energies

MDPI

Article

IFC BIM Model Enrichment with Space Function Information Using Graph Neural Networks

Adam Buruzs *, Miloš Šipetić, Brigitte Blank-Landeshammer and Gerhard Zucker

AIT Austrian Institute of Technology, 1210 Vienna, Austria; milos.sipetic@ait.ac.at (M.Š.); brigitte.blank-landeshammer@ait.ac.at (B.B.-L.); gerhard.zucker@ait.ac.at (G.Z.)
* Correspondence: adam.buruzs@ait.ac.at; Tel.: +43-664-8890-4316

Abstract: The definition of room functions in Building Information Modeling (BIM) using IfcSpace entities is an important quality requirement that is often not fulfilled. This paper presents a three-step method for enriching open BIM representations based on Industry Foundation Classes (IFC) with room function information (e.g., kitchen, living room, foyer). In the first step, the geometric algorithm for detecting and defining IfcSpace entities and injecting them into IFC models is presented. After deriving the IfcSpaces, a geometric method for calculating the graph of connections between spaces based on accessibility is described; this information is not explicitly stored in IFC models. In the final step, a graph convolution-based neural network using the accessibility graph to classify the IfcSpace entities is described. Local node features are automatically extracted from the geometry and neighboring elements. With the help of a Graph Convolutional Network (GCN), the connection and spatial context information is utilized by the neural network for the classification decision, in addition to the local features of the spaces which are more commonly used. To evaluate the classification accuracy, the model was tested on a set of residential building IFC models. A weighted version of the common GCN was implemented and tested, resulting in a slight improvement in the classification accuracy.

Keywords: BIM; IFC; architecture model enrichment; machine learning; IfcSpace

Citation: Buruzs, A.; Šipetić, M.; Blank-Landeshammer, B.; Zucker, G. IFC BIM Model Enrichment with Space Function Information Using Graph Neural Networks. *Energies* **2022**, *15*, 2937. https://doi.org/10.3390/en15082937

Academic Editor: Francesco Nocera

Received: 8 February 2022
Accepted: 14 April 2022
Published: 16 April 2022

1. Introduction

BIM modelling is the de facto standard in architecture, engineering, and the construction (AEC) industry. Various proprietary closed software packages are used to create BIM models. For the sake of interoperability, the buildingSMART consortium created the IFC (Industry Foundation Classes) standard [1], which is the established open data format for the BIM industry. IFC provides a rich vocabulary for the specification and classification of building components, as well as the properties needed for alphanumeric information. Unfortunately, in practice, the IFC files are most often incomplete. There exist solutions for the automatic checking of building code compliance, such as the Solibri Model Checker, but these tools employ hard-coded rules that require exact and complete BIM model specifications. In practice, incorrect design, lack of information provided by the user in the design phase, or the use of various local languages for labelling elements strongly limits the applicability of these automatic model checkers. For the validation of models and for further calculations, planning, or for cost estimations, the semantic enrichment of the IFC models is needed to inject this missing information into the IFC model.

In our work, we focus on the semantic enrichment of spaces (IfcSpace entities) with the function of the corresponding room. The exact spatial definition of the space is an important piece of information for stakeholders, for example, to quickly create a list of apartments with exact size information in residential buildings. Intended usage and occupancy of spaces is an important piece of information for facility managers: corridors, halls, and basement spaces require less ventilation and heating than office rooms or apartments.

Construction companies benefit from correctly labelled spaces by being able to calculate various cost factors and procurement plans more precisely (for example, calculating the required amount of tiling from the aggregated surface area of toilets and bathrooms).

IfcSpace entities are graphically not visible when rendering the BIM model; our analysis showed that the IfcSpace entity for a room is often omitted in the modeling process. However, even in higher quality models—where IfcSpaces are defined and thus PropertySets could be defined—the PropertySets are often left blank, leaving IfcSpace entities with only the space boundaries and without the room function defined.

IfcSpace entities are important elements that are required for HVAC simulations and control applications [2], and are needed for thermal zoning in thermal dynamic simulations [3]. The intended room function influences the different parameters that are used in the simulation, such as temperature, humidity, or air exchange setpoints and ranges (e.g., lower temperature setpoints in bedrooms). Similarly, storage rooms, corridors outside of apartments, and garages have lower or no heating requirements. Fine-grained control of related parameters of the building model in the design phase results in lower estimated energy needs, and, if those controls are correctly implemented in the commissioned building, also in a lower final energy consumption. Additionally, having the spaces automatically derived and their functions determined saves time and reduces the need for manual, error-prone work. Exact thermal simulations also require second-level space boundaries that can be derived from the defined spaces, as described in [4]. Room functionality is also an important information for facility managers throughout the lifecycle of the building.

Our paper structure is as follows: In Section 2, the state of the art of machine learning-based automatic enrichment of BIM models is reviewed, especially with regards to room-function classification. In Section 3, a geometric method is presented to automatically detect spaces in BIM models, if they are missing. We present a method for extracting connections between spaces and for building a graph of accessibility. Furthermore, a Graph Neural Network (GNN)-based classification algorithm is described to classify these spaces. This classification is based on this accessibility graph and the calculated geometric features that are extracted from the raw BIM Models. We also present a modification of the Graph Convolutional Network to strive for higher accuracy. In Section 4, the results of the classification are presented as an accuracy analysis and a confusion matrix. The paper is closed with Section 5.

2. Related Work

Machine learning methods have gained increasing popularity in recent years in the field of automated BIM processing. Krijnen and Tamke [5] used neural networks to classify buildings into residential and non-residential categories based on floor plans.

The semantic enrichment of IFC BIM Models has been addressed in multiple studies; Sacks et. al. developed a rule-based system that evaluates IF–THEN types of rules based on prepared features [6,7]. To extract geometric relations that are not explicitly contained in IFC files, they used Query Language for 4D Building Information Models (QL4BIM [8]).

Bloch and Sacks [9] examined the use of semantic enrichment in BIM models; they compared neural network-based rules against hand-crafted rules for IfcSpace classification (building rooms) and concluded that the neural networks based on geometrical features and spatial connections strongly outperformed the rule-based method in terms of accuracy.

Koo et al. [10] applied support vector machines (SVM) to classify building elements into eight of the most frequent IFC categories (Column, Beam, Slab, Wall, Covering, Door, Window, Railing). They used simple geometric features (bounding box size, volume, gyration, etc.) that were calculated using the opencascade geometry engine [11], as well as features derived from IfcRelations (number of related Walls/Windows). They trained their SVM classifier with six building IFC models and reached a classification accuracy of over 90%. The same team later used multi-view convolutional neural networks (MVCNN) to classify wall and door elements in BIM models [12]. The MVCNN [13] generates 2D renderings of objects, and practically transforms the 3D object recognition problem into the

well-studied 2D recognition problem, which can be solved efficiently using convolutional neural networks [14].

Lumio [15] et al. used deep learning methods to classify images generated from BIM model external views into three categories (residential, industrial, and other buildings). Besides classical SVM classifiers, they also tried deep neural networks (ResNet, MobileNet), which are very successful in machine vision tasks. They found that these deep neural nets considerably outperformed SVMs for this task and reached over 90% accuracy in predicting the building type based on the generated 2D images.

Edmunds et al. [16] created the IFCNet benchmark dataset for IFC element classification. They also tested the MVCNN architecture for classifying these elements; it showed very good results (over 85% accuracy). However, in their method, the context information of the elements is not used, and the elements are classified based on their geometry alone.

Many scientific and engineering datasets have an inherent structure that can be represented with a graph. Graph applications range from social networks, citation databases, molecular chemical datasets, recommendations, predicting infectious diseases, etc. Advanced neural network models for prediction and analysis tasks over graph data have attracted considerable research attention in recent years. Applications of deep learning methods on graph datasets include node classification, graph classification, link prediction, and graph generation tasks [17]. The methods applied on graph datasets include graph convolutional networks, recurrent networks, autoencoders, adversarial methods, and reinforcement learning. A common advantage of applying these methods to node classification is that they inherently incorporate graph network information into the node classification besides the single node information represented with the node features. A general mathematical framework for the application of deep learning on graphs is given by Bronstein et al. [18], where the analogy between graphs and manifolds is shown, and the extension of the successful Convolutional Networks to graphs is derived.

Kipf and Welling [19] proposed a graph based neural network model with a simple layer-wise propagation rule:

$$H^{l+1} = \sigma(\widetilde{A}H^{l}W^{l}) \quad (1)$$

where H^l is the input of the l-th layer, W^l are the weights of the layer, and the $\widetilde{A}$ matrix is calculated from the adjacency matrix, thereby establishing connections between adjacent nodes in one layer. It was shown how this formula can be interpreted as an approximation of a spectral graph convolution; therefore, the layer was named graph convolution. Additionally, the application of a graph convolutional network was demonstrated on a node classification task.

IFC BIM models contain numerous relations among their elements; thus, a graph of elements can be extracted with the IfcRelationships acting as edges. Additional relationships can be derived by applying different geometric reasoning methods. For example, using the calculated geometric distance between the extracted element shapes, a space adjacency graph structure can be derived by connecting spaces that are geometrically close to one another. After such graphs are derived, it becomes possible to apply graph neural networks to train models for inductive reasoning, such as element classification.

Several recent studies have dealt with building elements classification based on floor plans: Su et al. classifies structural elements (walls, windows, doors, stairs) based on floor plans with graph neural networks [20]. A recent paper by Paudel et al. [21] solved room-type classification based on the floor plan data of single apartments using various graph neural networks. Hu et al. [22] presented room-type classification based on the floor plans of university buildings using random forest and graph convolutional networks. All these studies report high (around 80%) classification accuracy. However, one has to note two things: 1. By selecting the model training and test set, the information concerning the parent building of each apartment is not preserved; it is likely that one floor of a building is included into the training set, while another floor of the same building belongs to the test set. One can assume that there are considerable similarities between the floors of the same building. 2. The number of common room types is relatively low.

3. Materials and Methods

We present a method where two types of information are extracted from IFC files to serve as input for a neural network-based supervised classification trained on labelled data: First, numerical features calculated from the geometric shapes of IFC elements are extracted. Second, the accessibility between spaces is calculated. The spatial proximity and traversability of spaces can be best described by a graph. The spaces are represented as nodes and connected with an edge if a person can walk from one space to the other. This condition is satisfied if spaces touch without a boundary between them or are connected via a door or staircase. The proposed graph convolutional network merges both numerical features and graph structure to train the weights of a neural network to classify the nodes in this graph. This method is not only promising for spaces, but also for other kinds of classifications and automatic IFC model quality assessments.

3.1. Space Extraction from IFC Elements

The IFC standard provided the IfcSpace (https://standards.buildingsmart.org/IFC/RELEASE/IFC4/ADD2_TC1/HTML/link/ifcspace.htm, accessed on 15 January 2022) entity for defining spaces or rooms in BIM models. IfcSpace elements can be defined and exported from closed BIM software packages. The correct definition of the spaces is important for subsequent analyses such as area calculation or apartment size determination. Further quality checks can also leverage the properties of the IfcSpace entities.

In our approach we started from the walls, which we extracted separately from each floor. First, the *Opencascade* TopoDS [23] shape was extracted from each wall with the help of *ifcOpenShell* [24]. Then, the oriented bounding box for each wall and the projection of that bounding box onto a horizontal plane was calculated. With this projection, two-dimensional rectangles were obtained for each wall. Using the *shapely* python package, these two-dimensional shapes were processed; the convex hull was calculated from the polygons and then the geometric difference of the convex hull and the rectangles corresponding to the walls was calculated. This difference resulted in a list of polygons that were closed polygons encompassed by the wall-rectangles. Polygons that were touching the convex hull had to be removed from this list, because these polygons were not contained within the building bounding walls. The polygons were then extruded along the z-axis (resulting in IfcExtrudedAreaSolid objects), assuming vertical walls and using the height of the storey that we calculated as the median of the wall heights belonging to the given IfcStorey. The obtained IfcSpace entities could then be written back into the original IFC file with the *ifcopenshell write* function, thus augmenting the initial BIM model with IfcSpaces. This procedure is visualized on Figure 1. This method assumes vertical walls and horizontal slabs. The processing of slanted slabs and roofs was out of scope of this paper. A further limitation of the method is that incorrectly modelled walls can also lead to problems with space detection (for example, when small gaps exist between the walls).

3.2. Accesssibility Graph Calculation

An accessibility graph is used to encode relations among spaces in the building. Adjacent graphs are commonly used in different kinds of analysis of spatial layouts, e.g., using the space syntax theory [25]. In architecture, this is often done to analyze the layout, connectedness, and relations between the spaces of the building. Another application of accessibility graphs is the automated calculation of emergency exit pathways; the maximum distance from each room to a fire-safe room is regulated by the Austrian building code (https://www.oib.or.at/de/oib-richtlinien/richtlinien/2019/oib-richtlinie-2, accessed on 15 January 2022). Using the accessibility graph, the shortest route to reach a fire-safe place can be easily calculated utilizing standard graph algorithms (for example, implemented in the *networkx* python package [26]). Adding length measures to the space-graph edges (distance between rooms), the path length can be obtained in meters.

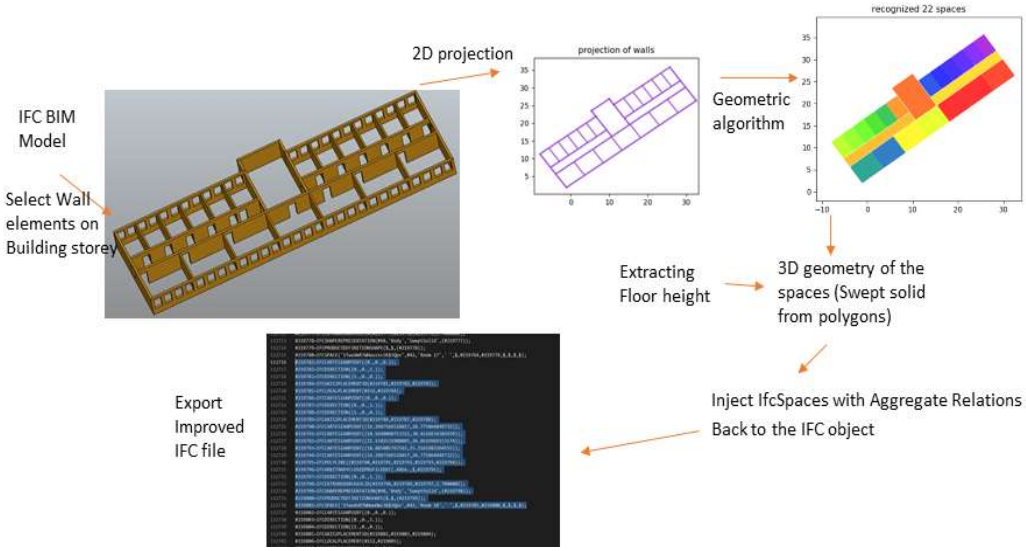


Figure 1. The visualization of the IfcSpace detection workflow.

To classify spaces, we filtered the information that is available in the IFC BIM Model; only Spaces, Doors, and Stairs were extracted. In order to place the Space objects into context, we calculated which rooms were connected, i.e., directly traversable from one to another by humans. To create a graph of accessibility, we calculated which spaces were directly reachable from other spaces through doors, or were in direct spatial contact (spaces are touching). Here, we calculated the spatial proximity with the help of the open source *Opencascade* geometry library (*pythonocc* interface [23])—see Figure 2. There are four types of space–space connections that we checked:

- two spaces are touching (zero distance);
- two spaces are connected through a door (IfcDoor);
- two spaces are connected through an opening (unpopulated IfcOpeningElement);
- two spaces are connected through stairs (IfcStair).

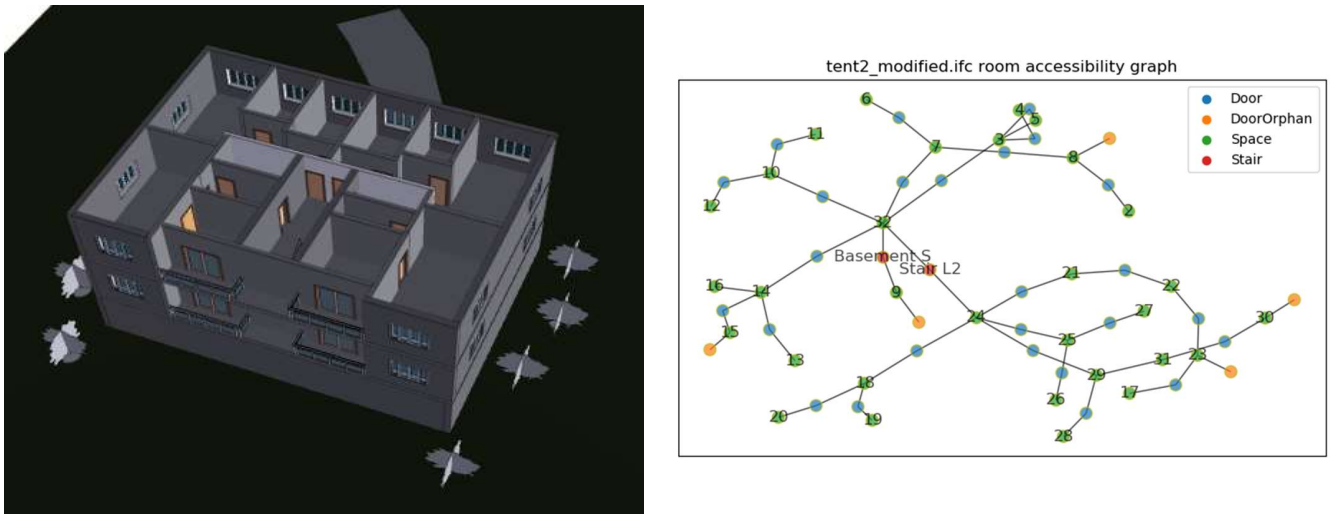


Figure 2. A building model and the corresponding calculated accessibility graph. Orphan doors were the doors with only one connected space (external or terrace doors).

If any of these four connections existed between two spaces, we created an edge between the two spaces in the graph created from the spaces as nodes.

3.3. Extracting Features

To perform the classification, certain geometric features were extracted from the shapes of the spaces and doors. These geometric properties were then used as features for supervised learning. Raw geometric features for the shapes of spaces were calculated using the the *Opencascade* G_prop package (https://dev.opencascade.org/doc/refman/html/class_g_prop___g_props.html, accessed on 15 January 2022). These raw features included:

- Total Volume;
- Principal moments of inertia;
- Gyration radius in the 3 axes in the principal coordinate system;
- Total Volume;
- Static moments in x, y, z directions;
- Height of the shape (in vertical z direction);
- Dimensions of the oriented bounding box of the space shape.

Additionally, the number of windows, doors, openings, and stairs belonging to a space was calculated using geometric proximity. In order to speed up the relatively slow *Opencascade* distance calculation between shapes, we first calculated a distance lower limit with the help of bounding box corners—with much lower computation time and complexity—and only if the distance between the bounding boxes lied below a certain limit were the time-consuming exact geometric distances calculated between the *Opencascade* shapes. The shape representation type of the space (Brep, Surface Model, or Swept Solid) was added as a categorical feature.

As a next step, the normalized features were calculated from these raw features. These included volume/height as the area of the space (in case of horizontal upper slabs), the ratio of the maximum and minimum bounding box dimension, or the ratio of the volume of the bounding box divided by the volume of the space—giving only one for cuboid shaped spaces.

Using these features, a classifier was trained to label each room with one of the following 15 labels: Bathroom, Toilet, Living Room, Kitchen, Bedroom, Corridor, Staircase, Foyer, Storage, Terrace, Office, Kitchen/LivingRoom, Living Room/Bedroom, Other, or Technical Room. Additionally, if the space definition was incorrect in the original file (for example whole floor was marked as one space), the space was labelled as "DELETE". We also let the network predict such incorrect spaces.

3.4. Neural Networks on Graphs

A multi-layer preceptor model (MLP) was implemented to serve as a baseline reference. MLP does not use the available graph information; it consists of two hidden layers and accepts all available numerical and categorical features as input, and outputs the predicted probabilities for each of the 16 room types.

Our first graph-based reference neural network model was the Graph Convolutional Network from Kipf [19]. It operates on a matrix of N nodes, each with F number of features. The Activation in hidden layer l can be given as:

$$h_{l+1} = \mathrm{f}(h_l, A) = \sigma(Ah_lW_l) \tag{2}$$

where A is the normalized adjacency matrix (with the size of $N \times N$); A_{ij} is nonzero if and only if i and j nodes are connected. W_l is the trainable weight matrix in the hidden layer l. The size of W_l does not depend on the size of the graph, just on the number of features F and on the size of the hidden layers. These W_l weights are shared between the nodes, just like the convolution matrix is shared by convolutional networks. For the first layer, H_0 is initialized with the input features. The activation function σ is usually implemented with the tanh function.

With these layers, the prediction for a two-layer GCN, as an example, can be written as:

$$Z_{i\gamma} = softmax\left(\sum_{\kappa}\sigma\left(\sum_{j\beta}A_{ij}\sigma\left(\sum_{k\alpha}A_{jk}X_{k\alpha}W^{0}_{\alpha\beta}+b_{\beta}\right)W^{1}_{\beta\kappa}+b_{\kappa}\right)W^{D}_{\kappa\gamma}\right) \tag{3}$$

Here, we only get contributions to the output if i and j are indices of neighboring spaces and j and k are also neighbors, so k is second neighbor to i. The Greek indices run over the number of features. The trainable bias vectors are denoted by b. The formula shows that in case of two GCN layers for the prediction of the class of a given element, anything up to the second neighbors of this element contribute (for 1-layer GCN, only the first neighbors show a contribution). $X_{k\alpha}$ is the feature α for node k; this is the first layer input $X = h_0$.

$W^{D}_{\kappa\gamma}$ is the weight matrix of the last dense softmax layer. The softmax function returns a vector $Z_{i\gamma}$ for each space i with the size equaling the number of classes and sums up to 1 (so $\gamma = 1 \ldots 15$ for the 15 room categories). The elements of this vector can be treated as probabilities of classes; the one with the highest "probability" is the predicted class.

The GCN in the stellargraph [27] python package is implemented for transductive inference (classifying the unlabeled nodes in a known graph), which does not naturally generalize to unseen nodes. This problem of inductive inference (predicting labels of the nodes of a previously unseen graph) was solved by saving the learned weight matrices (W_l) and creating a new stellargraph GCN instance from these previously learned weights for an unseen graph without re-training.

In the course of training the GCN model we have realized that it often does not outperform the simple MLP network in the inductive setup. Combining the two predictions was attempted with the hope that an ensemble model would perform better. In order to merge the predictions of the two models, we calculated a joint prediction. For this, we simply summed up the output probability matrix of the two classifiers and determined the joint prediction as the argmax of the single lines of this averaged probability matrix. This method is often referred to as ensemble averaging.

3.5. Weighted Graph Convolutional Network Architecture

As the classification accuracy of the GCN for the inductive setup was not satisfactory, a modification of the GCN architecture was attempted.

For the particular problem of assigning labels to IfcSpaces, the weakness of the GCN is that the importance of the single site vs. neighbor nodes features cannot be adjusted, i.e., they are simply averaged. In the original GCN, the l-th layer output h^{l+1}_{j} is obtained from the layer input ${h^{l}}_{k\alpha}$ with the formula in Equation (2):

$$h^{l+1}_{j\beta} = \sigma\left(\sum_{k\alpha}A_{jk}h^{l}_{k\alpha}W^{l}_{\alpha\beta}\right) \tag{4}$$

where $j, k = (1 \ldots N)$ are the node indices, the Greek letters are the feature indices, A_{jk} is the adjacency matrix that depends only on the graph structure, and ${W^{l}}_{\alpha\beta}$ are the trainable weights in layer l. We proposed an extended layer, with separate weight matrix for the single site elements (diagonal of the adjacency matrix) and separate weights for the off-diagonal elements, corresponding to the neighbors:

$$h^{l+1}_{j\beta} = \sigma\left(\sum_{k\alpha}\mathrm{A}_{jj}{h^{l}}_{j\alpha}W^{diag}_{1\alpha\beta} + A_{jk}{h^{l}}_{k\alpha}W^{off-diag}_{1\alpha\beta}\right) \tag{5}$$

Here, we have two matrices ($W^{diag}_{\alpha\beta}, W^{off-diag}_{\alpha\beta}$) instead of one, which were both separately trained on the data (to optimize the cross-entropy loss function).

With this novel architecture, a higher accuracy for the transductive learning was achieved, and we also obtained higher inductive accuracy, as shown in the last column of Table 1.

Table 1. Inductive Inference evaluation of the Neural Network models: MLP = simple multi-layer preceptor neural network, GCN: Graph convolutional network of Kipf and Welling [11]. GCN_MLP_joint mixture of MLP and GCN classifier, WGCN = novel Weighted Graph Neural Network Model.

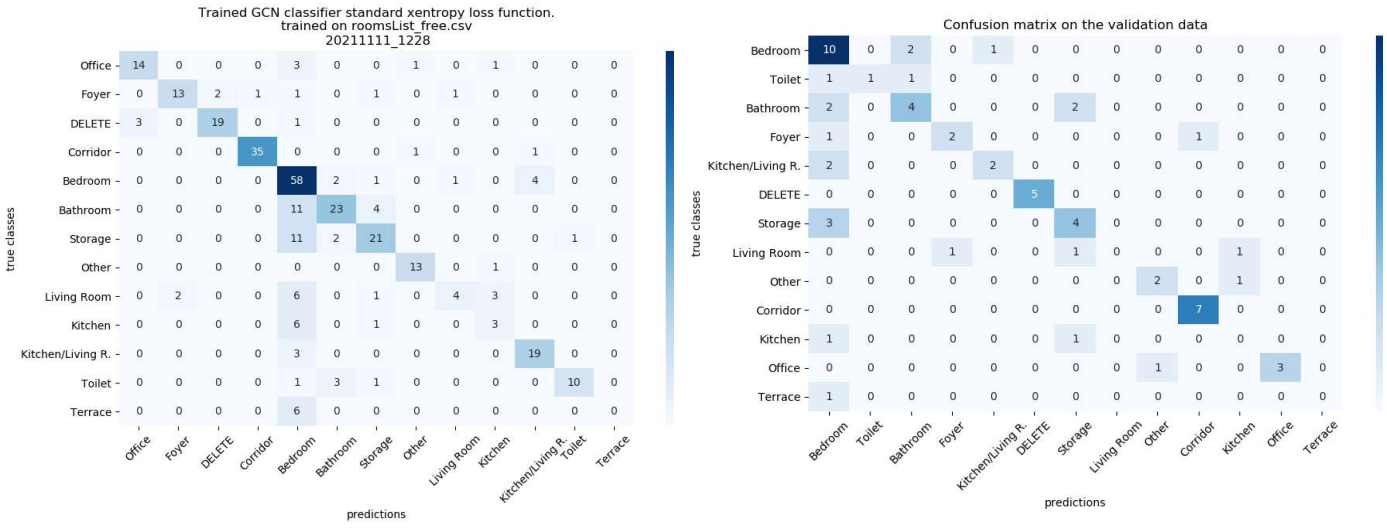
File	MLP	GCN	GCN_MLP_Joint	WGCN (Ours)
Duplex_A_20110907_optimized.ifc	20.0%	25.0%	30.0%	35.0%
FJK-Project-Final-gen.ifc	42.9%	50.0%	35.7%	42.9%
IFC_Schependomlaan_most_spaces.ifc	39.8%	28.9%	39.8%	39.8%
KIT_Smiley-West_Arch.ifc	51.4%	34.3%	64.3%	78.6%
SGD_Munkerud_Arch-1.ifc	23.3%	58.3%	46.7%	50.0%
Tent2_modified.ifc	46.6%	30.0%	28.6%	53.3%
Total accuracy	38.3%	37.6%	45.0%	50.0%

4. Results

We created a labelled dataset of spaces for eight IFC models of residential buildings that were freely available on the internet. The source of the IFC files were archives of the Ghent University IFC repository and archives of the *duraark* repository (https://www.re3data.org/repository/r3d100012506, accessed on 15 January 2022). For these BIM models, we used 13 room types.

For training and validation, we used the 320 spaces from the IFC models and randomly selected 80% of these rooms as a training set; the remaining 20% of the nodes we used as validation set. We optimized the neural network weights for 1500 iterations with the Adam optimizer using the *stellargraph/tensorflow* GCN implementation. The training time was under 1 min on an intel i9 CPU.

In Figure 3, the resulting confusion matrix of the classification is shown. We can see that bedrooms, the most common room types, were also the most often misclassified room types. A 62% classification accuracy was reached on the validation dataset.



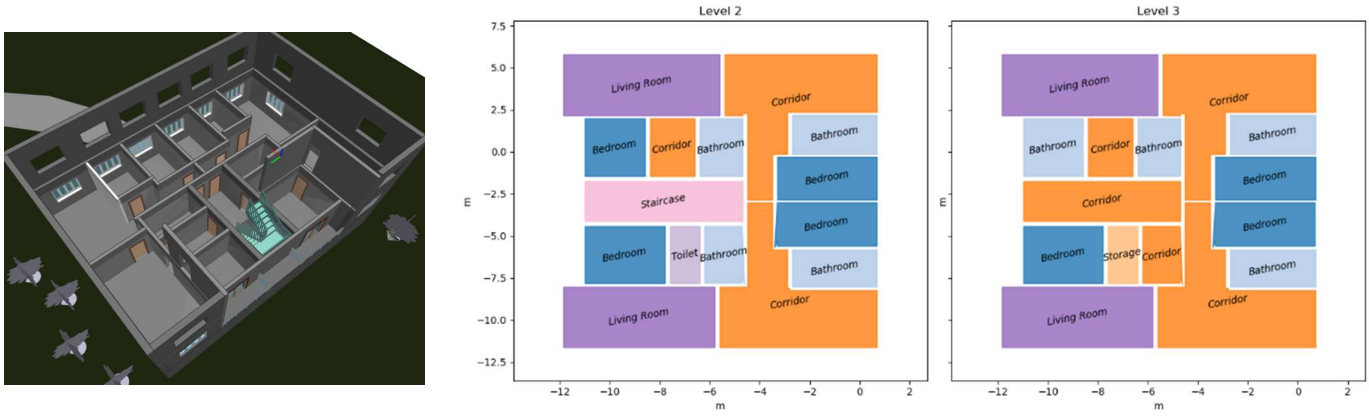
(**a**) (**b**)

Figure 3. Confusion matrix for classification (**a**) of all the training and validation data, for which we obtained an accuracy of 72%; (**b**) For the validation nodes alone (which were 20% of the total nodes), we obtained an accuracy of 62%.

The models were tested in an inductive setup as well; the models were trained on 1500 spaces of nine proprietary IFC files, then they were tested on six other unrelated IFC

files that were neither seen nor processed during the training. With this independence of the training and testing dataset we reached the most realistic accuracy assessment. The obtained accuracy for this test is shown in Table 1. With GCN, we reached an average classification accuracy of 50%, where the accuracy ratio strongly varied between the considered building models.

One example space classification result can be seen on Figure 4.



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

Figure 4. (**a**) An IFC model visualized with BIMVision software (https://bimvision.eu/, accessed on 15 January 2022); (**b**) The output of our WGCN space classifier for the spaces on the two floors of the building model.

Figure 5 shows the confusion matrix for the inductive inference test. Some prediction errors are understandable; for example, the network often mistakes Corridors for Foyers (14 times). The IfcSanitaryTerminal elements (used to represent elements such as drains, water closets, or sinks) were not used as space features, as around half of the used models did not contain such elements. Without this feature, it was hard for the model to differentiate between bathrooms and storage rooms (21 cases).

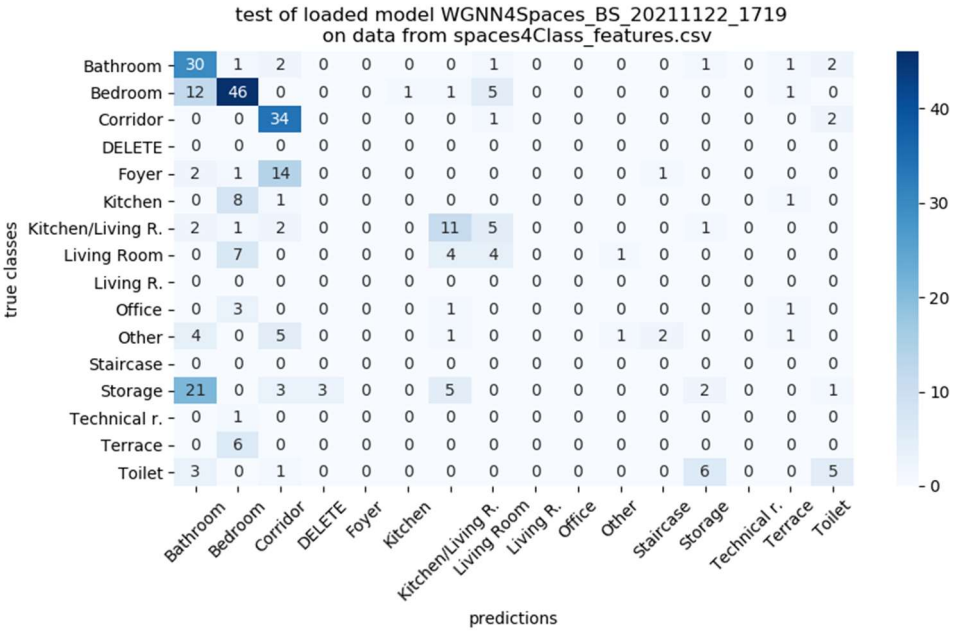


Figure 5. Confusion matrix for the inductive inference test of the WGCN model. The model was trained on a proprietary set of IFC files with 1500 spaces and validated on 283 other spaces from other IFC models as a realistic inference test.

The relatively low classification accuracy stemmed from the high number of room-types—some of them having just a few examples in the training set. Some room-types are hard to differentiate without additional information (like Storage rooms vs. Toilets); we also see that Foyers and Corridors are often confused by the neural model. They have a similar room function, although the difference is only that Foyers are within the apartments. Therefore, it made sense to merge some room types and recalculate both the confusion matrix and accuracy. We grouped the following room types to create the following three aggregate classes:

("Corridor", "Foyer", "Staircase", "Other")
("Kitchen/Living Room", "Living Room", "Living Room/Bedroom")
("Storage", "Toilet", "Technical room")

With this change, we reached an accuracy of 64%; the resulting confusion matrix is shown in Figure 6.

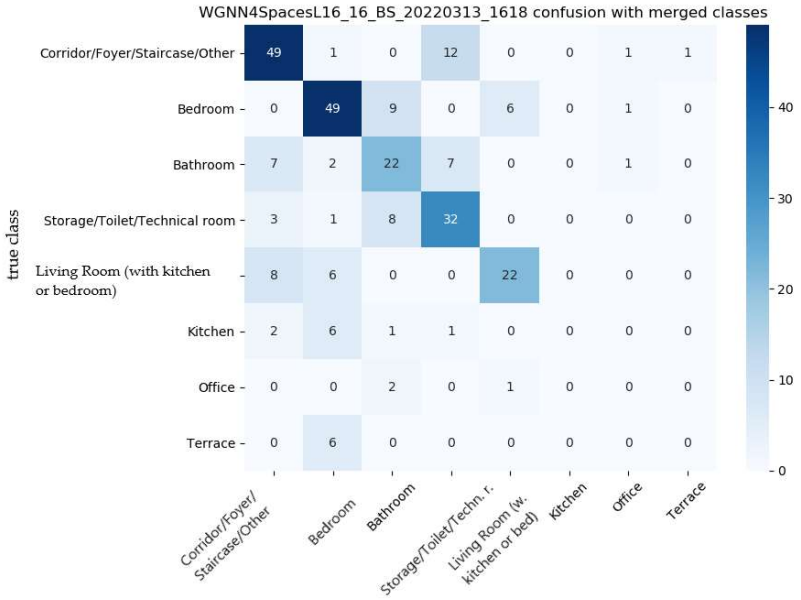


Figure 6. Confusion Matrix of the WGN classification with layer size of two, after merging classes. The accuracy reached 64%.

5. Conclusions and Outlook

A study for classifying IFC spaces with calculated geometrical features, accessibility graphs, and a graph-based machine learning model was presented here. The presented results demonstrate the operation of the workflow; however, there is still room for accuracy improvement before production deployment. For future work, the classification accuracy shall be improved by fine tuning the current workflow—there is an improvement potential that could be achieved by increasing the number of regarded features. Additionally, the inclusion of additional IFC elements such as IfcSanitaryTerminal would improve classification accuracy to differentiate spaces that have those elements (such as toilets, bathrooms, kitchens) from those which usually do not (living rooms, bedrooms, storage). We also did not use the information on whether a space was outside or inside the building, as this would require a time-consuming geometric calculation. However, having such information would be very useful to differentiate between, e.g., a terrace and a bedroom. Another huge quality improvement is expected by using higher quality labelled input data for training the neural networks. For this study, only a couple of IFC files were available, where the spaces within one IFC model were highly correlated (i.e., spaces in different floors or within one floor had very similar geometry). Even a data stock of several thousand spaces would yield only a limited number of independent samples compared to the variety

in existing buildings. It also has to be noted that in low-detailed IFC models (e.g., with the duct system missing from the model), sometimes the labelling of the space type was not obvious for human experts either. For example, toilets and storage rooms have very similar features; they have similar geometry, are windowless, and are typically located towards the core of the building. Additionally, the present analysis does not involve parent apartment-related information (e.g., small apartments usually have only one bathroom, or living rooms are usually the biggest rooms of an apartment). To incorporate such knowledge rules in the classification decision, first the individual apartments would need to be detected. Based on the segmentation of the spaces into apartments and public areas, new features could be calculated, which could be very useful for the space classification. This apartment information inference and processing should be a subject of a follow-up research project. Still, the presented study shows that it is possible to achieve an automated classification of an incomplete BIM model (i.e., one that is missing the classification of rooms), and that the method has the potential to automate the room classification process. It is expected that a similar approach can be used whenever BIM models are handed over from one stakeholder to another, e.g., in the case of issuing a building permit based on the BIM model data.

Author Contributions: Conceptualization, A.B. and G.Z.; methodology, A.B. and M.Š.; software development, A.B. and M.Š.; validation and software testing, B.B.-L.; data curation, B.B.-L. and M.Š.; writing—original draft preparation, A.B.; writing—review and editing, M.Š., B.B.-L. and G.Z.; visualization, A.B.; supervision, A.B. and G.Z.; project administration, A.B. and G.Z.; funding acquisition, A.B. and G.Z. All authors have read and agreed to the published version of the manuscript.

Funding: This research was funded by Forschungsförderungsgesellschaft (FFG), Austria, grant number 880874.

Institutional Review Board Statement: Not applicable.

Informed Consent Statement: Not applicable.

Data Availability Statement: Not applicable.

Conflicts of Interest: The authors declare no conflict of interest.

References

1. ISO International Organization for Standardization. Industry Foundation Classes (IFC) for Data Sharing in the Construction and Facility Management Industries. ISO. Available online: https://www.iso.org/standard/70303.html (accessed on 31 January 2022).
2. Sporr, A.; Zucker, G.; Hofmann, R. Automated HVAC Control Creation Based on Building Information Modeling (BIM): Ventilation System. *IEEE Access* **2019**, *7*, 74747–74758. [CrossRef]
3. Hitchcock, R.J.; Wong, J.; Consulting, H. Transforming ifc architectural view bims for energy simulation: 2011. In Proceedings of the Building Simulation 2011: 12th Conference of International Building Performance Simulation Association, Sydney, Australia, 14–16 November 2011; p. 7.
4. Lilis, G.N.; Giannakis, G.I.; Rovas, D.V. Automatic generation of second-level space boundary topology from IFC geometry inputs. *Autom. Constr.* **2017**, *76*, 108–124. [CrossRef]
5. Krijnen, T.; Tamke, M. Assessing Implicit Knowledge in BIM Models with Machine Learning. In *Modelling Behaviour: Design Modelling Symposium 2015*; Thomsen, M.R., Tamke, M., Gengnagel, C., Faircloth, B., Scheurer, F., Eds.; Springer: Cham, Switzerland, 2015; pp. 397–406. [CrossRef]
6. Belsky, M.; Sacks, R.; Brilakis, I. Semantic Enrichment for Building Information Modeling. *Comput.-Aided Civ. Infrastruct. Eng.* **2016**, *31*, 261–274. [CrossRef]
7. Sacks, R.; Ma, L.; Yosef, R.; Borrmann, A.; Daum, S.; Kattel, U. Semantic Enrichment for Building Information Modeling: Procedure for Compiling Inference Rules and Operators for Complex Geometry. *J. Comput. Civ. Eng.* **2017**, *31*, 04017062. [CrossRef]
8. Daum, S. QL4BIM. 2021. Available online: https://github.com/SimonDaum/QL4BIM (accessed on 23 November 2021).
9. Bloch, T.; Sacks, R. Comparing machine learning and rule-based inferencing for semantic enrichment of BIM models. *Autom. Constr.* **2018**, *91*, 256–272. [CrossRef]
10. Koo, B.; La, S.; Cho, N.-W.; Yu, Y. Using support vector machines to classify building elements for checking the semantic integrity of building information models. *Autom. Constr.* **2019**, *98*, 183–194. [CrossRef]
11. Open CASCADE Technology | Collaborative Development Portal. Available online: https://dev.opencascade.org/ (accessed on 1 February 2022).
12. Koo, B.; Jung, R.; Yu, Y. Automatic classification of wall and door BIM element subtypes using 3D geometric deep neural networks. *Adv. Eng. Inform.* **2021**, *47*, 101200. [CrossRef]

13. Su, H.; Maji, S.; Kalogerakis, E.; Learned-Miller, E. Multi-view Convolutional Neural Networks for 3D Shape Recognition. *arXiv* **2015**, arXiv:1505.00880.
14. LeCun, Y.; Bengio, Y.; Hinton, G. Deep learning. *Nature* **2015**, *521*, 436–444. [CrossRef] [PubMed]
15. Lomio, F.; Farinha, R.; Laasonen, M.; Huttunen, H. Classification of Building Information Model (BIM) Structures with Deep Learning. In Proceedings of the 2018 7th European Workshop on Visual Information Processing (EUVIP), Tampere, Finland, 26–28 November 2018; pp. 1–6. [CrossRef]
16. Emunds, C.; Pauen, N.; Richter, V.; Frisch, J.; van Treeck, C. IFCNet: A Benchmark Dataset for IFC Entity Classification. *arXiv* **2021**, arXiv:2106.09712.
17. Zhang, Z.; Cui, P.; Zhu, W. Deep Learning on Graphs: A Survey. *IEEE Trans. Knowl. Data Eng.* **2020**, *34*, 249–270. [CrossRef]
18. Bronstein, M.M.; Bruna, J.; LeCun, Y.; Szlam, A.; Vandergheynst, P. Geometric deep learning: Going beyond Euclidean data. *IEEE Signal Process. Mag.* **2017**, *34*, 18–42. [CrossRef]
19. Kipf, T.N.; Welling, M. Semi-Supervised Classification with Graph Convolutional Networks. *arXiv* **2017**, arXiv:1609.02907.
20. Song, J.; Yu, K. Framework for Indoor Elements Classification via Inductive Learning on Floor Plan Graphs. *ISPRS Int. J. Geo-Inf.* **2021**, *10*, 97. [CrossRef]
21. Paudel, A.; Dhakal, R.; Bhattarai, S. Room Classification on Floor Plan Graphs using Graph Neural Networks. *arXiv* **2021**, arXiv:2108.05947.
22. Hu, X.; Fan, H.; Noskov, A.; Wang, Z.; Zipf, A.; Gu, F.; Shang, J. Room semantics inference using random forest and relational graph convolutional networks: A case study of research building. *Trans. GIS* **2021**, *25*, 71–111. [CrossRef]
23. PythonOCC | Open CASCADE Technology. Available online: https://dev.opencascade.org/project/pythonocc (accessed on 16 November 2021).
24. IfcOpenShell. Available online: http://ifcopenshell.org/ (accessed on 16 November 2021).
25. Kim, H.; Jun, C.; Cho, Y.; Kim, G. Indoor Spatial Analysis Using Space Syntax, The. *Int. Arch. Photogramm. Remote Sens. Spat. Inf. Sci.* **2008**, *37*, 1065–1070.
26. Hagberg, A.A.; Schult, D.A.; Swart, P.J. Exploring Network Structure, Dynamics, and Function using NetworkX. In Proceedings of the 7th Python in Science Conference, Pasadena, CA, USA, 21–22 August 2008; pp. 11–15.
27. StellarGraph Machine Learning Library. CSIRO's Data61. 2022. Available online: https://github.com/stellargraph/stellargraph (accessed on 2 February 2022).