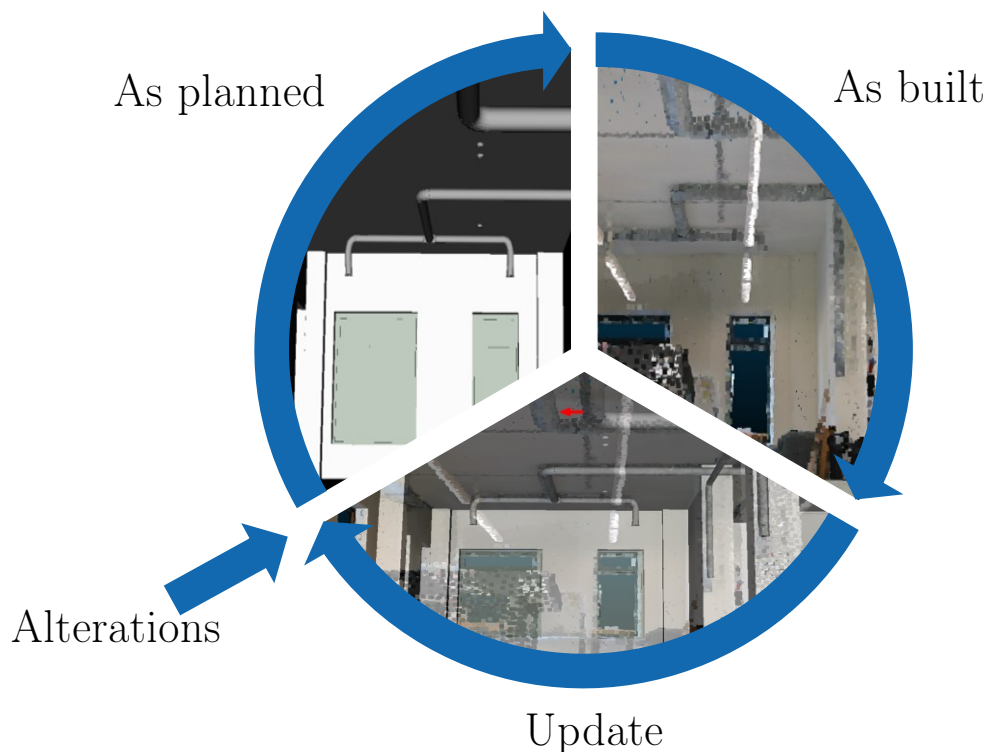


Master Thesis Encoding of geometric shapes from Building Information Modeling (BIM) using graph neural networks

Author(s): Fiona Collins ETH-Nr. 13-816-046
Supervisor: Prof. Daniel Hall
Prof. André Borrmann, Alexander Braun, Martin Ringsquandl
Date: 11. September 2020





Eidgenössische Technische Hochschule Zürich
Swiss Federal Institute of Technology Zurich

Declaration of originality

The signed declaration of originality is a component of every semester paper, Bachelor's thesis, Master's thesis and any other degree paper undertaken during the course of studies, including the respective electronic versions.

Lecturers may also require a declaration of originality for other written papers compiled for their courses.

I hereby confirm that I am the sole author of the written work here enclosed and that I have compiled it in my own words. Parts excepted are corrections of form and content by the supervisor.

Title of work (in block letters):

Encoding of geometric shapes from Building Information Modeling (BIM) using graph neural networks

Authored by (in block letters):

For papers written by groups the names of all authors are required.

Name(s):

Collins

First name(s):

Fiona Claire

With my signature I confirm that

- I have committed none of the forms of plagiarism described in the '[Citation etiquette](#)' information sheet.
- I have documented all methods, data and processes truthfully.
- I have not manipulated any data.
- I have mentioned all persons who were significant facilitators of the work.

I am aware that the work may be screened electronically for plagiarism.

Place, date

11.09.2020, Zürich

Signature(s)

F Collins

For papers written by groups the names of all authors are required. Their signatures collectively guarantee the entire content of the written paper.

Abstract

Facility management companies typically struggle with the building data that they receive upon project handover after construction completion. Unstructured, incomplete, or uncertain data, large document size and non-conformance to operational requirements are amongst the most common challenges. Restructuring, altering, or even remodelling of the digital building data is tedious, time-consuming, and often results in a considerable loss of information, higher project costs and lower operational efficiencies. Methods bundled under the term of *semantic enrichment* attempt to imitate the trained eye of architects and engineers to deduce implicitly available information in BIM models. Machine learning approaches have proven to handle such tasks similarly well or even better than rule-based inference methods. *Geometric deep learning* recently shows promising results for classification and segmentation tasks of non-Euclidian data such as 3D meshes or graphs. Capturing geometric features from Building Information Modeling (BIM) object geometries in the building unlocks further semantics and thereby adds to provide the required information needed during building operation.

The work formulated in this thesis starts from the observation of the described industry need, embeds it into a well-defined scientific context and proposes a novel toolset leveraging the power of Graph Convolutional Neural Network (GCN)s. Aiming to contribute to solving significant bottlenecks in BIM information processing, the work is validated with the industry partner Siemens AG, and implementation possibilities are discussed.

Using project handover data such as as-planned BIM models and as-built point clouds, the outcome of this project can be summarized in three contributions. A certainty estimate on the semantic correctness of BIM objects, a performance assessment on the correctness of the as-planned BIM geometries and finally a prediction of a semantic label for items unavailable in the as-planned BIM but visible in the as-built point cloud. The designed and/or trained GCN architectures proved feasible and show average prediction accuracies of 0.80 for classification and 0.77 for semantic segmentation tasks.

The results and validation talks with the industry partner show that the proposed solutions could significantly increase the amount of digitally available building information and thereby lead to a more efficient and data-driven building operation.

Keywords: BIM, Facility Management, Semantic integrity, Semantic enrichment, Geometric deep learning, Graph convolutional networks

Zusammenfassung

Viele Facility-Managementunternehmen haben Probleme mit Projektdaten, die sie bei der Projektübergabe nach Fertigstellung des Baus erhalten. Unstrukturierte, unvollständige oder ungewisse Daten, Dokumentgrösse und die Nichtübereinstimmung mit betrieblichen Anforderungen gehören zu den häufigsten Herausforderungen. Die Umstrukturierung, Änderung oder sogar Neumodellierung der digitalen Gebäudedaten ist mühsam, zeitaufwendig und führt oft zu einem erheblichen Informationsverlust, höheren Projektkosten und geringerer Betriebseffizienz. Unter dem Begriff *Semantic enrichment* gebündelte Methoden versuchen, das geschulte Auge von Architekten und Ingenieuren zu imitieren und implizit enthaltene Information in BIM-Modellen abzuleiten. Es hat sich gezeigt, dass Ansätze des maschinellen Lernens solche Aufgaben ähnlich gut oder sogar besser bewältigen können als regelbasierte Inferenzverfahren. Ansätze bekannt unter dem Begriff *geometric deep learning* zeigen in jüngster Zeit vielversprechende Ergebnisse in Klassifizierungs- und Segmentierungsaufgaben von nicht-euklidischen Daten wie 3D-Polygonnetzen oder Graphen. Die Erfassung geometrischer Merkmale aus BIM-Objektgeometrien im Gebäude ermöglicht weitere Semantik und trägt somit dazu bei, die während des Gebäudebetriebs benötigten Informationen bereitzustellen.

Diese Arbeit geht von der Beobachtung des beschriebenen Industriebedarfs aus, bettet ihn in einen klar definierten wissenschaftlichen Kontext ein und schlägt neuartige Hilfsmittel vor, die die Stärken von GCNs zunutze machen. Mit dem Ziel, einen Beitrag zur Lösung signifikanter Engpässe in der BIM-Informationsverarbeitung zu leisten, wird die Arbeit mit dem Industriepartner Siemens AG validiert, und Umsetzungsmöglichkeiten werden diskutiert.

Es werden Projektübergabedaten wie as-planned BIM Modelle und as-built Punktwolken verwendet. Das Ergebnis dieser Arbeit lässt sich in drei Teilbereiche gliedern. Eine Einschätzung der semantischen Korrektheit von BIM Objekten, eine Bewertung zur Korrektheit der as-planned BIM Geometrien und schließlich eine Vorhersage des semantischen Labels für Objekte, die in der as-built Punktwolke jedoch nicht im as-planned Model ersichtlich sind. Die entworfenen und/oder trainierten GCN-Architekturen erwiesen sich als durchführbar und zeigen durchschnittliche Prognosegenauigkeiten von 0,80 für Klassifikations- und 0,77 für semantische Segmentierungsaufgaben.

Die Ergebnisse und Validierungsgespräche mit dem Industriepartner zeigen, dass die vorgeschlagenen Lösungen die Menge der digital verfügbaren Gebäudeinformationen deutlich erhöhen und dadurch zu einem effizienteren und datengesteuerten Gebäudebetrieb führen könnten.

Acknowledgments

A big thank-you goes to Prof. Borrmann and the entire Chair of Computational Modeling and Simulation at TU Munich for welcoming me with open arms and allowing me to conduct this research in an environment of trust, mutual support and technical expertise. I want to especially thank Alexander Braun and his Digital Twin research group for always having an open door for feedback, discussion and convivial moments. Many thanks also go to Martin Ringsquandl from Siemens who supported me in a steep learning curve in technical machine learning topics. Both my supervisors always motivated me to look beyond the conventional and give room to my ideas, which I am very grateful for.

I would like to thank my official supervising professor at ETHZ Prof. Dr. Daniel Hall, who always supported me with a very open mind for innovation. When I came to his door with the self-suggested idea for this thesis, he did everything to point me in the right directions to carefully shape my thesis.

Last but not least I want to thank, Bryan Quak, Bernhard Birkner and Bernhard Müller who made my experience in Munich a memorable one despite the unfortunate circumstances of COVID-19.

Contents

Abstract	v
Zusammenfassung	vii
Acknowledgment	ix
1. Introduction	1
1.1. Thesis setting	2
1.2. Thesis reading guide	2
1.3. Motivation	2
1.4. Related work	3
1.4.1. 3D digital building data	3
1.4.2. BIM 3D-shape classification	6
1.4.3. From Point Clouds to BIM	7
2. Theoretical background and research gap	10
2.1. Preliminaries on convolutional neural networks	10
2.1.1. Universal approximation theorem	10
2.1.2. Neural networks	11
2.1.3. Key ideas of CNNs	14
2.2. Learning on irregular domains	15
2.2.1. Point cloud characteristics	15
2.2.2. Deep learning on point clouds	17
2.2.3. GCN for point cloud semantic segmentation	19
2.3. Preliminaries on graph convolutional neural networks	21
2.3.1. Basic definitions and notations	21
2.3.2. Graph construction	23
2.3.3. Spatial graph convolution	23
2.3.4. Graph coarsening	26
2.4. Research gap and research question	27
3. Methods and research guidelines	28
3.1. Guideline	28
3.1.1. Project methodology	29
4. Application case 1: BIM misclassification detection using geometric encodings	30
4.1. Problem Statement	30
4.2. Datasets	30
4.2.1. Dataset assembly	30
4.2.2. Domain knowledge for dataset variation	31

4.2.3. ModelNet10 dataset	32
4.2.4. Data preprocessing	33
4.3. Approach	34
4.3.1. Trainings setting and architecture	36
4.4. Experiments and Results	38
4.4.1. Benchmark	38
4.4.2. GCN performances on BIM dataset	39
4.4.3. Graph construction mesh vs. k-nn	41
4.4.4. Feature selection	42
5. Application case 2: Point cloud semantic segmentation for facilitated as-built model update	44
5.1. Problem statement	44
5.1.1. Datasets	45
5.1.2. Approach	49
5.1.3. Experiments and results	51
6. Discussion and Limitations	54
6.1. Technical discussion	54
6.1.1. BIM shape classification	54
6.1.2. Point Cloud Semantic Segmentation (PCSS)	56
6.2. Limitations and future work	57
6.3. Business opportunities for facility management	58
6.4. Contribution	60
7. Conclusion	62
Appendices	A1
A. Appendix Case 1	A1

List of Figures

1.1.	BIM maturity levels	4
1.2.	Geometry representation in IFC	5
1.3.	Information loss in construction project lifecycle	6
1.4.	Steps from raw point clouds to BIM	8
2.1.	A fully connected layer in a deep network	12
2.2.	Neural network with dropout	14
2.3.	Point cloud convolution problem	16
2.4.	Point cloud convolution equations	16
2.5.	Grouping strategies in PointNet++	19
2.6.	Superpoint Graph (SPG) framework	20
2.7.	L2 distances in metric and feature space - DGCNN	20
2.8.	Undirected and directed graph (Joshi, 2020)	21
2.9.	Graph propagation (University of Stanford, 2020)	24
2.10.	Computation graph (University of Stanford, 2020)	24
2.11.	Computation graph with shared parameters (University of Stanford, 2020)	25
2.12.	Schematic of graph coarsening (Loukas, 2020)	26
3.1.	CIFE Horseshoe research method - Formalizing Construction knowledge	28
3.2.	Sequential task formulation	29
4.1.	Geometry variations with LOD	32
4.2.	Graph construction variations	35
4.3.	From sampled points to connected graph; IfcStairs	36
4.4.	GCN model architecture for classification	37
4.5.	Parameter optimization: K-nn	40
4.6.	Classification report for best performing architectures on BIM dataset (K=5)	40
4.7.	Confusion matrix for best performing GCNCat architecture on BIM dataset (K=5)	41
4.8.	K-nn vs. mesh graph performances	42
5.1.	Discrepancies as-planned vs as-built visualized	44
5.2.	Labelled point cloud dataset Aspern (piping systems)	48
5.3.	DGCNN model architectures	50
6.1.	Revit prototype for semantic label enrichment	59
6.2.	Thesis contribution illustrated	61
A.1.	K-nn vs. mesh graph performances - per calss	A5

List of Tables

- 4.1. Assembled BIM dataset - train and test split 33
- 4.2. Overall accuracies compared for benchmark dataset 38
- 4.3. Comparing model complexities 39
- 4.4. Overall accuracies for base dataset and datasets with added features 42

- 5.1. Intersection over union for 8 class point cloud dataset 51
- 5.2. Intersection over union for 4 class point cloud dataset 53

- A.1. Dataset ModelNet10 - train and test split A4
- A.2. Benchmark model performances on ModelNet10 A5

Algorithms and program code

5.1. BIM face extraction	47
5.2. Point cloud labelling	48

Abbreviations and terms

2D	Two-Dimensional
3D	Three-Dimensional
as-built	Designation of a model reflecting the "as-built" state of a building after the completion of construction
as-planned	Designation of a model reflecting the "as-planned" state of a building before the start but certainly before the completion of construction
AEC	Architectural, Engineering and Construction
AI	Artificial Intelligence
API	Application Programming Interface
BAS	Building Automation Systems
BIM	Building Information Modeling
CAD	Computer Aided Design
CIFE	Centre for Integrated Facility Engineering, Stanford University
CNN	Convolutional Neural Networks
FM	Facility Management
FPS	Furthest Point Sampling algorithm
GCN	Graph Convolutional Neural Network
GUID	Global Unique Identifier
GRU	Gated Recurrent Unit
HVAC	Heating, Ventilation and Cooling
IFC	Industry Founded Classes
IOU	Intersection over Union
K-nn	K- Nearest Neighbours
LOD	Level of Detail
ML	Machine Learning
MLP	Multi-Layer Perceptron
MRG	Multi-Resolution Grouping
MSG	Multi-Scale Grouping
PCD	Point Cloud Data
PCS	Point Cloud Segmentation
PCSS	Point Cloud Semantic Segmentation
RANSAC	Random Sample Consensur
SLAM	Simultaneous Localization and Mapping
SPG	Superpoint Graph

1. Introduction

The built environment has, in many ways, a direct influence on the socio-economic structure and the quality of life of its surroundings and people inhabiting it. According to the United Nations, it gives a home to approximately 55% of the world's population today with a steeply rising tendency. On the city scale, the built environment must be diverse and interconnected enough to prevent the appearance of segregated or isolated clusters. On the infrastructure level, it should cope with the non-linear demand of a growing population. On the building scale, it should aim to provide the best possible environment for its inhabitants. Too many challenging requirements?

Next to the natural environment where everyone likes to take a resourceful breath of fresh air now and then, a world without the built environment would no longer be imaginable. The thrive of making it welcoming, enjoyable and healthy is thus at the top of priorities of the modern world.

However, the downsides of its indispensability manifests in many ways across all scales. The intensity of its use is reflected in the global yearly energy consumption of building and construction sector alone, accounting for 36% of global final energy use and 39% of energy-related CO₂ (Tomliak, 2017). Similarly when observing numbers at building scale, wherein people spend 60 to 90% of their life (Jantunen et al., 2011), the World Health Organization reports sobering numbers of health risks related to air pollution of indoor environments (WHO, 2014).

The built environment needs a transformation to be on track with the global climate ambitions formulated during the Paris Agreement. Fortunately, many opportunities exist to use efficient and novel technologies and thereby profiting from the learnings and recent developments of other sectors. Connectivity, integration and digital assets are concepts which start to resonate in the building industry, and many stakeholders are contributing to the innovations of the building process. Ideally, each phase from design over construction to operation could benefit from a sophisticated information management system, avoid redundant tasks, exploit digital techniques and use previous project knowledge for making better project choices.

In Switzerland, more than 85% of all buildings were constructed before 2000 and until today do not have any digital representation. Building models issuing from the BIM adoption phase are often incomplete and faulty. The smoothness of the way towards a fully connected and integrated built environment is therefore still highly dependant on many human and technological factors.

This thesis is a small contribution to the digitalization process and started from an observation in a facility management company struggling with a common challenge:

Customers request new operational use cases for building energy and asset management. Digital or non-digital data reaching the facility management after planning and construction, however, lacks integrity, standardization and

completeness and makes the fulfilment of such use cases a considerable business risk.

1.1. Thesis setting

Motivated by international exchange and the host universities' unique expertise, this thesis became a collaboration between ETH Zurich and TU Munich. Continuous supervision is given by the Digital Twinning research group of the Chair of Computational Modelling and Simulation at TU Munich.

On a technical level, it continues the story of a previous master thesis performed at the Institute of Construction Infrastructure Management (IBI) ETHZ *Crowd to Cloud: Simplified automatic reconstruction of digital building assets for Facility Management* (Selvakumaran, 2019).

Whereas the previous thesis investigated means of reality capture and point cloud generation, this thesis assesses in its second part new techniques of point cloud processing and information extraction.

Initially, this thesis finds its motivation at the heart of the industry partner Siemens where topics like Digital Building Twins and automated means for their generation and update have become highly relevant in recent times. The Designation of a model reflecting the "as-planned" state of a building before the start but certainly before the completion of construction (as-planned) BIM models and as-built Point Cloud Data (PCD) is provided by Siemens Building Technologies in Switzerland. Their expertise as a major building operation company allows for an industrially sounded research process. The supervision and technical exchange is offered by the research group of Siemens Cooperate Technologies in Munich.

1.2. Thesis reading guide

The red line in this thesis is lead by the CIFE Horseshoe research method Fischer (2006) described further in section 3. It aims to guide researchers in formalizing construction knowledge in the academic context.

This thesis starts with motivation from industry insights (Section 1.3), then investigates the context in academic literature (Section 1.4), before providing background knowledge (Section 2) on a tool-set (Section 3) to contribute to the solving of the problem, and finally proposes two contributions to respond to the industries needs (Section 4 and 5).

The work is discussed and validated with the industry partner, and the findings are discussed in section 6.

1.3. Motivation

The motivation and conceptualization of this work have their origin in the close observation of the typical methods and practices of a facility management company. Accurate and reliable data collecting during the design, planning and construction phases is essential for

the smooth operation and maintenance of built assets (Atkin and Brooks, 2015). Therefore, the handover of the as-built data to the operation phase upon the projects' completion is key to further management of the assets. Often the handover occurs in an unstructured way and results in labour-intensive and error-prone rework processes (Patacas et al., 2020; Jang and Collinge, 2020). Uncertainty and incompleteness demand tedious verification and validation work. There can be many differences between the as-built models and the expectations of a facility manager (Jang and Collinge, 2020). The misalignment is mostly due to a lack of communication and formulation of Facility Management (FM) requirements throughout project development phases (Pishdad-Bozorgi et al., 2018). According to Jang and Collinge (2020) and confirmed by practical observations, producing an as-built model may be more demanding for communication than producing a non-BIM as-built model for operation.

Typical FM use cases like predictive maintenance, building performance optimization, space usage optimization, and rely on structured and adequately indexed data. Inaccurate or uncertain information at handover, software interoperability issues and unclear requirement definitions ultimately result in higher project costs, lower operational efficiencies or even the unavailability of certain operational use cases.

On the other hand advances in building automation techniques as smart Heating, Ventilation and Cooling (HVAC) systems or sophisticated fire and access security systems have been at operation for the past years with (using the terminology of Jang and Collinge (2020)) non-BIM as-built models and thereby generating big amounts of organized building usage data. As a result of that, first data analytic teams are forming at the heart of facility management companies to capture the value of this data and foster predictive maintenance and asset usage optimization.

There is, however, still one key component missing. Each data point is tightly tied to the device it issues from and a sophisticated building automation system characterized by a highly engineered system network masters the communication between sensing and actuating devices. Except for the schematic network topology, which can be seen as a non-BIM as-built model of the operation phase, the exact location of the devices is not known. This information needs to come from BIM.

This thesis aims to highlight the added value that can be extracted from BIM models by researching the geometrical information of the modelled building elements. The results will hopefully contribute to an increased interest of the facility management in extracting building data from BIM and its inherent geometrical components and provide a method for the work.

1.4. Related work

1.4.1. 3D digital building data

About 20 years ago when first building projects were carried out with Computer Aided Design (CAD) software rather than a pencil, precision in planning has increased. Throughout the years, the software increasingly provided the drawers and planners with precise information management by allowing the planner to assign objects to particular layers and therefore facilitating the collaboration between stakeholders (Russell and Elger, 2008;

1. Introduction

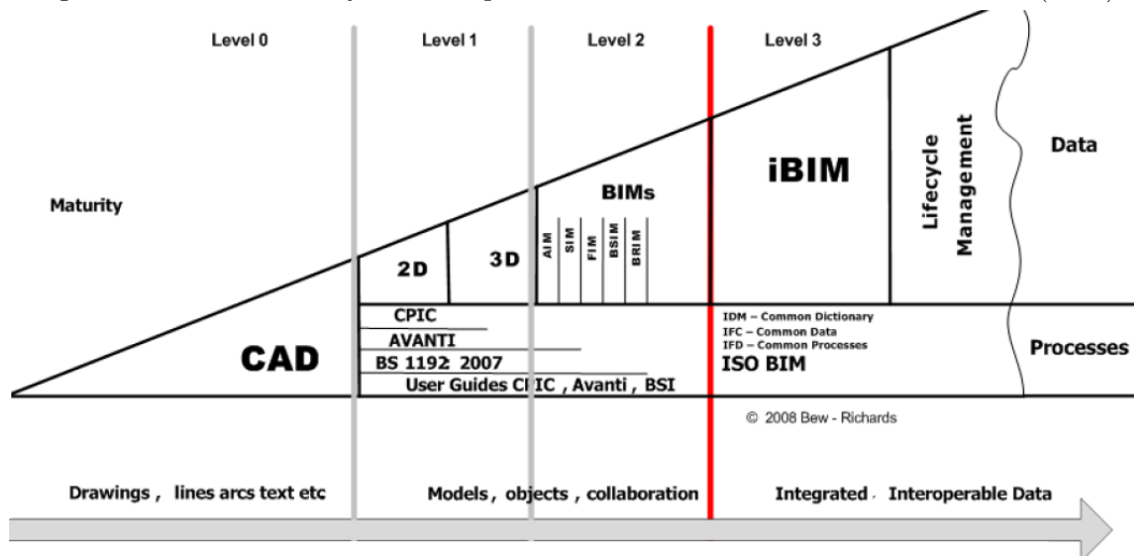
Migilinskas et al., 2013). Today BIM software products such as AutoDesk Revit, SketchUp, ArchiCAD, Vectorworks or Bentley AECOsim are helping designers, planners and constructors to visualize the building progress and design before its construction (Eastman et al., 2011). The organization of information analogue to layers in CAD has become more sophisticated and depends on the software. Depending on the degree of flexibility in the software, meaningful relationships between geometries can be established, metadata can be attached, and they can be organized in hierarchical and nested structures.

CAD has thus transformed the construction process from mere geometrical modelling to the concept of joint information modelling allowing for geometrical, numerical, algebraic and associative dependencies between objects (Ignatova et al., 2015).

With the increasing specialization of the involved parties, the growing number of stakeholders around the construction process and the growing amount of information each actor generates (Mazairac and Beetz, 2013), it requires expert knowledge to use BIM correctly. Continuous and efficient data exchange between the stakeholders has become more critical than ever before. Multidisciplinary repositories are necessary to merge models from different disciplines and to generate specialized sub-views (Mazairac and Beetz, 2013). Even with average-sized projects, the amount of data can quickly grow to a point when a data management strategy is needed. Therefore, literature often compares the role of a BIM manager to the one of an IT specialist for the three following reasons:

- Object oriented structure of BIM (Ignatova et al., 2015)
- Need of understanding of parametric modelling (Ignatova et al., 2015)
- Need of data management for accessibility, scalability and version control (Pătrăucean et al., 2015)
- Lack of systematic data structure and modelling (Ford et al., 1995; Pătrăucean et al., 2015)

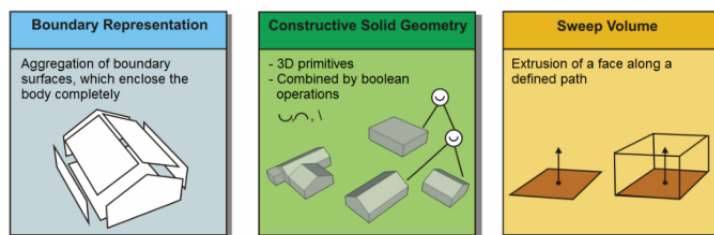
Figure 1.1.: BIM maturity levels Department of Business Innovation and Skills (2011)



Next to the growing importance of interconnected information management, geometry is still an essential part of BIM. Wu et al. (2019) states that pre-construction visualization improves the collaboration between the different stakeholders and allows for quality inspection. A BIM model with accurately modelled and placed geometry promises to provide real-scale measurements as they would be later on during construction and allows for clash detection between the different planning specialists.

Industry Founded Classes (IFC) is widely accepted as the open, platform-independent data exchange standard (ISO 16379) for BIM (Ji et al., 2011; Wu et al., 2019) and allows the transfer of models or sub-views of models both on semantic and geometrical level (Koo et al., 2019). It constitutes an essential element for an integrated and inter-operable data environment in the BIM road-map shown in Figure 1.1 and provides a common standard language (Pătrăucean et al., 2015). All physical objects are part of the base class `IfcProduct`. Products may have associated materials, shape representations, and placement in space, all of which are essential and sensitive components for an accurate as-built model. IFC represents geometry as explicit, extrusions or CSG (Constructive Solid Geometry) as shown in Figure 1.2 and does only since the release of IFC4 allow for some parametric exchanges between software. Tessellated geometry can be included in IFC4 and aims for a more compact and lossless representation of shapes from commonly used geometry file formats (buildingSMART International, 2020).

Figure 1.2.: Geometry representation in IFC Tobiáš (2015)



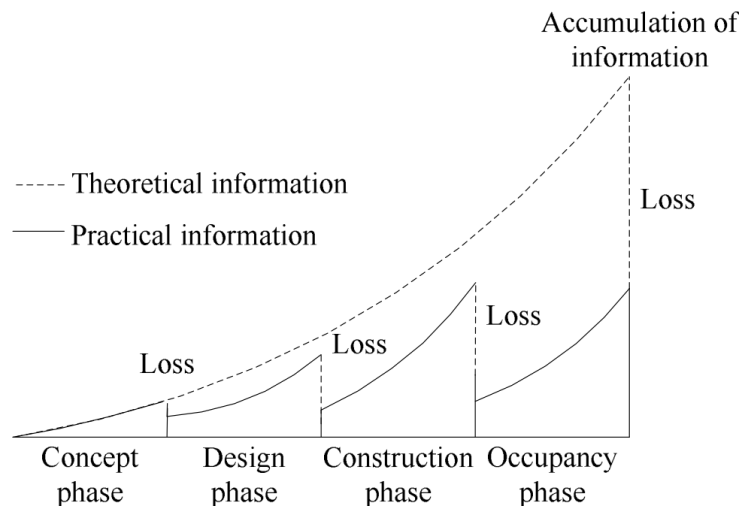
Despite many articles (Ma et al., 2020; Migilinskas et al., 2013) claiming that BIM has widely been applied in the construction industry, it is essential to note that such models are not impeccable. Figure 1.3 shows the problematic of information loss at every project hand over between phases and highlights the difference between the actual information of a building and the one captured by the current construction process. In this theses such incomplete BIM models are referred to as as-planned models of the building. They do not necessarily reflect the as-built state of the building. The difference between the two commonly appears once the physical building has started and the BIM model lags in updates. The availability of an as-built model entails correctness in its geometrical modelling and placements. If the as-built model state is not achieved during the planning and construction phase, the BIM should be corrected at a later stage. Borrmann et al. (2018) reminds that in the region of Bavaria, Germany the practical adoption of BIM is staggering and that many BIM models developed during the BIM adoption phase (level 1 and 2 in Figure 1.1) do not represent the as-built state of the building. In Switzerland, Ashworth (2015) states that in 2015 44% of FM survey respondents felt that BIM will impact on FM in the next 5 years. But only a small amount of the participating stakeholders have already

1. Introduction

had some experience with BIM. More recently Gao and Pishdad-Bozorgi (2019) state in a review about BIM enabled facility operation and maintenance, that interoperability is still a challenge. For FM an accurate as-built model state is essential since the daily business consists of operating, maintaining, locating, replacing and optimizing assets. Pătrăucean et al. (2015) mentions the problem to be complex and rooted in a technology gap for reality capture. They suggest that, given the economic impact, the automatizing of as-built BIM model acquisition from point clouds is a key objective for industrial, academic and governmental parties in the industry. Laser scanners are widely used for reality capture by collecting information on the object geometry present in the building. The generated point cloud provides a large amount of precise spatial data in short time (Macher et al., 2017) and is capable of capturing highly complex scenes (Zhang et al., 2019).

The existing building stock of which no digital representation exists could benefit from as-built modelling workflows in refurbishing, maintenance or deconstruction projects (Volk et al., 2014). Additionally, BIM models could bring significant benefits for tasks like restoration, documentation, maintenance, quality control or energy and space management Macher et al. (2017); Tang et al. (2010a)

Figure 1.3.: Information loss in construction project lifecycle Wenfa (2008)



1.4.2. BIM 3D-shape classification

While BIM continuously grows in its maturity, models with varying quality will be a common sight. Digital Twin applications aiming to make use of the interoperable and integrated data environment promised at level 3 of the BIM road-map (see Figure 1.1 must deal with unstructured, poorly standardized and highly variable BIM data. The difficulty of achieving BIM data integrity can be summarized to the bellow mentioned points and are is the origin of the data loss depicted in Figure 1.3:

- Technical problems arising during the conversion process between different BIM software tools causes data loss or errors (Kim et al., 2019).
- Human error; for example in the typing of product input or choice of abbreviations

- Utilization of dummy geometry from earlier project/LOD stages or other non-BIM modelling software (Kim et al., 2019; Krijnen, 2019)
- Mismatching geometric modelling paradigms (e.g. vendor specific) (Krijnen, 2019)
- Usage of unrelated ontologies (Krijnen, 2019)
- Insufficient hardware for data storage and processing (Krijnen, 2019)

It is common to see missuses of IFC class entities either for reasons of human mistake or lack of expertise using the authoring software. The IFC standard is still adapting to a fast-growing and hugely complex industry. Misuse of IFC classes can also arise from a custom industry entity for which no standardized IFC entity exists. Wu and Zhang (2019) states that such missuses prevent successful deployment of automation in BIM-supported tasks. Misclassified IFC entities translate into incorrect semantic information and can lead to significant rework times or even the creation of a new model for downstream application such as FM tools (Jang and Collinge, 2020; Wu et al., 2019).

Semantic enrichment is one good research topic aiming to solve the mentioned problems of interoperability and human mistakes in the BIM process (Bloch and Sacks, 2018). Motivated by the promising advantages of an integrated data environment, researches have made efforts to automatically recognize, validate and cluster and enrich BIM data (Kim et al., 2019). Bloch and Sacks (2018) proves a method of machine learning to be more performing than a rule-based approach for the task of room type classification. Kim et al. (2019) developed a prototype (see Figure 6.1) for BIM object recognition based on a machine learning approach on multiple 2D views of the 3D object geometry. Krijnen and Tamke (2015) investigates how machine learning can help automatically deduce the implicitly available information in BIM. They use geometrical and connectivity features to generate architectural insights of buildings to facilitate design and management decisions. The authors attempt searching for misclassified elements and describe their work as a starting point for future research.

(Koo et al., 2017) uses this suggestion and performs a geometrical object classification on IFC types and IFC subtypes. The geometrical features remain indirect or in other words, interpreted features of the underlying geometry.

1.4.3. From Point Clouds to BIM

The general process of developing as-built BIM from point clouds involves four main steps shown in Figure 1.4: data acquisition, preprocessing, object segmentation and finally 3D modelling in a BIM authoring software. This process, also called *Scan-to-BIM*, can have its use in two different application areas. Firstly, it serves the update of outdated BIM models from the construction phase. Such discrepancies can arise when additional renovations are made to the building, initially planned parts are left out in construction, or the placements differ from the plan. Secondly, Scan-to-BIM finds its application in virgin projects, where buildings are digitalized from scratch. Having, even a basic, BIM model of existing buildings seems to be significant for use cases such as restoration, documentation, maintenance, quality control or energy and space management (Volk et al., 2014; Juan and

1. Introduction

Hsing, 2017). However, Volk et al. (2014) states that despite well established BIM processes for new buildings, the majority of existing buildings is not maintained, refurbished or deconstructed with BIM.

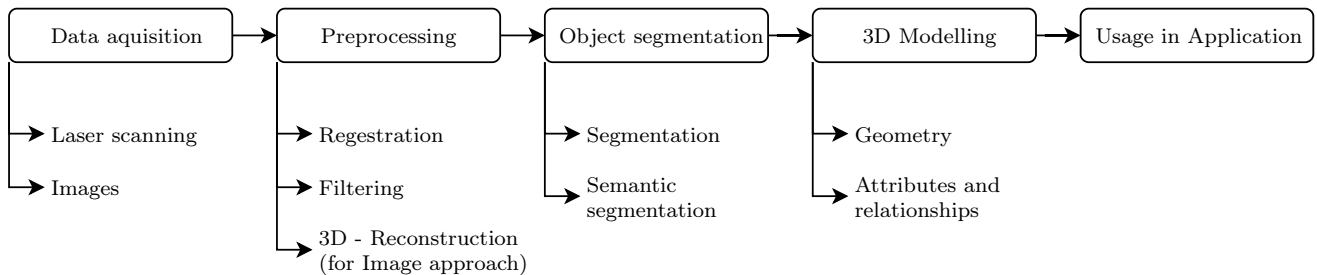


Figure 1.4.: Steps from raw point clouds to BIM

The data acquisition can be made in two different ways, outputting both a point cloud. The first and significantly more precise method uses mobile laser scanners to generate point clouds through an emitted laser pulse. Sensing devices capture the reflected laser pulse and measure the pulses' time of flight and laser wavelength. The second, image-based approach, uses computational algorithms such as SLAM to reconstruct a 3D point cloud from images. The 3D reconstruction is a famous research topic in itself. RGB-D cameras fall under the category of image-based approaches. The camera, however, embeds a depth sensor which associates a distance value to each pixel value. For that, it illuminates the object with infra-red light or LED and analyzes the reflected light patterns. This thesis sets its focus on point clouds obtained by laser scanners.

If the desired area is large, the scan needs to be performed in several steps and results in individual point clouds in the scanner's local coordinate frame (Huber and Hebert, 2003). The process of aligning them into a standard global coordinate system is called registration and remains a mainly semi-automated process (Tang et al., 2010b). A specialist identifies the approximate locations of specialized targets placed on the environment as an aid for registration (Tang et al. (2010b), NavVis (2020)). The preprocessing step may involve custom filtering or removal of unwanted point data. Typically points from moving objects, points from the surrounding non-desired environments, reflections, or sensor artefacts are filtered. Depending on the desired format, the point clouds may be converted to surface data like a triangular surface mesh by various meshing techniques (Tang et al., 2010b).

The registered point clouds are unordered and unstructured (see Section 2.2.1 for details on point cloud characteristics). As a help for subsequent 3D modelling, the point cloud gets segmented; meaning points are grouped into non-overlapping regions with one or more similar features, a process commonly known as Point Cloud Segmentation (PCS). An additional classification of those points involves the assignment of a label to each point. The simultaneous application of segmentation and classification is called PCSS. Standard methods for PCS are edge-based, region growing, shape model fitting including techniques like Random Sample Consensus (RANSAC) and cluster-based methods (Xie et al., 2019). Hybrid approaches combine two or more of these methods. As with most feature-based methods, these algorithms are mainly based on strict hand-crafted features from geometric constraints and statistical rules. They, therefore, require fine-tuning of different parameters to suit the captured scene (Xie et al., 2019). Unsupervised clustering methods usually do

not rely on such predefined patterns. A strict selection of seed points to start segmentation is not required. K-means, for example, is a widely used unsupervised clustering method and differs from region growing with respect to the cluster centres. In K-means, every point is compared to every cluster centre in each step. The point assignment to a cluster is followed by an update of the cluster itself and changes its centre.

In the modelling step, the segmented yet unstructured point cloud data is transformed into semantically meaningful BIM geometries with relationships amongst each other. Hichri et al. (2013) defines the modelling of an as-built BIM to include the three following aspects: geometrical modelling of the component, attribution of categories and material properties and finally establishing the relations between them. This step is still a mostly manual one. Modelling experts do, however say that a thorough point cloud segmentation facilitates their job considerably. (Selvakumaran, 2019)

2. Theoretical background and research gap

Generating a realistic, highly detailed semantic representation of real-world building objects is the goal of many researchers and industry partners. The amount of stakeholders around a building is growing, and so is the related informational complexity (Section 1.4.1). The tackling of this challenge requires a cross-domain skill-set of informatics, building engineering and data management.

This section is the result of a steep learning curve in the topic of machine learning on graphs and puts, the findings applicable to previous work mentioned in Section 1.4, to paper.

Section 2.1 gives background information on deep learning and CNNs and aims to equip the reader with the understanding of the basic learning techniques. In Section 2.3 this knowledge is then extended to the non-Euclidean domain meaning 3D meshes and graphs. Section 2.4 highlights the research gap identified by taking reference to the previous related work in Section 1.4 and proposes a contribution in Section 2.1 and Section 2.3 by using the newly equipped techniques described in this Section.

2.1. Preliminaries on convolutional neural networks

2.1.1. Universal approximation theorem

From a high level, mathematical perspective any machine learning architecture aims to find a function f that maps a given input x to output y . The ideal f operates on attributes of x by applying transformations and aggregations to reach y . The goal is to find function f that explains the underlying distribution in the dataset. f can thereby become arbitrarily complex.

The Universal Approximation Theorem gives a theoretical foundation to neural networks. It states that no matter what the real underlying function f is, there is a neural network, given the appropriate weights, capable of representing it. How to construct the ideal weights is left open, the theorem merely states that such a construction is possible. In other words, the theorem implies that there must be a neural network with as little as one hidden layer containing a sufficient but finite number of neurons capable of approximating any continuous function f . The derivation of the theorem is technical and the *why* it holds is not entirely understood by the community. Further discussion of the theorem thus goes beyond the scope of this thesis.

The theorem assumes that infinitely many neurons can be added to a single layer neural net. In practice, more neurons lead to a quick increase of computation demand since it would imply to find the best configuration amongst an infinite set. Alternatively, it is

often found that adding depth to the network by stacking layers subsequently, achieves good *enough* results while avoiding an explosion of computation power. Stacked layered networks allow the finding of patterns within patterns.

Generalization

The purpose of a neural network is to generalize. In other words, this means deducing a complex mapping function f from the given dataset and trusting it to perform well on unseen data. The underlying condition for generalization is that the dataset used for fitting the function f and the unseen data originate from the same underlying unknown distribution. By training the weights of function f this distribution is approximated. Neural networks are known to perform well on data that satisfies this condition, fail however as soon as applied to data lying beyond the direct range of the initial dataset.

Practically this means that a neural network generalizes only to data for which it has successfully identified the distribution in the initial training set.

2.1.2. Neural networks

Basic fully connected layer

The simplest form of a neural network is a single-layer, fully connected network. A certain input denoted as $\mathbf{z}^i \in \mathbb{R}^m$ is transformed by a set of trainable weights $\mathbf{W} \in \mathbb{R}^{n \times m}$, a bias $\mathbf{b} \in \mathbb{R}^n$ and a non linear activation function φ to form the output $\mathbf{z}^{i+1} \in \mathbb{R}^n$ (equations 2.1, 2.2). Every element of the input vector, or in the terminology of neural networks called neuron, z^i is connected by the functions 2.1 and 2.2 to every neuron in the output z^{i+1} . How much the input neuron influences the output neuron is learnt by the trainable weight and bias matrices. If the connection is activated depends on function φ .

$$y = W \cdot z^i + b \tag{2.1}$$

$$z^{i+1} = \varphi(y) \tag{2.2}$$

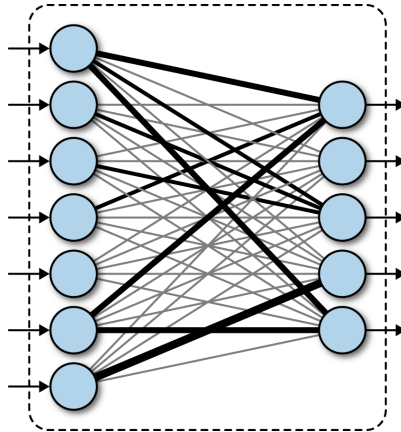
Since the output of 2.1 can be anything between $-\mathbf{inf}$ and \mathbf{inf} it must be fit in a standardized range by the activation function φ . There is a broad range of different activation functions, all having their custom implications on the output. However, they all allow for a threshold-based decision making, i.e. neuron A is activated if the values exceed a certain threshold. In this work, no experiments on varying activation functions have been made. A detailed description of activation functions and their implications is out of the scope of this thesis.

Convolutional Neural Networks

The neural networks used nowadays have increased in depth, mainly thanks to the increased amount of easily available computational resources. Equations 2.1 and 2.2 are stacked multiple times each output generating the input of the next layer.

2. Theoretical background and research gap

Figure 2.1.: A fully connected layer in a deep network Ramsundar and Zadeh (2018)



However, fully connected neural networks become very big when the dimensionality of the input increases. In other words, they do not scale very well to tasks like image segmentation. In well-performing architectures, the feature dimensions after the input layer $\mathbf{z}^i \in \mathbb{R}^m$ increase very quickly to higher dimensions. Using fully connected layers in this context leads to an explosion of parameters.

Convolutions neural networks solve this problem by applying a convolutional operator and variations of down-sampling techniques like pooling. The convolutional operator combines two signals $f(\mathbf{x})$ and $g(\mathbf{x})$ and is formally defined as in Equation 2.3. Images can be interpreted as a 2D discrete signal, which are usually represented as grid-shaped matrices. The matrix or pixel features form the d -dimensional vector \mathbf{x} .

Convolutional kernels of predefined sizes slide over the input with a given stride. The kernel itself contains a set of trainable weights which do not change during the convolution. The fixed (but trainable) weights is also referred to as the concept of weight sharing. Each input patch will be mapped to the output with the same weights. This relies on the assumption that if a patch feature is useful to compute in one spatial location, then it will also be at all other positions.

$$(f * g)(\mathbf{x}) = \iint_{\tau \in \mathbb{R}^d} f(\tau)g(\mathbf{x} + \tau)d\tau \quad (2.3)$$

For convolutional networks, the amount of parameters/weights in each layer does not depend on the input size any more but on the amount of input and respective output feature channels. The concept of convolution reduces the number of parameters to be trained. Convolutional networks have achieved astonishing results proving that the images neighbourhood information is usually sufficient to derive good features about elements in the input. Some points mentioned here are key ideas responsible for the success of Convolutional Neural Networks (CNN)s, and their importance is highlighted again in Section 2.1.3.

Pooling

Pooling layers are another important component of CNNs and are responsible for the periodical down-sampling process. As for the activation functions there are also various different pooling functions. This work focuses on the most common pooling method max pooling. The input is partitioned into non-overlapping sub-regions and for each the maximum per feature channel is computed. The underlying hypothesis is that the exact locality of the features is less important than their rough location relative to other features.

With pooling the spatial size of the input is progressively reduced and therefore the amount of trainable parameters as discussed above. The spatial reduction inline with the increase of feature channels balances the computational intensity of the network and prevents the amount of parameters from exploding. A lowering of parameter numbers is also commonly understood as a measure against overfitting since an infinite amount of parameters could simply memorize the samples and lead to poor network generalization.

Regularization strategies

There are several regularization strategies to prevent the model from overfitting. Especially, deep and large neural networks may have more internal degrees of freedom than present in the true underlying distribution of the training data. A trained model architecture relying on such weights for predictions becomes useless since there is no longer a guarantee that the model has not learnt 'false' or 'nonexistent' characteristics from within the dataset.

The process of learning parameters that can't be justified by the underlying training dataset is called overfitting.

Techniques preventing overfitting are designated as regularization techniques. One common way to control the risk of overfitting is Dropout. Dropout randomly drops neurons from a network according to a specifically assigned probability. At every training iteration, new nodes are selected at random and ignored in training. For validation of the trained model, all nodes are activated, and no dropout is applied. Figure 2.2 illustrates the process of dropping nodes. Empirically, dropout proves to be a powerful tool for regularization of network training (Ramsundar and Zadeh, 2018). Intuitively dropout forces every neuron to learn significant features and prevents the network from purely memorizing the training dataset.

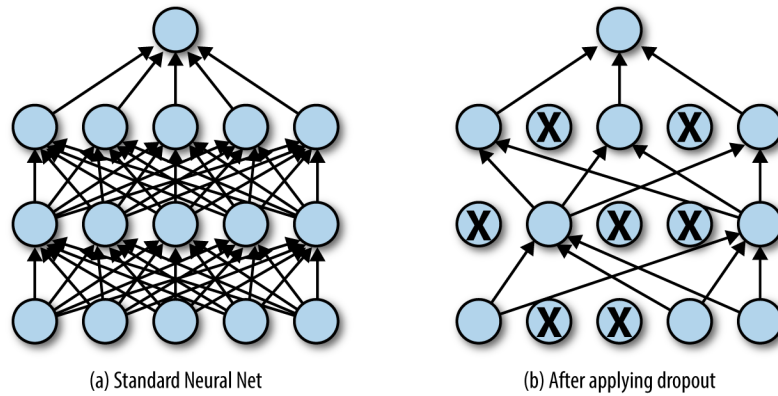
Many other regularization strategies exist, and they are a hot topic in research too. This thesis will however, not go into more detail on this topic.

Loss function and backpropagation and optimization

This thesis is limited to supervised learning techniques and will thus not elaborate on loss functions for unsupervised learning. In supervised machine learning, the loss function is at the root of the training process. The final layer of the neural network outputs a probability distribution of possible class predictions. This output is converted by the loss function to form a minimization problem, also called the objective function. Intuitively the loss function is an indicator of how much precision is lost if the predicted label is used instead of the true label. The aim is to achieve a label as close as possible to the true label

2. Theoretical background and research gap

Figure 2.2.: Neural network with dropout Ramsundar and Zadeh (2018)



such that the loss converges towards zero. The overall problem can be formulated as the solving of the objective function.

For multi-class classification problems the cross entropy loss (2.4) is most commonly used.

$$L(y, \hat{y}) = - \sum_j y_j \log(\hat{y}_j) \quad (2.4)$$

Let \hat{y} be the output of the prediction model and y be the true label. \hat{y} is the probability of the input belonging to a particular class and ranges between 0 and 1 for each class j . Due to the logarithmic behaviour of the function, the cross-entropy loss increases rapidly as the predicted probability decreases. Predicting a training sample of true class label j with a probability as low as 0.012 would result in a high loss.

Given that the connections between the neurons are differentiable at most points, gradient-based optimization methods could be used to minimize the loss. However, such techniques are generally very slow and were replaced by the backpropagation in modern deep learning. The invention of backpropagation has contributed to the immense rise in application possibilities of deep learning. The chain rule from differential calculus (Equation 2.5) is key to the loss propagation back through the network. Backpropagation leverages individual precomputed components from the forward pass and optimizes thereby the training process computationally. The gradients of the loss function with respect to the local parameters using Equation 2.5, determine the level of weight adjustments.

$$\frac{\partial y}{\partial x} = \sum_j \frac{\partial y}{\partial z_j} \frac{\partial z_j}{\partial x} \quad (2.5)$$

2.1.3. Key ideas of CNNs

The convolutional operation was one of the key inventions leading to the success of machine learning in vision tasks. The components essential for this rapid development can be summarized in the following points:

- **Weight sharing:** The kernel contains a set of trainable parameters/weights which remain the same while the kernel slides over the input. This has major benefits concerning memory consumption since the amount of trainable weights is decoupled from the input dimension.
- **Sparse connections:** Sparse connections (kernel size) instead of fully connected connections (input size) reduces the runtime of input processing significantly.
- **Translation equivariance:** The property of weight sharing within the kernels allows the model to detect a known object irrespective of its position within the image. It means that a translation in the input also leads to the equivalent translation in the feature space. This property becomes especially useful in applications such as object detection or image segmentation where an object might occur multiple times.
- **Translation invariance:** This property is the result of the pooling operation. In most cases, max pooling is used whereby the output of the layer is replaced by a summary (here the maximum value) of the surrounding neighbourhood. Even if the input is translated slightly, the result after pooling operation is not affected.

Ideally, these the representation power of CNNs could be applied to point clouds. Section 2.2 describes why this is not so straight forward and what methods exist to deal with the challenges involved.

2.2. Learning on irregular domains

Point clouds are unordered, irregular data which makes it challenging to apply convolution (Wu et al., 2019). Conventional CNN are designed to operate on structured, regular grid data such as images and leverage the key components described in section 2.1.3 to achieve astonishing results. 3D data is often represented as PCD and does not fit into the regular grid format. Applying the ideas of CNNs in to the 3D context without further thoughts would result in two main problems: a variance to ordering and desertion of shape. Section 2.2.1 goes further into detail on what the most distinctive characteristics of PCD are and illustrates the problem of conventional CNNs on PCD along with an example. In section 2.2.2 existing approaches of deep learning on point cloud data are shown.

2.2.1. Point cloud characteristics

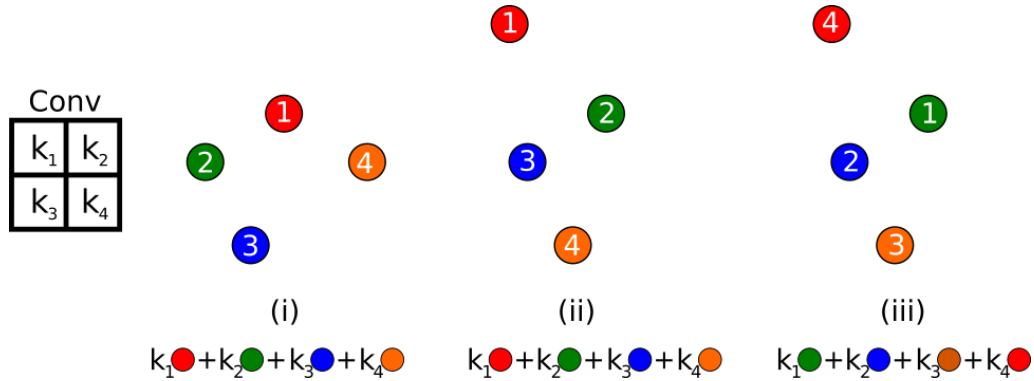
While designing and applying learning algorithms on point clouds, it is crucial to pay close attention to the following traits.

- **Unordered structure:** 3D PCD typically lacks a regular structure as known from 2D image data. Images represent a fixed sized pixel grid where the distances between pixels are always the same. In point clouds, the order amongst points and the relative positions to each other varies. The distances between the points are not fixed. Point clouds originating from laser scans can manifest huge variations in point density within the same dataset.

2. Theoretical background and research gap

- **Neighborhood similarities:** Point clouds are subspaces of R^3 for which the distance metric is not to be neglected. A neighborhood of points forms a meaningful subset and a deduced meaningful feature representation should capture such properties. Similarly as for 2D data, the feature learning must be able to take neighborhood interactions into account.
- **Rotation and translation invariance:** Feature representations should be invariant to rotation or translation transformations of the whole point set. It could be argued that, by applying certain prior domain knowledge to the problem setting, this characteristic could or could not be respected. For example: A wall and a slab can look very similar when rotated by 90% and for mere classification problems the point cloud rotation could therefore be a meaningful feature. However, when a scene contains points from both classes, the model should be capable of learning rotation invariant features too and be able to differentiate the two classes. This property is also similarly put to topic in 2D learning.
- **Data sparsity:** In 3D PCD the volume in which the data resides is highly sparse. The majority of the volume is is not occupied by any point but still relevant to the context. At the same time some regions in the volume might show high density point accumulation, whereas other regions might have a low point density. Naively overlying a 3D grid structure over the point cloud would cause unnecessary computational effort since most voxels will be empty volumes.

Figure 2.3.: Point cloud convolution problem (Thalinea, 2020)



What these listed traits imply when applying convolution to point clouds is illustrated in an example inspired by Thalinea (2020). The point clouds (i), (ii) and (iii) in Figure 2.3 each contain four points each of which has a color encoding for an associated feature. The kernel $(k_1, k_2; k_3, k_4)$ means to be applied. The numbers in each point cloud give the ordering for the convolution. The resulting computations would look as in Figure 2.4.

Figure 2.4.: Point cloud convolution equations (Thalinea, 2020)

$$\begin{aligned} \text{Conv(i)} &= k_1 \text{red} + k_2 \text{green} + k_3 \text{blue} + k_4 \text{orange} \\ \text{Conv(ii)} &= k_1 \text{red} + k_2 \text{green} + k_3 \text{blue} + k_4 \text{orange} \\ \text{Conv(iii)} &= k_1 \text{green} + k_2 \text{blue} + k_3 \text{orange} + k_4 \text{red} \end{aligned}$$

The equations imply that $\text{Conv}(i) = \text{Conv}(ii)$ however when looking at the shape of point cloud (i) and (ii) it is clear that they do not represent the same shape. The convolution leads to a shape distortion. Further, according to the equations $\text{Conv}(ii) \neq \text{Conv}(iii)$ although when looking at the point cloud the point locations as well as their feature correspondence is identical. The convolution violates the trait of permutation invariance described earlier.

2.2.2. Deep learning on point clouds

The overall goal of learning algorithms on point clouds is Three-Dimensional (3D) scene understanding. Various applications in geographical information systems, computer vision and pattern recognition sparked the interest of the research community (Zhang et al., 2019). Also for the BIM road map they promise a broad range of application possibilities .

There are various ways to deal with the irregularity of the domain. Recent developments can be summarized as voxel-based (Maturana and Scherer, 2015), multi-view (Su et al., 2015), set (Qi et al., 2017) and graph-based (Bronstein et al., 2017) methods. Zhang et al. (2019) groups the first two methods into the category of indirect methods for point cloud semantic segmentation because they require processing the PCD into an alternative format. Set and graph-based methods fall into the category of direct methods, operating directly on the raw points themselves.

Indirect learning methods

In voxel-based methods, the point space is discretized into volumetric representations to bring the data into a regular 3D density grid. Similar convolution methods to 2D images can then be applied. Discretization inevitably leads to inefficiencies due to the sparsity characteristic of point clouds (Landrieu and Simonovsky, 2018; Wang et al., 2019). Higher voxel resolution respectively results in a cubical increase in computational power. The approach also tends to discard details if the discretization is too coarse.

Multi-view-based approaches convert the 3D PCD into a set of virtual 2D snapshots (Multi-views). The segmentation of those images is later re-projected with mathematical approximations to the PCD. However, converting point clouds to images comes again with information loss and requires performing surface reconstruction, a problem seemingly as hard as semantic segmentation Landrieu and Simonovsky (2018). It has not been properly investigated how many snapshots are needed and of which spatial distribution they should originate from to get optimal prediction performance Wang et al. (2019).

Other approaches such as (Klokov and Lempitsky, 2017) or (Riegler et al., 2017) try to optimize computations on sparse domains.

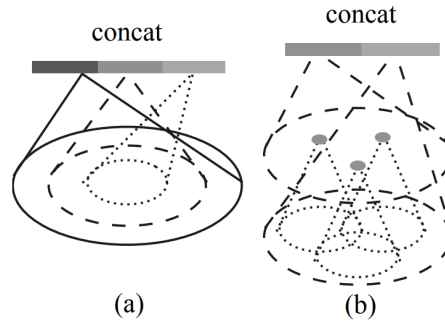
Direct learning methods

PointNet (Qi et al., 2017) solves the challenge of applying CNNs to 3D space and proposes an Machine Learning (ML) architecture for point cloud classification and semantic segmentation. PointNets' pioneering results are still used as a baseline for ML on PCD. The method respects the unique properties of point clouds and circumvents the problems illustrated in 2.2.1, by operating on each point individually for convolution and applying a symmetric aggregation function to guarantee permutation invariance. An in depth overview of the PointNet architecture is beyond the scope of this thesis and can be read in a previous thesis performed at TU Munich (Breu, 2019). The basic idea is to extract features from each point individually using a Multi-Layer Perceptron (MLP) and then aggregating all individual point features to a global point cloud signature with max pooling. Acting on each point individually and applying a symmetric aggregation function guarantees permutation invariance. To preserve translation and rotation invariance this network uses a spatial transformer network which learns a matrix to lift the absolute point coordinate features to a canonical space where translation and rotation invariance is given. However by its design PointNet does not capture local structures induced by the metric space in which the points reside. In other words, the importance of neighbouring points forming meaningful features is not captured by PointNet. Meaningful geometric features can be deduced from the spatial arrangement of single 3D points. The local neighborhood around each point must be adequately encoded such that the features can be propagate through the network (Weinmann et al., 2013).

The same authors later introduce a hierarchical network called **PointNet++** Qi et al. (2017) to include point neighborhood features. First they partition the PCD into overlapping local regions by a furthest point sampling algorithm in the metric space. Then a neighbourhood *ball* in the underlying metric space is formed. PointNet is applied to the points within each ball acting upon each point independently as described previously. Similar to conventional CNNs local features within the ball are extracted and aim to capture fine geometric structures from small neighbourhoods. The chosen pooling strategy in PointNet++ is max-pooling which chooses the highest distinct feature of all points in the neighbourhood to be propagated into subsequent layers. It can be argued that PointNet fails to pertinently capture the spatial arrangement of the neighboring points containing fine grained structural information. Points do not influence directly their neighboring points' features (Wang et al., 2019; Zhang et al., 2019). By max-pooling, global signature for the neighbourhood is generated. In 2.5 (a) it can be seen that the global signature is taken at multiple scales and then concatenated. The global neighbourhood signatures are further grouped into larger units and processed by the next hierarchical level to produce higher level features. Figure 2.5 (b) shows how the global neighborhood signatures are grouped over several resolutions. This process is repeated until features of the whole point set are obtained.

Many researchers have been aiming to improve the results of point cloud semantic segmentation by implementing variations of PointNet or PointNet++. The approaches are categorized by Zhang et al. (2019) into point-ordering, multi scale, feature fusion and GCN

Figure 2.5.: (a) Multi-Scale Grouping (MSG); (b) Multi-Resolution Grouping (MRG) (Qi et al., 2017)



methods. The detailed description of all methods lies beyond the scope of this thesis but the reader is recommended to refer to Zhang et al. (2019). As the title of this work suggests, the following sections focuses on GCNs which have shown to perform well in learning on point cloud data (Zhang et al., 2019; Xie et al., 2019).

2.2.3. GCN for point cloud semantic segmentation

Superpoint Graphs

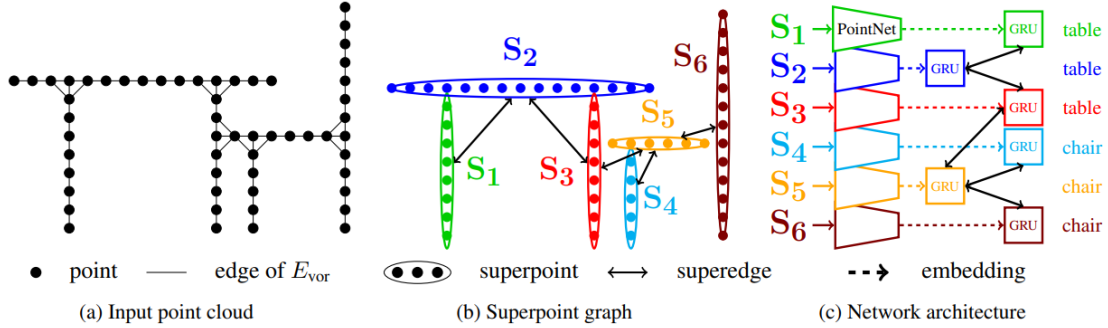
PointNet and adaptations of PointNet display good results but are next to the limited spatial neighborhood capture, limited by the size of the inputs they can handle at once since they are operating on each point individually (Landrieu and Simonovsky, 2018). Landrieu and Simonovsky (2018) proposes to represent large 3D PCD as a collection of interconnected simple shapes called superpoints. The graphs' nodes represent clusters of points with similar geometrical characteristics. It is assumed that all points in such a cluster belong to the same semantic object and represent simple shapes withing that object. The graphs' edges describe the adjacency relationship between such simple shapes and are characterized by rich edge features. Instead of classifying individual points, this approach considers geometrically simple sub-parts of objects as a whole. The authors argue that the SPG can describe in detail the relationship between adjacent sub-objects and objects which is crucial for including contextual information for classification. The size of the SPG is defined by the number of simple geometrical regions in a scene rather than the total number of points. PointNet is used for creating meaningful features for each superpoint. The global signature produced by PointNet becomes the embedding of the superpoints. A GCN propagates the embedding for contextual segmentation. The hypothesis that all points forming a superpoint belong to the same semantic class implies that all those points are labelled with the same predicted superpoint label.

Dynamic graph CNN

Wang et al. (2019) proposed the first general framework for applying GCN to point clouds. The used edge convolution generates edge features that describe the relationships between

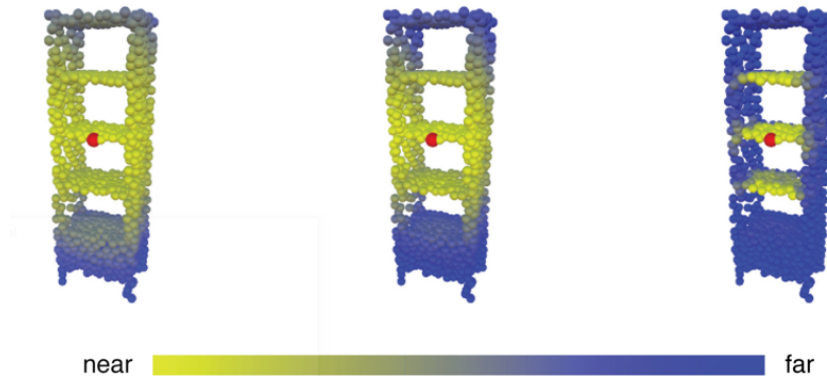
2. Theoretical background and research gap

Figure 2.6.: SPG framework (Landrieu and Simonovsky, 2018) (a) geometric partitioning of the point cloud; (b) building of the superpoint graph; (c) Each superpoint is embedded by a PointNet network. The embeddings are then refined in Gated Recurrent Unit (GRU)s by message passing along superedges to produce the final labeling



a point and its neighbours. An edge feature set for each center point is formed before those features get aggregated within each set to define the embedding of the center node for the next layer in the network. A detailed view of the model architecture can be seen on Figure 5.3. The most significant difference to other graph based methods is that DGCNN reconstructs the graph dynamically in every convolution layer. The edge construction is made in the metric space by gathering the K - Nearest Neighbours (K -nn) points and forming edges between them (this approach is further explained in section 2.3.2). DGCNN finds that a edge construction in the feature space further improves model performance. Indeed the authors are able to show that the newly formed neighborhood graph captures semantic similarities in the feature space. The color code in Figure 2.7 shows the distance found in the different model layers.

Figure 2.7.: DGCNN L2 distances in metric and feature space (Wang et al., 2019). (left) Euclidean distance in the input R^3 space; (Right) Distance in the feature space of the last layer



Graph attention convolution

Graph attention convolution (GAC) (Wang et al., 2019) brings forward a more structured feature learning process of 3D point clouds on graphs. The shared attention mechanism applies an extra matrix to the convolutional weights and allows to focus on the most relevant part of the neighbours for feature learning. The convolutional kernel of GAC can dynamically adapt to the structure of the objects by leveraging its attention weights. Specifically, this means the attentional weight of each neighbouring vertex is ideally object specific. The feature differences between vertex pairs is captured and the origin of attributing more attention to similar neighbours.

2.3. Preliminaries on graph convolutional neural networks

2.3.1. Basic definitions and notations

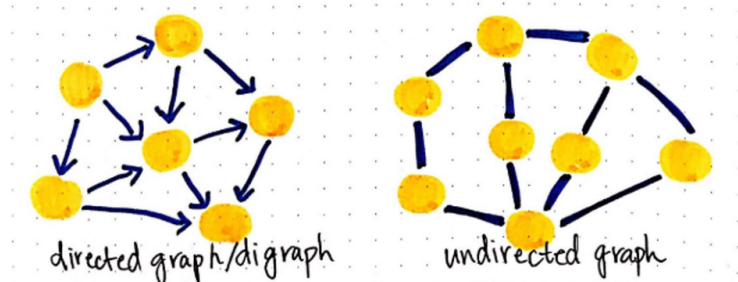
In this chapter, some basic definitions and notations of graph theory are described. The particular focus lays on how these notions allow for a generalization of conventional convolutional operators to unstructured datasets such as point clouds.

A graph can be seen as generalizations of a tree where the nodes can be connected in any possible way; a node can have an arbitrary amount of connections to other nodes and might or might not have a notion of direction in its connecting edges. Typically, one distinguishes between directed and undirected graphs where the edges either have a flow direction or not 2.8. In case the flow direction is defined from the origin node A to the destination node B, any so-called message passing is restricted to go in the direction of the edge. Directed graphs would typically be used to represent systems such as social or citation networks.

Intuitive examples of network graphs are Facebook and Twitter. In Facebook each profile is represented as a node and is liked by friend requests to other profiles in a bidirectional manner: person A sends a friend request to person B, and person B becomes upon acceptance automatically a friend of person A too. This is thus a bidirectional graph. When asked to follow a page on Twitter a link is only generated in this one direction resulting in a directional graph.

Geometric graphs as the ones used in this thesis are undirected graphs.

Figure 2.8.: Undirected and directed graph (Joshi, 2020)



The basic notion of a graph has its origin in mathematical graph theory, where it is defined as an interconnected collection of ordered pairs of objects. The pairs are denoted with v

2. Theoretical background and research gap

for vertices and e for its edges. The formal mathematical definition of a graph is visible in Equation 2.6.

$$\begin{aligned}\mathcal{G} &= (\mathcal{V}, \mathcal{E}) \\ \mathcal{G} &= (\mathcal{V}, \mathcal{E}, A) \\ \mathcal{G} &= (\mathcal{V}, \mathcal{E}, W)\end{aligned}\tag{2.6}$$

The ordered pair (V, E) is made up of two objects: a set of vertices and a set of edges. It is crucial to notice that both the sets are unordered, meaning that both sets can be arbitrarily permuted and still encode for the same graph. The edge set contains nested sets referencing the start and the end node, thereby encoding one edge at a time. In directed graphs, each edge definitions is ordered such that the first referenced node is the starting node.

When implementing a graph for means of computer science graphs are represented by an adjacency matrix where the entries represent the connectivity of the nodes (see Equation 2.7).

$$A_{i,j} = \begin{cases} 1, & \text{if link from node } i \text{ to node } j. \\ 0, & \text{otherwise.} \end{cases}\tag{2.7}$$

Note that for a directed graph the matrix is not symmetric. When dealing with big weakly connected graphs, the adjacency matrix tends to become sparse and difficult to work with. Specifically, for tasks like quickly retrieving all neighbours of a given node, a sparse matrix will perform poorly. Since real-world networks are mostly sparse, an alternative form of adjacency lists is often used.

A sparsity analysis of geometric graphs is out of scope of this thesis.

Furthermore, one distinguishes between weighted and unweighted graphs. Each edge $e_i \in E$ can be assigned with a weight giving a custom relevance to it. If there is a link between node i and j the binary adjacency matrix becomes a weighted one and $A_{i,j}$ becomes $W_{i,j}$ (see Equation 2.8).

$$W_{i,j} = \begin{cases} 1, & \text{if link from node } i \text{ to node } j. \\ 0, & \text{otherwise.} \end{cases}\tag{2.8}$$

Connected and unconnected graphs define if the graph consists of one single or more disconnected components. The corner cases are called bridge edge and articulation. In case a graph becomes disconnected when an edge is erased, this edge is called a bridge edge. If a graph becomes disconnected if we erase a node we call this node articulation node.

Each node in the graph $v_i \in V$ can be associated with a feature $f_i \in R^D$. Some datasets might not necessarily define the node connectivity and thus lack an adjacency or weight $W_{i,j}$ matrix (see Section 4.3).

2.3.2. Graph construction

When the edge weights $W_{i,j}$ are not present it is common to assign them by some function. A common way to assign weights of connecting vertices i and j is via a threshold function 2.9 (Gong, 2016).

$$W_{i,j} = \begin{cases} \exp\left(-\frac{[\text{dist}(i,j)]^2}{2\theta^2}\right), & \text{if } \text{dist}(i,j) \leq \mathcal{K}. \\ 0, & \text{otherwise.} \end{cases} \quad (2.9)$$

For some parameters θ and \mathcal{K} . The equation $\text{dist}(i,j)$ may represent a physical distance between vertices i and j or the Euclidian distance between two feature vectors describing i and j . Another strategy is to generate the graph from the existing faces in the input geometry. Mesh is a typical 3D surface representation format. 3D geometries, if not available as point clouds or parametric shapes, will be available in mesh. The vertex points of the mesh form the graph nodes and the faces the edge connectivity.

2.3.3. Spatial graph convolution

There are two approaches to graph learning; spectral graph convolutions and spatial graph convolutions. Spectral GCN rely on the eigendecomposition of the Laplacian matrix, meaning the Fourier basis needs to be recomputed for any changes in the graph. Eigendecomposition is highly complex and computationally intensive. Additionally, this eigendecomposition needs to be recomputed with every update, formation, modification to the graph. Spectral graph convolutions are therefore unsuitable for computer vision tasks. However, while being theoretically more elegant (spectral), these methods are routinely outperformed by spatial graph convolution networks (using message passing on nearest neighbours). Spectral methods do not generalize to previously unseen graphs (Klicpera et al., 2019). The scope of this thesis is thus limited to explore the graph convolution solely with the spatial graph convolutions. Spatial GCNs do not require an expensive decomposition, and the graph can quickly be updated without significant recomputation of the trained model. This makes spatial convolutions more generally applicable and more computationally efficient. Furthermore, frameworks such as Fey (2020b) and Fey (2020a) make spatial graph convolution networks easier to implement.

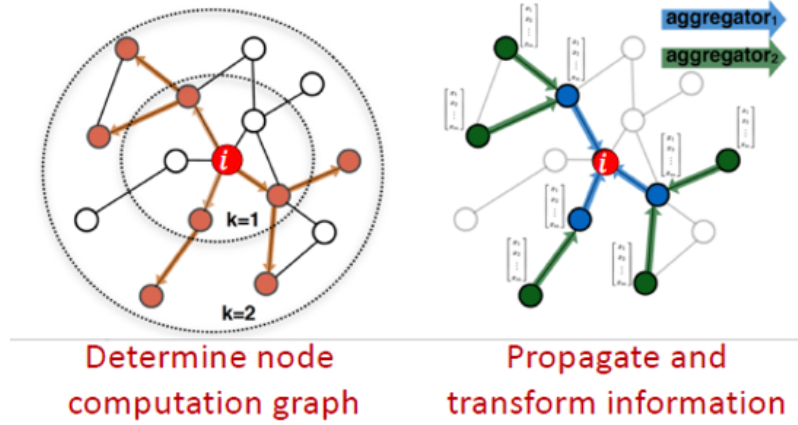
The aim is to learn how to propagate information across the graph and compute new meaningful node features. This can be done for all nodes in the original graph or a subset of nodes in which case a graph coarsening function is applied beforehand (see next chapter). The following computations can be generalized for all nodes in the given set.

Assume the graph $\mathcal{G} = (\mathcal{V}, A, X)$ where \mathcal{V} is a set of nodes $|\mathcal{V}| = n$, A is a binary adjacency

2. Theoretical background and research gap

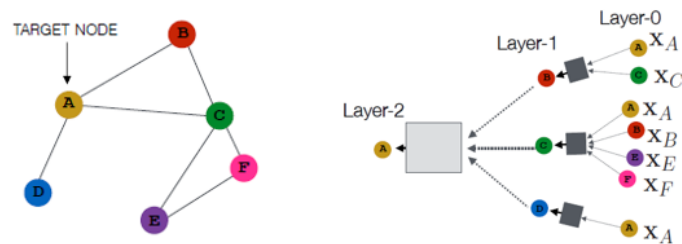
matrix and $X \in \mathbb{R}^{m \times |V|}$ is a matrix of node features. Every node is surrounded by a neighbourhood $N(v)$ to which it is connected over the set of undirected unweighted edges $e \in E$. The convolution can be decomposed into two steps, as illustrated in (2.9).

Figure 2.9.: Graph propagation (University of Stanford, 2020)



For a given target node v and its neighbourhood, ω_v a receptive field called computation graph is defined. The key idea is to generate and learn node embeddings (also called messages or hidden/latent representations) based on the local network neighbours. The grey boxes in 2.10 symbolize neural nets which learn how to aggregate the incoming messages to update the subsequent target node. The embedding of node A, for example, will be defined on it receiving information from its direct neighbours and finding the ideal aggregator to update its embedding. The computation graph is constructed based on the number of k -hop neighbours, and its hierarchies can be seen as the layers of the model. A deep model can be of arbitrary depth, meaning its layers can extend to an arbitrary depth of neighbours $\forall k \in \{1, \dots, K\}$. In every layer $\forall l \in \{1, \dots, L\}$ the corresponding neighbour's information gets transformed and aggregated to produce a feature-rich node embedding of the target node. Figure 2.10 illustrates the construction of a 2-hop computation graph of node A. Simultaneously, computation-graphs are defined for every node in the input graph based on their respective neighbourhood.

Figure 2.10.: Computation graph (University of Stanford, 2020)



Equations 2.10-2.12 formalize the described approach. The embedding of node v at layer k is denoted by h_v^k where the embedding of node v at layer $k=0$ is simply the nodes input features \mathbf{x}_v . 2.10. 2.11 formalizes the node update by applying a nonlinear transformation

σ (e.g. ReLu) to the sum of the weighted neighbourhood aggregation and the weighted embedding of node v from the previous layer \mathbf{h}_v^{k-1} . The trainable weight matrices \mathbf{W}_k and \mathbf{B}_k assign a weight to the two components and are layer specific. The aggregation operator \square (e.g. mean) bundles the neighbourhood embeddings of layer $k-1$ are. The output of 2.11 will serve as an input for layer $k+1$. 2.12 shows the final embedding after k layers of neighbourhood aggregation.

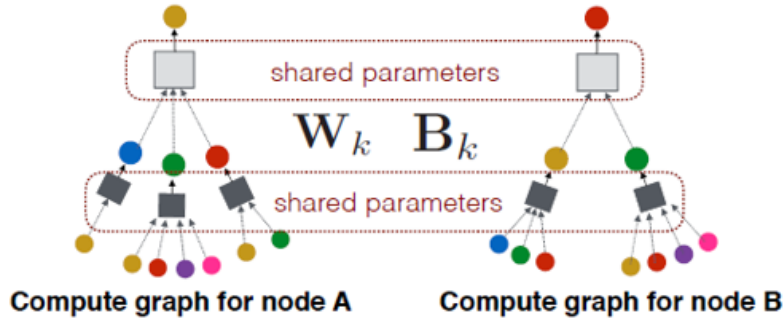
$$h_v^0 = x_v \quad (2.10)$$

$$\mathbf{h}_v^k = \sigma \left(\mathbf{W}_k \square \mathbf{h}_u^{k-1} + \mathbf{B}_k \mathbf{h}_v^{k-1} \right) \quad \forall k \in \{1, \dots, K\} \text{ and } \forall u \in N(v) \quad (2.11)$$

$$\mathbf{z}_v = \mathbf{h}_v^K \quad (2.12)$$

It is important to note that the aggregation parameters \mathbf{W}_k and \mathbf{B}_k are specific to one layer and are shared across all computation graphs as shown in figure 2.11. The sharing of weights can be compared to 2D convolutions where the same kernels/filters are slid over the entire image applying convolution to different parts of the image with the same weights (see Section 2.1.3. For graph convolutions this means that the number of model parameters does not depend on the input dimension $|\mathcal{V}|$ and the model can be generalized to unseen nodes easily. As mentioned in Section 4.3.1 this is one of the advantages of spatial convolutions over spectral convolutions. This property becomes especially useful in geometric graph learning since the number of input vertices can be very high.

Figure 2.11.: Computation graph with shared parameters (University of Stanford, 2020)



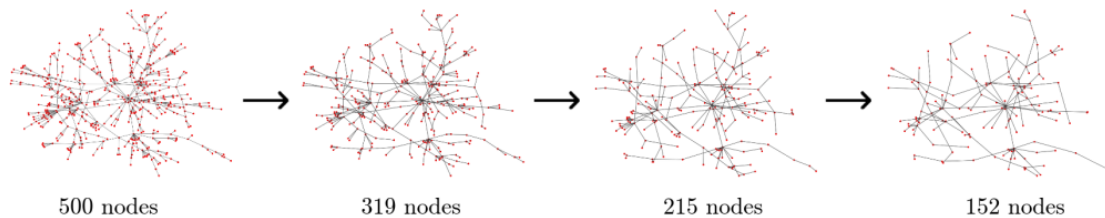
Furthermore, it should be pointed out that the varying the neighbourhood aggregation operator \square has a lot of attention from researchers. Many different approaches attempt to replace the operator and achieve better performance. One of the exciting fields is that of graph attention networks. Here each node u 's importance to node v is not anymore based on the structural properties of the graph and equally important across all messages, but gets assigned a trainable coefficient α_{vu} determining the importance of node u 's message to node v . For training, these embeddings are fed into any loss function, and a general stochastic gradient descent is run to train the weight parameters. The graph convolution network can then provide labels for every node.

To perform classification for groups of nodes(for segmentation) or even whole graph classifications, the following chapter of graph coarsening is needed.

2.3.4. Graph coarsening

The number of nodes and edges in graphs determines the computational complexity of graph algorithms. Every convolutional layer in a GCN entails $f_{\text{in}} \times f_{\text{out}} \times B$ convolutions, where $f_{\text{in}}, f_{\text{out}}$ are the in and output features and B is the amount of batches. Using GCN on large graphs can thus entail very long run times for training. Graph coarsening aims to simplify a given graph in a computationally inexpensive way. Figure 2.12 shows how an elaborate graph goes through several coarsening stages until it reaches a simple enough representation of itself. Coarsening operations are considered as successful if the global structure of the graph is maintained and only secondary details are lacking.

Figure 2.12.: Schematic of graph coarsening (Loukas, 2020)



There are several methods to coarsen a graph, and the topic is still widely ongoing in research. The most commonly used practice is described in this section and has proven to be effective experimentally.

Edge collapsing progressively reduces the number of nodes in a multilevel approach by collapsing selected edges. The two vertices connected by the selected edge then form together a multi-node. Each vertex of the resulting coarsened graph gets assigned a weight which depends on the original amount of vertices it now contains. The edges of the newly formed coarse graph similarly get associated with a weight, a measure of the number of initial edges they contain. Generally, this coarsening method is chosen because graph cut algorithms relying on a minimal edge cut approach would yield the same result if applied on the complete initial graph or the coarsened graph. This characteristic is useful for many graph applications which require graph partitioning. Please note that Figure 2.12 is for illustrative purposes and does not necessarily represent the result of the edge collapsing method.

Pooling

A more intuitive way of reducing the number of nodes in a graph is pooling. This operation is very similar to its equivalent in 2D CNNs. Effectively a subset of nodes is chosen before the features in that subset aggregate to a feature vector encoding the set. There are many different strategies for pooling on graphs for which no detailed explanation can be given in this thesis.

For classification either several subsequent pooling layers can be used or a global pooling can be one across all nodes. The final pooled embeddings allow for graph classification.

2.4. Research gap and research question

The overall goal of this thesis is to investigate how cutting-edge learning algorithms such as GCNs can facilitate the construction industry on its way towards increasingly digital workflows and interoperable data environments. Since the machine learning community has unlocked convolutional neural network methods to apply on 3D data, a continuously rising amount of work is dedicated to the topic. The sensational classification, as well as segmentation performances achieved on 2D data, motivates researchers to achieve likewise on 3D data. This motivation for technological innovation translates to the following driving research question:

1. In which ways can GCNs contribute to automating steps from as-planned to as-built model and thereby help to provide a more truthful digital representation of a building?

After extensive literature research, it is found that methods for improving the extensiveness and correctness of digital building models are various. Most focus on organisational changes, standardisation recommendations or improved means for collaboration. Only a few of them focus on the automatic data restructuring of available, completed construction data. Machine learning approaches begin to appear in the domain and show first promising results in tasks like semantic enrichment or BIM misclassification detection. However, the employment of such powerful algorithms still seems to stick to academia and is rarely found to create value in the industry. The next two, more specific questions are thus formulated as follows:

2. Can GCNs perform well on semantic enrichment of BIM models by recognising and classifying unknown BIM objects?
3. How could these results contribute to automating current workflows in the industry?

As described in section 2.2.2, the research community is highly invested in developing new methods for PCSS. The available datasets for testing and showing the effectiveness of the approaches are, however small and the available ones lack alignment to real industry needs. Application possibilities of PCSS in the industry are only briefly mentioned, concrete suggestions or prototypes on how to generate value have, to the knowledge of the author, not been elaborated on. The next question which this work aims to answer is the following:

4. How does a cutting-edge GCN model perform on an industrial dataset?
5. What results can contribute to facilitating challenges in the industry?

3. Methods and research guidelines

3.1. Guideline

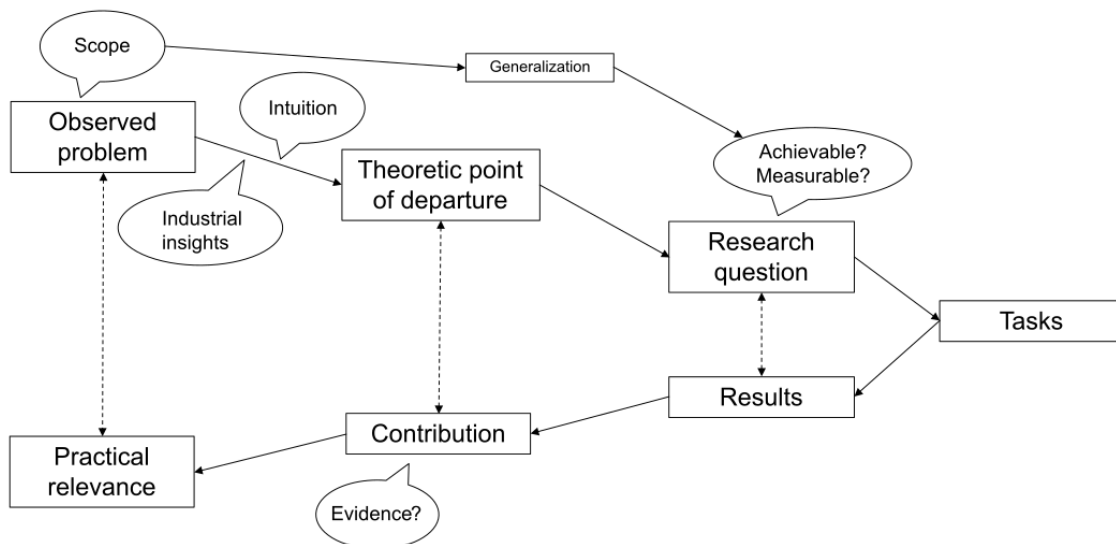
This thesis aims to follow the research methodology proposed by Fischer (2006). Slightly modified it addresses the challenge of carrying out scientifically sound research in a project-based industry like construction.

More specifically, this thesis gives special attention to the following recommendations.

- A problem which is closely observed in practice and formalized in a concise and quantifiable way is more likely to yield a productive research process. A problem definition formulated too vague and/or broad lacks clear criteria of success. Fischer (2006) gives the following example for a too vague problem definition: "4D modelling is too time-consuming."
- A research project collaborating with industry finds its significance in a test case where the new methods are deployed. Ideally, in a real project where engineers and constructors report on the value.

Figure 3.1 shows a modified version of the methodology, which serves as a guideline for this thesis.

Figure 3.1.: CIFE Horseshoe research method - Formalizing Construction knowledge



3.1.1. Project methodology

This thesis aims to address the research gap described in section 2.4 while making sure that the research results contribute to the solving of the industry need described in Section 1.

Specifically, this translates into a row of specific tasks sketched out in Figure 3.2. For both application cases, a similar task sequence is chosen: starting with data preparation and pre-processing, an iterative adaption of the model architecture, model evaluation and finally the transfer learning on an industrial dataset. The results are validated in a final step in a talk with the industry partner and the discussion on possible implementation possibilities. The PCD application case involved an additional task was the labelling of the industrial dataset.

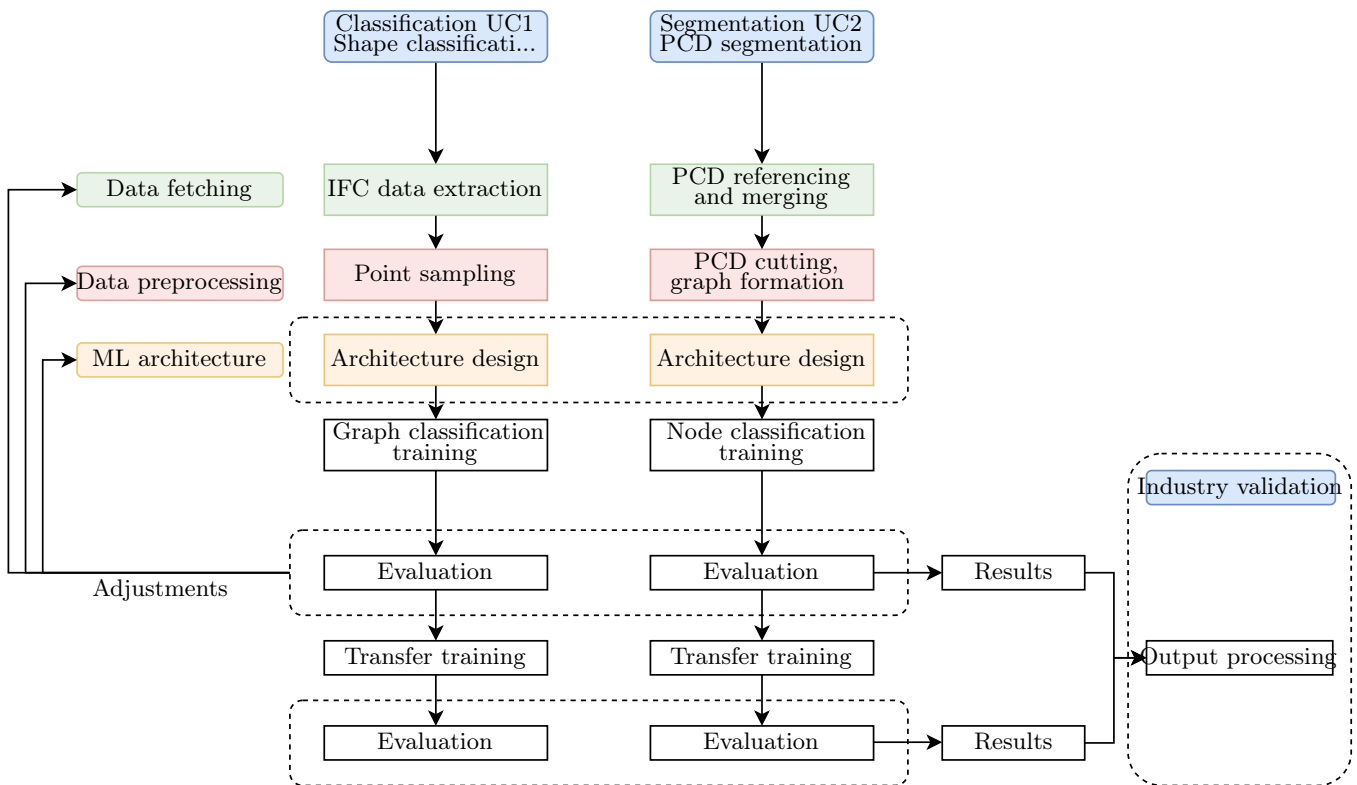


Figure 3.2.: Sequential task formulation

The technical methods and implementation choices are more carefully described in the approach section of corresponding application case (see Section for application case 1 and Section for application case 2).

4. Application case 1: BIM misclassification detection using geometric encodings

4.1. Problem Statement

Engineers in a facility management company aim to bring their operated building stock into an integrated data environment to increase the control of maintenance and operation costs and energy management (Ozturk, 2020). Only a cross-building uniform data structure will allow the platform to link data within a building or even across buildings. This section thus aims to provide a tool for BIM entity misclassification detection by employing novel geometric learning techniques.

4.2. Datasets

4.2.1. Dataset assembly

A dataset consisting of triangle mesh geometries of individual building components is created from a total of 22 IFC files. Several variations of this dataset are described in section 4.4 and make up the backbone of the research experiments. The dataset contains representations of the structural elements (IfcWall, IfcSlab, IfcColumn, IfcWindow, IfcDoor, IfcStair), the equipment classes (IfcFlowTerminal, IfcFlowSegment, IfcFlowFitting, IfcDistributionControlElement, IfcFlowController) and the interior equipment (IfcFurnishingElements). Each object type geometry is extracted from the 22 IFC files programmatically with help of the python native library IfcOpenShell (2020). The Application Programming Interface (API) of SimpleBIM (2020) is used to mark unique geometries in all IFCs to avoid extracting repetitive elements.

An overview of the class distribution can be seen in Table 4.1. Appendix A.1 shows some images of the mesh geometries contained in the dataset. For full visualization of the 3D shapes, the reader is referred to the attached documentation of this thesis.

Additional objects from the Siemens AG (2020) are added to enlarge small device geometries and make the dataset fit closer to the industries needs. The proprietary CAD tool provides planners and BIM executors a product library of Siemens building automation and fire safety devices.

As mentioned in section 2.1.1 a ML model is only as good as the data it has been trained on. It is, therefore, crucial to carefully compose a dataset and discuss the experiment results with two points in mind:

1. Limited dataset size: 22 IFC files are certainly too few for a construction-industry-wide conclusion. In construction projects, it is often decided to use a minimum of product type variations to reduce manufacturing, delivery and installation costs, i.e. having the same Door type for all Rooms in the building is favorable. This reduces the amount of objects available in the training even further. This point needs to be considered when examining the predictions for overfitting.
2. Limited dataset variation: Close attention is given to assembling the dataset with domain knowledge of the building industry 4.2.2. Model biases towards a specific building type are counteracted by increasing variation when assembling the dataset. However, since the scope of this thesis is merely a prototype, each model variation most likely lacks of representative samples in the dataset.

4.2.2. Domain knowledge for dataset variation

To maximize the variance in the dataset and therefore allowing for better generalization, special attention is given to the provenance of the IFC files. Even if the scope of this dataset is limited, the listed points can be seen as a guideline for future work.

Authoring software: Most BIM authoring tools rely on an open file exchange amongst each other and therefore allow for a configurable IFC export. Since every authoring tool has its own nuances of representing information and geometry the exported file varies in its characteristics. An example for this is the geometrical connection between two cornering walls: For a 90° Revit joins the walls by cutting each at an angle of equal size where as ArchiCAD allows for a custom wall joins. The resulting difference in geometry is important to consider. BIM models from different authoring tools are chosen, nevertheless the dominant origin of the files is Autodesk Revit (2020).

Building functionality: Office, residential, public or industrial facilities have unique design characteristics. The objects contained in the building differ (e.g. residential vs. office) and the scale of the object geometries could vary considerably. The final dataset contains geometries from all functionalities except for industrial buildings. It is estimated that industrial buildings represent a use case within themselves and should be considered separately or with special focus. The final dataset should thus contain little bias towards a specific building functionality and be able to generalize to all, excluding industrial buildings.

Level of Detail (LOD) and project phase: Depending on the required LOD of the construction project, the BIM models can differ significantly as shown in Figure 4.1. It is common to model a building firstly at a lower LOD level when the project is still in conceptual, planning or tendering phase. The higher LOD models follow during construction or even after the project completion. The final as-built model contains the highest detail and reflect the reality as well as possible. A well trained ML model, would ideally detect a door independently of the difference in LOD shape. A detailed analysis of the effects of

4. Application case 1: BIM misclassification detection using geometric encodings

LOD on the prediction performance is beyond the scope of this thesis. Nevertheless the dataset contains geometries from different LOD levels.

Families: It is common practise to define object families in a BIM authoring tool for planning facilitation. Thereby a geometry is drawn once and might have several semantic sub-types with only slight variation in the geometry. When fitting the object into the project, the same family is instantiated over and over again every time receiving a new instance Global Unique Identifier (GUID). Let’s take the example of doors. When doors are planned into the BIM model, the planner might have, depending on the project budget, the choice between 3 different door types; one xcm, one ycm and the third double-winged. He or she will then instantiate the three types of doors throughout the project without modifying the doors geometry. In big office buildings this could lead to a hundred doors of the same type geometry. Thus, when preparing the dataset, special attention is given to not select too many instances from the same type since it would otherwise cause the model to overfit highly on one specific geometry. However, Ifc does not consistently store the types in semantically and there is no easy way to guarantee the uniqueness of each geometry efficiently without comparing shape for shape. The final dataset could therefore contain some redundancies.


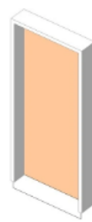
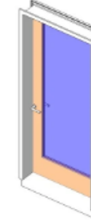
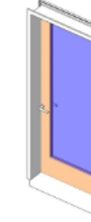

LOD 100	LOD 200	LOD 300/350	LOD 400	LOD 500
				
Concept phase	Conceptual design	Tender project	Construction state project	As built project

Figure 4.1.: Geometry variations with LOD (TeamCAD, 2020)

4.2.3. ModelNet10 dataset

This dataset consists of 12 highly-unbalanced classes of object geometries. The class distribution can be seen in Appendix A.2. The geometries are available as triangle meshes and thus need the same preprocessing step as the BIM geometries. This dataset or its more extensive version ModelNet40 are commonly used for performance evaluation of shape classification algorithms. ModelNet10 is used to verify the consistency and the functionality of the developed models before their application on the BIM datasets.

Additionally, since this dataset contains some overlapping classes with the BIM dataset, the hypothesis is made that a model trained on ModelNet10 could lead to an improved classification after being retrained on BIM datasets.

IFC Class	# unique geometries	Train dataset	Test dataset
IfcWall	384	1420	434
IfcWindow	256	1442	454
IfcFlowSegment	2539	1469	448
IfcFlowFitting	14418	2655	488
IfcDoor	572	1508	423
IfcFlowTerminal	1593	1516	449
IfcStair	302	1375	457
Ifccolumn	187	1440	439
IfcSlab	1448	1999	456
IfcFurnishingElement	2897	2732	431
IfcDistributionControlElement	1365	1853	449
IfcFlowController	729	1903	450

Table 4.1.: Assembled BIM dataset; The number of unique geometries show the class imbalance in the dataset. The train and test set are balanced by data augmentation and subsampling

4.2.4. Data preprocessing

Recently, CNNs can be applied to the 3D domain without requiring significant amounts of preprocessing steps (see 2.2.2). These advances in research allow this section to be much smaller than in other approaches. The only preprocessing step needed is the generation of 3D points from a triangular mesh geometry. The points can either originate from mesh faces directly or by sampling them at random from the underlying mesh faces.

- **FaceToEdge (Fey, 2020b)**: A mesh is typically encoded as vertex points with certain vertex connectivity defining the in-between faces. From this format, the graph is constructed. As seen in Figure 4.2 (a) the triangle vertices form the graph nodes, and the triangle faces the graph edges. It is important to note that the node degree, or the amount of edges connecting to each node, is not fixed in this case. In fact, as experiments with the BIM dataset show, the node degree can be as high as 1200 for complex geometries. The node degree could thus be an important feature to consider for this graph construction approach. One of the described experiments puts this to question and assesses the difference in performance when the node degree is added as an additional feature.

4. Application case 1: BIM misclassification detection using geometric encodings

- **PointSampling (Fey, 2020b)**: This approach uniformly samples a predefined number of points from the mesh faces according to their face area. Figure 4.2 (b) gives the corresponding illustration. A similar approach is often chosen in literature when operating on ModelNet10 dataset. Qi et al. (2017) finds that 1024 sampled points lead to good results. Thus, also in this work 1024 are sampled. To connect the nodes amongst each other and forming the graph edges a K-nn approach in the metric space is chosen (see section 2.3.2). Inherently the K-nn approach will cause all node degrees to be equal. This feature is thus not included in the feature vector.

For both approaches, the node positions are centred around the origin and normalized to the interval $(-1, 1)$. Since the dataset requires class balancing, random transformations such as rotation, translation and addition of noise are applied to the input geometries. These transformations allow for data augmentation and better generalization and are applied upon data loading. Specifically, random rotations of 360 and translations in a given interval also allow the respecting of the 3D data characteristics described in section 2.2.1. The model becomes rotation and translation invariant and avoids overfitting on the absolute spatial coordinates. Qi et al. (2017) and Wang et al. (2019) solve this problem by applying a spatial transformer network at the beginning of the architecture. However, despite being mentioned in the literature, when scrutinizing the implementation, the spatial transformer networks are often replaced by the same random translations rotations as described in this thesis.

When looking at Figure 4.2, it can be seen that the edges look very different between the two graph construction methods. The mesh graph manifests no nodes for planar surfaces (e.g. the body of a door) and might lack representation possibilities. The edge length is included in the graphs' edge weights $W_{i,j}$ according to Equation 2.9 to allow the mesh graph to counteract this shortcoming. This way, it is hoped that the graph encodings are more comparable

It is important to note that the geometry graph will look different after each construction with the PoinSampling approach, not however with teh FaceToEdge approach. The above-mentioned transformations happen when the geometries are loaded for training, at every training iteration. The second approach samples points each time at random, which results in a different graph every time. This property guarantees the respecting of permutation invariance of 3D data. With approach 1, the data augmentation for dataset balancing leads to multiple loading of the same graph. Of course, the transformations will still guarantee translation and rotation invariance. However, it might be that the model will overfit on a specific graph encoding because permutation invariance is not given.

4.3. Approach

This section aims to elaborate on the tasks sketched out in the project methodology in Figure 3.2 with technical implementation details.

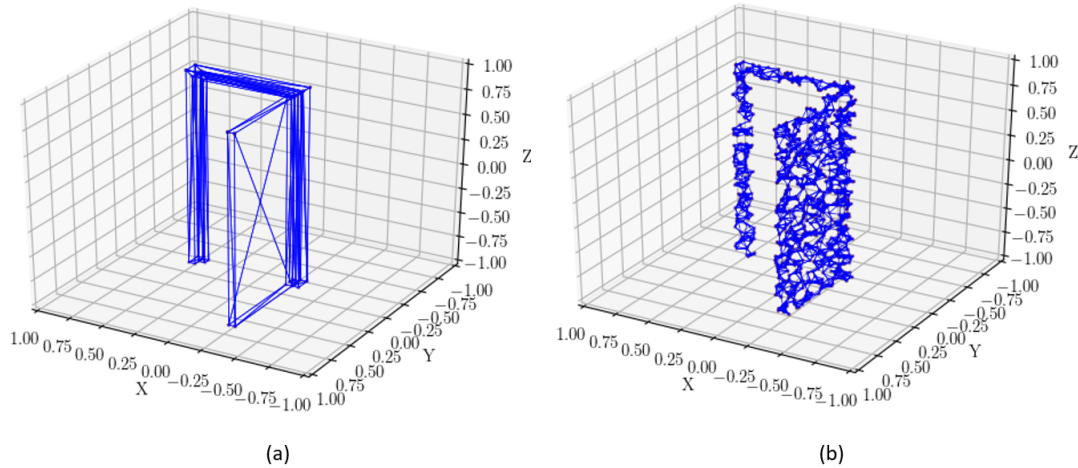


Figure 4.2.: Graph construction variations for IfcDoor. (a) Mesh vertices encode the graph nodes, mesh faces define the edge connections between the nodes; (b) Point sampling with edge definition by K-nn method

After assembling (4.2.1) and several preprocessing steps (4.2.4), an undirected, geometric adjacency graph $\mathcal{G} = (\mathcal{V}, \mathcal{E}, A)$ is connected for each object, see Figure 4.3 for illustration. A graph-based model architecture for BIM object classification is designed. Several experiments are performed, including variations in model architecture, data preprocessing and feature addition. Finally, the experiments are evaluated by the metrics of overall accuracy, f1 score, precision, recall and class confusion. For a reminder: precision shows how many selected elements are relevant (are of that class), recall shows how many relevant items are selected and the f1 score is the harmonic mean of the two. F1 score can be seen as an other measure of accuracy.

The set of vertices is defined by the amount of points sampled $|\mathcal{V}| = N_{\text{sampled}}$ or the amount of vertices in the initial mesh. Each vertex is a point with d -dim coordinates and F -dim point features. Point features could be, point coordinates in a canonical space, normal vectors with respect to the surrounding points, RGB colors from the underlying IFC-material, node degrees (in the case of a FaceToEdge graph construction) or neighborhood information of the adjacent BIM geometries. The experiments in this thesis are however limited to include the point coordinates and normal vectors. The feature additions performed in the experiments concatenate features at the global signature level and not at node level. If there is an edge $e = (i, j)$ linking vertices i and j , $\mathbf{W}_{ij} = d$, $\mathbf{A}_{ij} = 0$ otherwise, according to Equation 2.9.

It is essential to develop the new GCN architectures not on the assembled BIM dataset but on a dataset from literature. This allows for a performance assessment and a comparison with existing model architectures in literature.

4. Application case 1: BIM misclassification detection using geometric encodings

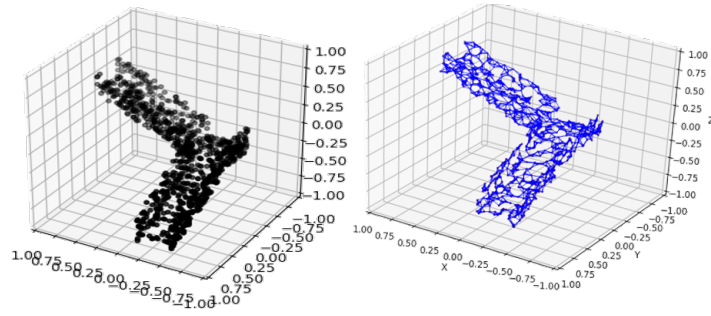


Figure 4.3.: From sampled points to connected graph; IfcStairs

4.3.1. Trainings setting and architecture

A multi-layer graph convolution architecture, as seen in Figure 4.4 is implemented and serves as a basis for various experiments in this section. The architecture itself has three variations. The first is the simplest form and consists of a 3-layer GCN where each graph convolution is followed by the activation function ReLU, and the first two convolution layers are preceded by a dropout of 0.2 during training (see Figure 4.4 (a) without blue). The model performance is found to be best with the global max pooling method. It takes the channel-wise maximum across the node dimension and leaves us with a global signature for the shape. This 1×265 -dim signature is passed onto a linear layer which maps onto the final classes. The cross-entropy loss is chosen, which forces the model to learn the mapping of the embedding nodes to C classes. For the graph convolution summation for neighborhood aggregation (see for background information) performs best. Three layers of graph convolution mean that each node receives feature messages from its 3-hop neighbours. 3-hops seems to be a good amount for these experiments. Adding further layers does not increase performance which could be explained by the fact that further layers cause feature redundancies to become important. A detailed sensitivity-analysis on the number of GCN layers is not performed in this thesis.

The second (GCNCat) and the third (GCNPool) variations have Qi et al. (2017)s reasoning on Multi-Scale Grouping (MSG) and Multi-Resolution Grouping (MRG) as a basis. The second model variation (GCNCat) shown in blue in Figure 4.4 (a) concatenates the features from GCN layer one with with the features from GCN layer two. Since after layer two, each node embedding will contain information about their second-hop neighbours too, this concatenation can be seen as a multi-scale feature grouping. It allows the third GCN layer to generate embeddings not only from the second-hop neighbourhood embeddings but also from the first-hop. This could allow the network to ignore the two-hop features and regenerate new, more relevant ones from the one-hop neighbourhood embeddings.

The third model variation (GCNPool) is shown in Figure 4.4 (b) in yellow, introduces a pooling layer in front of the last GCN layer. The pooling reduces the number of nodes to half by using the Furthest Point Sampling algorithm (FPS) method from PointNet++ (Qi et al., 2017). The method iteratively samples the most distant point with regard to

the rest of the points in the metric space. A ball of a certain radius then groups the surrounding points. Their features get aggregated to the centre node with the max-pooling method. The remaining nodes mean to contain the fine-grained structure of the geometric neighbourhood from the graph convolutions and the broader neighbourhood information from the FPS and aggregation method.

This model architecture aims to capture features from multiple resolutions like MRG suggests.

All three architectures use a global max pooling to form a final feature vector. A linear fully-connected layer maps the vector to the final output classes. Cross - entropy loss allows the models to learn the meaningful features for a good classification.

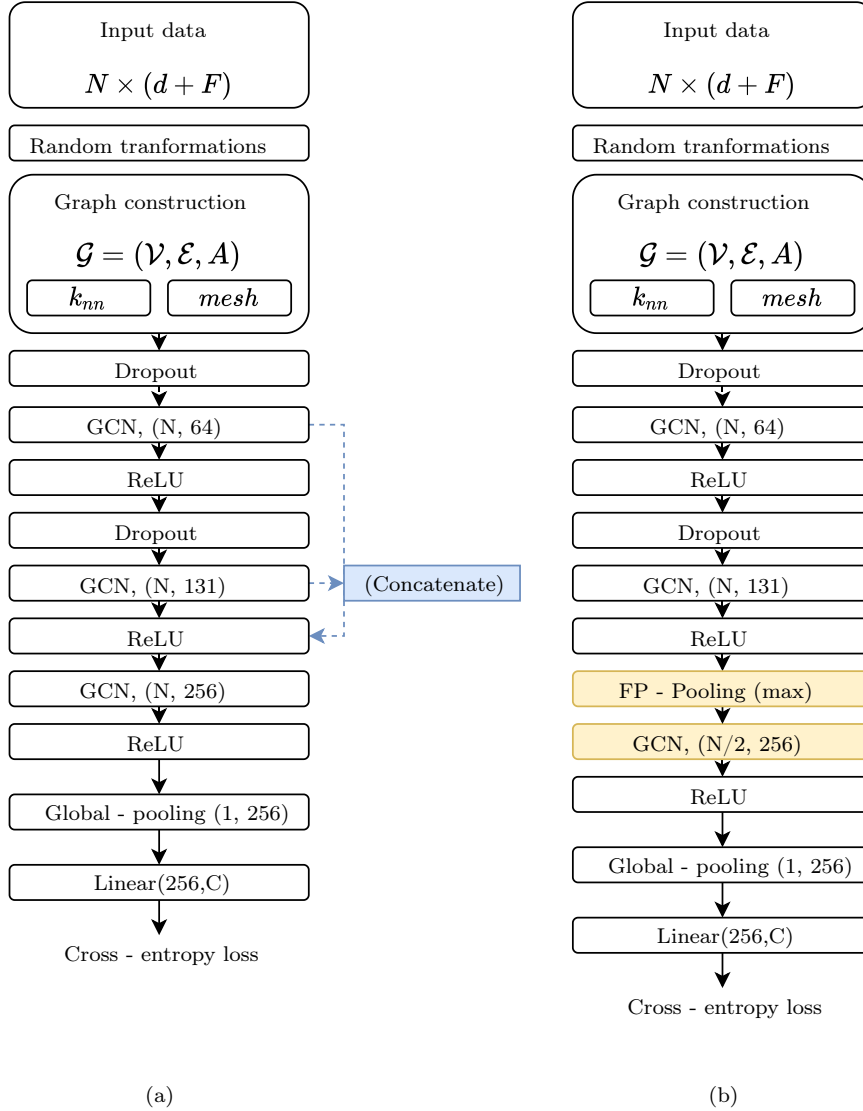


Figure 4.4.: GCN model architecture for classification. (a) Base GCN architecture, blue color shows the model version with multi-scale feature concatenation; (b) Model version with furthest-point pooling for multi-resolution grouping

4.4. Experiments and Results

4.4.1. Benchmark

The first training experiments are done on the known dataset ModelNet10. The set includes mesh geometries from objects types like Bathtub, Bed, Chair, Desk, Dresser, Monitor, Night-stand, Sofa, Table Toilet, which are all indoor objects but have except for the furnishing equipment little in common with the BIM dataset.

The dataset was used as a benchmark for evaluating the different model variations described in 4.3.1 before applying them to the BIM dataset. For reference, the PointNet classification results from Karaev (2020) are taken.

The GCN performance results are displayed together with the PointNet baseline in Table A.2. Merely the overall accuracies are shown for the benchmark experiment since it serves primarily for architecture validation purposes. The overall accuracies for each model architecture are reported in table 4.2. GCN results in a 7% overall accuracy increase compared to the baseline PointNet. GCNCat and GCNPool each perform by 9% better than PointNet for the overall accuracy. It can be seen that the GCN architectures generally perform slightly better than the PointNet baseline. The GCN variations with pooling and concatenation perform better than the base GCN model. For the detailed per-class accuracies, the reader is referred to the Appendix A.3. Most classes experience a classification performance increase. However, the classes Table, Desk and Bed perform slightly less for all GCN architectures. For the classes Desk and Bed, the GCN and GCNCat perform better than GCNPool but worse than PointNet. For the class Table, GCNCat performs the worst with a class accuracy of 0.73. The highest performance increase happens in class Night-Stand with 0.56 for PointNet, 0.60 for GCN, 0.78 for GCNCat and 0.87 for GCNPool.

Overall accuracy	
PointNet	0.82
GCN	0.88
GCNcat	0.90
GCNPool	0.90

Table 4.2.: Overall accuracies compared for benchmark dataset

Hyper-parameter optimization yielded the best results for a learning rate of 0.001 and a batch size of 30. These hyper-parameters are in line with choices made in other papers using ModelNet10.

As shown in Table 4.3 the amount of trainable parameters is significantly smaller for the GCN architectures than with PointNet and comes with a significant decrease in training time. Adding MSG with a concatenation, increases the amount of parameters slightly since

Model name	# trainable parameters	mean epoch time [s]
GCN	46666	7.8
GCNCat	64586	8.0
GCNPool	47434	20.0
PointNet++	1466956	300

Table 4.3.: Comparing model complexities - Trainable parameters and epoch execution time with batch size 30

there are more embeddings to propagate. A pooling layer drops the model complexity significantly while achieving better results.

4.4.2. GCN performances on BIM dataset

This experiment presents the model performances on the BIM dataset. Graph construction employs the PointSampling method. As suggested in literature for ModelNet10, 1024 points were sampled uniformly from the surface according to the triangle surface. A sensitivity-analysis for choosing the number of sampled points is beyond the scope of this thesis.

The node features included are point coordinates and normal vectors. As discussed in section 2.2.2 and 4.2.4 no spatial transformer network is added to the architecture. Random translations and rotations are applied to the samples instead, to prevent overfitting on the absolute coordinates i.e. be translation and rotation invariant. Hyper-parameter optimization shows that the three architectures on average perform best with a learning rate of 0.001 and batch size of 30.

Additionally, for this experiment, the hyper-parameter K-nn is optimized. It determines the degrees of connectivity for each node in the graph. By choosing K-nn to be five results in each node having a direct edge to 5 other nodes.

In this experiment, it has been found that too high node connectivity does not necessarily lead to higher prediction performance. Table 4.5 shows the performances for the three architectures with respect to the hyper-parameter K-nn and the mean epoch execution time.

Connecting each node to its five nearest neighbours shows to perform best for all architectures. Lower connectivity, as well as higher connectivity, leads to lower performance. The mean epoch runtime increases steadily with node connectivity. For the next experiments, K-nn will be set to be five.

Table 4.6 shows the accuracies and f1 scores for each class for the 3 model architectures. In Table 4.7 the confusion matrix for the best performing GCNCat architecture is shown. An average across model architectures, the highest performing classes (i.e. accuracy and f1 score above 0.85) are IfcColumn, IfcStair and IfcSlab. Similarly IfcFlowController and IfcDoor show high scores for GCN and GCNCat but a bit lower ones for GCNPool. IfcWall

4. Application case 1: BIM misclassification detection using geometric encodings

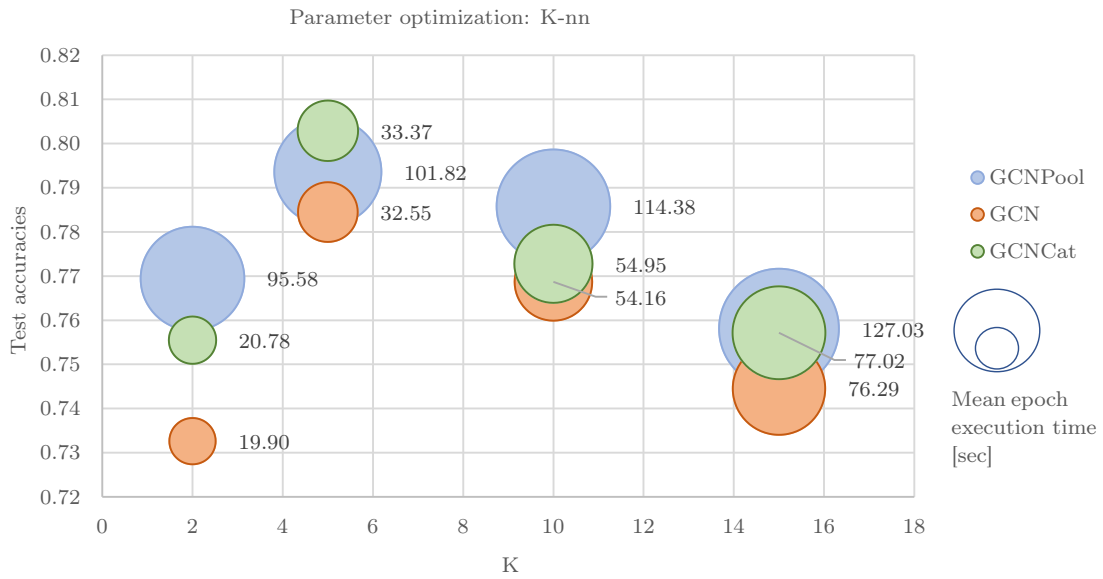


Figure 4.5.: Parameter optimization: K vs. test accuracy and execution time [sec]

	precision			recall			f1-score			acc		
	GCN	GCNCat	GCNPool	GCN	GCNCat	GCNPool	GCN	GCNCat	GCNPool	GCN	GCNCat	GCNPool
IfcColumn	0.80	0.87	0.89	0.96	0.89	0.88	0.87	0.88	0.88	0.96	0.89	0.88
IfcDistributionControlElement	0.67	0.82	0.84	0.79	0.75	0.73	0.72	0.78	0.78	0.79	0.75	0.73
IfcDoor	0.85	0.80	0.77	0.85	0.89	0.88	0.85	0.84	0.82	0.85	0.89	0.88
IfcFlowController	0.95	0.89	0.98	0.83	0.91	0.79	0.88	0.90	0.87	0.83	0.91	0.79
IfcFlowFitting	0.83	0.90	0.80	0.64	0.66	0.75	0.72	0.76	0.77	0.64	0.66	0.75
IfcFlowSegment	0.77	0.83	0.73	0.78	0.79	0.83	0.78	0.81	0.78	0.78	0.79	0.83
IfcFlowTerminal	0.66	0.70	0.81	0.89	0.89	0.90	0.76	0.78	0.85	0.89	0.89	0.90
IfcFurnishingElement	0.75	0.68	0.64	0.32	0.39	0.44	0.45	0.50	0.52	0.32	0.39	0.44
IfcSlab	0.81	0.75	0.76	0.87	0.89	0.82	0.84	0.82	0.79	0.87	0.89	0.82
IfcStair	0.94	0.86	0.85	0.86	0.93	0.92	0.89	0.90	0.88	0.86	0.93	0.92
IfcWall	0.84	0.78	0.76	0.76	0.83	0.85	0.80	0.80	0.80	0.76	0.83	0.85
IfcWindow	0.66	0.76	0.71	0.86	0.81	0.72	0.75	0.78	0.72	0.86	0.81	0.72

Figure 4.6.: Classification report for best performing architectures on BIM dataset ($K=5$)

shows scores of around 0.8 and an increased performance with GCNPool. The same tendency of increased performance with GCNPool is observed with classes IfcFlowFitting and IfcFlowSegment.

Classes where precision is low (red in the graph) but that accuracy high (green) need to be noted for reasons of overfitting, here IfcFlowTerminal and for GCNPool IfcDoor. Classes where recall is low but precision is considerably higher are a sign that the model penalized false positives more than false negatives. This topic will be discussed in the discussion.

The least performing class is IfcFurnishingElement in all aspects. The mean accuracy for all model architectures is 0.39. IfcFlowFitting and IfcDistributionControlElement perform a little better but reach average accuracies across models of 0.68 and 0.76 respectively. The other mean accuracies reach values above 0.8.

The confusion matrix 4.7 gives insight which class the wrongly classified samples were

		TRUE											
		IfcDCE	IfcFC	IfcFF	IfcFS	IfcFT	IfcCol	IfcFgElm	IfcStair	IfcDoor	IfcSlab	IfcWall	IfcWin
Predicted	IfcDCE	0.79	0.00	0.03	0.09	0.02	0.02	0.00	0.00	0.01	0.02	0.01	0.00
	IfcFC	0.10	0.83	0.02	0.00	0.03	0.00	0.00	0.00	0.00	0.00	0.02	0.00
	IfcFF	0.07	0.03	0.64	0.04	0.11	0.00	0.07	0.00	0.00	0.01	0.02	0.02
	IfcFS	0.01	0.00	0.01	0.78	0.05	0.08	0.01	0.00	0.00	0.03	0.00	0.02
	IfcFT	0.05	0.00	0.01	0.01	0.89	0.00	0.00	0.00	0.01	0.02	0.00	0.01
	IfcCol	0.01	0.00	0.00	0.00	0.00	0.96	0.00	0.00	0.00	0.00	0.03	0.00
	IfcFgElm	0.09	0.01	0.03	0.04	0.15	0.08	0.32	0.01	0.04	0.04	0.03	0.14
	IfcStair	0.00	0.00	0.00	0.00	0.01	0.00	0.00	0.86	0.00	0.08	0.00	0.04
	IfcDoor	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.85	0.00	0.01	0.13
	IfcSlab	0.02	0.00	0.01	0.02	0.02	0.00	0.00	0.03	0.00	0.87	0.01	0.01
	IfcWall	0.06	0.00	0.01	0.01	0.00	0.04	0.02	0.01	0.02	0.01	0.76	0.06
	IfcWin	0.00	0.00	0.01	0.00	0.04	0.00	0.01	0.01	0.06	0.00	0.00	0.86

Figure 4.7.: Confusion matrix for best performing GCNCat architecture on BIM dataset (K=5)

associated with. Overall it can be said that bigger structural elements are more often confused amongst each other and smaller elements in turn amongst themselves. A cross-group confusion happens mainly with IfcSlab and other classes. IfcDoor is classified as a IfcWindow in 9% and for a IfcWall in 4% of the cases and similarly vis-versa. IfcColumn is classified as a IfcFlowSegment in 4% of the samples and as IfcWall in 4% of the samples.

4.4.3. Graph construction mesh vs. k-nn

In this experiment the graph construction alternatives described in section 4.2.4 are investigated.

Since the graph is fundamentally different an extra experiment is made by adding the globally normalized degree of target nodes to the edge features for the mesh graph. The new feature is concatenated with the edge weight.

$$\mathbf{u}(i, j) = \frac{\deg(j)}{\max_{v \in \mathcal{V}} \deg(v)} \quad (4.1)$$

The experiment aims to show if the node degrees is a meaningful feature characterizing the mesh graph.

In Table 4.8 the results for this experiment are shown. For each model architecture the best performing model from the previous experiment is taken. The test accuracy is consistently lower for the mesh graph than for the sampled k-nn graph for all three architectures. For the base GCN architecture the reduction in performance is only 0.1%, however for the architecture variations GCNCat and GCNPool the mesh graph performs less by 0.4% and 0.17% respectively.

Adding the extra feature of node degrees does not increase performance, but reduces it further by 0.1% and 0.3% respectively for architectures GCN and GCNCat. For GCNPool the addition of the extra feature does not make any difference to the performance on the base mesh graph.

When looking into the individual classes (see Annex A.1 for detailed per class results) it appears that the mesh approach does perform better in some cases. IfcFlowController,

4. Application case 1: BIM misclassification detection using geometric encodings

IfcColumn, IfcStair, IfcDoor and IfcWindow have better f1 scores for the mesh graph approach. Thereby only IfcStair shows a significant difference in performance of 0.39%.

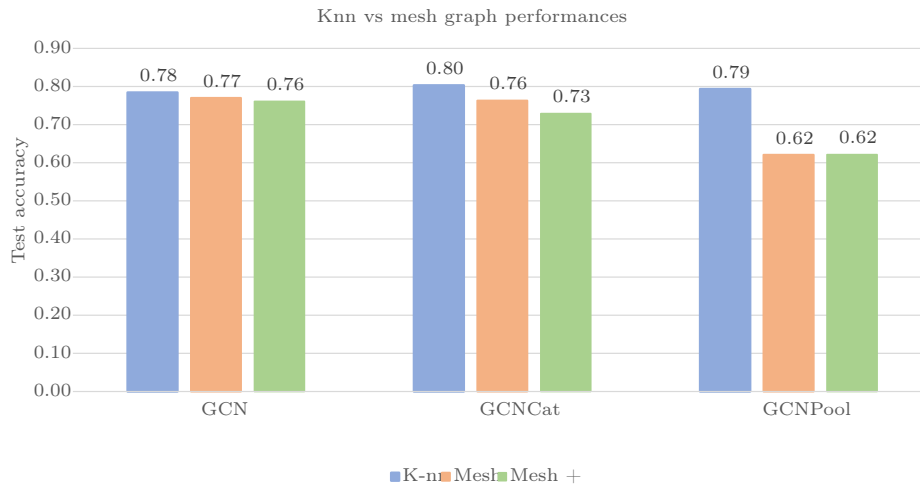


Figure 4.8.: K-nn vs. mesh graph performances for all model architectures. Mesh+ shows the results for when node degrees is added as an extra feature

4.4.4. Feature selection

In the previous experiments the model received information on the node position and the corresponding normal vectors. This experiment investigates the effect of adding additional features to the input. The overall geometric volume and surface area are added as features. Since the BIM shapes have extreme variance in their volume and area, i.e. IfcSlab vs. IfcFlowController, both features are log normalized. This fits the feature range to the interval of 0 and 1 while scaling big numbers more than small ones. Table 4.4 shows the improvements in performance for the dataset variations.

	Overall accuracy		
	base features	volume and area	neighbors
GCN	0.88	0.90	0.93
GCNcat	0.90	0.91	0.92
GCNPool	0.90	0.92	0.92

Table 4.4.: Overall accuracies for base dataset and datasets with added features. The overall accuracy is reported for each dataset variation. The results of for the dataset with neighborhood features must be considered with carfulness and consideration of the limitations

An additional experiment is attempted to include neighbourhood information in features. There are many limitations to this experiment and aims mostly to motivate the reader

for future work. For every BIM object the geometrically adjacent objects are added by one-hot-encodings as features. Here the true label of the adjacent objects is hypothetically available. This leads to a major performance increase for all classes. Geo-spatial queries are computationally very intensive and thus the preparation of the corresponding dataset would have been too time consuming. This experiment is therefore performed on a reduced dataset including only 5 IFC models.

In practise the true label of the surrounding geometries is not given. Instead, each shape would be represented by an embedding resulting from the developed instance graph of this work. In a higher-level context graph such embeddings could lead to a combination of local geometry information with geometric adjacency information.

5. Application case 2: Point cloud semantic segmentation for facilitated as-built model update

5.1. Problem statement

The discrepancy between the as-planned and the as-built BIM model is naturally present already before completion of construction. Processes on the building site can not always precisely happen as planned, and there is a need for time reservation to embed such inconsistencies. If this is not accounted for, the BIM model at hand-over will not reflect the building as is. The discrepancy risks to grow more significant, in case the building is in operation without that a responsible BIM manager has been appointed. Renovations and building alterations are frequent in the time after project hand-over, and BIM models can quickly become outdated and useless.

Unfortunately, engineers in a facility management company often face the issue of having outdated BIM data already on day one of their contract (see Figure 5.1). In this section methods for point cloud, semantic segmentation on an industrial dataset are investigated and aim to provide an idea for a toolset for a facilitated model update.

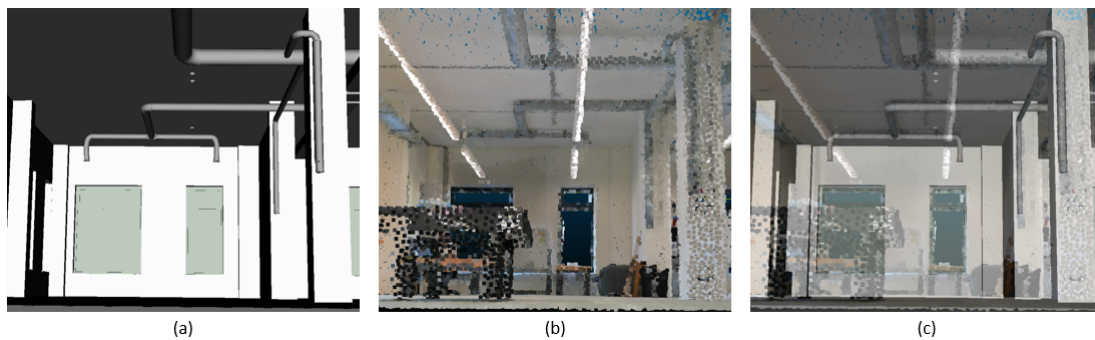


Figure 5.1.: Discrepancies as-planned vs as-built visualized for *Aspern Technologie Zentrum*. (a) as-planned BIM model; (b) as-built point cloud; (c) overlay for discrepancy visualization.

5.1.1. Datasets

Academic point clouds

PCSS is a growing topic in research and receives a lot of attention since 3D learning techniques on 3D data have become more accessible. New python libraries for deep learning on 3D graph data like *Pytorch Geometric* (Fey, 2020b) and *Torch Scatter* (Fey, 2020a) facilitate the implementation of deep learning architectures on point clouds and open the challenge to a broader public.

When deep learning architectures get deeper and more complex, a significant amount of training data is required to make the model performant and effective. The dataset should contain all variations that the final model shall perform predictions on. Following the principles stated in Section 2.1.1, a model should only be evaluated on data originating from the same underlying distribution it has also been trained on.

However, in the case of point clouds, it is a challenge to generate a relevant dataset. A large dataset requires a lot of workforces, material and financial resources (Zhang et al., 2019). There are various point cloud datasets in literature. However, only two represent indoor building scenes: the S3DIS dataset from the University of Stanford (Armeni et al., 2016) and the 3D point cloud dataset from the University of Zurich (Department of Informatics Visualization and Multimedia Lab, 2020). For the latter, it must be mentioned that the dataset does not contain any labels and is thus not usable for supervised training with goals for quantitative analysis.

The Stanford Large-scale 3D Indoor Spaces Dataset (S3DIS) contains 272 3D room scenes in 6 regions for semantic segmentation. The regions show different building storeys. The dataset was acquired by using the Matterport RGB-D scanner. There are 13 classes in a total of which 12 represent objects in indoor scenes (board, floor, table, door, chair, wall, column, bookcase, beam, window, sofa, ceiling) and one groups remaining points into a class called clutter. Each point in the scene is assigned to a semantic label corresponding to the classes.

Approaches like Czerniawski and Leite (2020)s' try use point clouds from other data acquisition methods such as images and apply methods of transfer learning to enlarge the size of the training dataset.

Nevertheless, a labelled data set acquired by a laser scanner is missing in academia.

Industrial point cloud Aspern

In industry point clouds become more important for building site progress monitoring, as-built model update and indoor navigation use cases. For building site progress monitoring a point cloud is acquired at several stages during construction, and the objects of interest are not limited to the ones visible at project completion but also include hidden elements such as piping systems.

This thesis uses industrial PCD from a research site in Aspern Seestadt in Austria. The campus is built to elaborate and test sustainable and innovative products and ideas in

5. Application case 2: Point cloud semantic segmentation for facilitated as-built model update

the fields of energy, environment, building technology and smart grids. Together many research partners from Wien Energie and Wiener Netze and Siemens aim to contribute to the solving of future challenges in the built environment.

One building, the Technology centre, was monitored with scannings during the construction process and the project initially included an as-planned to as-built BIM model update. In the end, this step only happened for the architectural discipline for the BIM model at hand-over. The other BIM disciplines such as mechanical, electrical and plumbing are omitted. The architectural BIM models from carcass construction stage were likewise not updated. Additionally, it is at the writer's knowledge that minor additions to the building were performed after hand-over. Lost in the hand-over process, such updates were not updated in the as-planned BIM model.

In the scope of this thesis, only the as-built point cloud acquired at project hand-over is considered. Additionally, to make the dataset workable and allow for an ideal learning curve, the point cloud is further reduced to include only one building story at a time. The point cloud has further been subsampled using the random subsampling tool in the Cloud Compare Software. The point clouds were acquired by the mobile laser scanner from NavVis M6 (NavVis, 2020).

As described in section 1.4.3 several preprocessing steps are involved before using raw PCD for semantic segmentation purposes. The point cloud used for this work was already reworked as follows:

- Post-processing: creates the coloured point cloud by merging picture information and point clouds. To the best of knowledge of the author, no Simultaneous Localization and Mapping (SLAM) anchor points are used, and the dataset was thus referenced and aligned manually by a responsible NavVis surveyor. The dataset is therefor not geo-referenced and does not align with the BIM model
- The NavVis surveyor performed filtering and removal of unwanted points like the surroundings or moving objects.
- The building surface is large, such that several individual scans acquired the point cloud. If anchor points are recorded, the alignment can be done automatically. Here however the point sets needed to be merged in the online software application from NavVis. This task was done by Siemens AG previous to the start of this thesis. The result is an alignment bundle containing the rotation matrices for each point set. It can be downloaded and used for the alignment of the raw point cloud data.

Dataset assembly

For supervised segmentation training tasks, the data has to be labelled, i.e. every point needs to know which semantic class it belongs to. During training the weights of the network are initialized and from then on they get iteratively updated to minimize the loss. The loss is a measure of how far away the prediction is to the true label. For further information, the reader is referred to the background Section 2.1.2. It is, therefore, necessary to have a high-quality dataset, such that the network has a fair learning process.

This thesis starts with processed, individual point cloud sets, as described in Section 5.1.1. The alignment bundle is used for the merger. Since the point cloud is not geo-referenced, it has to be aligned with the BIM model to allow for the as-planned, as-built comparison. This step was performed by the point-picking-alignment tool in the software Cloud Compare (2020).

The labels in the available dataset are generated by using the aligned IFC model geometries. Labelling a point cloud dataset is not a simple task and leads as pointed out below to several questions. The following points will be subject to discussion in section 6 and will guide the work of this section.

- Which and how many classes are relevant?
- What are the trade-offs between automated and manual labelling techniques?
- What characterizes good predictions in the case of PCSS. Is it a quantitative metric or is it the ability for a modeller to subsequently work upon the cloud

The developed labelling workflow bases itself on previous work done by Krijnen (2019). The algorithms are modifications of his work published on GitHub DURARK (2020).

The point labelling is performed by triangulating the IFC geometries and assembling the triangle faces in a spatial index structure. Spatial indices are specialized in storing spatial information and optimizing spatial queries upon that data. The index thus allows for a relatively fast query. The corresponding pseudo-code for extracting the face information is listed in Algorithm 5.1.

Algorithm 5.1: BIM face extraction

```

1 Input: IFC_file  $F$ , desired_entities  $E$ 
2 Output: spatial index of faces  $I_{faces}$ 
3
4 Set  $S \subseteq F$  // filter Ifc file by entities
5 Initialize spatial index  $I_{faces}$ 
6 For  $\forall S_i \in S$  with valid geometry:
7     Calculate shape triangulation
8     Extract individual face geometry
9     For each face with valid face geometry
10        Compute bounding box
11        Insert bounding box into spatial index
12 Return  $I_{faces}$ 

```

Following the generation of the spatial index, the algorithm iterates through all points in the input point cloud. A maximum distance must be defined, whereby this parameter can be a list of increasing distances. For each point in the input point cloud a bounding box is created according to the first distance value in the list. A spatial query on the spatial index allows finding all faces intersecting with the bounding box. If an intersection was found the point is labelled with the semantic IFC label of the face. If several intersections are found, the point gets labelled with the semantic IFC label of the nearest face. If no intersection has been found, the bounding box is increased by the next value of the given distance list. If still no intersections are found, the point is written to a different point cloud file collecting the remaining unlabelled points.

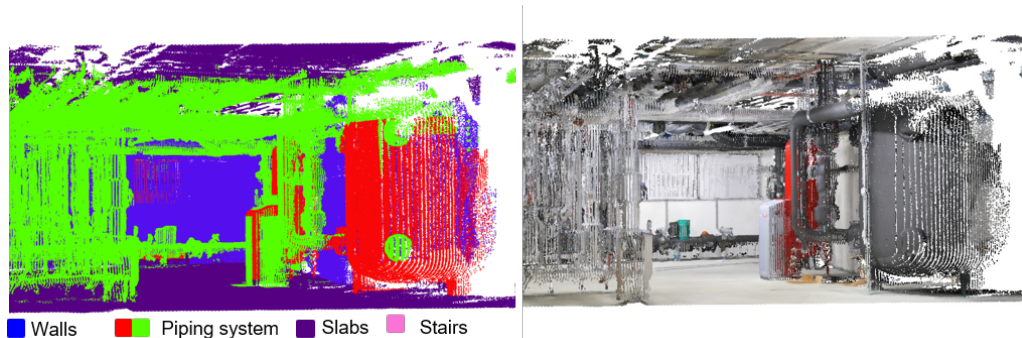
5. Application case 2: Point cloud semantic segmentation for facilitated as-built model update

Algorithm 5.2: Point cloud labelling

```
1 Input: Point cloud  $P$ , spatial index  $I_{faces}$ , max distances  $list_{dist}$ 
2 Output: Labelled point cloud file  $P_l$ , Remaining point cloud file  $P_{rest}$ 
3
4 For  $\forall P_i \in P$  :
5     For distances in  $list_{dist}$  increase point bounding box to  $bbox_i$ 
6         Intersect  $bbox_i$  with  $I_{faces}$ 
7         If intersection True:
8             If intersection is  $> 1$ :
9                 Find intersection closest to point
10                write point (coordinates and features) and label to  $P_l$ 
11                Break
12            If intersection False:
13                Continue increasing point bounding box to  $bbox_{i+1}$ 
14        If all intersections False:
15            write point (coordinates and features) and label to  $P_{rest}$ 
```

With the described process, on average, 85% of points per building storey are labelled. Figure 5.2 shows a snapshot of the labelled point cloud. The number of classes initially labelled corresponded to the IfcTypes present in the BIM model. These class types were then grouped into the following eight classes, pipes, walls, slabs, stairs, doors, column, tanks, windows. As will be seen in the experiment section below, this was modified afterwards. The deadset was split into training and test dataset, which included approximately 0.8 and 0.2 of all points respectively.

Figure 5.2.: Labelled point cloud dataset Aspern (piping systems)



Automated vs manual generation of point cloud datasets

As so often, automation comes with significant benefits regarding time and costs. The reduction in overall expenses comes as so often with a loss of flexibility and accuracy.

Manual labelling of point clouds is an extremely tedious task and requires adequate computational resources to render the relevant points. There is also a certain amount of uncertainty tied to manual approaches. An expert with an eye for BIM modelling would be favourable for this task. Even then, points can often not unambiguously be assigned to one semantic class. Inaccuracies or drifts in the point cloud cause uncertainty about the semantic label of some points.

Therefore, it is interesting to search for automated dataset generation or point cloud labelling techniques. One approach would be to generate an artificial point cloud from the BIM geometries by PointSampling as described in Section 4.2.4. Ma et al. (2020) have most recently published their work on that very topic. They prove that enriching the S3DIS dataset with artificial point clouds improves the segmentation Intersection over Union (IOU) by 7.1. It has to be noted that the artificial point cloud obtained by the PointSampling algorithm leads to point clouds without occluded surfaces. Hidden surfaces such as the intersecting faces of a slab and a wall are also represented in the point cloud.

An alternative strategy could be to simulate a mobile laser scanner with the same specifics of the real world scanner. GIScience Research Group (2020) have developed a tool where a different type of laser scanners with user-specific product specifics can be placed in a surface model and simulate outgoing laser pulses. Occluded surfaces will not be represented in the output point cloud, and the dataset will therefore be a more accurate artificial point cloud dataset. To the knowledge of the author, no approaches in literature have published results on the PCSS with this dataset preparation.

In this thesis, the advantage of having a closely aligned BIM model to reality is used to create a training dataset automatically. However, the approach has its own limitations. Some points will, inherently to the process, not be labelled adequately and extra manual work is needed. Specifically, the following cases lead to un- or wrongly labelled points.

- Major discrepancies between Designation of a model reflecting the "as-built" state of a building after the completion of construction (as-built) and as-planned models will result in remaining unlabelled points.
- Typically doors are opened in the scanning process to perform the areal scanning in one go and produce several overlapping point regions. The overlapping areas allow for an iterative correction in the SLAM calculation and reduce the drift in the resulting point cloud. Scanned open doors will however not be projected onto any BIM surface.
- Inaccurate as-planned BIM objects can also cause a false label assignment. For example, if a door is constructed 2m further than initially planned, true wall points will get assigned to the as-planned door and vis-versa.

It is thus essential to keep the limited labelling accuracy in mind when reading the following sections in this application case.

5.1.2. Approach

This section aims to elaborate on the tasks sketched out in the project methodology in Figure 3.2 with technical implementation details. The approach differs from the one in application case 1 insofar that it does not develop its own GCN architecture. Initially, it was planned to investigate different variations of GCN architectures for PCSS also, this, however, was finally set out of scope. The idea has to apply a modification of Graph-UNets (Gao and Ji, 2019) to the point cloud context. This GCN suggests a U-formed architecture, using several layers of GCN on various graph coarsening levels. Key is that

5. Application case 2: Point cloud semantic segmentation for facilitated as-built model update

the down-sampling is followed by an up-sampling, and the features of coarse and detailed graphs are combined.

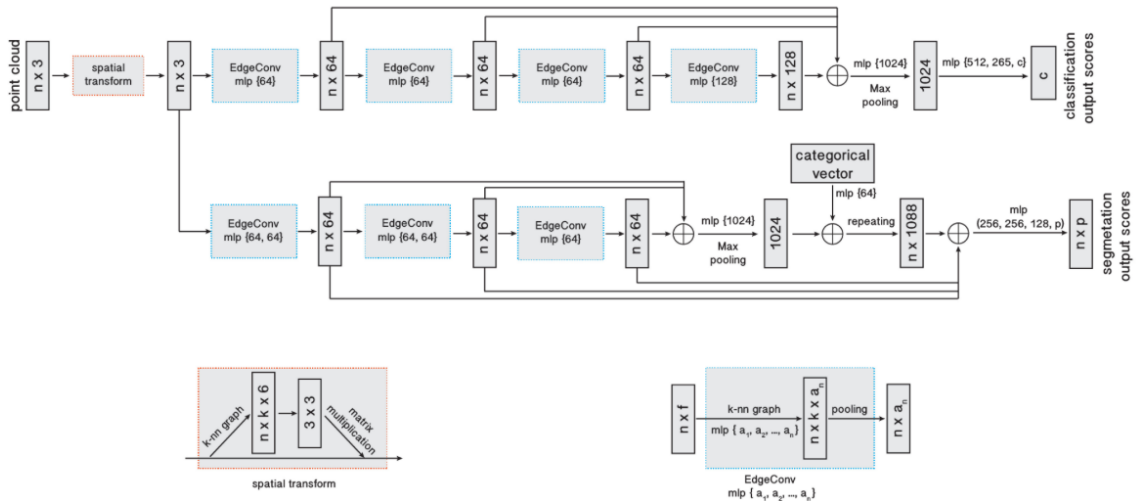
It was decided to focus on the development of an industrial dataset, the presenting of meaningful results and a well-sounded contribution to round off the project methodology. This decision is justified by the fact that point cloud datasets are scarce in academia and the availability of industrial PCD is of particular interest to the research community. Point segmentation of indoor systems such as plumbing, security or automation systems is of primary interest for the facility management. Since there is no publicly available laser-scan dataset including such elements, point cloud semantic segmentation methods and automated as-built model update are expected to receive further attention from the research community. The FM industries' opinion is that forms of PCSS remain too academic to deploy in productive solutions. The development of meaningful results on an industry dataset by leveraging existing methods is therefore considered to fill a research gap on its own.

The results are reported as IOU. Generally speaking, IOU is a measure of overlap between bounding boxes and is typically applied in object detection tasks. The detected area is compared to the ground truth object area, and the area of overlap is put in relation to the area of union. For point cloud segmentation the metric applies similarly.

Training setting and architecture

The proposed model architecture of Wang et al. (2019) is described in 2.2.3 and applied to this application case in the context described in this section. Zhang et al. (2019) and Ma et al. (2020) mention DGCNN to be a well performing GCN architecture and Ma et al. (2020) proves the generated results to be pertinent.

Figure 5.3.: The top architecture part serves for classification and the bottom part for segmentation. The bottom layer outputs per-point classification scores for p semantic labels (Wang et al., 2019)



As seen in Figure 5.3, the model has a classification and segmentation part that allows it to perform well for both tasks. The first layer takes a point cloud as input and shows to apply a spatial transformer network to allow for rotation and translation invariance. Thanks to this, the model can operate on the input coordinate features without overfitting to their exact values. In practice, this transformer network is not implemented. For segmentation the last layer outputs per point classification scores for p semantic labels (Wang et al., 2019). Similar as in 4.3 a cross entropy loss is chosen to train the model.

As suggested by the authors themselves, the K-nn parameter is chosen to be 5. No parameter optimization is performed. The hyper-parameters were optimized, and a learning rate of 0.001 and a batch size of 15 proved to perform best.

5.1.3. Experiments and results

In this section, the results of the segmentation process on the industrial dataset Aspern are shown.

1. DGCNN performances on Aspern point cloud

The initially labelled point cloud with eight classes as described in Section 5.1.1 did not lead to promising results. The reported Intersection over Unions for each class can be seen in Table 5.1. Structural elements such as walls and slabs show acceptable IOUs, whereas stairs, doors, columns and tanks report very bad results.

pipes	0.27
walls	0.60
slabs	0.58
stairs	0.03
doors	0.09
column	0.01
tanks	0.01
windows	0.20

Table 5.1.: Intersection over union for 8 class point cloud dataset

2. Dataset modifications

This experiment is formulated after the sobering results of the first experiment. It is decided to focus on the generation of meaningful results. The initial approach to the problem is revised, and an alternative dataset is generated. The number of classes was reduced to

5. Application case 2: Point cloud semantic segmentation for facilitated as-built model update

include the ones performing relatively well in experiment 1. Classes were grouped such that windows, doors, walls and columns represent one new class of walls. The tanks are added to the piping system class. An alternative approach would have been to increase the size of the dataset to include more instances of each class. This approach was briefly touched but for time reasons not pursued.

From an industrial point of view this reduction of classes does not come with decreased value creation. The reason for that is the importance of the semantic class of piping systems. For such installations there often is a need for an extensive BIM model review after project hand-over. Piping systems are expected to manifest an altered placement when constructed on a building site. An as-built point cloud segmentation for this class is thus of high interest to FM companies.

The remaining classes for this experiment are slab, wall, piping system and stairs. The thought behind keeping slabs, walls and stairs is to provide the necessary classes for downstream indoor navigation applications consuming the updated BIM model. In case pipes are updated, walls, slabs and stairs should be too, to meet the requirements of such a use-case. The results for the dataset with reduced class complexity are reported in column one of table 5.2.

It can be seen that using only four instead of eight classes increases the overall IOU values. The results show a significant accuracy increase for the classes piping systems and stairs. The overall IOU results mean around 0.77. This is an acceptable result but does not provide much value for application purposes. Visualization of the predicted point cloud labels is essential and will be subject to discussion in the limitations Section 6.2 of this report.

3. Using pretrained models for PCSS

(Ma et al., 2020) showed the effectiveness of using pre-trained models and retraining them on the relevant dataset. This is a common approach used whenever the datasets are of limited size and the datasets contain at least a few overlapping classes. In transfer training, the initial dataset should influence the model training such that, once applied to the second dataset, it can better identify similar features present in both datasets. Typically such approaches are used in industry to avoid long training times on new datasets. In that case, retraining is limited to a few epochs only to keep the training time and therefore, computational costs low. Here, however, the retraining was performed with the equal amount of epochs as for the initial training and accordingly not setting the computational resources as a limiting factor.

The DGCNN architecture is thus trained over 150 epochs on the S3DIS (Armeni et al., 2016). Area 6 is excluded from training and serves for test bases. The results of the pretraining are not reported in this report but claim to achieve similar results, as suggested in the article (Wang et al., 2019).

What is interesting to see is that the retraining on the Aspern dataset and evaluation on the test leads to slightly different IOU values visible in column 2 of Table 5.2. It can be seen that the classes present in both datasets such as walls and slabs experience a slight increase in performance of 0.8% and 2.1% respectively. The classes not present in the S3DIS

dataset such as stairs and piping systems cause the final IOU to decrease slightly for the pre-trained experiment. It can be debated if the increase in performance is significant or not in this context.

	IOU	
	Aspern	pretrained (S3DIS)
Piping system	77.4	64.2
Wall	78.3	79.1
Slab	83.1	85.2
Stair	72.2	68.7

Table 5.2.: Intersection over union for 4 class point cloud dataset. Results are shown for training and evaluation on the Aspern dataset. Additionally, results are shown when retraining a pre-trained DGCNN on S3DIS on the Aspern dataset using transfer learning.

6. Discussion and Limitations

In this section the results of both application cases from Sections 4 and 5 are discussed and the research questions from Section 2.4 are answered. In the first Section 6.1 the technical research questions 2 and 4 are discussed using the results of the GCN models. In Section 6.3, the industry-oriented research questions 3 and 5 are answered. Finally, Section 6.4 shows how the outcomes of this work answer the global research question number 1 and translate into three practical contributions.

6.1. Technical discussion

The two application cases motivated by industry insights offer an excellent insight into how the sometimes abstract academic methods find their meaningful application in the construction industry. As all ML methods do, these two application cases require a dataset before generating value for the industry. The project-based FM industry has a vast and various throughput in building data. Is all the more interesting to detach from the conventional academic datasets and test the developed methods on real-world datasets.

The assembled datasets have their flaws and faults since the focus was laid on an automatic dataset generation rather than manual data engineering. The underlying data is neither previously reworked for the purpose nor has there been preselection to favour specific input data. The input data originates from business as a usual setting within a FM company and represents real-world data. In-depth quality control of the dataset was not performed. In each case, there is only a short visual quality control made, and thus chances are there that the datasets contain unmeaning-full or even confusing geometries in the case of application case 1 or points for application case 2.

6.1.1. BIM shape classification

This dataset manifests geometry as extracted from IFC and illustrated well the challenge described above. `IfcFurnishingElement` class contains geometries from early building stages when dummy geometry is typically placed for proof of concepts. It seems that such dummy geometries remain in the models causing a the `IfcurnishingElement` class to contain a lot of rectangular objects. The class further represents a grate variety of geometries within-itself. A bookshelf, a chair and a desk do not have many geometrical structures in common, except maybe for the fact that they are standing on the floor.

In come cases it could also be that the BIM geometry is modelled as parts in the authoring software. When extracting the geometries from the IFC programmatically, by type, the

parts will reach the dataset as individual geometries, for which even for the human eye it is hard to assign the semantic label to.

Analysing the performances on ModelNet10 it can be said that the new models perform successfully. The general trend shows that GCN performs better than PointNet, looking at the individual class prediction accuracies however is interesting. It seems that GCN performs less on classes with big uniform horizontal surfaces and little less-fine grained geometric structure like Bed, Desk and Table. For these classes GCNPool further decreases performance. This could be a hint that the pooling reduces the most significant nodes of the flat surfaces to a point where they fail to influence the classification.

GCN has the advantage of being simple and still achieving comparable results to more complex architectures like PointNet. However it must be mentioned that the three GCN architectures are merely specialized for classification tasks whereas PointNet is also designed for semantic segmentation tasks on PCD. The developed architectures satisfy thus the need for light-weight, easily deployable but still performing models, do however not compete with the sophisticated architectures from literature.

Since GCNs argue to capture well fine grained geometrical features, it might be favorable to have a higher node connectivity to gather as many local traits as possible. The higher k , the higher the local connectivity of the graph. Wang et al. (2019) states that PointNet and similar approaches acting on points individually are a special case of GCN where $k = 0$. As anticipated, less than 5 nearest neighbors seems to fail in providing enough fine grained geometrical information. More than 5 neighbors probably leads to a over-connectivity, or in other worlds redundant information is propagated in the convolutions and therefore the prediction performance does no longer improve. The fine grained structural information that GCNs claim to capture better than PointNet models, seems to be captured well by connecting the 5 nearest neighbors in a 3-hop neighborhood convolution setting.

Ifc classes are a big topic of discussion in industry and are still expected to evolve. In fact, the standardisation discussion often lead to broad classes to satisfy all users needs. Also, the IFC class definition happens on a semantic level and not on a geometrical level. Thus, some considered IFC classes have higher theoretical geometrical variance; e.g. IfcFlowterminal, IfcFurnishingElement, IfcDistributionControlElement than others; e.g. IfcWall, IfcSlab, IfcFlowSegment. A IfcFowTerminal for example is defined as follows:

... defines the occurrence of a permanently attached element that acts as a terminus or beginning of a distribution system (e.g., air outlet, drain, water closet, sink, etc.). A terminal is typically a point at which a system interfaces with an external environment ... buildingSMART International (2020)

However a sink and a air outlet have very different geometries it does however make sense to group them semantically. In cases like this, a consideration of the surroundings should lead to a better object classification. Nevertheless, the predictions for this class are good. This could be because the model is overfitting on a special geometry type contained in the dataset. Since the dataset was assembled from a limited amount of IFCs of which only 4 were equipped with full HVAC systems the variation in geometries in such broad classes was probably not significant. In Table 4.1 it can indeed be seen that some classes have comparatively small amount of unique geometries. The model thus achieves good

6. Discussion and Limitations

results by learning the available geometries in the dataset and being tested on geometries not too different from the training samples. This could be counteracted by enlarging the dataset. This hypothesis is enhanced when looking at the precision and accuracy of the class `IfcFlowTerminal`. The precision, a measure of how many selected elements are relevant, is low whereas the accuracy is high. This is typically a sign for overfitting since the model classifies proportionally more elements as `IfcFlowTerminals` based on probably "un-real" features.

If classes show a low recall but a considerably higher precision it might be that the class is underrepresented in the input dataset.

In summary the first concrete research question can be answered here. Despite the mentioned limitation of a relatively small dataset, the three variations in GCN architecture prove to recognise and classify unknown BIM objects to a reasonable degree contributing to the semantic enrichment of BIM models from the lower maturity level as introduced in Figure 1.1.

6.1.2. PCSS

For PCSS the model training has only been performed for a small amount of semantic classes. As described in 5.1.2 this is not necessarily a shortcoming. Since PCD processing is still a complex topic in literature it is important to know the underlying need for the work performed on the data.

Point clouds can yield many sorts of information of which the simplest are floor plans, window area or simple measures of distance. More sophisticated information extraction methods are being developed currently by the research community, the final goal of which is to produce a fully semantic BIM model directly from point clouds.

Nevertheless, acquiring precise point clouds and applying manual, semi-automated or even automated information extraction from point clouds involves major workforce and financial and time resources. It is therefore recommended to know what the point clouds' use is and what information it should finally yield. When applying machine learning methods for PCSS to point clouds it should ideally be known what group of classes are of interest. Such requirements facilitate the data acquisition firstly and secondly also the dataset preparation for training.

It is thus, argued here that the piping systems and their structural environments are of major interest to the eFM facility and that the overall goal to suit a FM need has been accomplished despite only working with four classes. Piping systems are amongst the least up-to-date building elements in the as-built modelling process. An additional step towards facilitating the update of plumbing systems, even if it is a specialized one, is of importance. Further work could of course be invested in differentiating between individual plumbing elements. To suit the FM's need for digital building system representation additional elements such as sensors, lamps, sprinklers as other equipment should be considered.

The results of the pre-trained model experiment leave room for improvement. The pre-training does not show a significant performance increase. A poorly trained initial model could explain this. Despite the model performing well on test Area 6, cross-validation has

not been performed which could have lead to a less robust pre-trained model. This should have, however, not lead to enormous differences. Additionally, it can be argued that the generalization principle from Section 2.1.1 is not respected in the transfer learning process. Indeed the S3DIS and the Aspern datasets were acquired by different scanning technologies, once with an RGB-D camera and once with a laser-scanner. Although the model is retrained entirely, it could be that the underlying data from the S3DIS does offer any usefulness to the dataset of Aspern.

It has to be noted that the scope of the Aspern point has been reduced significantly. The results are to be considered with this in mind. Considering only one building storey and down-sampling the number of points in the cloud was necessary for the learning process of this thesis but could lead to significant feature loss. It is also hypothesized that this is the root of the relatively low performance reported on the eight classes in Table 5.1. The number of tanks, columns, doors and windows might be too few to let the training process capture equally meaningful features for all classes.

A visualization of the results would be necessary. Since the dataset is in itself not accurate, it would be interesting to see the per point predictions visualized. A qualitative analysis of the results might lead to different results than stated in section 5.

This lets a answer tot research question 4 be formulated. According to the evaluating metric commonly used in academia, the PCSS shows a good performance. However when considered the usability of these results, it must be said that they do not have must significance to the use case. A BIM modeller requires per-point prediction visualization in order to adequately update the as-planned model to as-built.

6.2. Limitations and future work

It must be noted that all experiment results are reported after one program execution. To make a more generalizable discussion the experiments should be executed several times and a result distribution should be reported.

The dataset size is for both application cases a point worth a discussion. Although the assembled BIM dataset as well as the Aspern point cloud dataset are of limited size and contain little variation (see 4.2.1 an 5.1.1) this thesis can serve as a guideline for assembling future datasets. More IFCs should be added, especially IFCs with equipped HVAC and Building Automation Systems (BAS) systems in the first case and more point clouds of storeys and buildings should be processed.

Currently, the BIM dataset includes geometries from the Siemens CAD library, which will make the trained model perform less on other companies' product types. The generalization principle described in Section 2.1.1 states that a trained machine learning model is not designed to perform well for data that lies beyond the underlying training distribution. Therefore, when applying this work to a different context, the models must be retrained.

Regarding the addition of features in the last experiment of application case 1 (Section 4.4.4) it must be stated that the potential of graphs has not fully been exploited. Instead of leveraging the powerful feature propagation of GCNs the features of the last experiment

6. Discussion and Limitations

were added before the final linear layer. It would be interesting to add those additional features to the nodes and edges in the graph encoding for more localized features. The explored features *volume* and *area* would be difficult to add at the node level. However, a measure of local shape compactness, curvature, planetary, isotropy (Weinmann et al., 2013) or similar could lead to exciting results.

Adding neighbourhood labels for the centre shape classification is an interesting concept worth exploring. In practice, the real labels would not be available, since the aim is to classify all objects in a building. Therefore the context information would be in the form of an embedding rather than a label. This concept would open the possibility of a high-level concept graph which would take the final embeddings of the presented models as an input. Combining the local geometry graph with a geometric concept graph and even the semantic knowledge graph from IfcOWL is an idea for future work.

Regarding the application case of PCSS the end to end workflow to a full as-built model, the BIM update must be performed. Some existing automated modelling tools exist, but there is no top off-the-shelf approach allowing for an application in this thesis. Since PCSS in the case of slabs and walls only gives the position of one face, the depth of the structural elements must be inferred. This challenge is subject to many discussion in literature and would be interesting to consider in future works aiming for an end-to-end application.

The model training has only been performed for a small number of semantic classes. As argued To generate real value, more classes would have to be covered.

When working with point clouds and machine learning, the data size influences the learning process. A fixed amount of points fit onto the GPU. A somewhat naive approach but still commonly used is the sliding window approach whereby slices of 1 by 1 by 1 meter cut the point cloud and ass the points to the GPU. Although some minor experiments have been performed, it was out of the scope of this thesis to investigate the effect of modifying this sliding window.

6.3. Business opportunities for facility management

Using results of both application use cases, this section shows how the findings could contribute to automating current workflows in a facility management company.

On the way towards an interoperable and integrated building environment, data from heterogeneous sources needs to be structured and made compatible. Appropriate semantic classes help to do so. Kim et al. (2019) developed a prototype for embedding the semantic classes found by their 2D CNN architecture in the BIM authoring software Revit (see Figure 6.1). A similar prototype can be imagined for the work shown in this thesis. Compared to Kim et al. (2019)s approach, the results from the developed GCNs would promise an improved prediction accuracy and most likely a reduced network size and training time. By implementing the concept of the context graph 4.4.4, suggestions on the neighbourhood geometries could be given. A door, for instance, not connected to any wall or space would be signaled to the planner for rework.

The contribution becomes more interesting when considered in a cross-building platform like the Siemens Building Twin. The aim is to bring all buildings to a homogeneous data-model on top of which operational use cases can be developed or enhanced. The increased data insights resulting from the size of the platform promise to supplement the information in the built environment further. A critical prerequisite for such a platform is the correct mapping of BIM to the platform ontology. Missing or misclassified BIM object semantics represents a considerable obstacle for such platforms. Currently high rework times hinder the platforms scale up.

The prediction models presented in this section could unlock a scalable and reliable on-boarding of new buildings. Tedious and cumbersome manual reworks can be reduced by letting the class predictions with high enough certainty assign the semantic label automatically.

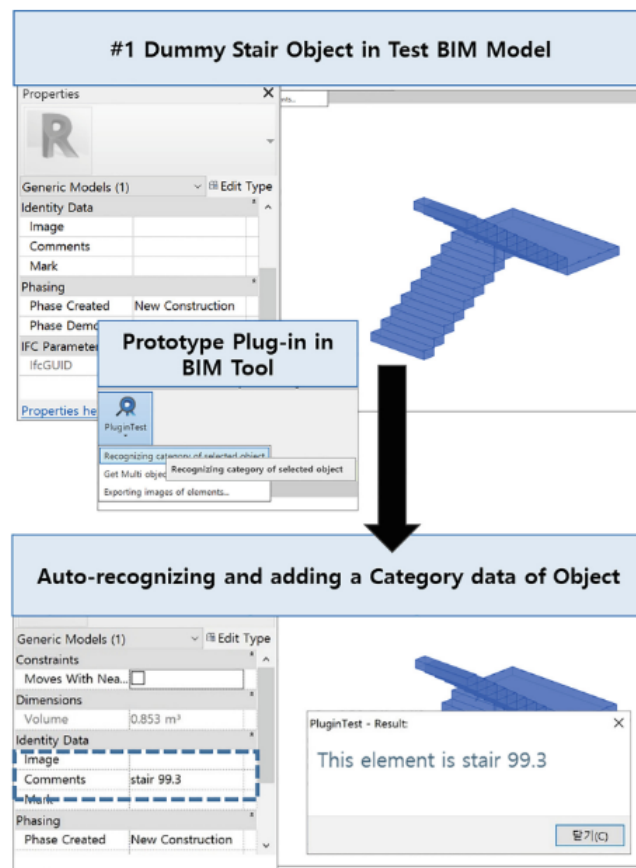


Figure 6.1.: Revit prototype for semantic label enrichment (Kim et al., 2019)

Point clouds are currently mainly acquired for visualization purposes. Indoor navigation, the coupling of visuals with manually placed points of interest and simple as-built measuring tasks can nowadays easily be performed. The most common tool for this is the indoor NavVis Viewer. BIM is, however, not included in the equation here. The moment when a fully semantic BIM model is linked to as-built point cloud data considerable time savings, uncertainty reduction and cost reductions will be the result. Point clouds, being the most

precise representation of the as-built building, offer way more data for information extraction than today possible. FMs owning such highly accurate point cloud data could be motivated for exploration for an as-planned, as-built comparison if provided with mature technology. As highlighted by Brilakis et al. (2010), such tools are critical to the bridging of the gap between as-planned and as-built. Piping systems have not been brought to topic in the context of PCSS yet. The work of this thesis proves that current academic methods could also generate their impact in an industry environment and could potentially be implemented as a prototype for the as-planned model update.

From a technical perspective, most of the code is developed to work in a Docker environment and could thus be deployed easily in a scalable cloud platform.

6.4. Contribution

The overall contribution of this work can be summarized in three points.

1. The self-designed GCN architecture offers a light-weight machine learning approach for BIM object classification. The models show their effectiveness in misclassification detection of as-planned BIM data and could provide FMs with a tool to verify project data at the hand-over. Alternatively, the trained models could also reclassify and restructure existing BIM data from buildings in operation for which the available BIM models have up-until now not found their use in daily operations. The found classification accuracies reaching on average 80% across classes are a good base for testing the automation possibilities for semantic enrichment in FM companies.
2. The second application case shows that the previously rarely explored semantic class of piping systems is detectable in point clouds to 77%. Although the work of this thesis excludes an automated visualization for the semantically segmented point clouds, the mean prediction accuracy of 0.77 motivates for further development of a prototype application. The update of piping placements in the as-planned model could profit from an automated detection in point clouds.
3. When generating the labelled point cloud dataset for training a counter is initiated to collect the number of points projected onto each BIM surface. The counter highlights BIM elements onto which comparatively many points have been projected. The results of the counter can be translated into a likely-hood for a given element to in the building.

The three points gather to a potential tool-set for an as-planned model update. Figure 6.2 shows an illustration of the three contributions. In Figure 6.2 (a) the semantic label prediction would show a class name for each BIM element with an achieved certainty of prediction. The colour code shows how the second contribution could lead to a highlighting of elements having very few points in their surroundings. Figure 6.2 (b) shows a predicted label for the set of points which do not lie in the proximity of any BIM object. Such a tool-set could facilitate FM companies in operating their buildings on a more truthful building data basis and reduce the risk of uncertainty in their operation.

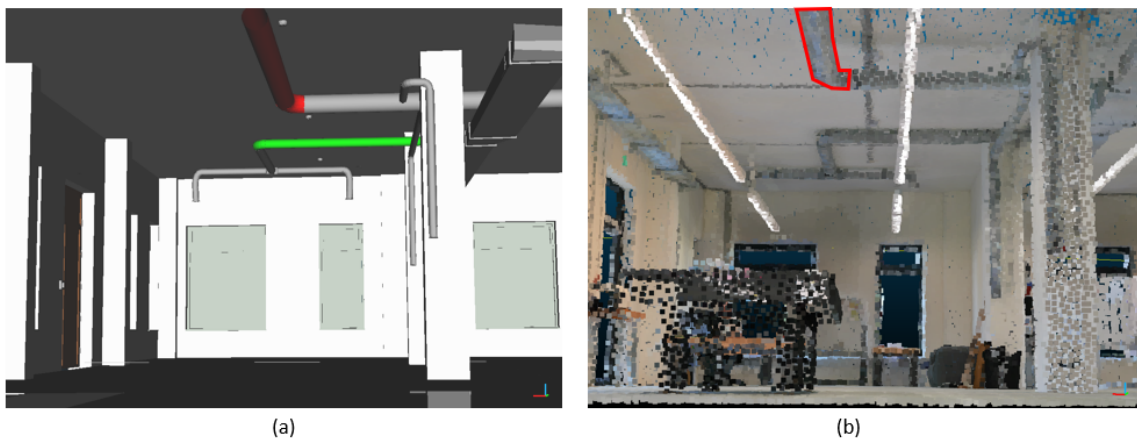


Figure 6.2.: Contribution illustrated. (a) Semantic label prediction (not shown) and object placement review; (b) Semantic segmentation with label prediction for unlabelled points

7. Conclusion

Although the implementation of BIM and its digital workflows is slowly establishing itself in industry, a lot of digital building data reaches the facility management company in an unstructured, incomplete and fragmented format. The error-prone hand-over processes between project stages are likely to increase the loss of information within a building project further. Building operation becomes more and more critical when thinking of the global transformation that the built environment will face soon. Indoor comfort, urban planning and global CO₂ monitoring are and will increasingly have increasingly detailed digital requirements for the building industry. It is all the more important that FM companies manage their operated buildings with a correct underlying digital representation (Digital Twin). Current BIM models reach the FM as-planned and do usually not represent the reality as-built. The development of tools aiming to bridge the gap between as-planned and as-built models requires cross-domain skills of scanning, computer vision and photogrammetry, machine learning, parametric object modelling (Brilakis et al., 2010) and BIM processes. Until today an end-to-end automation tool-set is missing, and the amount of incomplete as-built projects is rising.

The challenge of lacking as-built information takes two different forms. Once, it is a question of restructuring scattered data and generating usefulness from it. Methods bundled under the term semantic enrichment, for example, aim to render implicit information in BIM directly accessible for high-end query languages. Automatized approaches have the potential to transform data into information on large scales and allow for more informed decision making in building operation and management.

On the other hand, it is a question of capturing the unknown. Most of the built environment does not have a digital representation, although the benefits from it would be huge. Laser-scan technologies offer a precise way for capturing this reality, but the technology gap for the processing of the resulting point clouds is yet to be filled.

This work hopes to meet a practical need of the general facility management community by proposing a lightweight graph convolution architecture for adding semantic integrity to project records and a workflow for leveraging as-built information from point clouds for a later model update.

By leveraging the rapidly developing technologies in the fields of computer vision and machine learning, new levels of Building Information Modelling can find its ways into the highly rigid and project-based construction industry.

Bibliography

- Armeni, I., O. Sener, A. R. Zamir, H. Jiang, I. Brilakis, M. Fischer, and S. Savarese (2016). 3D semantic parsing of large-scale indoor spaces. *Proceedings of the IEEE Computer Society Conference on Computer Vision and Pattern Recognition 2016-Decem*, 1534–1543.
- Ashworth, S. (2015). BIM and FM in Switzerland: A survey of the perception of BIM by FM professionals in Switzerland. (March).
- Atkin, B. and A. Brooks (2015). *Total facility management*. John Wiley & Sons.
- Autodesk Revit (2020, accessed 01.09.2020). Revit 2020. <https://www.autodesk.com/products/revit/overview>.
- Bloch, T. and R. Sacks (2018). Comparing machine learning and rule-based inferencing for semantic enrichment of BIM models. *Automation in Construction 91*(December 2017), 256–272.
- Borrmann, J., W. Lang, and F. Petzold (2018). Digitales Planen und Bauen Schwerpunkt BIM. Technical report.
- Breu, T. (2019). Point Cloud Classification as a Support for BIM-based Progress Tracking. *Masterthesis - TUM Department of Civil, Geo and Environmental Engineering* (February).
- Brilakis, I., M. Lourakis, R. Sacks, S. Savarese, S. Christodoulou, J. Teizer, and A. Makhmalbaf (2010). Toward automated generation of parametric BIMs based on hybrid video and laser scanning data. In *Advanced Engineering Informatics*.
- Bronstein, M. M., J. Bruna, Y. Lecun, A. Szlam, and P. Vandergheynst (2017). Geometric Deep Learning: Going beyond Euclidean data. *IEEE Signal Processing Magazine 34*(4), 18–42.
- buildingSMART International (2020, accessed 01.09.2020). buildingSMART. <https://www.buildingsmart.org/>.
- Cloud Compare (2020, accessed 15.07.2020). Cloud Compare. <https://www.danielgm.net/cc/>.
- Czerniawski, T. and F. Leite (2020). Automated digital modeling of existing buildings: A review of visual object recognition methods. *Automation in Construction 113*(July 2019), 103131.
- Department of Business Innovation and Skills (2011). A report for the Government Construction Client Group Building Information Modelling (BIM) Working Party Strategy Paper. Technical Report March.

- Department of Informatics Visualization and Multimedia Lab (2020, accessed 15.07.2020). 3D point cloud research datasets. <https://www.ifi.uzh.ch/en/vmml/research/datasets.html>.
- DURAARK (2019, accessed 01.09.2020). IFCPPointCloud. <https://github.com/DURAARK/IFCPPointCloud>, commit 39056b3af4b12e378f8bb6f61749c26043655bd0.
- Eastman, C., P. Teicholz, R. Sacks, and K. Liston (2011). *A Guide to Building Information Modeling for Owners, Managers, Designers, Engineers, and Contractors*. Hoboken, New Jersey: John Wiley & Sons Inc.
- Fey, M. (2020, accessed 01.09.2020a). Pytorch Scatter. https://github.com/rusty1s/pytorch_scatter.
- Fey, M. (2020, accessed 01.09.2020b). PytorchGeometric. https://github.com/rusty1s/pytorch_geometric, commit 8a4830c5cfc637fcd8316c3d15e4e0f99b915bf7.
- Fischer, M. (2006). Formalizing construction knowledge for concurrent performance-based design. *Lecture Notes in Computer Science (including subseries Lecture Notes in Artificial Intelligence and Lecture Notes in Bioinformatics)* 4200 LNAI, 186–205.
- Ford, S., G. Aouad, J. Kirkham, P. Brandon, F. Brown, T. Child, G. Cooper, R. Oxman, and B. Young (1995). An information engineering approach to modelling building design. *Automation in Construction* 4(1), 5–15.
- Gao, H. and S. Ji (2019). Graph U-nets. *36th International Conference on Machine Learning, ICML 2019 2019-June*, 3651–3660.
- Gao, X. and P. Pishdad-Bozorgi (2019). BIM-enabled facilities operation and maintenance: A review. *Advanced Engineering Informatics* 39(November 2018), 227–247.
- GIScience Research Group (2020, accessed 01.09.2020). Helio. <https://github.com/GIScience/helios>, commit 2f391ec9497e5745890c2047451b2d9c15417453.
- Gong, S. (2016). Geometric deep learning. *SA 2016 - SIGGRAPH ASIA 2016 Courses* (September).
- Hichri, N., C. Stefani, L. D. Luca, and P. Veron (2013). Review of the « AS-BUILT BIM » approaches. *Computers in Human Behavior* 267(Cav), 1056–1061.
- Huber, D. F. and M. Hebert (2003). Fully automatic registration of multiple 3D data sets. *Image and Vision Computing* 21(7 SPEC.), 637–650.
- IfcOpenShell (2020, accessed 01.09.2020). IfcOpenShell. <https://github.com/IfcOpenShell/IfcOpenShell>, commit 55a54098dd1974dcfb5d8bf6c6c249886a528092.
- Ignatova, E., H. Kirschke, E. Tauscher, K. Smarsly, P. G. Modeling, and I. Modeling (2015). Parametric Geometric Modeling in Construction Planning Using Industry Foundation Classes. (July), 20–22.
- Jang, R. and W. Collinge (2020). Improving BIM asset and facilities management processes: A Mechanical and Electrical (M&E) contractor perspective. *Journal of Building Engineering* 32(May), 101540.

- Jantunen, M., E. F. Oliveira, P. Carrer, and S. Kephelopoulos (2011). *Promoting actions for healthy indoor air (IAIAQ)*.
- Ji, Y., J. Beetz, N. Nisbet, P. Bonsma, C. Katz, and A. Borrmann (2011). Integration of Parametric Geometry into IFC-Bridge. (January), 8.
- Joshi, V. (2019, accessed 01.09.2020). From Theory To Practice: Representing Graphs. <https://medium.com/basecs/from-theory-to-practice-representing-graphs-cfd782c5be38>.
- Juan, Y. K. and N. P. Hsing (2017). BIM-based approach to simulate building adaptive performance and life cycle costs for an open building design. *Applied Sciences (Switzerland)* 7(8).
- Karaev, N. (2019, accessed 01.09.2020). PointNet. <https://github.com/nikitakaraevv/pointnet>.
- Kim, J., J. Song, and J.-K. Lee (2019). *Recognizing and Classifying Unknown Object in BIM Using 2D CNN*, Volume 1028. Springer Singapore.
- Klicpera, J., S. Weissenberger, and S. Günnemann (2019). Diffusion Improves Graph Learning. (NeurIPS), 1–22.
- Klokov, R. and V. Lempitsky (2017). Escape from Cells: Deep Kd-Networks for the Recognition of 3D Point Cloud Models. *Proceedings of the IEEE International Conference on Computer Vision 2017-October*, 863–872.
- Koo, B., S. La, N. W. Cho, and Y. Yu (2019). Using support vector machines to classify building elements for checking the semantic integrity of building information models. *Automation in Construction* 98(October 2018), 183–194.
- Koo, B., B. Shin, and T. F. Krijnen (2017). Employing outlier and novelty detection for checking the integrity of BIM to IFC entity associations. *ISARC 2017 - Proceedings of the 34th International Symposium on Automation and Robotics in Construction* (July 2017), 14–21.
- Krijnen, T. (2019). *Efficient storage and retrieval of detailed building models: multi-disciplinary and long-term use of geometric and semantic construction information*. Number 2019.
- Krijnen, T. and M. Tamke (2015). Assessing Implicit Knowledge in BIM Models with Machine Learning. *Modelling Behaviour* (buildingSMART 2014), 397–406.
- Landrieu, L. and M. Simonovsky (2018). Large-Scale Point Cloud Semantic Segmentation with Superpoint Graphs. *Proceedings of the IEEE Computer Society Conference on Computer Vision and Pattern Recognition*, 4558–4567.
- Loukas, A. (2019, accessed 01.09.2020). Demystifying graph coarsening (intro). <https://andreasloukas.blog/2018/05/26/demystifying-graph-coarsening/>.
- Ma, J. W., T. Czerniawski, and F. Leite (2020). Semantic segmentation of point clouds of building interiors with deep learning: Augmenting training datasets with synthetic BIM-based point clouds. *Automation in Construction* 113(February), 103144.

Bibliography

- Macher, H., T. Landes, and P. Grussenmeyer (2017). From point clouds to building information models: 3D semi-automatic reconstruction of indoors of existing buildings. *Applied Sciences (Switzerland)* 7(10).
- Maturana, D. and S. Scherer (2015). VoxNet: A 3D Convolutional Neural Network for real-time object recognition. *IEEE International Conference on Intelligent Robots and Systems 2015-Decem*, 922–928.
- Mazairac, W. and J. Beetz (2013). BIMQL - An open query language for building information models. *Advanced Engineering Informatics* 27(4), 444–456.
- Migilinskas, D., V. Popov, V. Juocevicius, and L. Ustinovichius (2013). The benefits, obstacles and problems of practical bim implementation. *Procedia Engineering* 57, 767–774.
- NavVis (2020, accessed 01.09.2020). NavVis Developer Forum. <https://portal.navvis.com>.
- Ozturk, G. B. (2020). Interoperability in building information modeling for AECO/FM industry. *Automation in Construction* 113(December 2019), 103122.
- Patacas, J., N. Dawood, and M. Kassem (2020). BIM for facilities management: A framework and a common data environment using open standards. *Automation in Construction* 120(April), 103366.
- Pătrăucean, V., I. Armeni, M. Nahangi, J. Yeung, I. Brilakis, and C. Haas (2015). State of research in automatic as-built modelling. *Advanced Engineering Informatics* 29(2), 162–171.
- Pishdad-Bozorgi, P., X. Gao, C. Eastman, and A. P. Self (2018). Planning and developing facility management-enabled building information model (FM-enabled BIM). *Automation in Construction* 87(February 2017), 22–38.
- Qi, C. R., H. Su, K. Mo, and L. J. Guibas (2017). PointNet: Deep Learning on Point Sets for 3D Classification and Segmentation. In *The IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, Volume 2017-Janua, pp. 601–610.
- Qi, C. R., L. Yi, H. Su, and L. J. Guibas (2017). PointNet++: Deep hierarchical feature learning on point sets in a metric space. *Advances in Neural Information Processing Systems 2017-Decem*, 5100–5109.
- Ramsundar, B. and R. B. Zadeh (2018). *TensorFlow for Deep Learning*. O’Reilly Media, Inc.
- Riegler, G., A. O. Ulusoy, and A. Geiger (2017). OctNet: Learning deep 3D representations at high resolutions. *Proceedings - 30th IEEE Conference on Computer Vision and Pattern Recognition, CVPR 2017 2017-Janua*, 6620–6629.
- Russell, P. and D. Elger (2008). The Meaning of BIM: Towards a Bionic Building. *Architecture in Computro [26th eCAADe Conference Proceedings]*, 531–536.
- Selvakumaran, S. B. (2019). Master Thesis Crowd to Cloud : Simplified automatic reconstruction of digital building assets for Facility Management.

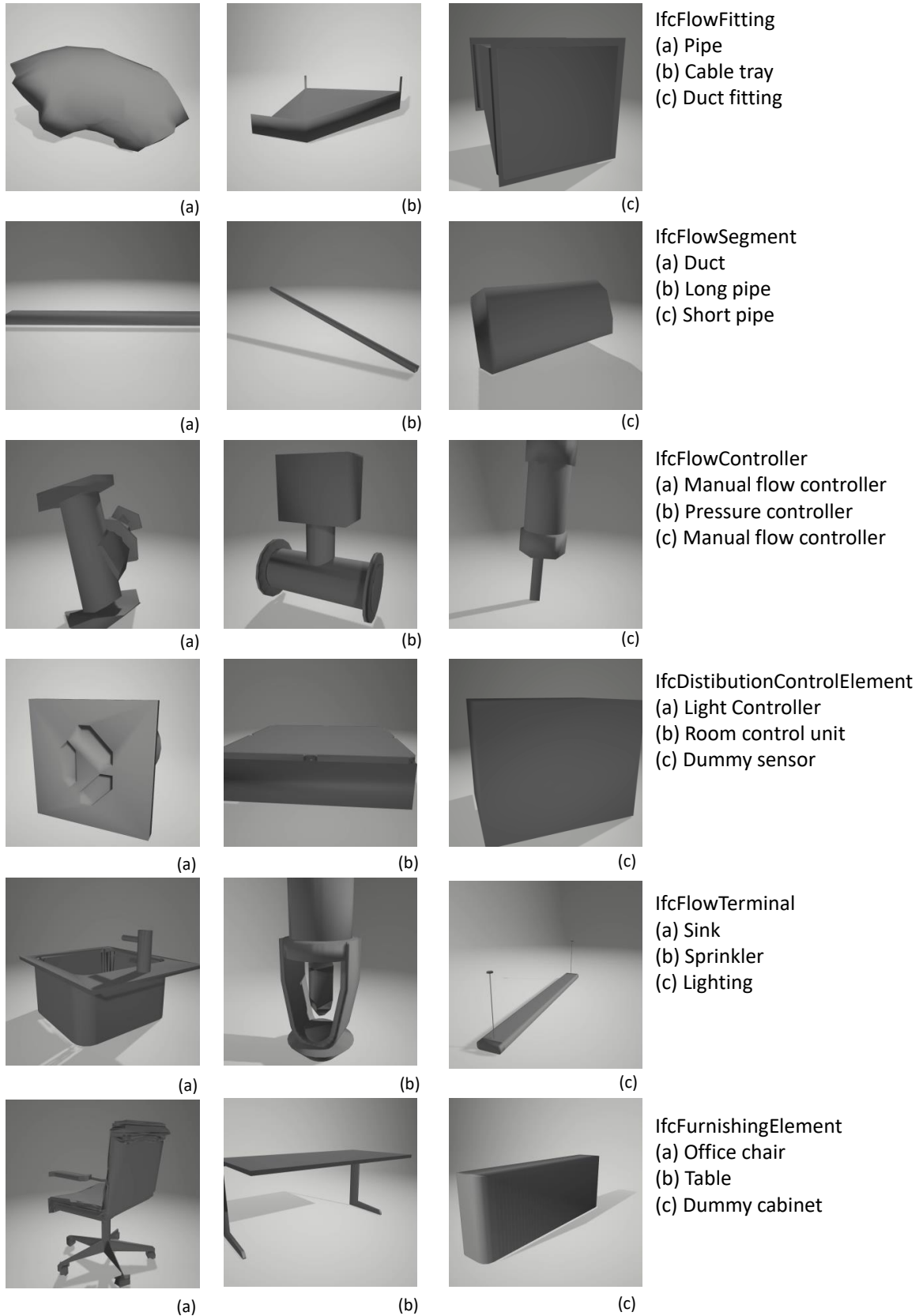
- Siemens AG (2019, accessed 01.09.2020). Siemens CAD Browser. <https://new.siemens.com/global/de/produkte/gebaeude/kontakt/bim-objects.html>.
- SimpleBIM (2020, accessed 01.09.2020). SimpleBIM. <https://simplebim.com/>.
- Su, H., S. Maji, E. Kalogerakis, and E. Learned-Miller (2015). Multi-view convolutional neural networks for 3D shape recognition. *Proceedings of the IEEE International Conference on Computer Vision 2015 Inter*, 945–953.
- Tang, P., D. Huber, B. Akinci, R. Lipman, and A. Lytle (2010a). Automatic reconstruction of as-built building information models from laser-scanned point clouds: A review of related techniques. *Automation in Construction* 19(7), 829–843.
- Tang, P., D. Huber, B. Akinci, R. Lipman, and A. Lytle (2010b). Automatic reconstruction of as-built building information models from laser-scanned point clouds: A review of related techniques. *Automation in Construction* 19(7), 829–843.
- TeamCAD (2020, accessed 15.08.2020). TeamCAD 2020. <https://www.teamcad.rs/index.php/en/news/330-what-is-lod>.
- Thalineau, R. (2019, accessed 01.09.2020). Deep learning with point clouds. <https://www.qwertee.io/blog/deep-learning-with-point-clouds/>.
- Tobiáš, P. (2015). An Investigation into the Possibilities of BIM and GIS Cooperation and Utilization of GIS in the BIM Process. *Geoinformatics FCE CTU* 14(1), 65–78.
- Tomliak, K. (2017). *Towards a zero-emission, efficient, and resilient buildings and construction sector*.
- University of Stanford (2020, accessed 15.07.2020). CS224W: Machine Learning with Graphs. <http://web.stanford.edu/class/cs224w/>.
- Volk, R., J. Stengel, and F. Schultmann (2014). Building Information Modeling (BIM) for existing buildings - Literature review and future needs. *Automation in Construction* 38(October 2017), 109–127.
- Wang, L., Y. Huang, Y. Hou, S. Zhang, and J. Shan (2019). Graph attention convolution for point cloud semantic segmentation. *Proceedings of the IEEE Computer Society Conference on Computer Vision and Pattern Recognition 2019-June*, 10288–10297.
- Wang, Y., Y. Sun, Z. Liu, S. E. Sarma, M. M. Bronstein, and J. M. Solomon (2019). Dynamic graph Cnn for learning on point clouds. *ACM Transactions on Graphics* 38(5).
- Weinmann, M., B. Jutzi, and C. Mallet (2013). Feature relevance assessment for the semantic interpretation of 3D point cloud data. *ISPRS Annals of the Photogrammetry, Remote Sensing and Spatial Information Sciences* 2(5W2), 313–318.
- Wenfa, H. (2008). Information lifecycle modeling framework for construction project lifecycle management. *Proceedings - 2008 International Seminar on Future Information Technology and Management Engineering, FITME 2008*, 372–375.
- WHO (2014). Burden of disease from household air pollution for 2012. Summary of results. *World Health Organization* 35(February), 2012–2014.

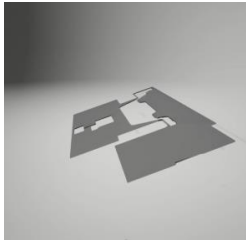
Bibliography

- Wu, J. and J. Zhang (2019). Introducing Geometric Signatures of Architecture, Engineering, and Construction Objects and a New BIM Dataset. In *Computing in Civil Engineering 2019: Visualization, Information Modeling, and Simulation - Selected Papers from the ASCE International Conference on Computing in Civil Engineering 2019*, Number September, pp. 264–271.
- Wu, W., Z. Qi, and L. Fuxin (2019). PointCONV: Deep convolutional networks on 3D point clouds. *Proceedings of the IEEE Computer Society Conference on Computer Vision and Pattern Recognition 2019-June*, 9613–9622.
- Wu, Z., S. Pan, F. Chen, G. Long, C. Zhang, and P. S. Yu (2019). A Comprehensive Survey on Graph Neural Networks. *XX(Xx)*, 1–22.
- Xie, Y., J. Tian, and X. X. Zhu (2019). A Review of Point Cloud Semantic Segmentation. pp. 1–51.
- Zhang, D., F. He, Z. Tu, L. Zou, and Y. Chen (2019). Pointwise geometric and semantic learning network on 3D point clouds. *Integrated Computer-Aided Engineering* 27(1), 57–75.
- Zhang, J., X. Zhao, Z. Chen, and Z. Lu (2019). A Review of Deep Learning-Based Semantic Segmentation for Point Cloud. *IEEE Access* 7, 179118–179133.

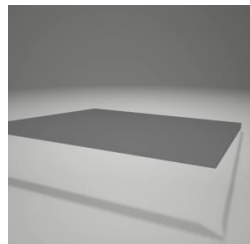
A. Appendix Case 1

A.1. BIM shape dataset

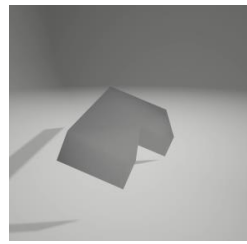




(a)



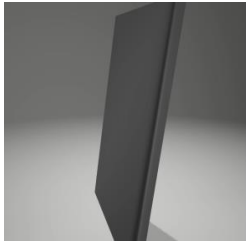
(b)



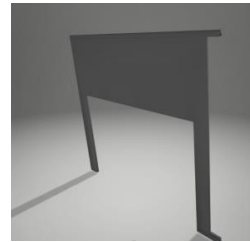
(c)

IfcSlab

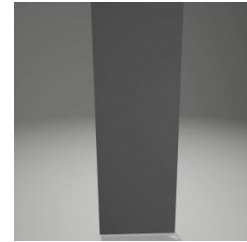
- (a) Whole story slab
- (b) Rectangular, thin slab
- (c) Thick, angled slab



(a)



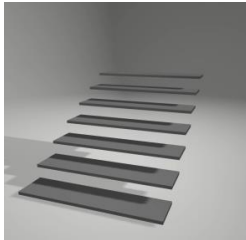
(b)



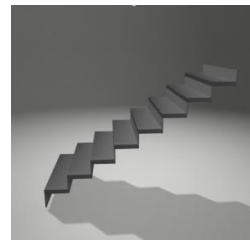
(c)

IfcWall

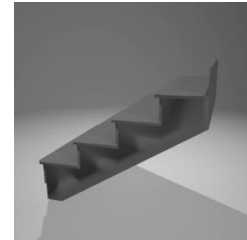
- (a) Double wall
- (b) Shaped wall
- (c) Narrow wall



(a)



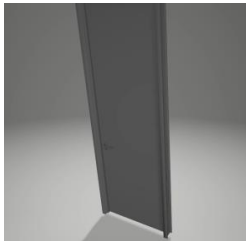
(b)



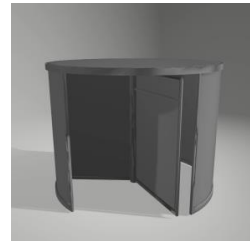
(c)

IfcStair

- (a) Separated stair
- (b) Connected stair
- (c) Stair with underlining



(a)



(b)



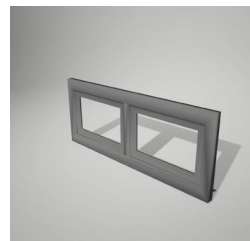
(c)

IfcDoor

- (a) Narrow door
- (b) Revolving door
- (c) Double door



(a)



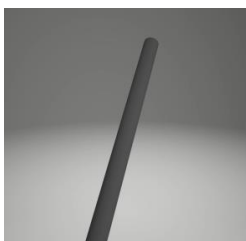
(b)



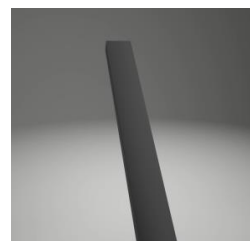
(c)

IfcWindow

- (a) Single window
- (b) Low window
- (c) Double window



(a)



(b)



(c)

IfcColumn

- (a) Round, long column
- (b) Rectangular column
- (c) Round short column

A.2. ModelNet10 dataset

ModelNet10 classes	unique geometries	Train dataset	Test dataset
monitor	156	388	85
toilet	615	436	97
sofa	989	518	91
bathhtub	286	328	94
nightstand	286	434	95
dresser	565	361	81
table	286	415	91
desk	780	345	91
bed	492	391	98
chair	444	375	85

Table A.1.: Dataset ModelNet10; The number of unique geometries show the class imbalance in the dataset. The train and test set are balanced by data augmentation and subsampling

A.3. Benchmark model performances on ModelNet10 - PointNet, GCN, GCNCat, GCNPool

A.4. Per class performances knn vs. mesh

	PointNet	GCN	GCNcat	GCNPool
Bathtub	0.93	0.91	0.97	0.95
Bed	0.92	0.90	0.94	0.83
Chair	0.97	0.98	0.97	0.99
Desk	0.82	0.79	0.80	0.69
Dresser	0.71	0.95	0.88	0.85
Monitor	0.89	0.99	0.97	0.98
Night stand	0.56	0.60	0.78	0.87
Sofa	0.87	0.96	0.95	0.99
Table	0.93	0.83	0.73	0.84
Toilet	0.96	0.91	0.95	0.97
Overall accuracy	0.82	0.88	0.90	0.90

Table A.2.: Benchmark model performances on ModelNet10. Per class accuracies are reported

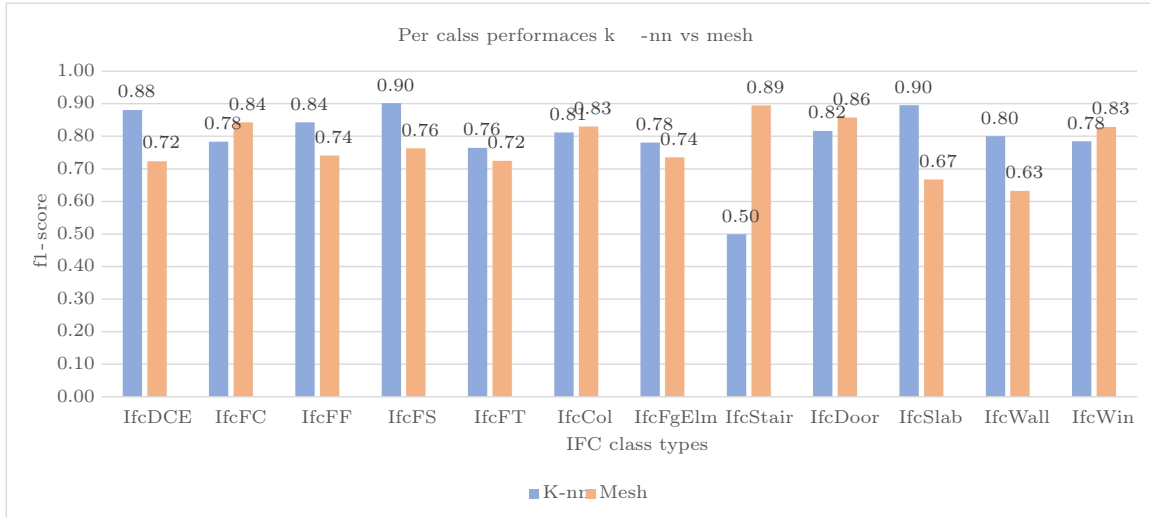


Figure A.1.: K-nn vs. mesh graph performances - per calss