical shape, reproducing particular architectural styles [105]. Other early works are the vector segment conversion from line drawings [106], the hand-sketched plan interpretation [107], and the recognition of symbols and structural textures from printed or hand-drawn plan sketches [108]. Despite these examples, this preliminary research did not analyze the plans concerning the semantics and functional interaction of the elements, for example, the relationship between walls and rooms or that openings (doors and windows) are usually embedded between two wall segments. Furthermore, these examples did not consider raster floor plans, which are common when storing and distributing to customers [3], or processed simplified sketches. Therefore, the scope was restricted to analyzing vector-based CAD files or retrieving individual elements from simple-format plans.

Among the first works that considered the analysis directly on raster plans is Ryall *et al.* [109]. They proposed an early semi-automatic room segmentation method, which finds regions using a proximity metric. Despite its significant drawbacks, such as retrieving false positives from slab shafts, doors, or staircases, it serves as a first approach to extract objects directly from images, proving that algorithms can recognize the building semantics and constraints even if they are not apparent from a low-level standpoint.

A major improvement to Ryall's work happened with the contributions of Tombre's group that studied the automatic reconstruction of 3D structures from scanned plans [72, 110, 111], whose main idea is illustrated in Figure 2.3. Their approach estimated tiling the high-resolution images, dividing them into independent and overlapped patches to overcome memory issues, and segmenting the pixels of thin and thick lines by morphological filtering [112] after separating graphics and text. The overall process considers two kinds of walls represented by thick parallel or single lines. Doors are sought by detecting arcs, windows by finding small loops, and rooms by even bigger loops. Finally, the segmented pixels are skeletonized to assemble a vectorized format, which leads to the 3D reconstruction of a single level [72]. Moreover, a multi-level building reconstruction is possible if floors are matched by finding special symbols like corners, staircases, pipes, and bearing walls [110].

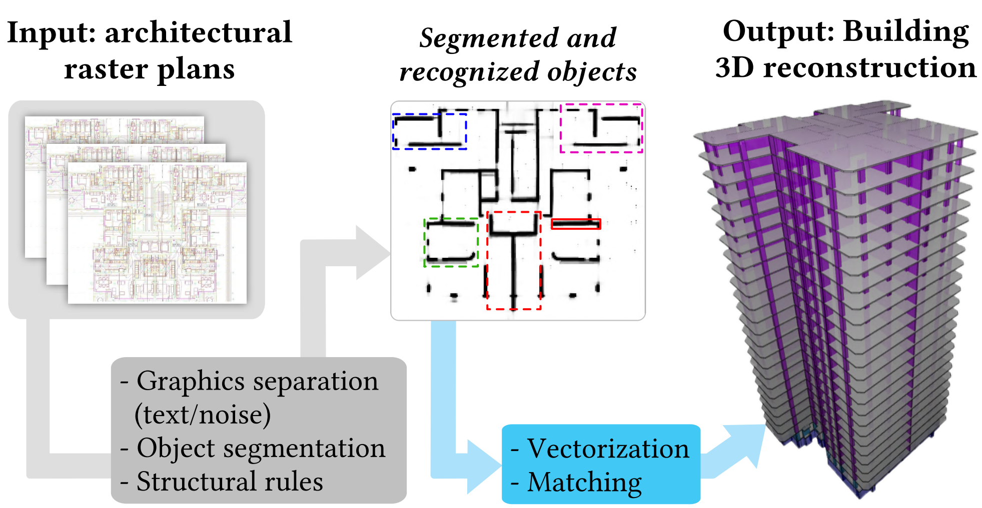


Figure 2.3: Reconstruction method of a 3D building structure from rasterized input plans.

Tombre's group has also intensively studied several rule-based methods for symbol detection, text separation, and graphics vectorization [112–115]. These proposed pipelines completely revolutionized floor plan analysis, contributing methods to assemble and reconstruct the overlying topologic and semantic constraints embedded in floor plans. However, the implemented symbol detection strategies are oriented to one specific and limited notation, same as their 3D building reconstruction method. Thus, a hypothetical change of the floor plan style might imply reconsidering parts of the algorithms, requiring a new set of threshold values for each case.

Or *et al.* in [116] also focused on 3D model generation from rasterized plans for one-story buildings. After separating text and vectorizing the graphical layer with QGAR tools [73], they manually detected object symbols unrelated to the plan structure, such as cupboards, sinks, among others. Once the remaining lines only belong to walls, doors, and windows, a set of polygons is generated using each vectorized image's polyline. Walls are represented by thick lines, windows by rectangles inside them, and doors by arcs. Similarly, Gimenez *et al.* [2] proposed another method to reconstruct the 3D models from image plans. After separating graphics using Tombre's work [117] and QGAR tools, various building elements were detected based on structural rules, like assigning the wall to two parallel lines within a certain distance; finally, 3D building models were generated by properly assembling the vectorized elements. Even though the models achieved a good performance concerning their plan style, these methods have many predefined hyper-parameters, manual pre-processing heuristics, or assumed a specific notation for wall segments; thus, these methods lack generalizability.

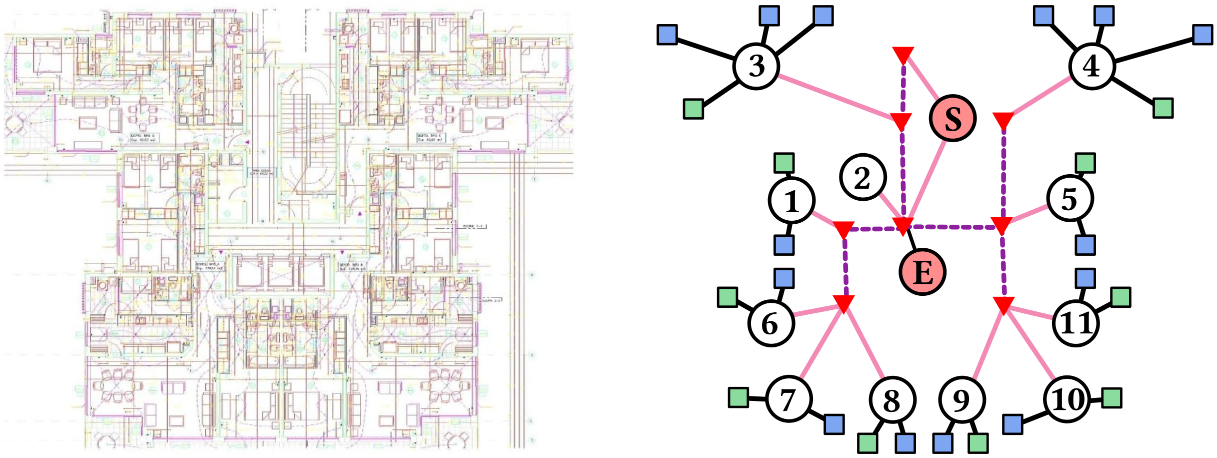
Macé *et al.* [9] also focused on extracting the structure from scanned plans and proposed an algorithm to detect rooms. Like previous examples, text/graphic pre-processing is performed with QGAR, followed by a thin/thick separation from graphic components based on coupling the Hough Transform (HT) [118] and image vectorization. The thick lines extracted from this algorithm are regarded as wall contours, which authors expected to be parallel, and are used as the candidates for the wall detection. Finally, doors and windows are identified to detect rooms through recursive decomposition until convex-shaped regions are found from the wall borders. Similar to previous works, this approach also considers manual thresholds and is limited to a specific notation; thus, the wall detector must be re-designed to deal with other plan styles.

Mace's work was later expanded by Ahmed *et al.* [43, 119], where they introduced new processing steps like wall edge extraction and boundary detection, designed for plan retrieval tasks. Their process starts with the wall detection and text/graphics segmentation [120] to separate graphical components into thin, thick, and, as a novelty, medium lines. Walls are assembled from thick and medium ones, while thin lines are considered to form symbols; components outside the convex hull of the outer walls were also removed. Then, doors, windows, and rooms were spotted using SURF [121], which is a method that provides an adequate discriminative translation, rotation, and scale-invariant representation of symbols. Finally, the text inside the rooms was used for their labeling. According to its distribution, the authors further enhance this method by splitting rooms into many parts as labels are inside them, vertically or horizontally [62]. It is important to note that these works [43, 62, 119, 120] consider some structural and semantic information as they assembled the wall contours of each room, labeled them with their name, and verified their composition using the door and window positions. However, as before, these methods might have to be revisited when dealing with floor plans of different graphical conventions.

Several other studies have also considered a line representation to recognize structural elements from floor plans. Park and Kwon [7] recognized the main walls of apartments using the auxiliary dimension line, where windows can be retrieved as a subproduct. Feltes *et al.*'s work [122] is capable of finding the object's corners in wall-line drawing images by filtering out unnecessary points without changing the overall structure, especially those that appeared through over-segmentation of diagonal lines; also, wall-gap filling is possible using a heuristic criterion. Tang *et al.* [123] automatically generated vector drawings by applying various filters, such as gradient, length, gap-filling, line-merging, and connectivity under several millimeter sizes, assuming walls are represented by parallel lines in both vertical and horizontal axis. Pan *et al.* [124] detected walls and windows considering empirical rules regarding their pixel layouts, where the user must adjust the method's thresholds; bearing walls corresponded to black areas, non-bearing walls to unfilled parallel rectangles, and windows are composed of three to four closer parallel lines. De [125] also assumed that only walls are illustrated as thick black lines in a floor plan layout. Thus, thick and thin lines can be distinguished using a morphological transformation; thick lines can be considered walls, whereas arc lines represent doors. On the other hand, in an effort to overcome the lack of a standard notation, de las Heras *et al.* [11] presented an unsupervised wall segmentation method that assumes walls as repetitive rectangular elements, placed in orthogonal directions, filled with the same pattern and naturally distributed across the plan. Although assumptions might work over a specific notation, they do not consider semantical relationships or require new rules to adopt for other plan styles.

Graph-based solutions also have been presented to describe the underlying structure of floor plans. Sharma *et al.* [126] proposed a room layout segmentation and adjacent room detection algorithm to represent the layout as an undirected graph. The model was developed to retrieve similar plans from a large database by calculating a matching score that considered fine-grained features computed from an assembled Room Adjacency Graph (RAG), where the room area and furniture types were identified [38]. Similarly, Barducci *et al.* [127] described floor plan images by building a RAG, identifying room purpose from the furniture recognized by graph matching, but without considering textual labels. Their work was further expanded by Goncu *et al.* [45], extending the wall, door, and room identification. Walls were binarized, straight-line segments were identified by the Hough transform (HT) and polygonized with the Ramer–Douglas–Peucker algorithm [128]. HT was also used to detect arcs, which were later assigned to doors, as previous examples did.

Figure 2.4 illustrates an example of a graph model from a complex rasterized floor plan. The circular numbered nodes represent apartments, red nodes indicate the stairs (S) and elevators (E), and the red inverted triangles stand for hall joints. The squared nodes belong to bedrooms (blue) and dinner rooms (green). Finally, edges represent the connectivity between elements.



(a) Rasterized floor plan. (b) Graph model.

Figure 2.4: Example of a graph model from a rasterized input plan.

Different low-level geometrical vectorization methods have also been developed to obtain the objects from generic line drawings, for instance, by separating and skeletonizing layers of homogeneous thickness [129] or by an energy-based approach [130]. Nevertheless, recognizing floor plan elements without considering their semantic relationship is error-prone, as each element can be a constraint for other objects. For example, walls delineate the perimeter of rooms, defining the layout and conveying essential information to detect other structural elements, like doors, windows, or openings [11]. For such reasons, and as a means to avoid the elaboration of complex recognition rules, learning-based models have been widely studied, especially in the last five years. Learning models, detailed in the following section, allow for extracting the complex and hidden relationships embedded in plan documents directly from the training data, synthesizing the experience of several architects and structural engineers, construction regulation, and human creativity.

Table 2.2 details the reviewed rule-based methods within floor plan recognition, considering the datasets used (Table 2.1) and the four categories of tasks, such as (1) *Graphics separation*, (2) *Object recognition*, (3) *Vectorization*, and (4) *Structural modeling*.

2.2.3 Learning-based methods

2.2.3.1 Machine learning in floor plan analysis

Since rule-based methods were introduced in floor plan analysis, many tasks have been solved with reasonable accuracies, such as recognizing the structural objects and symbols. Nevertheless, methods have still been confined to a few concise, simplified, and abridged versions of architectural plans [10]. The lack of a standard notation and the limited number of public datasets forced the pipelines to deal with different styles, for example, by applying new rules that were hard to implement or needed in some cases to fine-tune certain variables requiring expert design knowledge. These drawbacks, especially those related to the lack of public data, limited development within the area.

Table 2.2: Rule-based research, sorted by year, considering its tasks and datasets used.

Reference (year)	Dataset (number of plans used)	Strategy	G. Sep.[a]	Object recognition				Vect.[d]	Mod.[e]
				Wall	Door/W.[b]	Room	OCR/Dim.[c]		
[109] (1995)	Defined in paper (1)	Proximity field	–	–	–	✓	–	–	–
[72, 110, 111] (1997)	Defined in paper (2)	Tiling, Morphological operations, Skeletonization, Feature matching	✓	✓	✓	✓	–	✓	–
[7] (2003)	Defined in paper (1)	Auxilary dimension line, Binarization	✓	✓	✓	–	✓	✓	–
[116] (2005)	Defined in paper (-)	QGAR, Segment matching, Predefined rules	✓	✓	✓	–	–	✓	–
[9] (2010)	CVC-FP (80)	QGAR, HT, image vectorization, recursive decomposition	✓	✓	✓	✓	–	✓	–
[120] (2011)	CVC-FP (90)	Morphological operations, connected component analysis	✓	✓	–	–	✓	–	–
[119] (2011)	CVC-FP (80)	Text/graphics segmentation, line separation, SURF	✓	✓	✓	✓	✓	✓	–
[62] (2012)	CVC-FP (80)	SURF, post-processing room split, predefined rules	✓	–	–	✓	✓	✓	–
[127] (2012)	SESYD (1000), FPLAN-POLY (42)	Graph matching, adaptive thresholding, morphological operations, HT	✓	✓	✓	✓	–	–	✓
[11] (2013)	CVC-FP (122)	Predefined rules	✓	✓	–	–	–	–	–
[122] (2014)	CVC-FP (90)	Corner detection and filtering, wall gap closing	–	✓	–	✓	–	✓	–
[45] (2015)	CVC-FP (90)	Adaptive thresholding, HT, Ramer-Douglas-Peucker, Voronoi partition, RAG	✓	✓	✓	✓	✓	✓	✓
[126] (2016)	SESYD (1000)	Boundary extraction, morphological operations, graph spectral embedding	✓	✓	✓	✓	–	–	✓
[2] (2016)	CVC-FP (90)	Text and Geometry separation, HT, QGAR, predefined rules	✓	✓	✓	–	✓	✓	✓
[123] (2017)	Defined in paper (-)	Rule-based filters	✓	✓	✓	–	–	✓	–
[124] (2017)	Defined in paper (100)	OTSU binarization, predefined rules	✓	✓	✓	–	–	–	–
[38] (2018)	ROBIN (510)	Topological adjacency graph, furniture categorization	–	–	✓	✓	–	–	✓
[125] (2019)	Defined in paper (80)	OTSU binarization, thin/thick morphological separation, skeletonization	✓	✓	✓	–	–	✓	–

[a] Graphical separation [b] Door/Window/Furniture/Others [c] OCR or object dimensions were recognized [d] Vectorization [e] Modeling (Graph, other)

However, since 2017, an explosion in research of learning-based methodologies (Figure 2.1) happened alongside the increase of public datasets (Table 2.1) and general-purpose models within the computer vision field. In contrast to the rule-based methods previously detailed, learning-based pipelines automatically learn the relationship between floor plan elements by exploiting new low-level and high-level features directly from hundreds of validated floor plans, improving results while simplifying the analysis. However, learning methods require a larger volume of data for training and parameter tuning, which can be challenging to access, extremely expensive, or unnecessary if only a few concise plans are required to be processed.

Among the first approaches, de las Heras and Sánchez [131] proposed a syntactic model for architectural floor plan interpretation. A stochastic image grammar over an And-Or graph was inferred to represent the hierarchical, structural, and semantic relations between floor plan elements, thus comprising architectural knowledge. This grammar was augmented with three different probabilistic models, learned from a training set, to account for the frequency of these relations. Then, a parser with a pruning strategy was used for the plan recognition. Walls and doors were detected using Mace's rule-based method [9], windows were extracted using a bag of patches approach, and rooms were assembled by joining each element with incident neighbors. Despite its recognition results, this work introduced a learnable model for interpreting a plan; however, rule-based methods were still needed to detect the structural objects. To overcome the last issue and push learning-based algorithms to become style-independent, the group later proposed a machine learning procedure [54] to study and recognize floor plan elements, thus avoiding the need for complex ad-hoc rules for each notation.

In 2014, de las Heras *et al.* presented a style-invariant, automatic method that uses a Support Vector Machine Bag of Visual Words (SVM-BOVW) to detect the pixel boundaries of the structural elements [54]. BOVW is a technique that describes an image as a set of visual words or topics created by clustering similar low-level image features extracted from training data [132]. With such a method, the authors later presented an improved pipeline which consisted of two steps: a statistical pixel-level patch-based segmentation, and structural recognition [133, 134]. The image patches were classified into three types (doors, walls, and windows) using the BOVW model. In addition, the pipeline recognizes room boundaries in the floor plan by finding closed regions surrounded by vectors in a planar graph of structural entities. Even though these works achieved a remarkable advance in architectural floor plan analysis, the models were still tuned to each particular graphical style in the CVC-FP dataset [75], using different parameters for each wall type. Thus, they cannot be generalized to arbitrary scenarios.

A similar approach based on the SVM-BOVW model was proposed by de las Heras *et al.* [135], but using an unsupervised segmentation as a preliminary pipeline step to avoid expensive and time-consuming image labeling [11]. In this work, a template-matching technique is done by finding parallel and closer lines to seek the wall-segment candidates; those were also ranked considering a score based on assumptions regarding the plan style. Finally, a patch-based SVM-BOVW learns the candidate's appearance and refines the initial segmentation. Although the method can be applied to several un-labeled styles, the walls must abide by strict assumptions. Furthermore, the semantic relationship between segments is ignored, as only the drawing style is considered when querying elements. For instance, if walls and furniture have a similar line notation, both are segmented, independent of their semantic representation.

An unsupervised statistical approach was also presented by de las Heras *et al.* [136]. In that work, they introduced an attributed graph grammar that represents the floor plan layout by incorporating structural and semantic relations within the building objects learned stochastically from annotated data. The stochastic model embedded in the grammar allows for inferring contextual relations between architectural elements, adapting the methodology to the variability while analyzing different plans. Their parsing method relies on their previous SVM-based pipeline to recognize walls and doors [135], considering the standard rule-based Hough transform (HT) method. Although this contribution summarizes the techniques proposed by the group to assemble a complete floor from a style-invariant model, it relies on complex learning rules, and assumes a particular format for wall recognition.

After SVM-based models, different algorithms have been proposed in recent years to improve performance, simplify the analysis, and generalize recognition to more plan styles and formats. Mewada *et al.* [53], for instance, introduced a framework based on the α-shape algorithm [137] to extract room shapes from binarized images, calculating and classifying their properties, such as room's width, length, area, and type, using a linear regression model. Other works have also presented learning-based models for classification; however, rule-based algorithms were still needed to recognize the objects. Guo and Peng [138], for example, segmented walls considering their color gradient, eliminating noise by adjusting a threshold. They used a pre-trained VGG-16 network [139] (Figure 2.5) to extract features of the floor plan, inspired by transfer learning, whose goal is to extract information from related tasks to assist in solving new ones that lack valid training sets. Later, the wall shapes were classified with a multi-layer perceptron into rectangle, square, L-shape, and irregular classes. Another recent example is the work from Park and Kim [140], which assembles a 3D model of the building using rule-based methods to recognize the horizontal and vertical walls, further using the learning-based TensorFlow object detection API to detect the wall junctions, openings (door/window), and rooms. The results from junctions and walls were used later to assemble a graph representation of the plan layout employing five generation rules, allowing their approach to vectorize the elements and reconstruct their 3D representation.

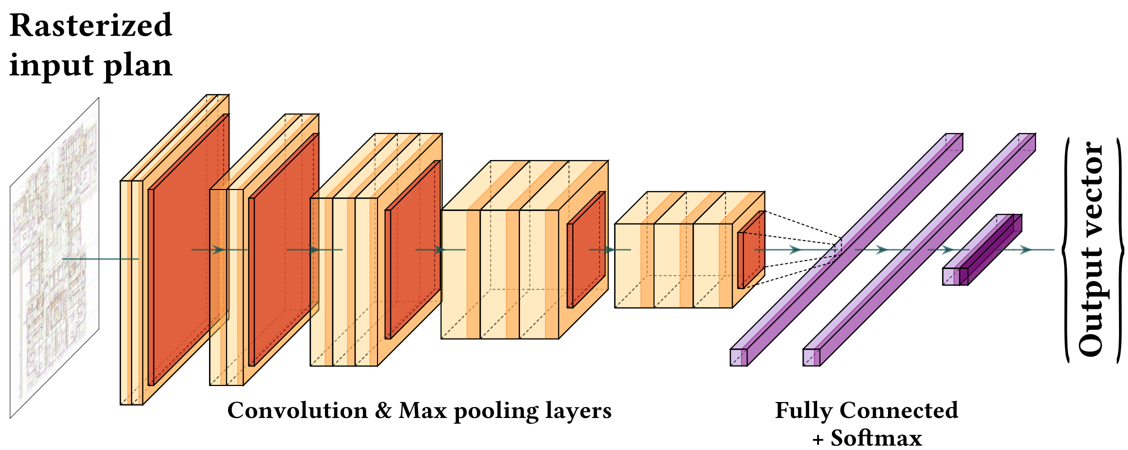
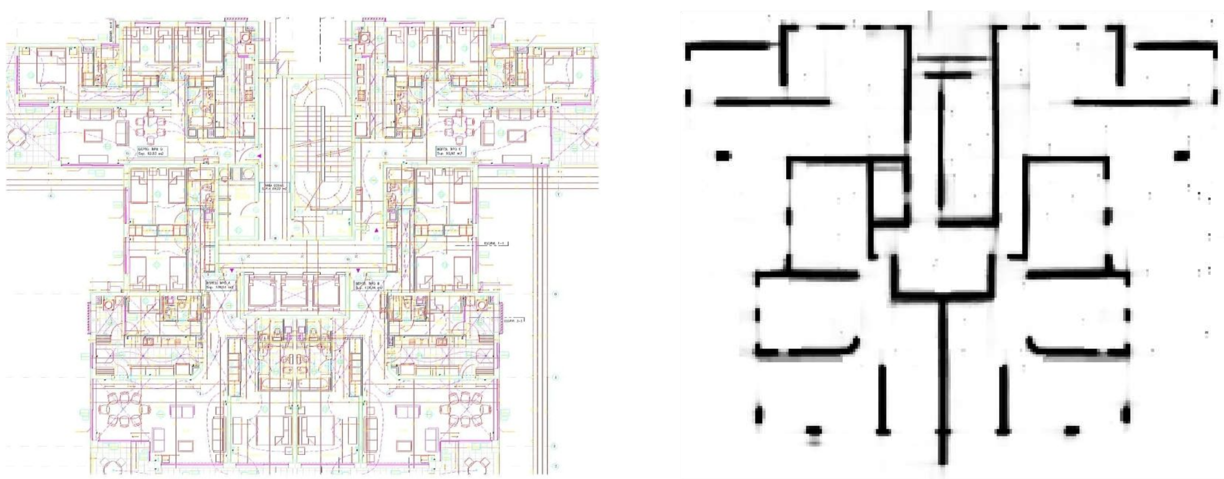


Figure 2.5: VGG model architecture, which extracts features from a rasterized floor plan and outputs a vector that can be used to predict or classify several elements.

A Positive Unlabeled (PU) learning-based approach was presented by Evangelou *et al.* [141] to retrieve walls similar to a manual query by the user, exploring object recognition from unlabelled plans as a means to avoid the expensive annotation task. In PU learning, a binary classifier learns in a semi-supervised way from positive or unlabelled data points, where the assumption is that the unlabeled data can contain both positive and negative examples. It is typically used when labeled data is unavailable, has many outliners, or the training dataset contains many false negatives [142]. In the context of the proposed method, the query serves as the positive example of the particular wall template to be matched, whereas the filtered candidate Regions of Interest (ROIs) of each floor plan are unlabelled.

Despite being a single object retrieval model, this SVM-based PU approach improves the performance concerning the BOVW [54].

Fuzzy rule-based systems (FRBS) have also been studied within floor plan analysis. Fuzzy logic is an intelligent controller that simulates human behavior by incorporating *If–then rules* into the system, thus including human experience and knowledge [143]. Leon-Garza *et al.* [132] introduced two Type-1 FRBS models that use fuzzy logic and similarity of image patches to add context information, an approach inspired by the BOVW [144] and the patch-based segmentation process proposed by de las Heras *et al.* [133]. One model used only pixel-level information (color intensity) and the other pixel-level and context information to segment floor plans for wall retrieving. An interval Type-2 FRBS model was also presented by Leon-Garza *et al.* [145], which does not need a pre-process step to remove noise from the image, and outperformed Type-I models in terms of the Intersection over Union (IoU), a standard metric for segmentation problems [133, 146]. Although FRBS models are simple to implement, have low computational cost, are transparent, explainable, and modifiable by end-users (architects or engineers) [147], they still suffer from common issues present in other floor plan analysis models. In this case, they are hard to generalize to other styles after learning and rely upon low-level pixel information to compute features, such as the color intensity.



(a) Rasterized floor plan. (b) Segmented walls.

Figure 2.6: Example of segmented walls from a floor plan image.

While a wide variety of learning algorithms have been presented in recent years within floor plan analysis research, those that have achieved state-of-the-art results come with the development of DL technology, especially neural networks [86]. In this way, the role of learning-based models has expanded as graphic separation can be omitted from raw plan images, and rule-based recognition rules were abridged, as models were trained to infer them directly from a broad variety of styles [10]. Among DL models, object segmentation is one of the tasks that has led research in computer vision recently [97] and can be formulated as a classification (semantic) or partition problem (instance). Semantic segmentation performs pixel-level labeling with a set of object categories for all image pixels, such as wall, window, or room, by identifying the spatial feature of the object and reflecting it in the results. Meanwhile, instance segmentation extends the classification scope further by detecting and delineating each object of interest in the image [148]. As an example, Figure 2.6 illustrates the segmented walls of a floor plan image, where it can be noticed that results are subject to noise and other artifacts, making the recovery of, for instance, the polygon or the precise contour shape, a non-trivial task.

In the following subsection, the proposed DL models are revised, explaining their approaches in floor plan analysis to recognize, classify, and vectorize structural objects and rooms. Table 2.3 resumes all learning-based works, considering the datasets used and the four categories of tasks, such as (1) *Graphics separation*, (2) *Object recognition*, (3) *Vectorization*, and (4) *Structural modeling*.

Table 2.3: Learning-based research, sorted by year, considering its tasks and datasets used.

Reference (year)	**Dataset (number of used plans)**[a]	**Strategy**	**Aug.**[b]	**G. Sep.**[c]	**Object recognition**				**Vect.**[f]	**Mod.**[g]
					Wall	**Door/W.**[d]	**Room**	**OCR/Dim.**[e]		
[131] (2011)	CVC-FP (25)	And-Or graph, predefined rule	–	–	✓	✓	✓	–	–	✓
[133] (2011)	CVC-FP (90)	BOVW	–	✓	✓	–	–	–	–	–
[134] (2013)	CVC-FP (100)	SVM-BOVW	–	✓	✓	–	–	–	–	–
[54] (2014)	CVC-FP (122)	SVM-BOVW	–	✓	✓	✓	✓	–	–	✓
[135] (2014)	CVC-FP (122)	SVM-BOVW	✓	✓	✓	–	–	–	–	–
[136] (2015)	CVC-FP (122)	Stochastic attributed graph grammar	–	✓	✓	✓	✓	–	–	✓
[80] (2017)	R-FP (500), CVC-FP (122)	FCN-2s, Faster R-CNN	–	–	✓	✓	–	✓	–	–
[1] (2017)	R2V (770/100)	CNN, modified ResNet-152	✓	–	✓	✓	✓	–	✓	–
[149] (2018)	LIFULL (1635/500/500)	FCN	–	–	✓	✓	✓	–	–	✓
[150] (2018)	Defined in paper (100/15)	Pix2PixHD	–	–	–	–	✓	–	–	–
[3] (2018)	EAIS (255/35/35)	U-Net + PixelDCL	–	–	✓	✓	–	–	–	–
[151] (2018)	Defined in paper (135)	Faster R-CNN	✓	–	–	✓	–	–	–	–
[138] (2018)	Defined in paper (800/200)	Predefined rule, VGG-16, MLP	–	✓	✓	–	–	–	–	–
[152] (2018)	LIFULL (20140/2000)	Multi-task VGG-16	–	–	–	–	✓	–	–	✓
[49] (2019)	R2V (715/100), R3D (179/53)	VGG, RCF, DeepLabV3+, PSPNet	–	–	✓	✓	✓	–	–	–
[82] (2019)	CubiCasa5K (4200/400/400)	Modified ResNet-152	✓	–	✓	✓	✓	–	✓	–
[28, 153] (2020)	EAIS (247/25/47), R-FP (500)	DeepLabV3+	✓	–	✓	✓	–	–	✓	✓
[83] (2020)	BTI (700)	U-Net + PixelDCL, Faster R-CNN	✓	✓	✓	✓	–	–	✓	–
[52] (2020)	PFP (1514/40)	U-Net, ResNet, Transformers	✓	✓	–	–	–	–	✓	–
[53] (2020)	CVC-FP (90)	α-shape, linear regression	–	✓	–	–	✓	–	–	–
[97] (2020)	Korea LH (2400/1030)	DeepLabV3+	✓	–	✓	✓	✓	–	–	–
[8] (2020)	CubiCasa5K (480/60)	FCN-2s, DeepLabV3+	–	–	✓	–	–	–	–	–
[27] (2020)	CVC-FP (122)	Mask-R-CNN	✓	–	✓	✓	✓	–	✓	–
[154] (2020)	Defined in paper (3500/500/1000)	YOLOv3	–	–	–	✓	✓	–	–	–
[50] (2020)	R2V (815), R3D (232)	GAN	–	–	✓	✓	✓	–	–	–

[a] Format: (total), (train/test), (train/val/test) [b] The dataset considered data augmentation [c] Graphical separation [d] Door/Window/Furniture/Others [e] OCR or Dimensions were recognized [f] Vectorization [g] Modeling (Graph, other)

Table 2.3: Learning-based research, sorted by year, considering its tasks and datasets used (continuation).

Reference (year)	Dataset (number of used plans)[a]	Strategy	Aug.[b]	G. Sep.[c]	Object recognition				Vect.[f]	Mod.[g]
					Wall	Door/W.[d]	Room	OCR/Dim.[e]		
[10, 64] (2021)	EAIS (400/50), CVC-FP (122)	Pix2Pix, multi-task DL	✓	–	✓	–	–	–	✓	–
[71] (2021)	ZSCVFP (8800/2000)	EdgeGAN, GNN	–	–	✓	–	–	–	✓	✓
[103] (2021)	RUB (74)	GNN	–	–	–	✓	–	–	–	✓
[68] (2021)	CubiCasa5K (200/200), Defined in paper (7)	GNN	✓	✓	✓	✓	✓	–	✓	✓
[85] (2021)	RFP (5600/1400), R3D (232), CubiCasa5K (5000)	YOLOv4, DeepLabV3+, FCN	–	–	✓	✓	✓	✓	✓	–
[155] (2021)	CubiCasa5K (5000)	Cascade Mask-R-CNN	–	–	✓	–	✓	–	–	–
[156] (2021)	LIFULL (3800/500/500)	DeepLabV3+	–	–	✓	✓	✓	–	–	✓
[141] (2021)	CVC-FP (122), R3D (215)	Bagging SVM PU-Learning	–	–	✓	–	–	–	–	–
[132] (2021)	Defined in paper (–)	Type-I FRBS	–	✓	✓	–	–	–	–	–
[145] (2021)	Defined in paper (–)	Interval Type-2 FRBS	–	–	✓	–	–	–	–	–
[37] (2021)	ROBIN/REDA (5610)	Predefined rules, Faster R-CNN, YOLO	✓	✓	✓	✓	–	–	–	–
[140] (2021)	Defined in paper (30/30)	Predefined rule, TensorFlow Object detection API	–	–	✓	✓	✓	–	✓	✓
[86] (2021)	RuralHomeData (700/100), R2V (770/100), CubiCasa5K (800/100)	VGG-16, U-Net, SSD	–	–	✓	✓	✓	✓	–	✓

2.2.3.2 Deep learning models

Among deep learning techniques (DL), Convolutional Neural Networks (CNN) have been widely employed within floor plan analysis to automatically extract advanced features, enhancing the recognition of several structural objects [85]. CNNs are a standard supervised learning algorithm, generally used in computer vision due to their intrinsic relationship with two-dimensional tensor processing, such as the pixel matrix of an image [157]. CNNs have a topology composed of convolutional layers, non-linear processing units, and sampling layers. The first one applies a convolution operator on the input through a kernel matrix (also known as filters), transforming the data so that certain features become more dominant in the output. The kernel matrices, commonly used in image processing, can be manually defined to perform different tasks such as edge detection, blurring, or contrast change; however, those trained in a CNN model extract more abstract non-trivial features. The convolutional layers' output is later assigned to a non-linear processing unit (activation function), which helps in the abstraction capacity while learning and provides non-linearity in the feature space, generating diverse activation patterns for different responses, facilitating the learning of semantic differences between the data. The activation function output is usually followed by a sampling layer (subsampling or oversampling), summarizing the results, and keeping the input invariant to geometric distortions [22].

CNNs have had a significant adoption for detection, segmentation, classification, generation, and image recovery tasks [158]. For such reasons, they have been widely used to exploit new features hard to capture considering manual rules, as exemplified in Figure 2.7. Although CNNs have proved to be powerful in image classification and segmentation, they have two main disadvantages. First, there is a lack of interpretability of how the model works for end-users [159], and training requires a lot of labeled data for the models to be capable of generalizing correctly [132]. Thus, the development of such procedures led the research community to create new, large-scale datasets, which started to be publicly published after the first works tackled CNNs (2017), as shown in Table 2.1.

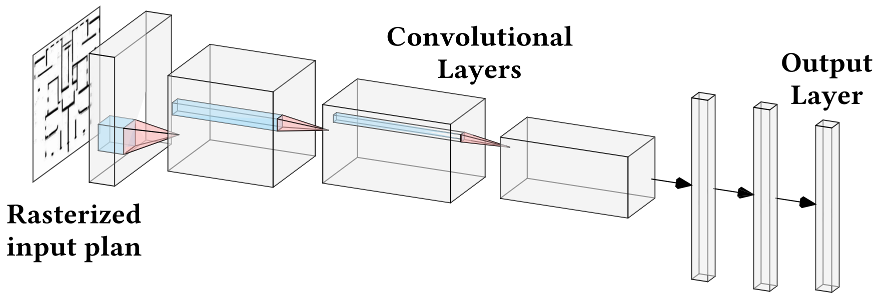


Figure 2.7: Generic CNN-based model that automatically retrieves features from a rasterized plan, for example, to segment walls or classify its objects.

Within DL, models can be discriminative or generative-based. Discriminative models (section 2.2.3.2.1) learn the conditional probability distribution of the classes (e.g., wall or background), that is, the decision boundary, to make predictions on the unseen data in tasks such as classification, regression, or segmentation; therefore, their ultimate objective is to separate one class from another. Conversely, generative models (section 2.2.3.2.2) learn the joint probability distribution, that is, the distribution of the individual classes in a dataset, to return a probability for a given example. Generative learning algorithms tend to model the underlying patterns or distribution of the data points, and, unlike discriminative models, they are also capable of generating new data points. Figure 2.8 illustrates the explored DL models within floor plan research, which are detailed in the following paragraphs.

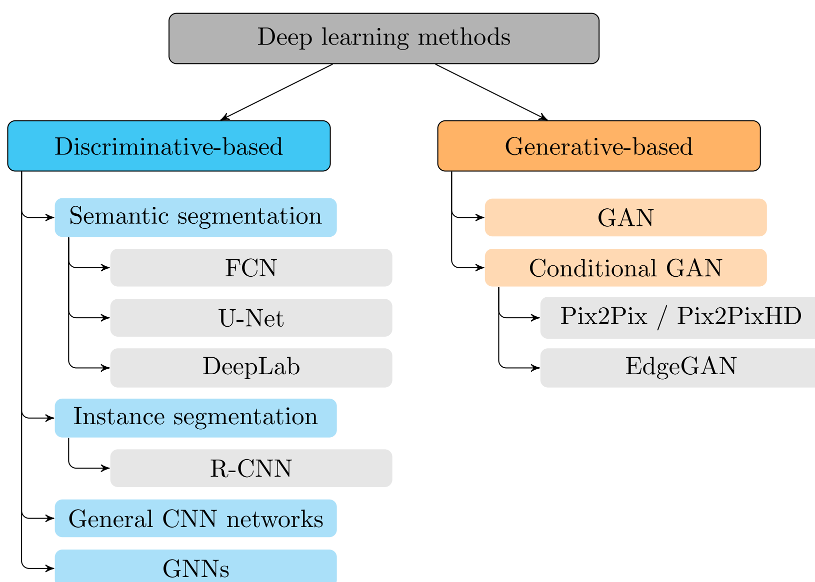


Figure 2.8: Deep learning methods explored within floor plan analysis research.

2.2.3.2.1 Discriminative-based models

Among discriminative-based models, the semantic segmentation FCN [146], U-Net [51], DeepLab [160], and instance segmentation model R-CNN [161] have been used. FCNs or Fully Convolutional Networks are composed of two main sections: encoder (contraction) and decoder (expansion). The encoder section is used to capture the context of the image. It comprises several convolutional and max-pooling layers, which reduce the input image size by subsampling with kernel stride, capturing finer grain structures from the input image as they have a smaller receptive field [149]. In opposition, the decoder section comprises many feature channels that enable precise localization through the transposed convolutions, propagating context information to higher resolution layers, giving the segmented output from the generated classification feature maps.

Similar to FCNs, in U-Net (Figure 2.9), the decoder also combines the feature and spatial information through a sequence of up-convolutions and concatenations with high-resolution features obtained from the encoder, improving localization and reconstruction of the segmented output image while keeping the underlying structure. Therefore, the expansive path is symmetric to the contracting part, yielding a u-shaped architecture [51]. Likewise, DeepLab is a semantic segmentation model which employs a pre-trained CNN to get encoded feature maps from the input and a decoder to reconstruct the segmented output image. Among their different versions, DeepLabV3+ has achieved state-of-the-art results, famous for its stacked atrous (i.e., dilated) convolutions, enlarging the kernel's field-of-view to extract long-distance features. Finally, the instance segmentation R-CNN is a family of models which produces a set of bounding boxes for each object in the image, referred to as Regions of Interests (ROIs), where the position and category (e.g., wall) are inferred using neural networks.

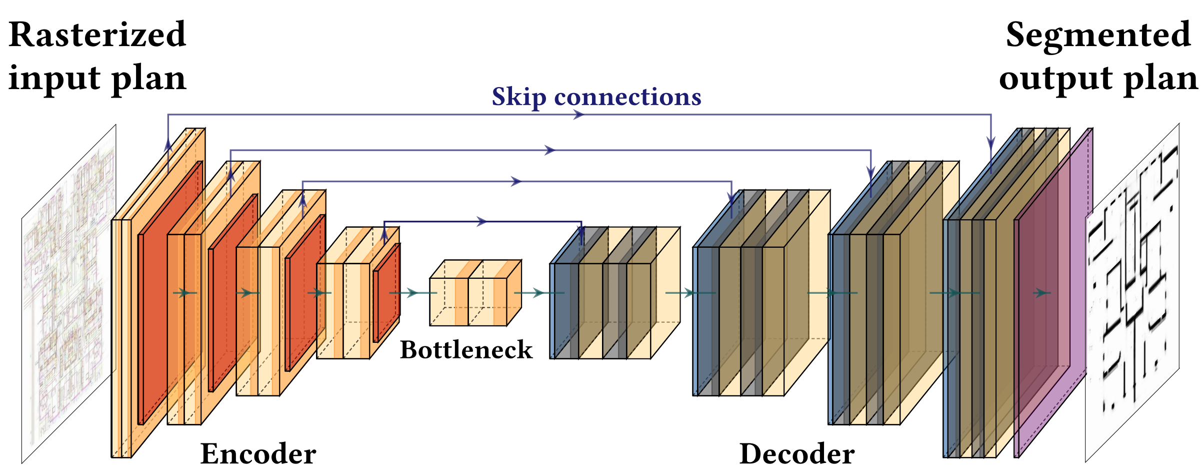


Figure 2.9: A U-Net model which segments the walls from a rasterized floor plan image. Layer legend: *(yellow)* convolutional block, *(orange)* max-pool, *(blue)* up-sampling, and *(purple)* softmax.

Concerning the discriminative semantic segmentation problem in floor plan analysis, Dodge *et al.* [80] were the first to propose an FCN-2s model to segment walls and Faster R-CNN to detect objects such as doors, among other five classes. They also implemented OCR to recognize the room size and place furniture scaled to the scene; the wall segmentation experiments conducted in Dodge's work demonstrated the superiority of a CNN-based approach compared with some traditional patch-based models that use standard shallow classifiers like support vector machines [82], while also proving that CNNs can handle various drawing styles. Yamasaki *et al.* [149] also presented a fully convolutional end-to-end FCN network to label pixels of 12 different object classes. For this purpose, a semantic segmentation was performed, taking as input the images of apartment floor plans, in which spatial relations between elements and room boundaries were ignored; the classified pixels formed a graph to model the structure and measure the structural similarity for apartment retrieval.

A U-Net approach was introduced by Yang *et al.* [3], where the authors also employed the pixel deconvolutional layers PixelDCL [162] to avoid checkerboard artifacts while segmenting walls and doors. This work was extended by Surikov *et al.* [83], who detected objects with the Faster R-CNN model and proposed statistical methods to vectorize walls, doors, and windows. Morphological operations were used to remove border defects, component filtration to remove connected objects, and the Ramer-Douglas-Peucker algorithm to extract and simplify the room contours. Egiazarian *et al.* [52] obtained the line primitives from floor plan drawings, using U-Net for pre-processing (to eliminate background, imperfections, and fill missing parts); then, the resulting images were split into patches to independently estimate the line and curve primitives with a feed-forward Artificial Neural Network (ANN). Each patch is encoded with a ResNet-based feature estimator [163] and decoded using Transformer blocks [164] that allow for varying the number of output primitives per patch. Predicted primitives were later refined and aligned to the raster image through an optimization procedure. Finally, Lu *et al.* [86] adopted a joint deep neural network approach to extract elements and text simultaneously from an architectural floor plan image, whose were also split into patches for overcoming information loss due to downsampling. A VGG-16 encoder was considered to get a common feature map and extract latent features of the input image. Then, a U-Net model was used to predict the mask and class of architectural elements, and a pre-trained fast Single Shot Detector (SSD) [165] was considered to retrieve the bounding boxes of room types' text. Predicted per-pixel classes were optimized to remove boundary noise and assign unlabeled adjacent ones, for example, in pixels belonging to a wall that was blurred or partitioned into smaller but connected elements. These classes then fed a mixed-integer quadratic programming algorithm to designate a rectangle for each room beside its type recognized by OCR, leading to the assembly of a room layout graph and the 3D reconstruction of the building.

DeepLab semantic segmentation models have also been widely used among deep learning approaches. Jang *et al.* [28, 153] segmented walls and doors using the DeepLabV3+ model; centerline [166] and corner [167] algorithms were proposed to vectorize the walls and doors, leading to the assembly of a node-edge graph to describe their position, connectivity, and thickness obtained by a moving kernel method. Seo *et al.* [97] also used DeepLabV3+ to recognize walls, windows, doors, and room types from eight classes; data augmentation techniques were further studied to improve the model results in terms of the IoU metric. Yamada *et al.* [156] conducted semantic segmentation with the DeepLabV3+ model to recognize objects from 14 classes, which was later used to assemble a graph in a rule-based procedure for apartment retrieval. Nodes were created by extracting regions with a particular area, and edges were created between rooms adjacent to the same door or directly adjacent to each other. Finally, Zhu *et al.* [8] compared different training strategies to parse complex floor plans considering the FCN-2s and DeepLabV3+ models for wall segmentation, with VGG-16 as a backbone.

Within instance segmentation models, Faster R-CNN (Figure 2.10) and YOLO, as well as other anchor-based frameworks, have been used to detect the building elements, as these propose and combine numerous boxes to detect and classify the objects, such as walls, doors, or windows. However, if these general-purpose frameworks are used without further post-processing, the ground-truth inflated boxes and the lack of suitable annotation to describe the complex geometrical characteristic of architectural primitives lead to problems in the localization of sloped and curved walls. Thus, instance segmentation models can only replace some modules of the conventional pipeline. Anchor-free frameworks, such as CenterNet [168] and CornerNet [169], cannot solve this problem either. For such reasons, only anchor-based frameworks were explored within floor plan analysis [71].

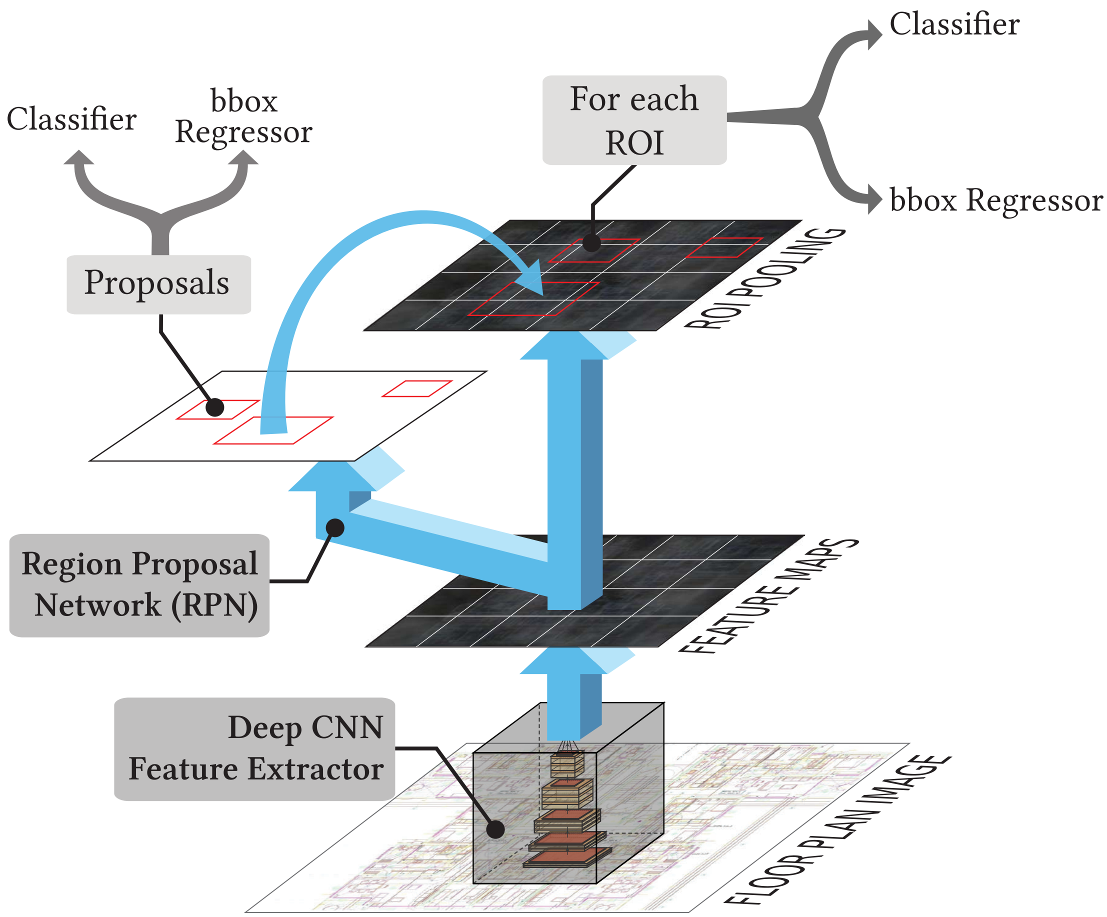


Figure 2.10: Instance segmentation Faster R-CNN model [69] that considers a floor plan image as input and predicts the position of the objects inside region proposals.

From anchor-based frameworks, Wu *et al.* [27] used Mask-R-CNN [170] to vectorize the walls by finding a rectangle proposal representing each segment's width, thickness, angle, and location. After simplifying and merging the proposals, an optimization model adjusts its vertex coordinates to resolve inconsistencies from adjacent rectangles such as overlaps and gaps conform to the topological constraints. Although the complex wall layout was described as simply connected segments, the rectangle-based modeling is able to reduce the shape complexity of the segmented regions and can represent the polygons with high accuracy while retaining the connection topology [21]. Murugan *et al.* [155] segmented walls and rooms using the Cascade Mask R-CNN model [171]; wall corners were also detected with a Keypoint Mask R-CNN to improve results after post-processing. The YOLOv3 model [172] was employed by Wang *et al.* [154] to detect doors and windows, alongside the classification of eight types of rooms with the C4.5 decision tree. C4.5 is a tree-like structure method that minimizes the measure of entropy (or impurity) by separating the dataset into smaller classes. Conversely, Khade *et al.* [37] proposed a scale-invariant algorithm to remove doors, segment walls, and trace the outer shape of the floor plan for Content-Based Image Retrieval (CBIR). Furniture objects from 12 different classes were also detected and classified, wherein Faster R-CNN has a better performance than the YOLO model.

Recently, Lv *et al.* [85] presented a framework that combines the multi-modal information of the floor plan, such as room structure, type, symbols, text, and scale, to recognize and reconstruct its elements. The anchor-based model YOLOv4 [173] is employed to detect the ROIs alongside the text, number, and symbols containing semantic and contextual information like room types, dimensions, or areas. Twelve object classes, and the endpoints of doors, windows, and doorways, were extracted with the DeepLabV3+ [160] model. In terms of the model training, the affinity field loss [174] was used to incorporate structural reasoning into semantic segmentation, despite the standard cross-entropy loss that lacks spatial discrimination ability to distinguish between similar or mixed pixels, outperforming previous works [1, 49]. Scale calculation was also implemented to retrieve the size of each object; for such an aim, dimension lines were detected by obtaining its endpoints with a modified FCN network, and matched with the recognized length texts by YOLO. Finally, a room vectorization algorithm was proposed that considered room contour and wall centerline optimization, leading to the 3D reconstruction of each floor plan image.

Some works do not consider a segmentation pipeline but rather propose CNNs to capture spatial features to reconstruct the objects. For instance, Liu *et al.* [1] introduced a deep learning CNN model to vectorize the plans. The pixel-wise semantic ResNet-152 network was applied to detect junction points of interior and exterior walls, considering a Manhattan assumption, that is, it only can recognize horizontal or vertical walls due to the use of a template matching technique. These detected objects fed an Integer Programming (IP) method to construct the vector data by finding the optimal primitive pair that correctly represented walls and openings such as doors or windows, leading to the assembly of the rooms. Despite their drawbacks, the major finding was that deep neural networks could act as an effective precursor to the final post-processing heuristics to restore the floor plan elements, including their geometry and semantics. Liu's work was further extended by Kalervo *et al.* [82], who also proposed a modified ResNet-152 model to detect wall junctions, rooms, and icons, obtaining better results as they applied a trainable module [175] for tuning the relative weights between the multi-task loss terms; similarly, these outputs were employed to vectorize the floor plan. Another example is Zeng *et al.* [49], who proposed a deep multi-task neural network to predict room-boundary objects (walls, doors, or windows) and room types. A shared VGG encoder [139] was used for feature extraction and two separate VGG decoders to perform both tasks, recognizing individual elements considering their spatial relationship and a room-boundary guided attention mechanism to enhance the pixel classification performance of the floor plan image. The results were compared against the Richer Convolutional Features (RCF) edge detection model [176], DeepLabV3+, and PSPNet [177] segmentation networks, obtaining better results.

Graph Neural Networks (GNN) have also been studied to model and classify the floor plan objects, describing a way to express the nodes' order and connectivity learned from the dataset structure [178, 179]. GNNs have undergone rapid development in recent years as convolution was introduced to update the latent node vector (Graph Convolutional Networks, GCN) or by studying graph operations such as aggregation or combination powered by deep neural networks [180]. Like other DL models, GNN extract and compares a unique embedding vector of each entity in the target dataset to predict a result as close as possible to the label data [68]. The scope of application for GNN varies, including nodes, edges, graphs, and subgraphs, and has been widely applied in the area, for example, to generate floor plans [35] or for architectural symbol-detection tasks [39].

Among GNN approaches, Simonsen *et al.* [102] implemented a GNN-based model to classify the nodes of a large rasterized CAD image as door or non-door. On the other hand, Song and Yu [68] developed a framework to vectorize the floor plan objects considering a GNN for object classification. First, a pre-processing task erased texts and binarized the raster plan; the processed image is then vectorized, relying on its closed regions, and converted to a region adjacency graph according to their adjacent relationship with neighboring polygons. The graph is then fed to an inductive learning-based GNN, which compares multiple floor plan graphs and performs node classification by analyzing inherent features and the relationships, such as the distance. Despite its good performance while classifying elements, the proposed GNN approach, unlike those CNN-based, is not robust to noise and resolution changes.

2.2.3.2.2 Generative-based models

Ever since Goodfellow *et al.* [181] presented the Generative Adversarial Network (GAN) in 2014, there has been tremendous development in generative models and neural style transfer [182, 183]. By providing training data in pairs, the algorithm finds the most suitable parameters in the network so that the discriminator has the least potential to distinguish the generated data from the original one [150]. GAN has sprouted many branches, including conditional GAN [184, 185], Wasserstein GAN [186], or Pix2Pix [187], and has been used successfully in image translation, style migration, denoising, superresolution and repair, image matting, semantic segmentation, and dataset expansion [188, 189].

From related work, one of the GAN applications is for recognizing structural objects. Zhang *et al.* [50] created direction-aware, learnable, and additive kernels to optimize the recognition of complex and irregular walls through the context module and convolutional blocks of a multi-task GAN-based neural network, improving accuracy and segmentation results of the objects (wall, door, window, and rooms). Despite this example, most researchers considered GANs for image style transfer, as it offers the capability to homogenize the level of detail from varied types of drawings, leading to the recognition of primitives from complicated and overlapping graphics.

Recently, image style transfer models have improved remarkably with the development of GANs; among them, the deep networks such as Conditional GANs (cGAN) [184, 185], CycleGAN [190], and DiscoGAN [191] have gained a great reputation. cGANs and CycleGAN transfer images into different styles while preserving the underlying structure, whereas DiscoGAN focus primarily on their texture [64]. The cGAN model assumes that labeled pairs exist within the dataset, turning the original generation process into a conditional one.

In this aspect, labeled data, such as one-hot vectors, 2D images, or even 3D models, provides hints to guide the training process; once it runs toward an unexpected direction, punishment will be given to correct its tendency according to the additional information [150]. Thus, cGAN learn the forward mapping, that is, $y = G(x)$, where x belongs to the input, y to the output, and G to the generative model. Conversely, CycleGAN and DiscoGAN aim to transfer the style between domains even when their images are not paired [64], learning from a two-cycle mapping, i.e., $x = F(y') = F\Big(G(x)\Big)$ and $y = G(x') = G\Big(F(y)\Big)$, with the input x and output y unpaired. Although CycleGAN have a wider range of general-purpose applications [71] as they do not require a pixel-level annotation for the images, which can be extremely expensive, the lack of large-scale datasets imposes a difficult restriction for its usage within floor plan analysis. Therefore, only conditional GANs have been used so far.

One important milestone of GAN for image translation is the Pix2Pix model introduced by Isola *et al.* [187], developed from cGAN [185] using an encoder-decoder architecture for the generator, for example, the U-Net model. Pix2Pix was designed to become a general-purpose solution to translate an image between two domains with the same settings, corresponding, in other words, to a pixel-by-pixel mapping. For instance, Isola's group originally employed Pix2Pix to generate: (1) a real photo from a partly-damaged one, (2) a colorful map from a black-and-white map, and (3) an image with texture and shadow from a linear sketch [150]. Based on Pix2Pix, Wang *et al.* [192] presented Pix2PixHD, expanding its capabilities to handle high-resolution image synthesis and semantic manipulation (from original 256×256 to 2048×1024 pixels) by introducing a new robust adversarial learning objective together with new multi-scale generator and discriminator architectures [71].

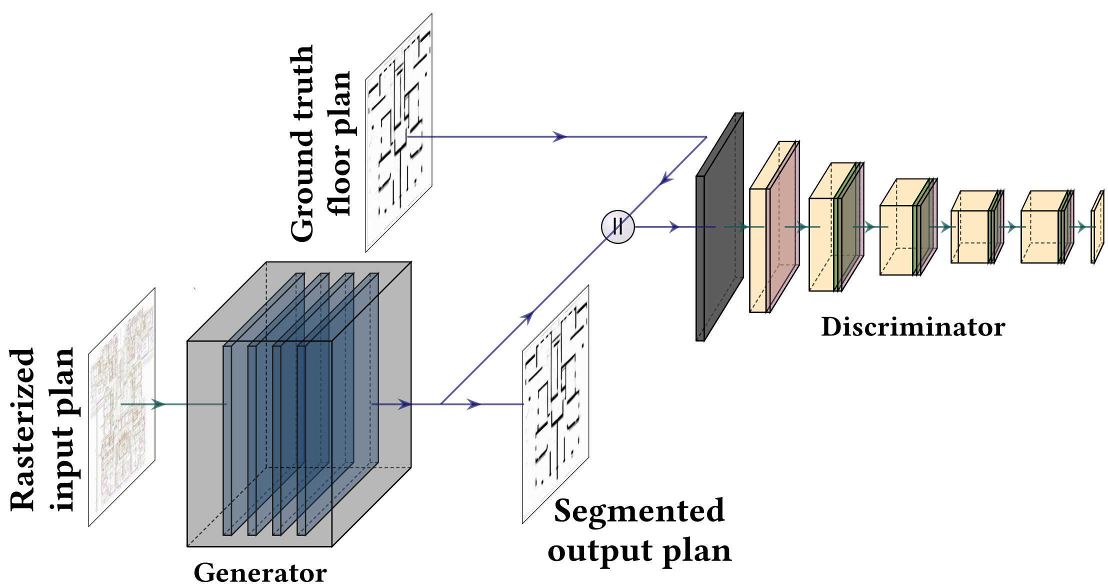


Figure 2.11: Pix2Pix model that translates the rasterized floor plan image style into a segmented format.

Concerning the recognition and generation of floor plans, Huang and Zheng [150] introduced an application of Pix2PixHD [192] to detect rooms from 8 classes and then colorize them to generate a new image. In this example, the cGAN translate the raster plan to a segmented style using annotated pairs, classifying each pixel while preserving the image's underlying structure. Pix2Pix was also adopted by Kim *et al.* [10, 64] to transform plans into a unified format [187]; in their study, a multi-task DL network transferred the style and simultaneously extracted the wall junction features (Liu *et al.* [1]), considering a Manhattan assumption. These outputs were used to assemble the wall's vector format through a combinatorial optimization that represents a structure similar to the style-transferred plan, while satisfying the semantic constraints from the floor layout.

Finally, Dong *et al.* [71] developed an edge extraction GAN, named EdgeGAN, to detect walls based on Pix2Pix. EdgeGAN projects the floor plans into a Primitive Feature Map (PFM); each channel contains some lines representing one category of primitives, leading to the vectorization of walls in an end-to-end manner. Two inspection modules were also proposed to check the connectivity and consistency of PFM based on the Subspace Connective Graph (SG). The first module contains four criteria that correspond to the sufficient conditions of a fully connected graph. The second module classifies the category of all subspaces via one single graph neural network, which should be consistent with the text annotations in the original floor plan.

2.3 Challenges and opportunities

Automatic floor plan analysis has witnessed remarkable progress over the last few years with the help of DL models. Numerous innovative concepts have emerged, such as identifying wall joints to facilitate plan vectorization, utilizing generative image-to-image networks for plan conversion into a standardized format, or employing tailored loss functions to incorporate structural reasoning during segmentation. These novel ideas have contributed to enhancing recognition metrics compared to traditional rule-based models. However, despite this progress, certain challenges remain unresolved. In the subsequent paragraphs, we outline and deliberate upon these challenges, offering insights that can steer future advancements in the domain.

Standardization of result analysis. Although each reviewed article contemplated the evaluation of its models, the lack of a standard procedure makes it difficult to compare with other similar works. Several metrics have been used even to check the results of the same tasks, and many use custom ones that fit their specific purposes. Table 2.4 groups the typical metrics used throughout reviewed works; typically, segmentation results were evaluated in terms of the intersection over union (IoU) [146], pixel/class accuracy, and the Jaccard Index (JI) proposed by de Las Heras *et al.* [54]. By contrast, works that detected objects (e.g., walls, doors, windows) used the mean average precision (mAP), the recall & precision, the match score (MS), detection rate (DR), and recognition accuracy (RA) [193], or considered a confusion matrix.

Besides the multiple evaluation metrics used, there is no shared annotation for complicated floor plan datasets, which are fundamental barriers to compare learning-based approaches [10]. Private datasets, popular in the last few years, further complicate this issue [145]. Thus, there is an urgent need to standardize how analysis is performed on each task. A common metric, which also requires a standard representation of the plan annotation, allows for comparing the models and choosing the one with better results for a particular plan style and task.

Table 2.4: Common metrics used to evaluate floor plan results.

Evaluation metric	Article
Intersection over Union (IoU)	[3, 8, 28, 49, 50, 52, 80, 82, 83, 85, 86, 97, 102, 132, 145, 156, 194]
Pixel/Class Accuracy	[1, 3, 49, 50, 53, 54, 68, 71, 80, 82, 85, 86, 102, 138, 140, 149, 152, 154–156]
Jaccard Index (JI)	[2, 11, 80, 134, 135, 141]
Mean Average Precision (mAP)	[37, 83, 86]
Precision	[27, 49, 102, 151, 152, 154, 155, 194]
Recall	[1, 27, 49, 82, 102, 134, 135, 140, 151, 152, 155, 194]
Match Score (MS)	[10, 27, 54, 62, 119, 122, 136]
Detection Rate (DR)	[9, 10, 62, 119, 122, 136]
Recognition Accuracy (RA)	[10, 62, 119, 122, 136]
Confusion Matrix	[2, 28, 71, 150, 152, 154]

New public datasets. Like in many other computer vision tasks, datasets play an essential role within automatic floor plan analysis. These documents define the geometrical, topological, and semantical information of plan objects in a highly correlated fashion, following strict restrictions such as usability, layout, and regulation [2, 3]. New datasets can provide researchers with more possible styles for the models to handle, especially if future learning-based methodologies are toward a style-independent trend.

Another major problem regarding datasets is that most current public ones consider only houses or apartments (Figure 2.2); however, it is usual for architectural and structural engineering offices to design and process multi-unit plans, which incorporate multiple apartments yielding the entire shape of the building, accounting also for the corridors, staircases, parking lots, and common areas, making them diametrically different from conventional single-unit plans due to a massive scale change. Incorporating such samples can expand the scope of floor plan analysis, enabling the process of different sources among the industry. Figure 2.12 illustrates a sample of a multi-unit raster floor plan; unlike those presented in the reviewed datasets (Table 2.1 and Figure 2.2), this plan has complex walls, more furniture, and new semantics. For example, some walls separate two rooms of different apartments, which together constitute the perimeter of the building. This new level of complexity is not explicit, but it is only apparent when processing the plan as a whole.

Figure 2.12: Example of a rasterized multi-unit floor plan [21].

Non-supervised DL models. Annotating floor plans is difficult and expensive. There is no standard notation, and some examples offer ambiguous situations that are difficult even for experts [9, 54]. For example, consider a plan in black-and-white where walls and beams have the same annotation; in such a case, these objects are only differentiable considering a structural perspective. Hence, non-supervised models enable analysis without annotating the floor plans. Currently, few works have proposed unsupervised methods [11, 135, 136]; nevertheless, they fall into strict assumptions or rely on complex learning rules. In this sense, DL can help to learn these relationships and structural reasoning for recognizing new complex objects for upcoming plan styles and can enable the analysis of unexploited datasets.

Combination of rule-based and learning-based methods. While learning-based algorithms have revolutionized the plan recognition area, they still have problems in solving tasks that require a deterministic response or fine-grained detail. Therefore, the combination of rule-based models and learning can offer the best of both worlds. The former solves fine details that are difficult to capture by a DL model because they are infrequent or require a high refinement level, such as polygon resolution, the detailing of certain sections of complex geometry, or the recognition of custom objects. The latter allows for solving common problems that require specific rules such as segmentation or vectorization. Both mechanisms are not mutually exclusive and can be leveraged.

Trending applications within the industry. As floor plans are one of the key products in architectural firms or structural engineering offices, the algorithms that can analyze and process them in batches have many applications, as they allow for automating pipelines in recognition, vectorization, modeling, or searching in large databases.

One of the most active research areas belongs to BIM and 3D reconstruction, as these technologies help to improve productivity and reduce costs in different stages of the building lifecycle, especially in the early ones, requiring less paperwork to visualize or edit the projects. Also, in recent years, governments and private companies have started a more data-driven approach because models are composed of several elements that contain information about their properties and relationships with others, facilitating interdisciplinary work [195]. Despite benefits, BIM and 3D models are costly and time-consuming to produce [13], particularly if the only available documentation is 2D scanned images of their paper floor plans [132]. Therefore, one critical short-term research challenge in the renovation scope is to devise effective and reliable methods and tools to reconstruct the digital models of existing buildings [2]. In BIM and 3D reconstruction areas, algorithms have been developed since the early 2000s to recognize and vectorize building shapes from walls, beams, and slabs [111]. Zhao *et al.* [15] recently implemented a framework to assemble a BIM representation from CAD files, employing revised DL techniques such as Faster R-CNN and YOLO. In this aspect, the accuracy and generalization ability to process plans in different styles are critical aspects of research.

Indoor data models, maps, and spatial information is another widely studied area of application, with a globally growing market that is predicted to expand from $2.6bn in 2017 to $43bn by 2025 [102]. For such reason, research has been conducted on generating indoor spatial information from various data such as LiDAR (Light Detection and Ranging), BIM, and 2D floor plans [28]. As rasterized floor plans are more accessible compared to other sources but discard semantic and topological metadata [8], the revised algorithms of this review can be employed to re-assemble this representation by spotting, retrieving, and vectorizing the foundational components of indoor maps automatically, like doors, walls, corridors, and furniture, or by detecting the usage of rooms, avoiding time-consuming human labor. Naturally, automatic procedures come with several challenges, mainly caused by the diversity of floor plans and the flexibility of preferences in visual styles, symbols, and topology [27].

Building search retrieval is another example of an application with growing interest. Because of the increasing demand for apartment search, the emergence of online platforms has made this task easier. Nevertheless, most only provide information regarding location, monthly rent, or room size, but little information on plan structure [149], which turns the searching process into a tedious task [37]. For such reasons, the research community has developed several tools to simplify this process. Examples include the use of graphs to query similar floor plans among large databases [12, 37, 38], search based on hand-made sketches [41–43], the use of natural language to describe the plan layout [24], audio feedback mechanisms to help visually-impaired people navigate floor plans [45], the development of VR experiences for customers to explore real estates [25], or the use of AI networks for valuate them [44]. Due to the massive amount of data and the variety of styles, reviewed machine learning solutions, like CNNs and GNNs, have been used extensively to describe, extract, and query the meaningful features of the floor plans, allowing developers to create accessible and easy-to-use tools for customers, enhancing the overall design experience.

Structural analysis is another area where floor plan research algorithms can be applied. New machine learning models can be trained to automatically assemble a structural floor plan from an architectural image, predicting new walls and computing its members' thickness, length, and displacement [22, 23]. For such reason, there is a considerable need for processed datasets that consider a wide range of architectural styles and layouts to train these algorithms. By this means, discriminative DL models, for example, R-CNN, can be employed to transform rasterized plans into a rectangle-based representation [21, 196] to compute features, avoiding expensive manual labeling. These upcoming solutions can simplify the decision processes, reduce costs, and improve productivity, while also adding value to the already manufactured plans, which can now be employed to develop data-driven models and improve the production lines of the structural engineering offices.

Chapter 3

Wall polygon retrieval and vectorization

Within the area of automatic floor plan analysis, learning-based methods have become increasingly popular in recent years due to their superior accuracy and generalizability compared to traditional approaches while processing rasterized floor plans. However, the scarcity of public raster datasets with various styles and sufficient quantity hinders the development of new models, as current ones only contain a single apartment or house, limiting the analysis of large-scale plans usually designed in architectural and structural offices. In order to address that issue, this chapter, which also has been submitted to *Automation in Construction* journal [197], presents a novel multi-unit floor plan dataset comprising 954 high-resolution images of residential buildings with annotated walls and slabs as polygons, enabling large-scale plan analysis. Additionally, we implement an automatic wall vectorization method that uses a learning discriminative-based semantic segmentation U-Net model to retrieve wall objects, followed by a deep learning model that predicts the segmented primitives, providing a baseline for future comparison of automatic wall segmentation results.

3.1 Dataset

3.1.1 Motivation

Among automatic floor plan methods, datasets have played a significant role because of the lack of standard notation while describing their constitutive elements in terms of geometry, composition, drawing method, and labeling; therefore, designed models have incorporated specific techniques to handle each particular plan style [58], mainly in rule-based approaches. The lack of design conventions within the industry is particularly remarkable in architectural plans, where in the best case, only 70% of the graphical information complies with a standard rule [72]. An example of the dataset variability might be found when describing wall segments; even though they can be represented with straight parallel lines, the color, line pattern, thickness, labeling (e.g., its construction material or dimensions), the polygon fill, and many other properties can vary, making their retrieval challenging to generalize. Additionally, the inherent nature of the documents further compounds this complexity, since each plan element intricately interacts with others, for example, in the structural relationships among walls and beams, the correlation between walls and non-structural components like doors or windows, or the inclusion of furniture within the plan's space.

Datasets require the annotation of objects such as walls or rooms to be considered helpful, depending on the context and those that meet the specific needs of each case study. For example, a structural engineering application might only recognize walls and slabs for assembling a digital building model [22], ignoring other objects like furniture. Regardless of the case, annotating floor plans, despite other document types, is a complex and expensive task, as it requires high-level expertise to recognize the different elements due to ambiguity in notation [9, 54]. Moreover, even though several practical tools have been developed to annotate them conveniently [74, 75], it is difficult to do so because there is no way to guarantee the same annotations from different experts, especially for complicated plans [10].

Although many datasets have been proposed so far in several styles (Figure 2.2), they typically consist of plans for apartments or houses. Nevertheless, it is usual in a production environment that the plan generated by the architectural or engineering office yields the complete layout of a building, named a multi-unit plan, which is a significant conceptual difference from the current data [58]. A multi-unit floor plan shows not only an apartment's geometric and topological layout but also the sum of all floor' apartments, their connections, and relationships, yielding an additional abstraction layer [58].

For these reasons, we present a novel multi-unit floor plan dataset of residential buildings, presenting new wall topological configurations by including architectural layouts that are not present in the current single-unit plans, such as parking lot walls, large spans of more than 10 meters, elevator shafts, corridors, entrance lobbies, among others. Our dataset was specifically curated to prioritize the recognition of walls, as they serve as critical landmarks for identifying other components, such as beams or openings like doors or windows [54]. In addition, walls are the main ones responsible for providing the torsional and lateral stiffness in the mechanical response of buildings [198]; therefore, they are essential not only from the information retrieval point of view but also from the underlying structural aspect of the automatic floor plan analysis.

3.1.2 MLSTRUCT-FP: A novel multi-unit floor plan dataset

The Machine Learning Structural Floor Plan (MLSTRUCT-FP) dataset comprises 954 colorized high-res multi-unit rasterized plan images from 165 Chilean residential building projects [21]. These buildings were designed by 52 different architectural offices between 2004 and 2018; thus, plans exhibit a wide range of drawing styles, icons, and custom labels, along with wall systems featuring a non-uniform distribution and complex cross-sections. This typology differs from the structural systems commonly found in countries like the US [199]; consequently, plans from the MLSTRUCT-FP dataset often comprise complex walls with redundant sections, usually placed for architectural reasons [198]. Figure 3.1 illustrates a selection of plan crops from the proposed dataset, where multiple line styles and wall types can be observed. For example, walls are represented within empty parallel lines of multiple colors and thicknesses (Cases *a, c-i, k, m*), filled with dashed lines or grids (Cases *b, j, l, n, o*); some have textured floor tilings (Cases *a, d, g, i, m*), furniture (Cases *a, d-g, i, o*), grid and axes lines (Cases *b, c, e, f, j, n, o*), dimension lines (Cases *b-o*), room labels (Cases *a, c, g-i, m*), and non-structural icons, such as door or windows (Cases *c-e, g-m, o*).

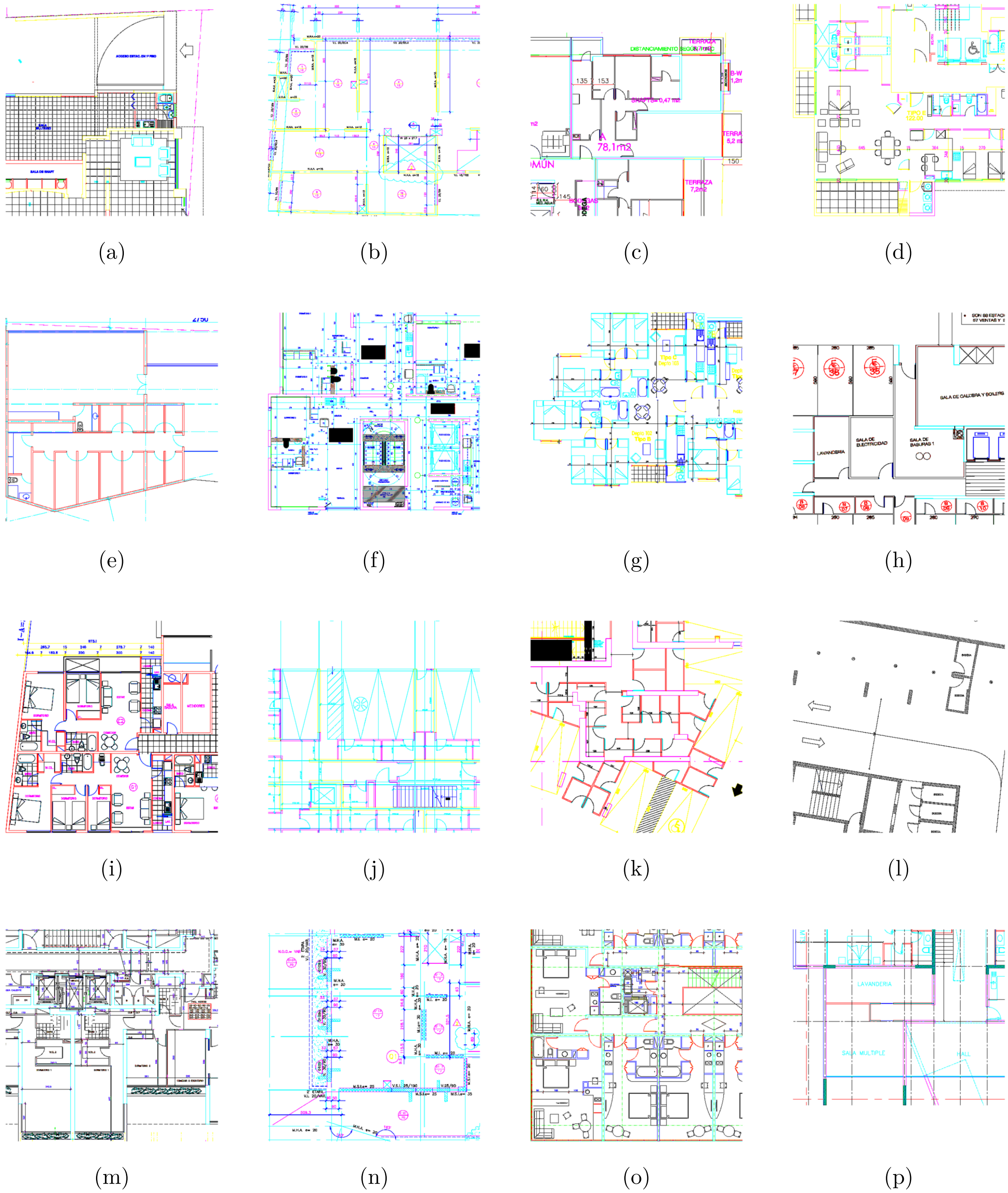


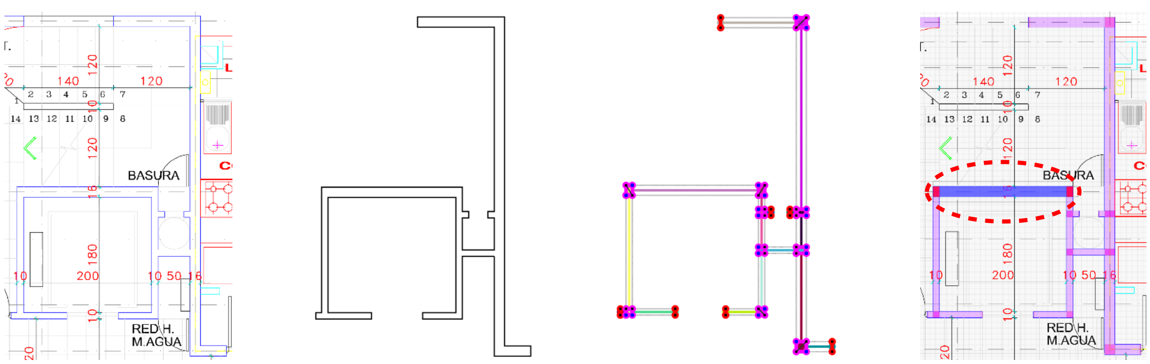
Figure 3.1: MLSTRUCT-FP floor plan examples.

The MLSTRUCT-FP dataset comprises the rasterized plan images and labels of the wall segments and slab contour polygons with dimensions in meters (m), so each image also has its scale in meters per pixel (m/px). The plans' images are stored in transparent PNG format, whose width ranges between 6360 px and 9650 px, with an average of 9450 px, and their height ranges between 6300 px and 9500 px, with an average of 6700 px. In contrast, wall and slabs labels are stored as polygon data within a JSON file, accounting for each object geometry (coordinate positions, length, thickness, scale) and topology (wall, slab, and floor

IDs), enabling the composition of complex hierarchical queries to obtain the distribution of these elements with different rules.

We considered wall labels because these objects are one of the most important regarding floor plan analysis [58], as they define the main layout of the building, delimit the room perimeter and convey essential information to detect other elements, such as doors, windows, or beams [11]. Also, wall recognition is helpful across the spectrum of architecture, engineering, and construction as they provide data for design, analysis, and cost estimation, among others [12]. The detailing of wall objects into basic rectangular segments was used because of three fundamental reasons [196]: (1) they allow a detailed description of the building geometry and topology; that is, they account for the exact member position and connection alongside the floor plan [76], (2) the simplicity regarding their computing implementation, and (3) their applicability in and optimization models, as rectangles can be represented using images within discriminative-based CNN [22] or as region proposal boxes (ROI) in instance segmentation models such as Faster R-CNN [69].

The assembly process of the floor plan dataset is detailed in Figure 3.2. First, the image of each plan was collected from the digital CAD model (Figure 3.2.a) alongside the wall contour polygon (Figure 3.2.b); plans were corrected because of layer issues, and sensitive data was removed. Each retrieved polygon was processed to disassemble the complex geometry into connected rectangular segments stored on a graph-based structure (Figure 3.2.c). Finally, the segments (graph edges) were located in each plan's correct position (Figure 3.2.d). For automation purposes, it was considered that each wall corner must be orthogonal with a tolerance of 10°, that there cannot be intersections between edges, and that the thickness of each wall must be limited to a maximum of 0.5 m. Invalid cases were manually edited to ensure every rectangular segment was valid and well-positioned within their respective plan.

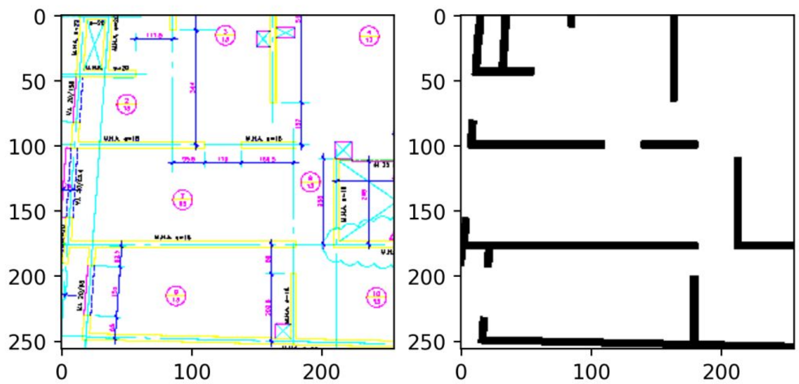


(a) (b) (c) (d)

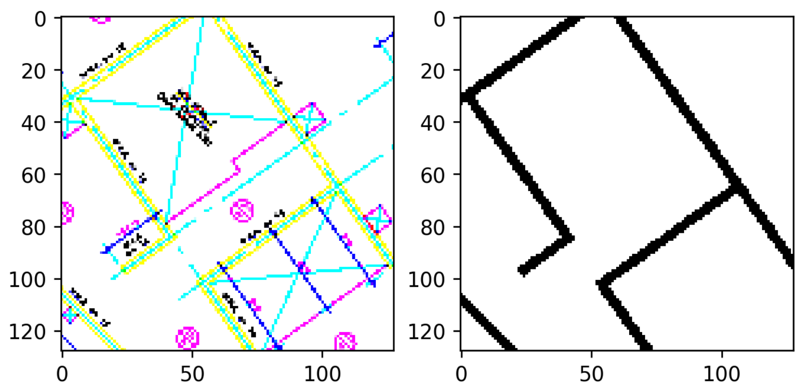
Figure 3.2: Example of the MLSTRUCT-FP' wall assembly process – (a) retrieval of the floor plan image, (b) wall contour polygon retrieval from its CAD model, (c) wall polygon dissasembly into a rectangular segments graph, (d) modeling of the wall, where a rectangle is highlighted [21].

The proposed dataset, publicly available on GitHub [200], encompasses a Python API designed to create the binary wall representation image for a given floor plan, providing the ground truth to develop wall segmentation models; additionally, it offers methods to generate crops and apply transformations, such as rotation and scaling, which are commonly employed techniques to increase plan variability within architectural floor plan research. Regarding the

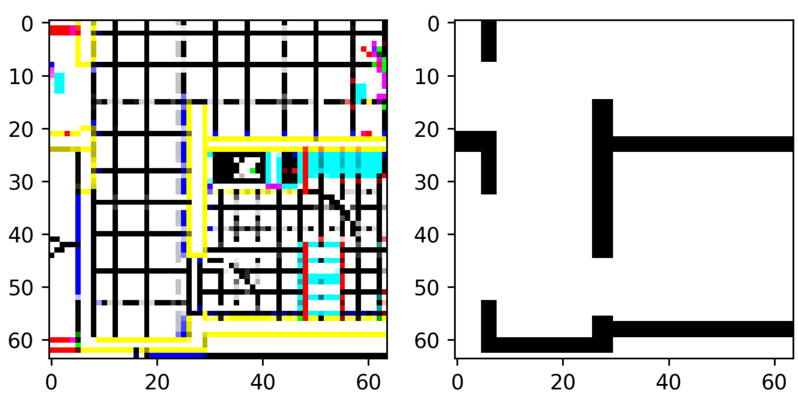
API implementation and details, refer to its GitHub repository, which describes its structure and provides examples for a quick start.



(a) 256×256 px crop of a 10×10 m region. Factor 0.039 m/px



(b) 128×128 px crop of a 7.5×7.5 m region with a 45° clockwise angle rotation. Factor 0.058 m/px



(c) 64×64 px crop of a 5×5 m region. Factor 0.078 m/px

Figure 3.3: Example of different crops from MLSTRUCT-FP dataset API [200], in terms of crop size, plan area extents, and rotation angles.

The plan crops can be created in any target pixel size, such as images of 64 or 256 px, for a particular plan area in meters. The factor meter/pixel (m/px) is essential when generating floor plan crops because larger factors lead to information loss. For example, a factor of 0.312 m/px (that is, a 32×32 px image for a 10×10 m area) has problems detailing walls whose thicknesses are less than 0.2 m; conversely, a factor of 0.156 allows for detailing a complete description of walls, but discards fine-grained details such as grids or non-structural labels [22]. For example, Figure 3.3 details different plan crops with distinct factors alongside their binary wall polygon image (ground truth) generated with the API. Figure 3.3.a depicts a high-resolution crop yielding a 10×10 m region, where labels and dimension lines remain intact; on the other hand, Figure 3.3.b details a 64×64 px image with massive information loss, especially regarding dimension lines, axes, and non-structural information; however, wall data remains intact due to the 0.078 m/px factor. That is, walls of 0.2 m (common within Chilean reinforced concrete buildings [198]) are still present as they use up to 3 px of the image. Regarding the API implementation and details, refer to its GitHub page [200], which describes the methods, structure, and provides examples for quick start.

With the aim of testing the MLSTRUCT-FP dataset, we propose a method that recognizes and vectorizes the wall polygons from each floor plan. The wall object recognition is performed by the learning-based U-Net segmentation model [51], which has extensively been used by related works in the object recognition task for several floor plan objects [3, 52, 83, 86], obtaining better results than other discriminative-based models used throughout architectural floor plan analysis [58], like FCN and DeepLab [83]. Furthermore, as a proof of concept, we vectorized the segmented output by employing the DL model from Egiazarian et al. work [52], allowing us to retrieve the wall primitives that constitute the output plan. The data processing, model training, evaluation, and results are detailed in the following section.

3.2 Wall segmentation and vectorization

In order to retrieve wall segments from the rasterized plan, in this thesis, we propose a method that automatically segments and vectorizes the elements by coupling an image processing method to split the input in small patches, segment each patch, and further re-assemble the plan for later vectorization through a DL model. In this way, it is possible to recover the primitives of complex plans for use in multiple applications within the design and construction industry, such as the automatic creation of digital models from PDF plans or automatic budgeting from a photo.

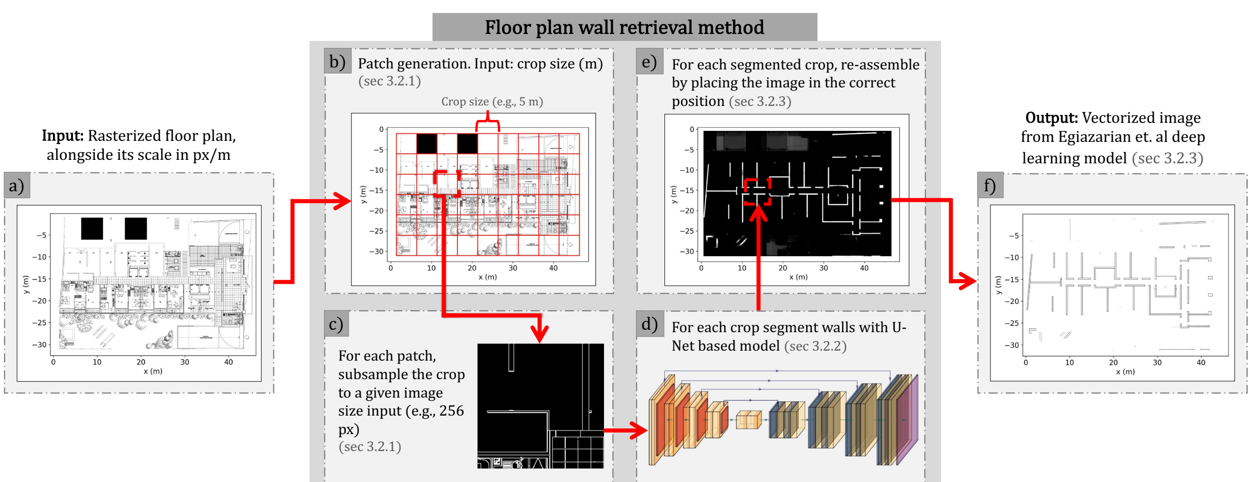
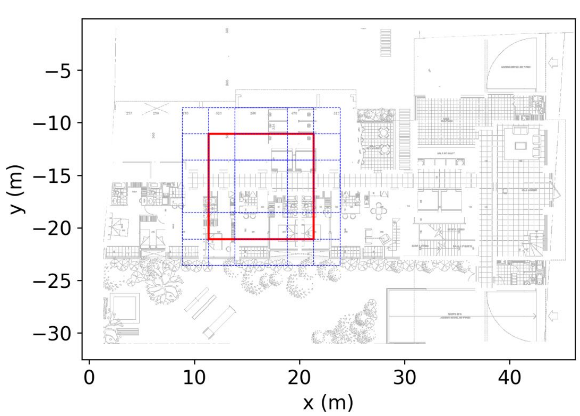


Figure 3.4: Schematic of the wall retrieval method from rasterized floor plans proposed in this thesis.

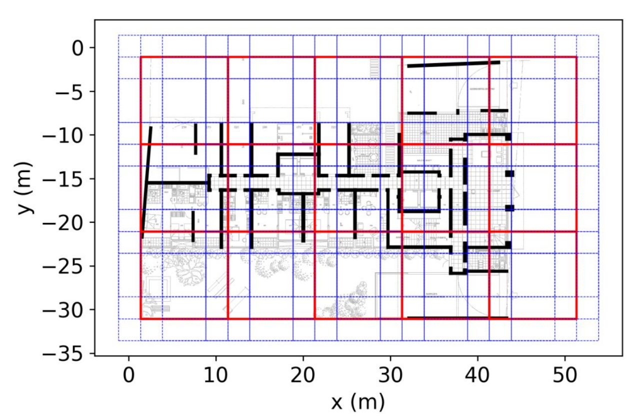
The proposed method, illustrated in Figure 3.4, details the six processing steps covered in the following sections. First, the plan image serves as the model input together with the scale in meters per pixel, which can be obtained by measuring the distance between two positions manually (Figure 3.4a); then, this image is divided into multiple patches of fixed size, for example, 5 meters, in order to be processed by a segmentative model (Figure 3.4b). Next, each patch is converted to black and white because the color does not contribute new information due to the absence of uniform design rules for MLSTRUCT-FP images; also, we reduce their dimension by subsampling to a fixed size because of memory issues (Figure 3.4c) [111]. Subsequently, the method processes each image with a U-Net-based model to retrieve the wall segments (Figure 3.4d), which are later placed in their correct position to assemble the entire floor plan (Figure 3.4e). Finally, the complete image is vectorized using a DL method retrieved from the literature review, thus obtaining the primitives that constitute each wall. The details of each step, as well as its implementation, are detailed in the following sections.

3.2.1 Data processing for floor plan wall segmentation

The initial phase of developing the segmentation model involves the generation of input and output pairs for training purposes. Since the minimum size of MLSTRUCT-FP dataset images exceeds 6300 px, with an average width of 9450 px, we decided to divide them into small patches with different resolution sizes to test the effect of information loss due to downsampling [83, 86]. Furthermore, for the additional purpose of testing different m/px factors, we split the plan into consecutive patches of fixed width in meters, with a translation offset of $\pm 25\%$ on each axis to capture the overlapping differences between contiguous patches. Figure 3.5.a details all generated 10×10 m patches from a given master position (zero translation), detailed in a red box, where nine patches can be observed for a ± 2.5 m translation, being $(0, 0)$ translation for x and y-axis, $(-2.5, -2.5)$, $(0, -2.5)$, $(2.5, -2.5)$, $(-2.5, 0)$, $(2.5, 0)$, $(-2.5, 2.5)$, $(0, 2.5)$, and $(2.5, 2.5)$. In contrast, Figure 3.5.b showcases a total of 135 patches, encompassing both master and translated instances.



(a) Example of the patch generation given a master position (illustrated in a red box), and the translated ones (illustrated in dashed blue)



(b) All patches generated from a single plan image. Wall polygon labels are depicted in black

Figure 3.5: Example of the patch generation with translation offset, for an area of 10×10 m.

In order to generate the cropped images, we use OpenCV [57] with the pixel area relation resampling method and a $\begin{pmatrix} -1 & -1 & -1 \\ -1 & 9 & -1 \\ -1 & -1 & -1 \end{pmatrix}$ filter, known as sharpen kernel, which emphasizes differences in adjacent pixel values in the context of large floor plans [22]. We stored the cropped images in black and white, given that the color does not contribute novel information because of the lack of standard styling rules within our dataset. Moreover, patches not covering the rasterized plan image (usually the ones around its perimeter) or those without wall segments are also removed. To assemble the train/test split, we consider 70% of the plan images for training (667-floor plans) and 30% for testing (287). The training data was furthermore split into 80% to train the wall segmentation model and the rest 20% for validation. Finally, no data augmentation was considered (rotation, scaling). Table 3.1 details all different data partitions, the image resolutions, and the number of patches.

Table 3.1: Patches generation cases used to train and evaluate the wall segmentation model.

Case	Patch crop (m)	Image size (px)	Factor m/px	No. of images
1	5	64	0.078	348,653
2	5	128	0.039	350,024
3	5	256	0.020	350,535
4	7.5	64	0.117	183,482
5	7.5	128	0.059	183,753
6	7.5	256	0.029	183,803
7	10	64	0.156	112,702
8	10	128	0.078	112,948
9	10	256	0.039	113,062
10	12.5	64	0.195	77,086
11	12.5	128	0.098	77,214
12	12.5	256	0.049	77,259
13	15	64	0.234	57,334
14	15	128	0.117	57,544
15	15	256	0.059	57,566
16	17.5	64	0.273	44,097
17	17.5	128	0.137	44,229
18	17.5	256	0.068	44,280

3.2.2 Deep learning wall segmentation model

Within learning-based models in automatic floor plan analysis, many algorithms have been presented to retrieve wall objects, such as SVMs [54] and fuzzy logic systems [145]; however, the ones that have achieved state-of-the-art results come along with deep learning technology [58] while also circumventing the need for intricate graphic separation rules, allowing one to employ the raw plan images to infer the recognition rules in the training step [10].

Among DL, the semantic segmentation U-Net [51] model has been extensively used to retrieve wall objects [3, 52, 83, 86], among other elements like doors, windows, and room shapes [58], obtaining better results than other discriminative-based models used throughout architectural floor plan analysis [58], like FCN and DeepLab [83], while also being simple to implement. For these reasons, we consider this architecture as a starting point to evaluate the performance of the patch-based method, while empirically testing how the wall segmentation behaves in large-scale plans.

The U-Net architecture, illustrated in Figure 2.9, can be described by two main components: an encoder (used to capture the image context) and a decoder (expansion, used to generate the segmented output). The encoder section focuses on capturing finer-grained structures from the input image by applying several convolutional and max-pooling layers, reducing the receptive field [149]. This approach facilitates the recovery of low-level features from the floor plans, such as the subtle relationships between the thin line segments that characterize walls, axes, or dimension lines. Conversely, the decoder allows reassembling the segmented plan by combining the intrinsic encoder features with higher-resolution feature channels through skip connections, enabling a better localization of the wall objects.

Figure 3.6 details the U-Net model architecture, implemented using the Python Keras library; the input image, belonging to each plan crop, is processed by the 5-level convolutional encoder, with sizes of 64, 128, 256, 512, and 1024 filters with a dropout regularization (ratio of 0.5) in the deep contractive levels. All convolutional layers have a kernel size of 3×3 and a stride of size 1×1, following the implementations of Yang et al. [3] and Surikov et al. [83]. The ReLU activator was used in all hidden layer outputs, except for the binary segmented wall image output, where a sigmoidal activation function is used instead. The pooling layers correspond to Average Pooling with a size of 2×2. The loss function, which measures the difference between the model output and the ground truth wall representation, corresponds to the binary cross-entropy $CE(y_i, \hat{y}_i) = -\frac{1}{N}\sum_{i=1}^{N} y_i \cdot \log \hat{y}_i + (1-y_i) \cdot \log(1-\hat{y}_i)$ [22]. Adam optimizer was used with a learning rate α of 10^{-4}, $\beta_1 = 0.9$, $\beta_2 = 0.999$, and $\varepsilon = 10^{-7}$. The weight initializer in all convolutional layers is He Normal [201]. L2 was used as a kernel regularizer with a factor of 10^{-4}. The training was carried out with mini-batches of size 2 in a maximum of 10 epochs [3, 83], accounting for 80% of the training partition and the other 20% for validation. Finally, we considered Intersection over Union (IoU) [146] as the evaluation metric because it is the most widely used to evaluate segmentation results among floor plan analysis learning-based models [58]. The model, alongside its examples and documentation can be found on its public GitHub repository [202].

All experiments were conducted on a personal computer with an Intel® Core™ i7-9750H (12M Cache, 4.5 GHz, six cores) CPU, 24GB DDR4-2666 RAM, and NVIDIA® GeForce RTX™ 2070 8GB GDDR6 GPU.

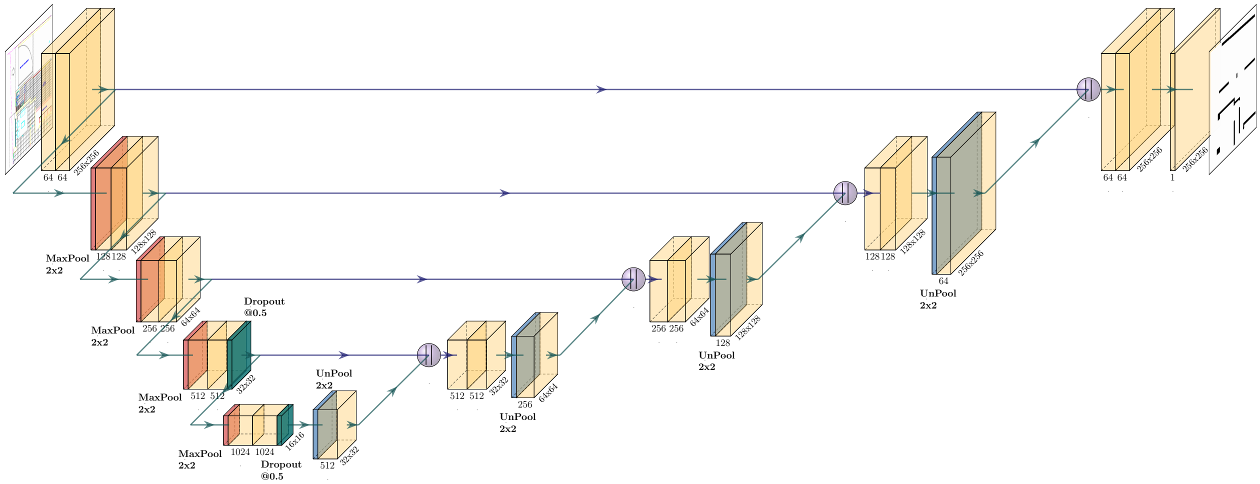


Figure 3.6: U-Net model architecture implementation, which takes each floor plan crop patch as input, and returns the segmented plan as output. Layer legend: *(yellow)* convolutional block, *(orange)* max-pool, *(blue)* up-sampling, and *(green)* dropout.

Table 3.2 details the model's performance for each generated test case; whereas Table 3.3 outlines the time associated for each train scenario. The highest achieved mean IoU corresponds to 0.77 for a 5×5 m crop and an image size of 256×256 px, belonging to the case with the lowest m/px factor. The results, which are in line with comparable wall retrieval models within single-unit plans [3, 28, 34, 49, 86], closely correlate with the m/px factor, obtaining a sigmoidal correlation with an R^2 of 0.9964; as the factor decreases, the IoU metric slowly approaches the maximum; on the contrary, as the factor increases (a higher crop area with the same image size or a lower image size with the same crop area), the model performance gradually decays. Figure 3.7 details this phenomenon, which is relevant since the scale factor has not been considered in related work, mainly due to the lack of large images or the absence of scale in the current datasets [58].

Table 3.2: Wall segmentation U-Net model results (mean IoU) for each test case, considering each plan crop and patch size combination.

Image size (px)	**Patch crop (m)**					
	5	7.5	10	12.5	15	17.5
64	0.63	0.50	0.44	0.39	0.36	0.33
128	0.76	0.71	0.63	0.57	0.52	0.47
256	**0.77**	0.76	0.74	0.72	0.6	0.65

Table 3.3: Training time in hours for each case.

Image size (px)	Patch crop (m)					
	5	7.5	10	12.5	15	17.5
64	16:46	7:26	4:25	3:06	2:09	1:37
128	20:46	10:51	6:10	4:24	3:06	2:20
256	**45:19**	24:03	14:03	9:24	7:00	5:28

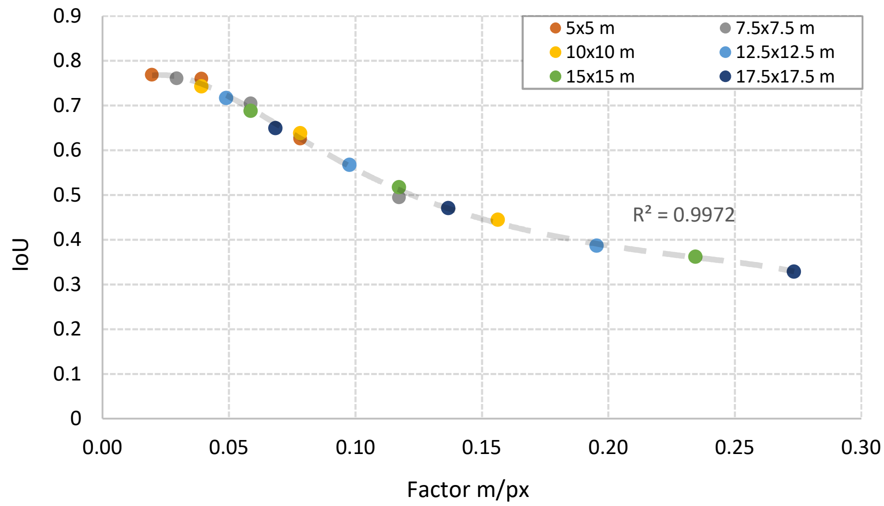


Figure 3.7: Mean IoU results for each m/px factor. Correlation in terms of the symmetrical sigmoidal 4PL function $y(x) = d + \frac{a-d}{1+\left(\frac{x}{c}\right)^b}$, parameters $a = 0.7836$, $b = 2.4175$, $c = 0.1082$, and $d = 0.29$.

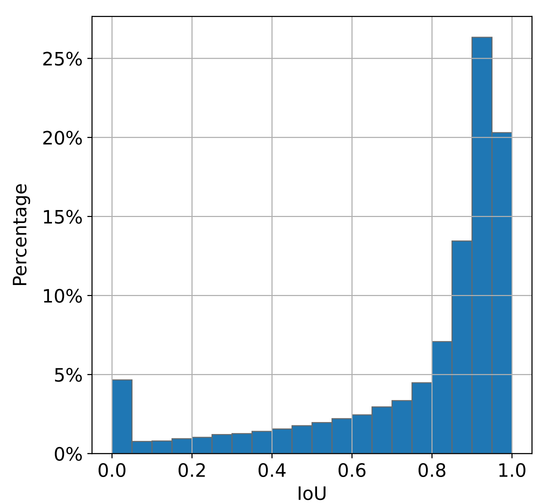
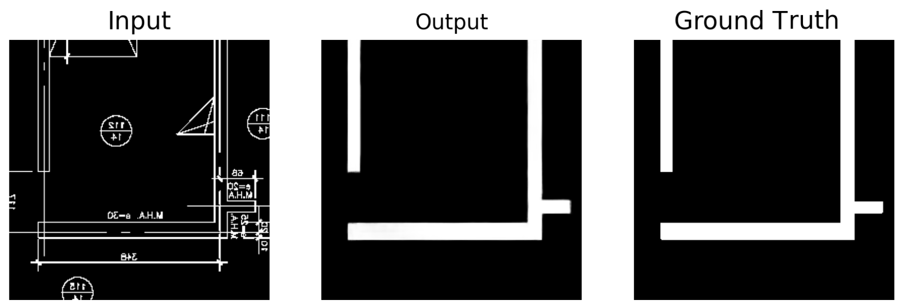
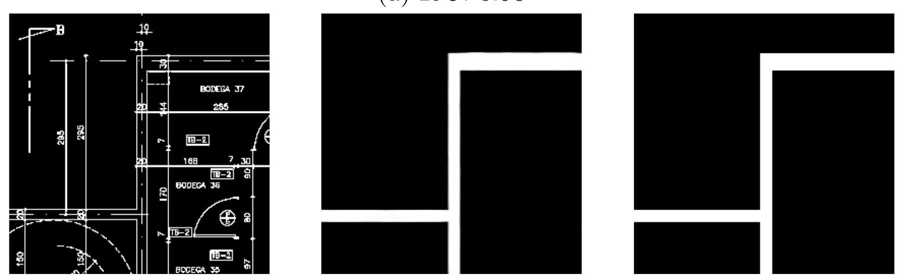


Figure 3.8: IoU histogram of the U-Net model results considering a 256×256 px image and 5×5 crop area, associated with 0.77 mIoU results in test.

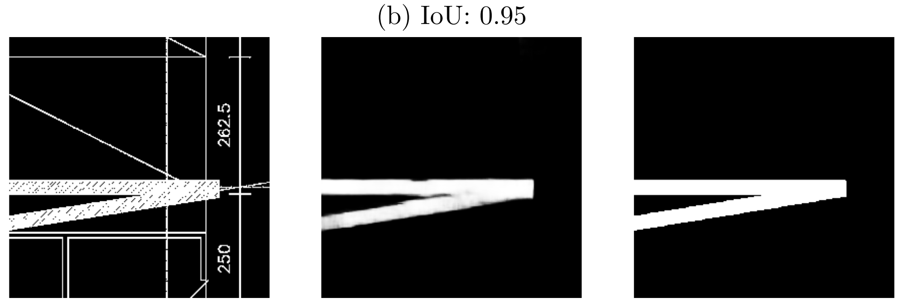
The distribution of results for the best training case is detailed in the histogram in Figure 3.8, which illustrates the IoU obtained as a percentage of the total test images (102,553 for a total of 287 plans); it can be observed that the results' mode is above the 0.77 average, around 0.9; however, almost 5% of the evaluations have an incorrect segmentation. Most of these cases are associated with plans characterized by distinctive, unique styles or cases that do not have walls, but the model incorrectly segmented them, usually in areas with large spans (e.g., parking lots) or around the structure's perimeter. Figure 3.9 illustrates the model results for different patches, showcasing the performance for a spectrum of several IoUs; cases a-c depict successful segmentation results for orthogonal and diagonal wall segments; conversely, case g shows an incorrect detection of a plan with an uncommon angle and wall styling.



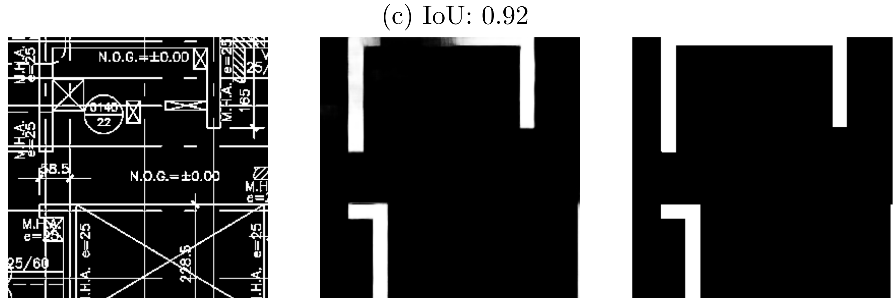
(a) IoU: 0.98



(b) IoU: 0.95

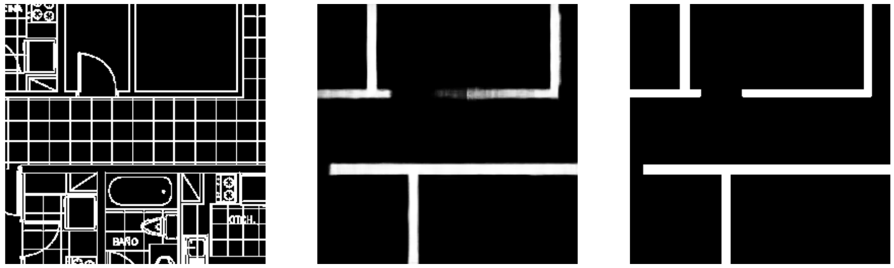


(c) IoU: 0.92

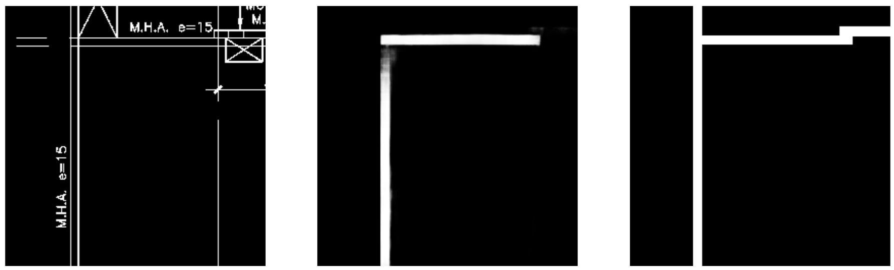


(d) IoU: 0.90

Figure 3.9: U-Net model results for different patches considering a 256×256 px image and 5×5 m crop area. Each image displays the input (patch crop), the model result (segmented wall), and the ground truth.



(e) IoU: 0.84



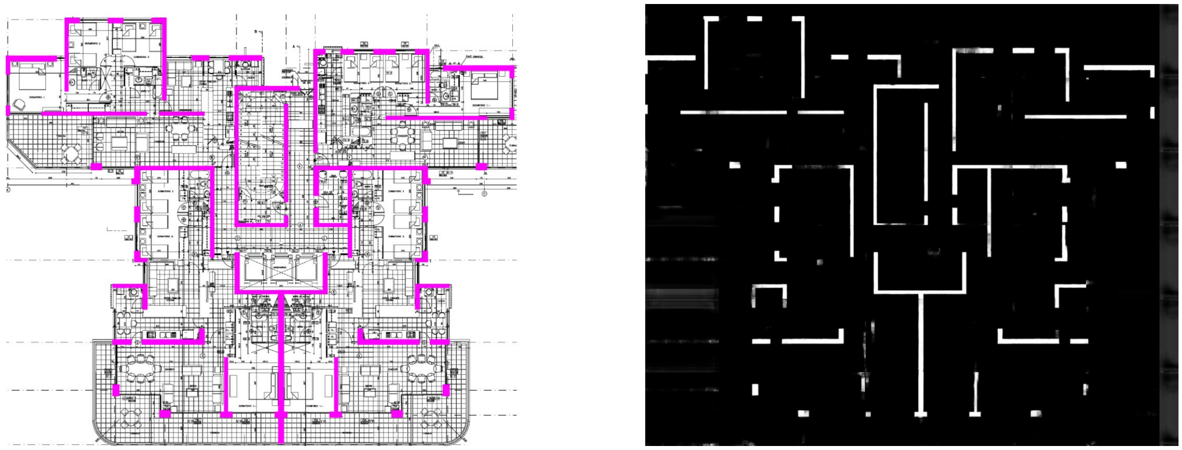
(f) IoU: 0.74

(g) IoU: 0.57

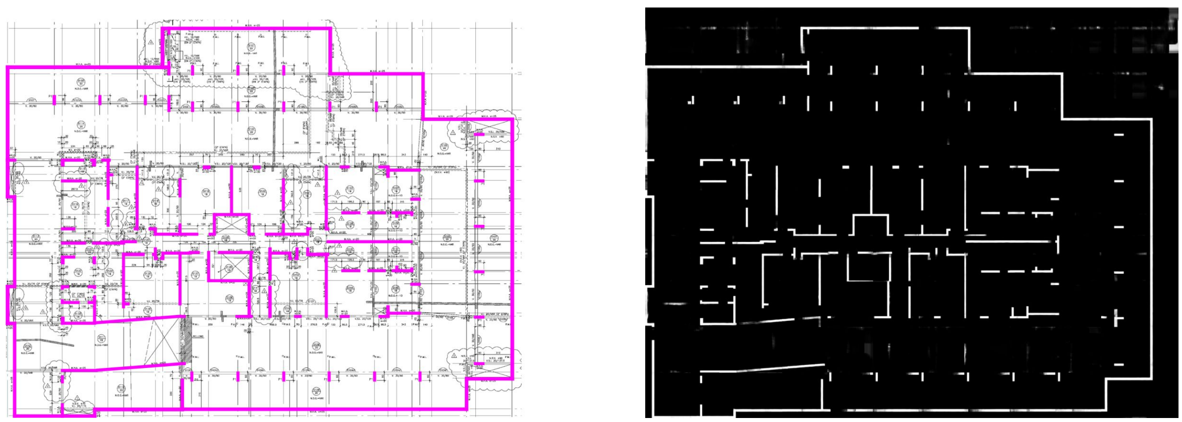
(h) IoU: 0.33

Figure 3.9: U-Net model results for different patches considering a 256×256 px image and 5×5 m crop area. Each image displays the input (patch crop), the model result (segmented wall), and the ground truth (continuation).

One typical limitation of pixel-level semantic segmentation learning-based models, such as U-Net, is the generation of blurry results, as illustrated in cases e-g of Figure 3.9, leading to recognition issues because of disconnected entities and the object recovery with irregular geometry [58]. A noteworthy consequence of incorrect recognition arises in room detection models, which strongly depend on structural elements to delimit the plan area topology; hence, disappearing elements can considerably affect the room formation processes [68]. On the other hand, the inflated ground-truth boxes from instance segmentation models like Faster R-CNN lack a suitable notation to describe the curved or sloped wall primitives, leading to localization issues if no complex post-processing is considered [71]. Therefore, despite its limitations, the implemented U-Net model enables object retrieval with flexible (pixel-based) labeling, being also simple to implement and train.



(a)



(b)

(c)

Figure 3.10: Segmentation results of the whole plan by assembling each processed patch in its correct position.

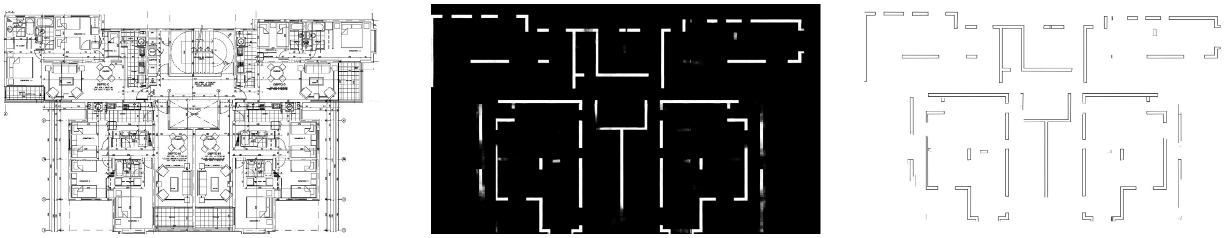
Finally, Figure 3.10 illustrates the complete plan segmentation by placing each patch in its correct position for a selection of three test floor plans. At the left of each example, the rasterized input plan can be observed alongside the wall label polygons in pink, whereas the segmentation results are illustrated on the right. In each case, the U-Net model can recognize the main structural walls with excellent results, yielding issues in complex scenarios where parallel lines are mistakenly recognized, especially nearby the parking lots (case c) and the line segments close to axis labels (case b). On the other hand, some beams are also incorrectly detected as walls, especially in areas with long spans (case b), mainly because beams are often distinguished from walls using a different line color. In the context of the complete plan processing, it is worth emphasizing that the entire model workflow, encompassing image filter and subsample, their partitioning into distinct patches, their subsequent processing using the trained U-Net model, and the final post-processing to integrate the results, takes an average time of 2.2 $\pm$ 0.8 seconds. This rapid evaluation timeframe underscores the model's applicability across various floor plan methodologies [58] without substantially disrupting their workflow, while allowing users to quickly evaluate the results.

3.2.3 Deep learning vectorization

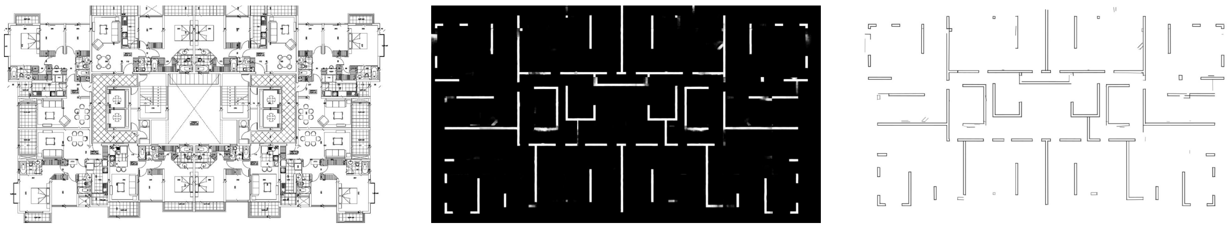
The final step of the wall retrieval from the rasterized floor plan images is the vectorization of the segmented wall output, which was carried out using the DL based method proposed by Egiazarian et al. [52] as a proof of concept, since this model is one of the few that has its implementation publicly available, in addition to allowing an input compatible with the output of our method. Their method obtains the line primitives from floor plan drawings using U-Net for pre-processing (eliminate background, imperfections, and fill missing parts); then, the resulting images are split into patches to independently estimate the line and curve primitives with a feed-forward Artificial Neural Network (ANN). Each patch is encoded with a ResNet-based feature estimator [163] and decoded using Transformer blocks [164] that allow varying the number of output primitives per patch. Predicted primitives are later refined and aligned to the raster image through an optimization procedure.

In order to employ their proposed method, we post-processed the segmented output images with the Sobel filter to detect the wall edges [202]. Then, we vectorize the filtered image using a pre-trained line-detector model [203], considering patches of 64×64 px with a 16 px overlap in 300 iterations. Figure 3.11 illustrates five examples of the polygon vectorization method, detailing the plan input image from the test partition, the segmented output, and the vector result. In particular, edge characterization is well achieved in most walls, except for some artifacts due to divergences in the segmentation process. The vectorization model can complete unjoined segments (case a, c); however, many are open, especially in outer walls or with a non-orthogonal angle, which requires post-processing to fix. It is important to note that the vector primitives are disconnected and, in some cases, repeated along the plan; therefore, to reconnect and filter them, it is necessary to use a post-processing mechanism that compares, for example, each situation with a set of rules that depend on a particular style in the geometrical description of the walls [17]. On the other hand, it is also evident that the U-Net pre-processing removes noise and imperfections from the segmented image [52], common in the plan image perimeter and those areas with large spans, contributing to the overall results of the vectorized model.

Since the polygons require complex post-processing, the results were verified only from a qualitative perspective. The correction and optimization of the resulting geometry is work that was beyond the scope of the thesis; therefore, it is an excellent topic for future work.

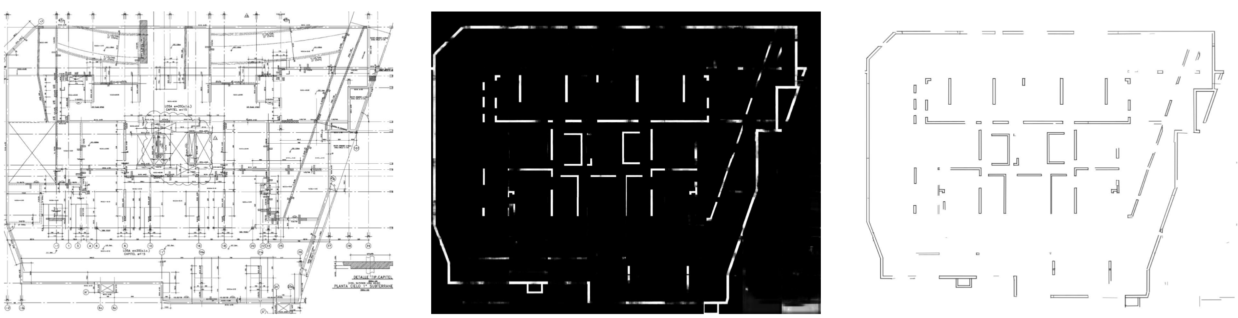


(a)

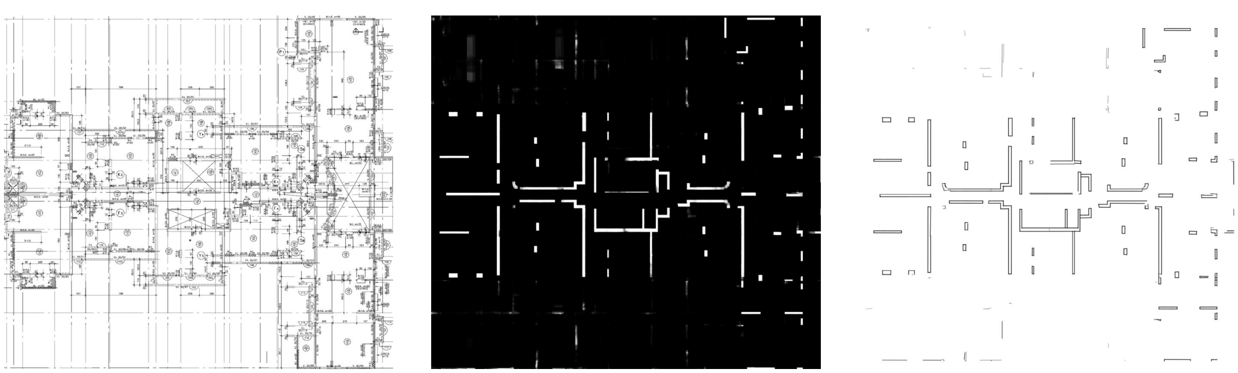


(b)

(c)



(d)



(e)

Figure 3.11: Vectorization results of the wall polygon from the segmented output for five complex floor plans.

Chapter 4

Conclusions

The automatic analysis of architectural floor plans is a discipline that has experienced a significant boom, mainly motivated by the need to increase productivity levels in the construction and design industries. In addition, the rise of learning models also highlights the need for new data to train and validate them. Therefore, in this study, we reviewed related work within architectural floor plan analysis that used rasterized images to automatically retrieve objects like walls, doors, windows, and rooms; then, we proposed a novel public multi-unit floor plan dataset (MLSTRUCT-FP) to add more variety to the available data. The following summarizes the significant findings related to each research question.

Concerning the revised methodologies, authors have traditionally considered rule-based methods that exploited low-level heuristics to retrieve the desired objects in the plans, generally by solving four common tasks: (1) *Graphics separation*, which removes undesirable elements from plans, (2) *Object recognition*, which recognizes the building elements from the image, (3) *Vectorization*, a process that transforms the objects to a vector form, and (4) *Structural modeling*, which transforms the floor objects to a mathematical model. Most methods that employ manual rules to solve these tasks are restricted to the datasets researchers used, as plans can have different styles, semantics, layouts, and inner correlations, limiting the range these rules can handle.

Since the plans are complex and diverse in style but difficult to access and produce, rule-based algorithms kept a limited development until learning-based algorithms were introduced. Unlike the previous ones, these methodologies automatically learn the relationship between the floor plan elements, exploiting low and high-level features directly from the training data, composed of dozens of validated plans from various styles. Thus, these documents' thresholds, limitations, and rules are inferred. From the learning approach, deep learning, especially those related to neural networks, has achieved state-of-the-art results with respect to the accuracy and other metrics such as the IoU. These models can compute complex and non-trivial features from the plan data, which are challenging to reproduce manually, and usually avoids graphical separation as the raw plan images can be used without further pre-processing.

Even though remarkable results have been achieved in the last few years, floor plan analysis is still considered an open task within computer vision. For instance, rule-based algorithms rely on particular plan styles, being hard to generalize to other formats. Conversely, learning-based models trained on various datasets might have great adaptability, but their outputs are usually blurry as they perform pixel-level segmentation or can have significant differences if an object is missing. Also, learning models require a high number of plans to train and generalize the results; and this can be extremely expensive or unnecessary if only a couple of plans have to be processed. For such reasons, we introduced a novel dataset, alongside an image processing technique, to help future developers build models that analyze complex multi-unit floor plans, common within architectural and structural engineering firms.

In particular, this dataset stands out for a wide range of geometric scenarios in the composition of its walls; these objects are of great importance when understanding the layout of an architectural plan since they delimit the shape and perimeter of a structure and allow for detecting additional elements such as beams, doors, windows, among others. For these reasons, we implemented a U-Net based-model to segment the walls from the raster images, obtaining excellent results despite the plan style complexity and their large image size. Notably, two key insights emerge: (1) the importance of cropping the input image to avoid loss of information due to subsampling, and (2) the importance of the image scale factor (pixel/meter) while processing the plan patches.

Despite being a simple model, U-Net allows for obtaining wall shapes with high accuracy, achieving a 0.77 mIoU, which is comparable regarding related wall retrieval research, serving as a comparison baseline. Moreover, it is a result that can be improved by the incorporation of more sophisticated models that consider the contextual situation of the elements and their interaction (for example, wall/room), or that employ an architecture focused on the characteristic granular features of floor plans, which remain an open problem. Future work must also focus on reducing the number of disconnected elements, refining the precision of the recognized geometry to match the distinctive regular patterns of wall elements, and improving fuzzy results near the primitives. With respect to the plan vectorization, the deep-learning model yielded good qualitative results following the proposed edge detection method. However, post-processing is required to correct the polygons and reassemble the disconnected primitives, which can be challenging to implement and evaluate.

Regarding future research directions, it's worth highlighting that the structured nature format of the dataset allows creating distinct queries to retrieve its polygons and images, for example, to generate new floor plan layouts, check their accessibility, calculate plan escape routes, among others. In addition, novel datasets can be created by manually labeling elements such as doors, windows, and furniture. The image processing pipeline also offers several improvement opportunities, for instance, in the definition of different kernel filters that improve results for downsampled plans of distinct styles, the calibration of the plan crop and target image size for a particular style, and the proposal of distinct segmentation and vectorization models to improve the recognition results, whose have sustained substantial growth over the last few years due to advances in discriminative and generative deep neural networks. All these applications open new veins in the research that will allow a more productive industry in the future years with the development of AI and the incorporation of more and better datasets and models.

4.1 Contribution

The contributions of this thesis can be summarized as follows:

1. The revision of the state of the art, which involved a comprehensive analysis of numerous articles, encompassing the conceptualization of the problem, its scope, objectives, and prospective goals. This extensive review spanned the initial advancements in this field from 1995 to early 2022. Consequently, it provides a valuable resource for future developers seeking models that align with specific requirements and tasks. The review has been published in the Automation in Construction (Q1) journal [58].

2. The creation of a new dataset of complex multi-unit plans, which enables the analysis of new floor plan configurations of Chilean residential buildings, opening up a new vein of possible research in the analysis and processing of data within the discipline. This dataset is publicly accessible on GitHub [200].

3. Segmentation and vectorization of complex multi-unit plans, comprising a large-scale image processing method, implementation of a DL model for segmentation, and proof-of-concept vectorization, both publicly available on GitHub [202]. The dataset, method, and model were also submitted to the Automation in Construction journal [197].

4.2 Future work

Plan analysis remains an open topic within the area of computer vision. Future goals include the development of generalizable models that enable the processing of drawings regardless of their style or scale. On the other hand, there is the open problem of recovering a more extensive variety of elements, not only walls or rooms but also other entities such as texts and dimensions, slabs, and textures.

In this sense, there are also goals within the generative area, providing new solutions to the engineering and architecture team that allow for solving new, sustainable, low-cost structural configurations that comply with legal requirements, safety, geometrical constraints, resource availability, and human labor; this requires new models, more data, and interdisciplinary coordination between multiple agents, from construction and design, urban planning, and sales.

New advances will enable reducing the technological gap that the industry has presented for many years, especially in Chile, which at the date of publication of this thesis, is experiencing one of the biggest crises of the last decades, with more than 300 construction-related companies bankrupt in the last two years alone. Therefore, it is necessary to create new tools to reduce costs, provide innovative solutions, and prevent losses. Moreover, it is an area that offers multiple opportunities in the long term since software is scarce and very expensive.

Our work represent only a tiny part of the innovations that can be achieved in this area. It provides both an analysis of what has been done and proposes new challenges and opportunities, thinking of a bright future for the industry and our country's development.

List of Terms

2D	Two-dimensional building retrieval and reconstruction, typically from raster and vector plans. 1, 2, 8, 28, 31
3D	Three-dimensional building retrieval and reconstruction, typically from CAD and BIM plans. vii, 1–3, 8, 12, 13, 18, 24, 26, 28, 31
AI	Artificial Intelligence. 4, 6, 7, 32, 50
ANN	Artificial Neural Network, is a computational model inspired by the human brain's neural structure, used for various machine learning tasks. 24, 47
API	API stands for Application Programming Interface, which allows different software applications to communicate and interact with each other. vii, 18, 21, 36–38
BIM	Building Information Modeling, is a digital representation of a building's characteristics used in architecture, engineering, and construction. 2, 3, 31
BOVW	Bag of Visual Words, is a technique that describes an image using clusterized low-level features. 17, 19, 20
C4.5	C4.5 is a popular decision tree algorithm used in machine learning and data mining. 26
CAD	Computer-Aided Design, is a technology that uses computer software to create, modify, and optimize design solutions for various industries. i, ii, vii, 1, 7, 11, 12, 27, 31, 36
CBIR	Content-Based Image Retrieval. 26
CenterNet	CenterNet is a deep learning architecture designed for object detection tasks, focusing on locating object centers and regressing bounding box parameters. 25
cGAN	Conditional Generative Adversarial Network, is a machine learning model that combines GANs with conditional information, allowing the generation of data based on specific conditions or input data. 27, 28
CNN	Convolutional Neural Network, is a deep learning model designed for processing and analyzing visual data, inspired by the visual cortex of living organisms. vii, 2, 20–24, 26, 27, 32, 36

CornerNet	CornerNet is a keypoint detection neural network that excels in locating object corners, especially for bounding boxes. 25
CycleGAN	Cycle-Consistent Generative Adversarial Network, is a type of GAN designed to learn mappings between two different domains of data, enabling the transformation of images from one domain to another while maintaining the original content. 27, 28
DeepLab	DeepLab is a Convolutional Neural Network architecture designed for semantic image segmentation. 20, 21, 23, 24, 26, 38, 41
DiscoGAN	Discovering Cross-Domain Relations Generative Adversarial Network, is a type of GAN which learns to translate images between two domains without paired examples. 27, 28
DL	Deep Learning. 3–5, 8, 19–22, 27, 29, 31, 32, 38, 39, 41, 47, 51
DR	Detection Rate measures the percentage of correctly detected objects among the total actual objects, assessing object detection algorithm performance. 29, 30
FCN	Fully Convolutional Network, is a deep learning architecture designed for pixel-level tasks like image segmentation. 20, 21, 23, 24, 26, 38, 41
FRBS	Fuzzy Rule-Based System, is an algorithm that employs fuzzy logic and rules for flexible decision-making in uncertain environments. 19, 21
GAN	Generative Adversarial Network, is a framework in machine learning consisting of two neural networks, a generator and a discriminator, trained in a competitive process to produce realistic synthetic data. 2, 20, 21, 27–29
GCN	Graph Convolutional Network, is a type of neural network designed for processing and analyzing data structured as graphs or networks. 27
GitHub	GitHub is a web-based platform that provides version control and collaboration tools for software development projects. 36–38, 41, 51
GNN	Graph Neural Network, is a machine learning model designed to process and analyze graph-structured data. 2, 21, 27, 32
HT	Hough Transform, is a technique for detecting shapes in images, commonly used for detecting lines and circles. 13, 14, 16, 17
IoU	Intersection over Union, is a metric used to evaluate the accuracy of object detection algorithms by measuring the overlap between predicted and actual object bounding boxes. vi–viii, 5, 19, 24, 29, 30, 41–44, 49, 50
IP	Integer Programming, is an optimization method solving linear programming problems with certain variables constrained to integer values. 26

JI	Jaccard Index measures the similarity between two sets by dividing the size of their intersection by the size of their union. 29, 30
JSON	JavaScript Object Notation. 35
Keras	Keras is an open-source high-level neural networks API written in Python, often used as a user-friendly interface to build and train deep learning models. 5, 41
L2	L2 kernel regularizer helps prevent overfitting by adding a penalty term to the loss function proportional to the squared magnitude of the model's weight values. 41
LiDAR	Light Detection and Ranging, is a technology that uses lasers to measure distances, creating precise 3D models for applications like mapping and autonomous vehicles. 31
mAP	Mean Average Precision, is a metric used to evaluate object detection models' accuracy by calculating the average precision for each class across different levels of precision. 29, 30
MLP	Multilayer Perceptron, is a class of neural networks composed of multiple layers of interconnected nodes, commonly used for supervised learning tasks. 20
MLSTRUCT-FP	Machine Learning Structural Floor Plan, is the multi-unit floor plan dataset proposed in this thesis. vii, 34–39, 49
MS	Match Score, is a metric used to assess the quality of object detection results, measuring the similarity between detected bounding boxes and ground truth boxes. 29, 30
multi-unit	Multi-unit floor plans refer to architectural layouts that encompass multiple individual living spaces within a single structure, often seen in apartment buildings or housing complexes. vii, 3, 7, 11, 30, 33, 34, 49–51
OCR	Optical Character Recognition, is a technology that converts scanned text or images containing text into editable and searchable data. 9, 16, 20, 21, 23, 24
OECD	Organisation for Economic Co-operation and Development, is an international organization promoting economic development and cooperation among member countries. 1
OpenCV	Open Source Computer Vision Library, is a library of programming functions mainly aimed at real-time computer vision. 5, 40
OTSU	Otsu's method is an image thresholding technique used to automatically determine optimal thresholds for image segmentation. 16
PDF	Portable Document Format. 38
PFM	Primitive Feature Map. 29

Pix2Pix	Pix2Pix is a deep learning model for image-to-image translation tasks, preserving structure while converting images between domains. vii, 20, 21, 27–29
PixelDCL	Pixel Decomposition Contrastive Learning, is a self-supervised framework for enhancing image understanding by separating content and style representations. 20, 24
PNG	Portable Network Graphics. 35
PSPNet	Pyramid Scene Parsing Network, is a deep learning architecture used for high-resolution image segmentation tasks by incorporating contextual information through different scales. 20, 26
PU	Positive Unlabeled learning is a machine learning approach for predicting positive instances using only positive and unlabeled data. 18, 19, 21
Python	Python is a versatile and widely used high-level programming language known for its readability and ease of use, commonly used in various applications including web development, data analysis, and artificial intelligence. 5, 36, 41
QGAR	Rule-based software for rasterized floor plan recognition. 13, 16
R-CNN	Region-Based Convolutional Neural Network, is a computer vision model for object detection that segments and classifies objects within images. vii, 9, 20, 21, 23–26, 31, 32, 36, 45
RA	Recognition Accuracy gauges the correctness of identified objects in a dataset, reflecting the precision of an object recognition model's predictions. 29, 30
RAG	Room Adjacency Graph. 14, 16
RCF	Randomized Consensus-based Framework, is an image processing technique that employs random sampling and consensus mechanisms to detect edges and boundaries in images. 20, 26
ReLU	Rectified Linear Unit, is an activation function commonly used in neural networks that replaces negative input values with zero, helping to introduce non-linearity and improve convergence during training. 41
ResNet	Residual Network, is a deep learning architecture using residual blocks to address vanishing gradient in deep neural networks. 20, 24, 26, 47
ROI	Region of Interest, refers to a specific area within an image or dataset that is selected for analysis due to its relevance or importance. 18, 23, 26, 36
SG	Shape grammar, is a rule-based computational design method for generating complex shapes and structures. 12, 29
SSD	Single Shot MultiBox Detector, is a real-time object detection algorithm that efficiently detects objects in images and videos. 21, 24
SURF	Speeded-Up Robust Features. 13, 16

SVM Support Vector Machine, is a classification and regression algorithm that finds a hyperplane to separate data into classes. 17–21, 40

TensorFlow TensorFlow is an open-source machine learning framework developed by Google, widely used for building and training various types of neural network models. 5, 18, 21

U-Net U-Net, is a convolutional neural network architecture designed for semantic segmentation tasks in image processing. i, ii, vi–viii, 3, 5, 20, 21, 23, 24, 28, 33, 38, 39, 41–45, 47, 50

VGG VGG is a deep convolutional neural network architecture used for image recognition. vii, 18, 20, 21, 24, 26

VR Virtual Reality, is a technology that immerses users in a simulated environment through computer-generated experiences. 2, 32

XML eXtensible Markup Language, is a text-based data format for hierarchical data organization and exchange. 7

YOLO You Only Look Once, is an object detection algorithm that predicts object bounding boxes and class probabilities in images. 9, 20, 21, 25, 26, 31

Bibliography

[1] Liu, C., Wu, J., Kohli, P., and Furukawa, Y., "Raster-to-vector: revisiting floorplan transformation," in 2017 IEEE International Conference on Computer Vision, (Venice, Italy), pp. 2195–2203, IEEE, 2017, doi:10.1109/ICCV.2017.241.

[2] Gimenez, L., Robert, S., Suard, F., and Zreik, K., "Automatic reconstruction of 3D building models from scanned 2D floor plans," Automation in Construction, vol. 63, pp. 48–56, 2016, doi:10.1016/j.autcon.2015.12.008.

[3] Yang, J., Jang, H., Kim, J., and Kim, J., "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, (Beijing, China), pp. 1–9, IEEE, 2018, doi:10.1109/CISP-BMEI.2018.8633243.

[4] Stokes, H. K., "An examination of the productivity decline in the construction industry," The Review of Economics and Statistics, vol. 63, p. 495, 1981, doi:10.2307/1935844.

[5] de Valence, G. and Abbott, M., "A review of the theory and measurement techniques of productivity in the construction industry," in Measuring Construction. Prices, Output and Productivity, ch. 12, p. 288, Routledge, 2015, doi:10.4324/9781315882925.

[6] Abdel-Wahab, M. and Vogl, B., "Trends of productivity growth in the construction industry across Europe, US and Japan," Construction Management and Economics, vol. 29, pp. 635–644, 2011, doi:10.1080/01446193.2011.573568.

[7] Park, J. and Kwon, Y.-B., "Main wall recognition of architectural drawings using dimension extension line," in Graphics Recognition. Recent Advances and Perspectives. Lecture Notes in Computer Science, vol 3088, pp. 116–127, Springer, Berlin, Heidelberg, 2003, doi:10.1007/978-3-540-25977-0_11.

[8] Zhu, R., Shen, J., Deng, X., Walldén, M., and Ino, F., "Training strategies for CNN-based models to parse complex floor plans," in Proceedings of the 2020 9th International Conference on Software and Computer Applications, (New York, NY, USA), pp. 11–16, ACM, 2020, doi:10.1145/3384544.3384566.

[9] Macé, S., Locteau, H., Valveny, E., and Tabbone, S., "A system to detect rooms in architectural floor plan images," in Proceedings of the 9th IAPR International Workshop on Document Analysis Systems, (New York, NY, USA), pp. 167–174, ACM Press, 2010, doi:10.1145/1815330.1815352.

[10] Kim, S., Park, S., Kim, H., and Yu, K., "Deep floor plan analysis for complicated drawings based on style transfer," Journal of Computing in Civil Engineering, vol. 35, p. 04020066, 2021, doi:10.1061/(ASCE)CP.1943-5487.0000942.

[11] de las Heras, L.-P., Fernandez, D., Valveny, E., Llados, J., and Sánchez, G., "Unsupervised wall detector in architectural floor plans," in 2013 12th International Conference on Document Analysis and Recognition, (Washington, DC, USA), pp. 1245–1249, IEEE, 2013, doi:10.1109/ICDAR.2013.252.

[12] Sharma, D., Gupta, N., Chattopadhyay, C., and Mehta, S., "DANIEL: a deep architecture for automatic analysis and retrieval of building floor plans," in 2017 14th IAPR International Conference on Document Analysis and Recognition, (Kyoto, Japan), pp. 420–425, IEEE, 2017, doi:10.1109/ICDAR.2017.76.

[13] Gimenez, L., Hippolyte, J.-L., Robert, S., Suard, F., and Zreik, K., "Review: reconstruction of 3D building information models from 2D scanned plans," Journal of Building Engineering, vol. 2, pp. 24–35, 2015, doi:10.1016/j.jobe.2015.04.002.

[14] Lu, Q. and Lee, S., "A semi-automatic approach to detect structural components from CAD drawings for constructing as-is BIM objects," in Computing in Civil Engineering 2017, (Reston, VA), pp. 84–91, American Society of Civil Engineers, 2017, doi:10.1061/9780784480823.011.

[15] Zhao, Y., Deng, X., and Lai, H., "Reconstructing BIM from 2D structural drawings for existing buildings," Automation in Construction, vol. 128, p. 103750, 2021, doi:10.1016/j.autcon.2021.103750.

[16] Yin, X., Wonka, P., and Razdan, A., "Generating 3D building models from architectural drawings: a survey," IEEE Computer Graphics and Applications, vol. 29, pp. 20–30, 2009, doi:10.1109/MCG.2009.9.

[17] Wen, Q. and Zhu, R.-G., "Automatic generation of 3D building models based on line segment vectorization," Mathematical Problems in Engineering, vol. 2020, pp. 1–16, 2020, doi:10.1155/2020/8360706.

[18] Machairas, V., Tsangrassoulis, A., and Axarli, K., "Algorithms for optimization of building design: a review," Renewable and Sustainable Energy Reviews, vol. 31, pp. 101–112, 2014, doi:10.1016/j.rser.2013.11.036.

[19] Zawidzki, M. and Szklarski, J., "Multi-objective optimization of the floor plan of a single story family house considering position and orientation," Advances in Engineering Software, vol. 141, p. 102766, 2020, doi:10.1016/j.advengsoft.2019.102766.

[20] Nguyen, A.-T., Reiter, S., and Rigo, P., "A review on simulation-based optimization methods applied to building performance analysis," Applied Energy, vol. 113, pp. 1043–1058, 2014, doi:10.1016/j.apenergy.2013.08.061.

[21] Pizarro, P. N. and Massone, L. M., "Structural design of reinforced concrete buildings based on deep neural networks," Engineering Structures, vol. 241, p. 112377, 2021, doi:10.1016/j.engstruct.2021.112377.

[22] Pizarro, P. N., Massone, L. M., Rojas, F. R., and Ruiz, R. O., "Use of convolutional networks in the conceptual structural design of shear wall buildings layout," Engineering Structures, vol. 239, p. 112311, 2021, doi:10.1016/j.engstruct.2021.112311.

[23] Lu, X., Liao, W., Zhang, Y., and Huang, Y., "Intelligent structural design of shear wall residence using physics-enhanced generative adversarial networks," Earthquake Engineering & Structural Dynamics, 2022, doi:10.1002/eqe.3632.

[24] Goyal, S., Chattopadhyay, C., and Bhatnagar, G., "Knowledge-driven description syn-

thesis for floor plan interpretation," International Journal on Document Analysis and Recognition, vol. 24, pp. 19–32, 2021, doi:10.1007/s10032-021-00367-3.

[25] Gerstweiler, G., Furlan, L., Timofeev, M., and Kaufmann, H., "Extraction of structural and semantic data from 2D floor plans for interactive and immersive VR real estate exploration," Technologies, vol. 6, p. 101, 2018, doi:10.3390/technologies6040101.

[26] Lewandowicz, E. and Lisowski, P., "Methodology to generate navigation models in building," Journal of Civil Engineering and Management, vol. 24, pp. 619–629, 2018, doi:10.3846/jcem.2018.6599.

[27] Wu, Y., Shang, J., Chen, P., Zlatanova, S., Hu, X., and Zhou, Z., "Indoor mapping and modeling by parsing floor plan images," International Journal of Geographical Information Science, vol. 35, pp. 1205–1231, 2020, doi:10.1080/13658816.2020.1781130.

[28] Jang, H., Yu, K., and Yang, J., "Indoor reconstruction from floorplan images with a deep learning approach," ISPRS International Journal of Geo-Information, vol. 9, p. 65, 2020, doi:10.3390/ijgi9020065.

[29] Pintore, G., Mura, C., Ganovelli, F., Fuentes-Perez, L., Pajarola, R., and Gobbetti, E., "State-of-the-art in automatic 3D reconstruction of structured indoor environments," Computer Graphics Forum, vol. 39, pp. 667–699, 2020, doi:10.1111/cgf.14021.

[30] Mathew, B. P. and Dharan, S., "Review on room layout estimation from a single image," International Journal of Engineering Research and Technology, vol. V9, 2020, doi:10.17577/IJERTV9IS060820.

[31] Kang, Z., Yang, J., Yang, Z., and Cheng, S., "A review of techniques for 3D reconstruction of indoor environments," ISPRS International Journal of Geo-Information, vol. 9, p. 330, 2020, doi:10.3390/ijgi9050330.

[32] Fang, H., Lafarge, F., Pan, C., and Huang, H., "Floorplan generation from 3D point clouds: a space partitioning approach," ISPRS Journal of Photogrammetry and Remote Sensing, vol. 175, pp. 44–55, 2021, doi:10.1016/j.isprsjprs.2021.02.012.

[33] Liu, C., Wu, J., and Furukawa, Y., "FloorNet: a unified framework for floorplan reconstruction from 3D scans," in Computer Vision – ECCV 2018. ECCV 2018. Lecture Notes in Computer Science, vol 11210, pp. 203–219, Springer, Cham, 2018, doi:10.1007/978-3-030-01231-1_13.

[34] Wu, W., Fu, X.-M., Tang, R., Wang, Y., Qi, Y.-H., and Liu, L., "Data-driven interior plan generation for residential buildings," ACM Transactions on Graphics, vol. 38, pp. 1–12, 2019, doi:10.1145/3355089.3356556.

[35] Hu, R., Huang, Z., Tang, Y., Van Kaick, O., Zhang, H., and Huang, H., "Graph2Plan: learning floorplan generation from layout graphs," ACM Transactions on Graphics, vol. 39, 2020, doi:10.1145/3386569.3392391.

[36] Azizi, V., Usman, M., Zhou, H., Faloutsos, P., and Kapadia, M., "Graph-based generative representation learning of semantically and behaviorally augmented floorplans," The Visual Computer, 2021, doi:10.1007/s00371-021-02155-w.

[37] Khade, R., Jariwala, K., Chattopadhyay, C., and Pal, U., "A rotation and scale invariant approach for multi-oriented floor plan image retrieval," Pattern Recognition Letters, vol. 145, pp. 1–7, 2021, doi:10.1016/j.patrec.2021.01.020.

[38] Sharma, D. and Chattopadhyay, C., "High-level feature aggregation for fine-grained architectural floor plan retrieval," IET Computer Vision, vol. 12, pp. 702–709, 2018, doi:10.1049/iet-cvi.2017.0581.

[39] Renton, G., Héroux, P., Gaüzère, B., and Adam, S., "Graph neural network for symbol detection on document images," in 2019 International Conference on Document Analysis and Recognition Workshops, (Sydney, NSW, Australia), pp. 62–67, IEEE, 2019, doi:10.1109/ICDARW.2019.00016.

[40] Rezvanifar, A., Cote, M., and Albu, A. B., "Symbol spotting on digital architectural floor plans using a deep learning-based framework," in 2020 IEEE/CVF Conference on Computer Vision and Pattern Recognition Workshops, (Seattle, WA, USA), pp. 2419–2428, IEEE, 2020, doi:10.1109/CVPRW50498.2020.00292.

[41] Shio, A. and Aoki, Y., "Sketch plan: a prototype system for interpreting hand-sketched floor plans," Systems and Computers in Japan, vol. 31, pp. 10–18, 2000, doi:10.1002/(SICI)1520-684X(200006)31:6<10::AID-SCJ2>3.0.CO;2-Y.

[42] Weber, M., Liwicki, M., and Dengel, A., "a.SCAtch - a sketch-based retrieval for architectural floor plans," in 2010 12th International Conference on Frontiers in Handwriting Recognition, (Kolkata, India), pp. 289–294, IEEE, 2010, doi:10.1109/ICFHR.2010.122.

[43] Ahmed, S., Weber, M., Liwicki, M., Langenhan, C., Dengel, A., and Petzold, F., "Automatic analysis and sketch-based retrieval of architectural floor plans," Pattern Recognition Letters, vol. 35, pp. 91–100, 2013, doi:10.1016/j.patrec.2013.04.005.

[44] Nkolika J., P., Okagbue, H. I., Obasi, E. C., and Akinola, A. O., "Review on the application of artificial neural networks in real estate valuation," International Journal of Advanced Trends in Computer Science and Engineering, vol. 9, pp. 2918 – 2925, 2020, doi:10.30534/ijatcse/2020/66932020.

[45] Goncu, C., Madugalla, A., Marinai, S., and Marriott, K., "Accessible on-line floor plans," in Proceedings of the 24th International Conference on World Wide Web, (Republic and Canton of Geneva, Switzerland), pp. 388–398, International World Wide Web Conferences Steering Committee, 2015, doi:10.1145/2736277.2741660.

[46] Riedinger, C., Jordan, M., and Tabia, H., "3D models over the centuries: from old floor plans to 3D representation," in 2014 International Conference on 3D Imaging, pp. 1–8, IEEE, 2014, doi:10.1109/IC3D.2014.7032583.

[47] Tabia, H., Riedinger, C., and Jordan, M., "Automatic reconstruction of heritage monuments from old architecture documents," Journal of Electronic Imaging, vol. 26, p. 011006, 2016, doi:10.1117/1.JEI.26.1.011006.

[48] Swaileh, W., Kotzinos, D., Ghosh, S., Jordan, M., Vu, N.-S., and Qian, Y., "Versailles-FP dataset: wall detection in ancient floor plans," in Document Analysis and Recognition – ICDAR 2021. ICDAR 2021. Lecture Notes in Computer Science, vol 12821, pp. 34–49, Springer, Cham, 2021, doi:10.1007/978-3-030-86549-8_3.

[49] Zeng, Z., Li, X., Yu, Y. K., and Fu, C.-W., "Deep floor plan recognition using a multi-task network with room-boundary-guided attention," 2019 IEEE/CVF International Conference on Computer Vision, pp. 9095–9103, 2019, doi:10.1109/ICCV.2019.00919.

[50] Zhang, Y., He, Y., Zhu, S., and Di, X., "The direction-aware, learnable, additive kernels and the adversarial network for deep floor plan recognition," arXiv, 2020,

arXiv:2001.11194.

[51] Ronneberger, O., Fischer, P., and Brox, T., "U-Net: convolutional networks for biomedical image segmentation," Medical Image Computing and Computer-Assisted Intervention – MICCAI 2015. Lecture Notes in Computer Science, vol. 9351, pp. 234–241, 2015, doi:10.1007/978-3-319-24574-4_28.

[52] Egiazarian, V., Voynov, O., Artemov, A., Volkhonskiy, D., Safin, A., Taktasheva, M., Zorin, D., and Burnaev, E., "Deep vectorization of technical drawings," in Computer Vision – ECCV 2020. ECCV 2020. Lecture Notes in Computer Science, vol 12358, pp. 582–598, Springer, Cham, 2020, doi:10.1007/978-3-030-58601-0_35.

[53] Mewada, H. K., Patel, A. V., Chaudhari, J., Mahant, K., and Vala, A., "Automatic room information retrieval and classification from floor plan using linear regression model," International Journal on Document Analysis and Recognition, vol. 23, pp. 253–266, 2020, doi:10.1007/s10032-020-00357-x.

[54] de las Heras, L.-P., Ahmed, S., Liwicki, M., Valveny, E., and Sánchez, G., "Statistical segmentation and structural recognition for floor plan interpretation," International Journal on Document Analysis and Recognition, vol. 17, pp. 221–237, 2014, doi:10.1007/s10032-013-0215-2.

[55] Keras Team, "Keras: deep learning for humans," 2021, https://keras.io (visited on 19 August 2023).

[56] Abadi, M., Agarwal, A., Barham, P., Brevdo, E., Chen, Z., Citro, C., Corrado, G. S., Davis, A., Dean, J., Devin, M., Ghemawat, S., Goodfellow, I., Harp, A., Irving, G., Isard, M., Jia, Y., Jozefowicz, R., Kaiser, L., Kudlur, M., Levenberg, J., Mane, D., Monga, R., Moore, S., Murray, D., Olah, C., Schuster, M., Shlens, J., Steiner, B., Sutskever, I., Talwar, K., Tucker, P., Vanhoucke, V., Vasudevan, V., Viegas, F., Vinyals, O., Warden, P., Wattenberg, M., Wicke, M., Yu, Y., and Zheng, X., "TensorFlow: large-scale machine learning on heterogeneous distributed systems," arXiv, 2016, arXiv:1603.04467.

[57] Bradski, G., "The OpenCV library," Dr Dobb's Journal of Software Tools, vol. 25, pp. 120–125, 2000, https://opencv.org/ (visited on 19 August 2023).

[58] Pizarro, P. N., Hitschfeld, N., Sipiran, I., and Saavedra, J. M., "Automatic floor plan analysis and recognition," Automation in Construction, vol. 140, p. 104348, 2022, doi:10.1016/j.autcon.2022.104348.

[59] Krippendorff, K., Content analysis: an introduction to its methodology. Sage, 2012, https://refhub.elsevier.com/S0141-0296(17)33552-6/h0105 (visited on 19 August 2023).

[60] Salehi, H. and Burgueño, R., "Emerging artificial intelligence methods in structural engineering," Engineering Structures, vol. 171, pp. 170–189, 2018, doi:10.1016/j.engstruct.2018.05.084.

[61] Tombre, K. and Lamiroy, B., "Pattern recognition methods for querying and browsing technical documentation," in Progress in Pattern Recognition, Image Analysis and Applications. CIARP 2008. Lecture Notes in Computer Science, vol 5197, pp. 504–518, Springer, Berlin, Heidelberg, 2008, doi:10.1007/978-3-540-85920-8_62.

[62] Ahmed, S., Liwicki, M., Weber, M., and Dengel, A., "Automatic room detection and

room labeling from architectural floor plans," in 2012 10th IAPR International Workshop on Document Analysis Systems, (Gold Coast, QLD, Australia), pp. 339–343, IEEE, 2012, doi:10.1109/DAS.2012.22.

[63] Ravagli, J., Ziran, Z., and Marinai, S., "Text recognition and classification in floor plan images," in 2019 International Conference on Document Analysis and Recognition Workshops, (Sydney, NSW, Australia), pp. 1–6, IEEE, 2019, doi:10.1109/ICDARW.2019.00006.

[64] Kim, S., Park, S., and Yu, K., "Application of style transfer in the vectorization process of floorplans (short paper)," 10th International Conference on Geographic Information Science, vol. 114, pp. 39:1–39:6, 2018, doi:10.4230/LIPIcs.GISCIENCE.2018.39.

[65] Guo, T., Zhang, H., and Wen, Y., "An improved example-driven symbol recognition approach in engineering drawings," Computers & Graphics, vol. 36, pp. 835–845, 2012, doi:10.1016/j.cag.2012.06.001.

[66] Santosh, K. C., "Complex and composite graphical symbol recognition and retrieval: a quick review," in Recent Trends in Image Processing and Pattern Recognition. RTIP2R 2016. Communications in Computer and Information Science, vol 709, pp. 3–15, Springer, Singapore, 2017, doi:10.1007/978-981-10-4859-3_1.

[67] Elyan, E., Jamieson, L., and Ali-Gombe, A., "Deep learning for symbols detection and classification in engineering drawings," Neural Networks, vol. 129, pp. 91–102, 2020, doi:10.1016/j.neunet.2020.05.025.

[68] Song, J. and Yu, K., "Framework for indoor elements classification via inductive learning on floor plan graphs," ISPRS International Journal of Geo-Information, vol. 10, no. 97, 2021, doi:10.3390/ijgi10020097.

[69] Ren, S., He, K., Girshick, R., and Sun, J., "Faster R-CNN: towards real-time object detection with region proposal networks," IEEE Transactions on Pattern Analysis and Machine Intelligence, vol. 39, pp. 1137–1149, 2017, doi:10.1109/TPAMI.2016.2577031.

[70] Redmon, J., Divvala, S., Girshick, R., and Farhadi, A., "You only look once: unified, real-time object detection," in 2016 IEEE Conference on Computer Vision and Pattern Recognition, (Las Vegas, NV, USA), pp. 779–788, IEEE, 2016, doi:10.1109/CVPR.2016.91.

[71] Dong, S., Wang, W., Li, W., and Zou, K., "Vectorization of floor plans based on EdgeGAN," Information, vol. 12, p. 206, 2021, doi:10.3390/info12050206.

[72] Ah-Soon, C. and Tombre, K., "Variations on the analysis of architectural drawings," in Proceedings of the Fourth International Conference on Document Analysis and Recognition, vol. 1, (Ulm, Germany), pp. 347–351, IEEE Computer Society Press, 1997, doi:10.1109/ICDAR.1997.619869.

[73] Rendek, J., Masini, G., Dosch, P., and Tombre, K., "The search for genericity in graphics recognition applications: design issues of the Qgar software system," in Document Analysis Systems VI. DAS 2004. Lecture Notes in Computer Science, vol 3163, pp. 366–377, Springer, Berlin, Heidelberg, 2004, doi:10.1007/978-3-540-28640-0_35.

[74] Russell, B. C., Torralba, A., Murphy, K. P., and Freeman, W. T., "LabelMe: a database and web-based tool for image annotation," International Journal of Computer Vision, vol. 77, pp. 157–173, 2008, doi:10.1007/s11263-007-0090-8.

[75] de las Heras, L.-P., Terrades, O. R., Robles, S., and Sánchez, G., "CVC-FP and SGT: a new database for structural floor plan analysis and its groundtruthing tool," International Journal on Document Analysis and Recognition, vol. 18, pp. 15–30, 2015, doi:10.1007/s10032-014-0236-5.

[76] Rusiñol, M., Borràs, A., and Lladós, J., "Relational indexing of vectorial primitives for symbol spotting in line-drawing images," Pattern Recognition Letters, vol. 31, pp. 188–201, 2010, doi:10.1016/j.patrec.2009.10.002.

[77] Delalandre, M., Valveny, E., Pridmore, T., and Karatzas, D., "Generation of synthetic documents for performance evaluation of symbol recognition & spotting systems," International Journal on Document Analysis and Recognition, vol. 13, pp. 187–207, 2010, doi:10.1007/s10032-010-0120-x.

[78] Liu, C., Schwing, A. G., Kundu, K., Urtasun, R., and Fidler, S., "Rent3D: floor-plan priors for monocular layout estimation," in 2015 IEEE Conference on Computer Vision and Pattern Recognition, (Boston, MA, USA), pp. 3413–3421, IEEE, 2015, doi:10.1109/CVPR.2015.7298963.

[79] Chu, H., Wang, S., Urtasun, R., and Fidler, S., "HouseCraft: building houses from rental ads and street views," in Computer Vision – ECCV 2016. ECCV 2016. Lecture Notes in Computer Science, vol 9910, pp. 500–516, Springer, Cham, 2016, doi:10.1007/978-3-319-46466-4_30.

[80] Dodge, S., Xu, J., and Stenger, B., "Parsing floor plan images," in 2017 Fifteenth IAPR International Conference on Machine Vision Applications, (Nagoya, Japan), pp. 358–361, IEEE, 2017, doi:10.23919/MVA.2017.7986875.

[81] National Institute of Informatics, J., "LIFULL HOME'S," 2021, https://www.nii.ac.jp/dsc/idr/en/lifull (visited on 19 August 2023).

[82] Kalervo, A., Ylioinas, J., Häikiö, M., Karhu, A., and Kannala, J., "CubiCasa5K: a dataset and an improved multi-task model for floorplan image analysis," in Image Analysis. SCIA 2019. Lecture Notes in Computer Science, vol 11482, pp. 28–40, Springer, Cham, 2019, doi:10.1007/978-3-030-20205-7_3.

[83] Surikov, I. Y., Nakhatovich, M. A., Belyaev, S. Y., and Savchuk, D. A., "Floor plan recognition and vectorization using combination unet, faster-rcnn, statistical component analysis and ramer-douglas-peucker," in Computing Science, Communication and Security. COMS2 2020. Communications in Computer and Information Science, vol 1235, pp. 16–28, Springer, Singapore, 2020, doi:10.1007/978-981-15-6648-6_2.

[84] Ministry of Land and Transport, K., "EAIS, electronic architectural administration information system," 2021, https://eais.go.kr (visited on 19 August 2023).

[85] Lv, X., Zhao, S., Yu, X., and Zhao, B., "Residential floor plan recognition and reconstruction," in 2021 IEEE/CVF Conference on Computer Vision and Pattern Recognition, (Virtual event), pp. 16712–16721, IEEE, 2021, doi:10.1109/CVPR46437.2021.01644.

[86] Lu, Z., Wang, T., Guo, J., Meng, W., Xiao, J., Zhang, W., and Zhang, X., "Data-driven floor plan understanding in rural residential buildings via deep recognition," Information Sciences, vol. 567, pp. 58–74, 2021, doi:10.1016/j.ins.2021.03.032.

[87] Rusiñol, M., Borràs, A., and Lladós, J., "FPLAN-POLY dataset," 2010, http://www.

cvc.uab.es/~marcal/FPLAN-POLY (visited on 16 April 2022).

[88] Delalandre, M., Valveny, E., Pridmore, T., and Karatzas, D., "SESYD dataset," 2010, http://mathieu.delalandre.free.fr/projects/sesyd (visited on 19 August 2023).

[89] de las Heras, L.-P., Terrades, O. R., Robles, S., and Sánchez, G., "CVC-FP dataset," 2015, http://dag.cvc.uab.es/resources/floorplans (visited on 19 August 2023).

[90] Liu, C., Schwing, A. G., Kundu, K., Urtasun, R., and Fidler, S., "R3D dataset," 2015, https://www.cs.toronto.edu/~fidler/projects/rent3D (visited on 16 April 2022).

[91] Chu, H., Wang, S., Urtasun, R., and Fidler, S., "SydneyHouse dataset," 2016, https://www.cs.toronto.edu/housecraft (visited on 19 August 2023).

[92] Dodge, S., Xu, J., and Stenger, B., "R-FP dataset," 2017, https://rit.rakuten.com/data_release (visited on 19 August 2023).

[93] Sharma, D., Gupta, N., Chattopadhyay, C., and Mehta, S., "ROBIN dataset," 2017, https://github.com/gesstalt/ROBIN (visited on 19 August 2023).

[94] Liu, C., Wu, J., Kohli, P., and Furukawa, Y., "R2V dataset," 2017, https://github.com/art-programmer/FloorplanTransformation (visited on 19 August 2023).

[95] Kalervo, A., Ylioinas, J., Häikiö, M., Karhu, A., and Kannala, J., "CubiCasa5K dataset," 2019, https://github.com/CubiCasa/CubiCasa5k (visited on 19 August 2023).

[96] Wu, W., Fu, X.-M., Tang, R., Wang, Y., Qi, Y.-H., and Liu, L., "RPLAN dataset," 2019, http://staff.ustc.edu.cn/~fuxm/projects/DeepLayout (visited on 19 August 2023).

[97] Seo, J., Park, H., and Choo, S., "Inference of drawing elements and space usage on architectural drawings using semantic segmentation," Applied Sciences, vol. 10, p. 7347, 2020, doi:10.3390/app10207347.

[98] Land and Housing Corporation, K., "Korea LH dataset," 2019, https://www.data.go.kr/en/data/15037046/fileData.do (visited on 19 August 2023).

[99] Goyal, S., Mistry, V., Chattopadhyay, C., and Bhatnagar, G., "BRIDGE: building plan repository for image description generation, and evaluation," in 2019 International Conference on Document Analysis and Recognition, (Sydney, NSW, Australia), pp. 1071–1076, IEEE, 2019, doi:10.1109/ICDAR.2019.00174.

[100] Li, T., Ho, D., Li, C., Zhu, D., Wang, C., and Meng, M. Q.-H., "HouseExpo: a large-scale 2D indoor layout dataset for learning-based algorithms on mobile robots," in 2020 IEEE/RSJ International Conference on Intelligent Robots and Systems, (Las Vegas, NV, USA), pp. 5839–5846, IEEE, 2020, doi:10.1109/IROS45743.2020.9341284.

[101] Li, T., Ho, D., Li, C., Zhu, D., Wang, C., and Meng, M. Q.-H., "HouseExpo dataset," 2020, https://github.com/TeaganLi/HouseExpo (visited on 19 August 2023).

[102] Simonsen, C. P., Thiesson, F. M., Philipsen, M. P., and Moeslund, T. B., "Generalizing floor plans using graph neural networks," in 2021 IEEE International Conference on Image Processing, (Anchorage, AK, USA), pp. 654–658, IEEE, 2021, doi:10.1109/ICIP42928.2021.9506514.

[103] Simonsen, C. P., Thiesson, F. M., Philipsen, M. P., and Moeslund, T. B., "RUB dataset," 2021, https://github.com/Chrps/MapGeneralization (visited on 19 Au-

gust 2023).

[104] Cherneff, J., Logcher, R., Connor, J., and Patrikalakis, N., "Knowledge-based interpretation of architectural drawings," Research in Engineering Design, vol. 3, pp. 195–210, 1992, doi:10.1007/BF01580842.

[105] Sönmez, N. O., "A review of the use of examples for automating architectural design tasks," Computer-Aided Design, vol. 96, pp. 13–30, 2018, doi:10.1016/j.cad.2017.10.005.

[106] Hori, O. and Tanigawa, S., "Raster-to-vector conversion by line fitting based on contours and skeletons," in Proceedings of 2nd International Conference on Document Analysis and Recognition (ICDAR '93), (Tsukuba, Japan), pp. 353–358, IEEE Computer Society Press, 1993, doi:10.1109/ICDAR.1993.395716.

[107] Aoki, Y., Shio, A., Arai, H., and Odaka, K., "A prototype system for interpreting hand-sketched floor plans," in Proceedings of 13th International Conference on Pattern Recognition, vol. 3, (Vienna, Austria), pp. 747–751, IEEE, 1996, doi:10.1109/ICPR.1996.547268.

[108] Lladós, J., López-Krahe, J., and Martí, E., "A system to understand hand-drawn floor plans using subgraph isomorphism and Hough transform," Machine Vision and Applications, vol. 10, pp. 150–158, 1997, doi:10.1007/s001380050068.

[109] Ryall, K., Shieber, S., Marks, J., and Mazer, M., "Semi-automatic delineation of regions in floor plans," in Proceedings of 3rd International Conference on Document Analysis and Recognition, vol. 2, (Montreal, QC, Canada), pp. 964–969, IEEE Computer Society Press, 1995, doi:10.1109/ICDAR.1995.602062.

[110] Dosch, P. and Masini, G., "Reconstruction of the 3D structure of a building from the 2D drawings of its floors," in Proceedings of the Fifth International Conference on Document Analysis and Recognition, (Bangalore, India), pp. 487–490, IEEE, 1999, doi:10.1109/ICDAR.1999.791831.

[111] Dosch, P., Tombre, K., Ah-Soon, C., and Masini, G., "A complete system for the analysis of architectural drawings," International Journal on Document Analysis and Recognition, vol. 3, pp. 102–116, 2000, doi:10.1007/PL00010901.

[112] Tombre, K., Ah-Soon, C., Dosch, P., Habed, A., and Masini, G., "Stable, robust and off-the-shelf methods for graphics recognition," in Fourteenth International Conference on Pattern Recognition, vol. 1, (Brisbane, Australia), pp. 406–408, IEEE Computer Society Press, 1998, doi:10.1109/ICPR.1998.711167.

[113] Ah-Soon, C., "A constraint network for symbol detection in architectural drawings," in Graphics Recognition Algorithms and Systems. GREC 1997. Lecture Notes in Computer Science, vol 1389, pp. 80–90, Springer, Berlin, Heidelberg, 1998, doi:10.1007/3-540-64381-8_41.

[114] Ah-Soon, C. and Tombre, K., "Network-based recognition of architectural symbols," in Advances in Pattern Recognition. SSPR /SPR 1998. Lecture Notes in Computer Science, vol 1451, pp. 252–261, Springer, Berlin, Heidelberg, 1998, doi:10.1007/BFb0033243.

[115] Tombre, K., "Analysis of engineering drawings: state of the art and challenges," in Graphics Recognition Algorithms and Systems. GREC 1997. Lecture Notes in

Computer Science, vol 1389, pp. 257–264, Springer, Berlin, Heidelberg, 1998, doi:10.1007/3-540-64381-8_54.

[116] Or, S.-h., Wong, K.-H., Yu, Y.-k., and Chang, M. M.-y., “Highly automatic approach to architectural floorplan image understanding & model generation,” Proceedings of Vision, Modeling, and Visualization 2005, pp. 25–32, 2005, https://www.cse.cuhk.edu.hk/~shor/paper/vmv05.pdf (visited on 19 August 2023).

[117] Tombre, K., Tabbone, S., Pélissier, L., Lamiroy, B., and Dosch, P., “Text/graphics separation revisited,” in Document Analysis Systems V. DAS 2002. Lecture Notes in Computer Science, vol 2423, pp. 200–211, Springer, Berlin, Heidelberg, 2002, doi:10.1007/3-540-45869-7_24.

[118] Duda, R. O. and Hart, P. E., “Use of the Hough transformation to detect lines and curves in pictures,” Communications of the ACM, vol. 15, pp. 11–15, 1972, doi:10.1145/361237.361242.

[119] Ahmed, S., Liwicki, M., Weber, M., and Dengel, A., “Improved automatic analysis of architectural floor plans,” in 2011 International Conference on Document Analysis and Recognition, (Beijing, China), pp. 864–869, IEEE, 2011, doi:10.1109/ICDAR.2011.177.

[120] Ahmed, S., Weber, M., Liwicki, M., and Dengel, A., “Text/graphics segmentation in architectural floor plans,” in 2011 International Conference on Document Analysis and Recognition, (Beijing, China), pp. 734–738, IEEE, 2011, doi:10.1109/ICDAR.2011.153.

[121] Bay, H., Ess, A., Tuytelaars, T., and Van Gool, L., “Speeded-up robust features (SURF),” Computer Vision and Image Understanding, vol. 110, pp. 346–359, 2008, doi:10.1016/j.cviu.2007.09.014.

[122] Feltes, M., Ahmed, S., Dengel, A., and Liwicki, M., “Improved contour-based corner detection for architectural floor plans,” in Graphics Recognition. Current Trends and Challenges. GREC 2013. Lecture Notes in Computer Science, vol 8746, pp. 191–203, Springer, Berlin, Heidelberg, 2014, doi:10.1007/978-3-662-44854-0_15.

[123] Tang, R., Wang, Y., Cosker, D., and Li, W., “Automatic structural scene digitalization,” PLOS ONE, vol. 12, p. e0187513, 2017, doi:10.1371/journal.pone.0187513.

[124] Pan, G., He, J., and Fang, R., “Automatic floor plan detection and recognition,” in 2017 2nd International Conference on Image, Vision and Computing, (Chengdu, China), pp. 201–205, IEEE, 2017, doi:10.1109/ICIVC.2017.7984546.

[125] De, P., “Vectorization of architectural floor plans,” in 2019 Twelfth International Conference on Contemporary Computing, (Noida, India), pp. 1–5, IEEE, 2019, doi:10.1109/IC3.2019.8844930.

[126] Sharma, D., Chattopadhyay, C., and Harit, G., “A unified framework for semantic matching of architectural floorplans,” in 2016 23rd International Conference on Pattern Recognition, (Cancún, México), pp. 2422–2427, IEEE, 2016, doi:10.1109/ICPR.2016.7899999.

[127] Barducci, A. and Marinai, S., “Object recognition in floor plans by graphs of white connected components,” in Proceedings of the 21st International Conference on Pattern Recognition, (Tsukuba, Japan), pp. 298–301, IEEE, 2012, https://ieeexplore.ieee.org/document/6460131 (visited on 19 August 2023).

[128] Ramer, U., “An iterative procedure for the polygonal approximation of plane curves,”

Computer Graphics and Image Processing, vol. 1, pp. 244–256, 1972, doi:10.1016/S0146-664X(72)80017-0.

[129] Hilaire, X. and Tombre, K., "Robust and accurate vectorization of line drawings," IEEE Transactions on Pattern Analysis and Machine Intelligence, vol. 28, pp. 890–904, 2006, doi:10.1109/TPAMI.2006.127.

[130] Li, M., Lafarge, F., and Marlet, R., "Approximating shapes in images with low-complexity polygons," in 2020 IEEE/CVF Conference on Computer Vision and Pattern Recognition, (Seattle, WA, USA), pp. 8630–8638, IEEE, 2020, doi:10.1109/CVPR42600.2020.00866.

[131] de las Heras, L.-P. and Sánchez, G., "And-or graph grammar for architectural floor plan representation, learning and recognition. a semantic, structural and hierarchical model," in Pattern Recognition and Image Analysis. IbPRIA 2011. Lecture Notes in Computer Science, vol 6669, pp. 17–24, Springer, Berlin, Heidelberg, 2011, doi:10.1007/978-3-642-21257-4_3.

[132] Leon-Garza, H., Hagras, H., Pena-Rios, A., Conway, A., and Owusu, G., "A fuzzy rule-based system using a patch-based approach for semantic segmentation in floor plans," in 2021 IEEE International Conference on Fuzzy Systems, (Luxembourg, Luxembourg), pp. 1–6, IEEE, 2021, doi:10.1109/FUZZ45933.2021.9494427.

[133] de las Heras, L.-P., Mas, J., Sanchez, G., and Valveny, E., "Wall patch-based segmentation in architectural floorplans," in 2011 International Conference on Document Analysis and Recognition, (Beijing, China), pp. 1270–1274, IEEE, 2011, doi:10.1109/ICDAR.2011.256.

[134] de las Heras, L.-P., Mas, J., Sánchez, G., and Valveny, E., "Notation-invariant patch-based wall detector in architectural floor plans," in Graphics Recognition. New Trends and Challenges. GREC 2011. Lecture Notes in Computer Science, vol 7423, pp. 79–88, Springer, Berlin, Heidelberg, 2013, doi:10.1007/978-3-642-36824-0_8.

[135] de las Heras, L.-P., Valveny, E., and Sánchez, G., "Unsupervised and notation-independent wall segmentation in floor plans using a combination of statistical and structural strategies," in Graphics Recognition. Current Trends and Challenges. GREC 2013. Lecture Notes in Computer Science, vol 8746, pp. 109–121, Springer, Berlin, Heidelberg, 2014, doi:10.1007/978-3-662-44854-0_9.

[136] de las Heras, L.-P., Terrades, O. R., and Llados, J., "Attributed graph grammar for floor plan analysis," in 2015 13th International Conference on Document Analysis and Recognition, (Tunis, Tunisia), pp. 726–730, IEEE, 2015, doi:10.1109/ICDAR.2015.7333857.

[137] Ganapathy, H., Ramu, P., and Muthuganapathy, R., "Alpha shape based design space decomposition for island failure regions in reliability based design," Structural and Multidisciplinary Optimization, vol. 52, pp. 121–136, 2015, doi:10.1007/s00158-014-1224-6.

[138] Guo, X. and Peng, Y., "Floor plan classification based on transfer learning," in 2018 IEEE 4th International Conference on Computer and Communications, (Chengdu, China), pp. 1720–1724, IEEE, 2018, doi:10.1109/CompComm.2018.8780679.

[139] Simonyan, K. and Zisserman, A., "Very deep convolutional networks for large-scale

image recognition," arXiv, 2014, arXiv:1409.1556.

[140] Park, S. and Kim, H., "3DPlanNet: generating 3D models from 2D floor plan images using ensemble methods," Electronics, vol. 10, p. 2729, 2021, doi:10.3390/electronics10222729.

[141] Evangelou, I., Savelonas, M., and Papaioannou, G., "PU learning-based recognition of structural elements in architectural floor plans," Multimedia Tools and Applications, vol. 80, pp. 13235–13252, 2021, doi:10.1007/s11042-020-10295-9.

[142] Bekker, J. and Davis, J., "Learning from positive and unlabeled data: a survey," Machine Learning, vol. 109, pp. 719–760, 2020, doi:10.1007/s10994-020-05877-5.

[143] Heo, S.-J., Chunwei, Z., and Yu, E., "Response simulation, data cleansing and restoration of dynamic and static measurements based on deep learning algorithms," International Journal of Concrete Structures and Materials, vol. 12, p. 82, 2018, doi:10.1186/s40069-018-0316-x.

[144] Peng, X., Wang, L., Wang, X., and Qiao, Y., "Bag of visual words and fusion methods for action recognition: comprehensive study and good practice," Computer Vision and Image Understanding, vol. 150, pp. 109–125, 2016, doi:10.1016/j.cviu.2016.03.013.

[145] Leon-Garza, H., Hagras, H., Pena-Rios, A., Conway, A., and Owusu, G., "An interval type-2 fuzzy-based system to create building information management models from 2D floor plan images," in 2021 IEEE International Conference on Fuzzy Systems, (Luxembourg, Luxembourg), pp. 1–7, IEEE, 2021, doi:10.1109/FUZZ45933.2021.9494464.

[146] Shelhamer, E., Long, J., and Darrell, T., "Fully convolutional networks for semantic segmentation," IEEE Transactions on Pattern Analysis and Machine Intelligence, vol. 39, pp. 640–651, 2017, doi:10.1109/TPAMI.2016.2572683.

[147] Barredo Arrieta, A., Díaz-Rodríguez, N., Del Ser, J., Bennetot, A., Tabik, S., Barbado, A., Garcia, S., Gil-Lopez, S., Molina, D., Benjamins, R., Chatila, R., and Herrera, F., "Explainable artificial intelligence (XAI): concepts, taxonomies, opportunities and challenges toward responsible AI," Information Fusion, vol. 58, pp. 82–115, 2020, doi:10.1016/j.inffus.2019.12.012.

[148] Minaee, S., Boykov, Y. Y., Porikli, F., Plaza, A. J., Kehtarnavaz, N., and Terzopoulos, D., "Image segmentation using deep learning: a survey," IEEE Transactions on Pattern Analysis and Machine Intelligence, pp. 1–1, 2021, doi:10.1109/TPAMI.2021.3059968.

[149] Yamasaki, T., Zhang, J., and Takada, Y., "Apartment structure estimation using fully convolutional networks and graph model," in Proceedings of the 2018 ACM Workshop on Multimedia for Real Estate Tech, (New York, NY, USA), pp. 1–6, ACM, 2018, doi:10.1145/3210499.3210528.

[150] Huang, W. and Zheng, H., "Architectural drawings recognition and generation through machine learning," in Proceedings of the 38th Annual Conference of the Association for Computer Aided Design in Architecture (ACADIA), (Mexico City, Mexico), pp. 156–165, 2018, doi:S0020-0255(21)00275-9/h0105.

[151] Ziran, Z. and Marinai, S., "Object detection in floor plan images," in Artificial Neural Networks in Pattern Recognition. ANNPR 2018. Lecture Notes in Computer Science, vol 11081, pp. 383–394, Springer, Cham, 2018, doi:10.1007/978-3-319-99978-4_30.

[152] Takada, Y., Inoue, N., Yamasaki, T., and Aizawa, K., “Similar floor plan retrieval featuring multi-task learning of layout type classification and room presence prediction,” in 2018 IEEE International Conference on Consumer Electronics, (Las Vegas, NV, USA), pp. 1–6, IEEE, 2018, doi:10.1109/ICCE.2018.8326163.

[153] Jang, H., Yang, J., and Yu, K., “Automatic wall detection and building topology and property of 2D floor plan (short paper),” in 10th International Conference on Geographic Information Science, (Dagstuhl, Germany), pp. 33:1–33:5, Schloss Dagstuhl–Leibniz-Zentrum fuer Informatik, 2018, doi:10.4230/LIPIcs.GIScience.2018.33.

[154] Wang, W., Dong, S., Zou, K., and LI, W.-s., “Room classification in floor plan recognition,” in 2020 4th International Conference on Advances in Image Processing, (New York, NY, USA), pp. 48–54, ACM, 2020, doi:10.1145/3441250.3441265.

[155] Murugan, G., Moyal, V., Nandankar, P., Pandithurai, O., and John Pimo, E., “A novel CNN method for the accurate spatial data recovery from digital images,” Materials Today: Proceedings, 2021, doi:10.1016/j.matpr.2021.05.351.

[156] Yamada, M., Wang, X., and Yamasaki, T., “Graph structure extraction from floor plan images and its application to similar property retrieval,” in 2021 IEEE International Conference on Consumer Electronics, (Las Vegas, NV, USA), pp. 1–5, IEEE, 2021, doi:10.1109/ICCE50685.2021.9427580.

[157] Goodfellow, I., Bengio, Y., and Courville, A., Deep learning. MIT Press, 2016, https://www.deeplearningbook.org (visited on 19 August 2023).

[158] Khan, A., Sohail, A., Zahoora, U., and Qureshi, A. S., “A survey of the recent architectures of deep convolutional neural networks,” Artificial Intelligence Review, vol. 53, pp. 5455–5516, 2020, doi:10.1007/s10462-020-09825-6.

[159] Zeiler, M. D. and Fergus, R., “Visualizing and understanding convolutional networks,” in Computer Vision – ECCV 2014. ECCV 2014. Lecture Notes in Computer Science, vol 8689, pp. 818–833, Springer, Cham, 2014, doi:10.1007/978-3-319-10590-1_53.

[160] Chen, L.-C., Zhu, Y., Papandreou, G., Schroff, F., and Adam, H., “Encoder-decoder with atrous separable convolution for semantic image segmentation,” in Computer Vision – ECCV 2018. ECCV 2018. Lecture Notes in Computer Science, vol 11211, pp. 833–851, Springer, Cham, 2018, doi:10.1007/978-3-030-01234-2_49.

[161] Girshick, R., Donahue, J., Darrell, T., and Malik, J., “Rich feature hierarchies for accurate object detection and semantic segmentation,” in 2014 IEEE Conference on Computer Vision and Pattern Recognition, (Columbus, OH, USA), pp. 580–587, IEEE, 2014, doi:10.1109/CVPR.2014.81.

[162] Gao, H., Yuan, H., Wang, Z., and Ji, S., “Pixel deconvolutional networks,” arXiv, 2017, arXiv:1705.06820.

[163] He, K., Zhang, X., Ren, S., and Sun, J., “Deep residual learning for image recognition,” in 2016 IEEE Conference on Computer Vision and Pattern Recognition, (Las Vegas, NV, USA), pp. 770–778, IEEE, 2016, doi:10.1109/CVPR.2016.90.

[164] Vaswani, A., Shazeer, N., Parmar, N., Uszkoreit, J., Jones, L., Gomez, A. N., Kaiser, L., and Polosukhin, I., “Attention is all you need,” arXiv, 2017, arXiv:1706.03762.

[165] Liu, W., Anguelov, D., Erhan, D., Szegedy, C., Reed, S., Fu, C.-Y., and Berg, A. C., “SSD: single shot multibox detector,” in Computer Vision – ECCV 2016. ECCV 2016.

Lecture Notes in Computer Science, vol 9905, pp. 21–37, Springer, Cham, 2016, doi:10.1007/978-3-319-46448-0_2.

[166] Huang, L., Wan, G., and Liu, C., "An improved parallel thinning algorithm," in Seventh International Conference on Document Analysis and Recognition, vol. 1, (Edinburgh, UK), pp. 780–783, IEEE Computer Society Press, 2003, doi:10.1109/ICDAR.2003.1227768.

[167] Harris, C. and Stephens, M., "A combined corner and edge detector," in Procedings of the Alvey Vision Conference 1988, (Manchester, UK), pp. 23.1–23.6, Alvey Vision Club, 1988, doi:10.5244/C.2.23.

[168] Duan, K., Bai, S., Xie, L., Qi, H., Huang, Q., and Tian, Q., "CenterNet: keypoint triplets for object detection," in 2019 IEEE/CVF International Conference on Computer Vision, (Seoul, Korea), pp. 6568–6577, IEEE, 2019, doi:10.1109/ICCV.2019.00667.

[169] Law, H. and Deng, J., "Cornernet: detecting objects as paired keypoints," International Journal of Computer Vision, vol. 128, pp. 642–656, 2020, doi:10.1007/s11263-019-01204-1.

[170] He, K., Gkioxari, G., Dollar, P., and Girshick, R., "Mask R-CNN," in 2017 IEEE International Conference on Computer Vision, (Venice, Italy), pp. 2980–2988, IEEE, 2017, doi:10.1109/ICCV.2017.322.

[171] Cai, Z. and Vasconcelos, N., "Cascade R-CNN: high quality object detection and instance segmentation," IEEE Transactions on Pattern Analysis and Machine Intelligence, vol. 43, pp. 1483–1498, 2021, doi:10.1109/TPAMI.2019.2956516.

[172] Redmon, J. and Farhadi, A., "YOLOv3: an incremental improvement," arXiv, 2018, arXiv:1804.02767.

[173] Bochkovskiy, A., Wang, C.-Y., and Liao, H.-Y. M., "YOLOv4: optimal speed and accuracy of object detection," arXiv, 2020, arXiv:2004.10934.

[174] Ke, T.-W., Hwang, J.-J., Liu, Z., and Yu, S. X., "Adaptive affinity fields for semantic segmentation," in Computer Vision – ECCV 2018. ECCV 2018. Lecture Notes in Computer Science, vol 11205, pp. 605–621, Munich, Germany: Springer, Cham, 2018, doi:10.1007/978-3-030-01246-5_36.

[175] Cipolla, R., Gal, Y., and Kendall, A., "Multi-task learning using uncertainty to weigh losses for scene geometry and semantics," in 2018 IEEE/CVF Conference on Computer Vision and Pattern Recognition, (Salt Lake City, UT, USA), pp. 7482–7491, IEEE, 2018, doi:10.1109/CVPR.2018.00781.

[176] Liu, Y., Cheng, M.-M., Hu, X., Wang, K., and Bai, X., "Richer convolutional features for edge detection," in 2017 IEEE Conference on Computer Vision and Pattern Recognition, (Honolulu, HI, USA), pp. 5872–5881, IEEE, 2017, doi:10.1109/CVPR.2017.622.

[177] Zhao, H., Shi, J., Qi, X., Wang, X., and Jia, J., "Pyramid scene parsing network," in 2017 IEEE Conference on Computer Vision and Pattern Recognition, (Honolulu, HI, USA), pp. 6230–6239, IEEE, 2017, doi:10.1109/CVPR.2017.660.

[178] Gori, M., Monfardini, G., and Scarselli, F., "A new model for learning in graph domains," in 2005 IEEE International Joint Conference on Neural Networks, vol. 2, (Montreal, QC, Canada), pp. 729–734, IEEE, 2005, doi:10.1109/IJCNN.2005.1555942.

[179] Scarselli, F., Gori, M., Ah Chung Tsoi, Hagenbuchner, M., and Monfardini, G., "The graph neural network model," IEEE Transactions on Neural Networks, vol. 20, pp. 61–80, 2009, doi:10.1109/TNN.2008.2005605.

[180] Wu, Z., Pan, S., Chen, F., Long, G., Zhang, C., and Yu, P. S., "A comprehensive survey on graph neural networks," IEEE Transactions on Neural Networks and Learning Systems, vol. 32, pp. 4–24, 2021, doi:10.1109/TNNLS.2020.2978386.

[181] Goodfellow, I. J., Pouget-Abadie, J., Mirza, M., Xu, B., Warde-Farley, D., Ozair, S., Courville, A., and Bengio, Y., "Generative adversarial nets," in Proceedings of the 27th International Conference on Neural Information Processing Systems - Volume 2, (Montreal, QC, Canada), pp. 2672–2680, MIT Press, 2014, doi:10.5555/2969033.2969125.

[182] Gatys, L., Ecker, A., and Bethge, M., "A neural algorithm of artistic style," Journal of Vision, vol. 16, p. 326, 2016, doi:10.1167/16.12.326.

[183] Jing, Y., Yang, Y., Feng, Z., Ye, J., Yu, Y., and Song, M., "Neural style transfer: a review," IEEE Transactions on Visualization and Computer Graphics, vol. 26, pp. 3365–3385, 2020, doi:10.1109/TVCG.2019.2921336.

[184] Mirza, M. and Osindero, S., "Conditional generative adversarial nets," in Proceedings of the 34th International Conference on Machine Learning - Volume 70, (Sydney, NSW,Australia), p. 7, 2014, arXiv:1411.1784.

[185] Odena, A., Olah, C., and Shlens, J., "Conditional image synthesis with auxiliary classifier GANs," arXiv, 2016, arXiv:1610.09585.

[186] Arjovsky, M., Chintala, S., and Bottou, L., "Wasserstein GAN," arXiv, 2017, arXiv:1701.07875.

[187] Isola, P., Zhu, J.-Y., Zhou, T., and Efros, A. A., "Image-to-image translation with conditional adversarial networks," in 2017 IEEE Conference on Computer Vision and Pattern Recognition, (Honolulu, HI, USA), pp. 5967–5976, IEEE, 2017, doi:10.1109/CVPR.2017.632.

[188] Creswell, A., White, T., Dumoulin, V., Arulkumaran, K., Sengupta, B., and Bharath, A. A., "Generative adversarial networks: an overview," IEEE Signal Processing Magazine, vol. 35, pp. 53–65, 2018, doi:10.1109/MSP.2017.2765202.

[189] Hong, Y., Hwang, U., Yoo, J., and Yoon, S., "How generative adversarial networks and their variants work," ACM Computing Surveys, vol. 52, pp. 1–43, 2019, doi:10.1145/3301282.

[190] Zhu, J.-Y., Park, T., Isola, P., and Efros, A. A., "Unpaired image-to-image translation using cycle-consistent adversarial networks," in 2017 IEEE International Conference on Computer Vision, (Venice, Italy), pp. 2242–2251, IEEE, 2017, doi:10.1109/ICCV.2017.244.

[191] Kim, T., Cha, M., Kim, H., Lee, J. K., and Kim, J., "Learning to discover cross-domain relations with generative adversarial networks," in Proceedings of the 34th International Conference on Machine Learning - Volume 70, (Sydney, NSW, Australia), pp. 1857–1865, JMLR.org, 2017, doi:10.5555/3305381.3305573.

[192] Wang, T.-C., Liu, M.-Y., Zhu, J.-Y., Tao, A., Kautz, J., and Catanzaro, B., "High-resolution image synthesis and semantic manipulation with conditional GANs," in 2018

IEEE/CVF Conference on Computer Vision and Pattern Recognition, (Venice, Italy), pp. 8798–8807, IEEE, 2018, doi:10.1109/CVPR.2018.00917.

[193] Phillips, I. and Chhabra, A., "Empirical performance evaluation of graphics recognition systems," IEEE Transactions on Pattern Analysis and Machine Intelligence, vol. 21, no. 9, pp. 849–870, 1999, doi:10.1109/34.790427.

[194] Mishra, S., Hashmi, K. A., Pagani, A., Liwicki, M., Stricker, D., and Afzal, M. Z., "Towards robust object detection in floor plan images: a data augmentation approach," 2021, doi:10.3390/app112311174.

[195] Zak, J. and Macadam, H., "Utilization of building information modeling in infrastructure's design and construction," IOP Conference Series: Materials Science and Engineering, vol. 236, p. 012108, 2017, doi:10.1088/1757-899X/236/1/012108.

[196] Pizarro, P. N., Massone, L. M., and Rojas, F. R., "Simplified shear wall building model for design optimization," Journal of Building Engineering, vol. 76, p. 107368, 2023, doi:https://doi.org/10.1016/j.jobe.2023.107368.

[197] Pizarro, P. N., Hitschfeld, N., and Sipiran, I., "MLSTRUCT-FP: A large-scale multi-unit floor plan dataset for architectural plan analysis and recognition (submitted)," Automation in Construction, aug.

[198] Lagos, R., Lafontaine, M., Bonelli, P., Boroschek, R., Guendelman, T., Massone, L. M., Saragoni, R., Rojas, F., and Yañez, F., "The quest for resilience: the Chilean practice of seismic design for reinforced concrete buildings," Earthquake Spectra, vol. 37, no. 1, pp. 26–45, 2021, doi:10.1177/8755293020970978.

[199] National Institute of Standards and Technology (NIST), "Comparison of U.S. and Chilean building code requirements and seismic design practice 1985–2010," NIST GCR 12-917-18, 2012, https://nehrp.gov/pdf/nistgcr12-917-18.pdf (visited on 19 August 2023).

[200] Pizarro, P. N., "MLSTRUCT-FP dataset," 2023, https://github.com/MLSTRUCT/MLSTRUCT-FP (visited on 19 August 2023).

[201] He, K., Zhang, X., Ren, S., and Sun, J., "Delving deep into rectifiers: surpassing human-level performance on ImageNet classification," in 2015 IEEE International Conference on Computer Vision (ICCV), (Santiago, Chile), pp. 1026–1034, IEEE, 2015, doi:10.1109/ICCV.2015.123.

[202] Pizarro, P. N., "MLSTRUCT-FP benchmarks," 2023, https://github.com/MLSTRUCT/MLSTRUCT-FP_benchmarks (visited on 19 August 2023).

[203] Egiazarian, V., Voynov, O., Artemov, A., Volkhonskiy, D., Safin, A., Taktasheva, M., Zorin, D., and Burnaev, E., "Deep vectorization of technical drawings (implementation)," 2020, https://github.com/Vahe1994/Deep-Vectorization-of-Technical-Drawings (visited on 19 August 2023).