- Suggested Citation: Wu, J., and Zhang, J. (2022). "Model validation using invariant signatures and logic-based inference for automated building code compliance checking." *Journal of Computing in Civil Engineering*, 36(3), 04022002.
- For final published version, please refer to ASCE database here: https://doi.org/10.1061/(ASCE)CP.1943-5487.0001002

Model Validation Using Invariant Signatures and Logic-Based Inference for Automated Building Code Compliance Checking

Jin Wu S.M.ASCE[1]; and Jiansong Zhang, Ph.D., A.M.ASCE[2*]

Abstract

Fully automated building code compliance checking (ACC) requires accurate information extraction from both building information models (BIMs) and building code chapters, and equally (if not more) importantly, a precise matching between the two. While research on information extraction has been extensively conducted for ACC, there is a lack of investigation of automated and practical information mapping between the extracted information, from BIMs to building code requirements. To address this gap, the authors proposed a new method for BIMs model validation, to validate an input IFC model with regard to building code concepts. This validation method was supported by creating invariant signatures of architecture, engineering, and construction (AEC) objects that capture the geometric nature of the objects. Target concepts from building codes are classified into four categories: (1) explicit concepts, (2) inferable concepts, (3) user-assisted concepts, and (4) system defaults. Identification algorithms are developed for all four categories based on the invariant signatures of AEC objects. An experiment was conducted to test the proposed method on validating five real commercial project models with selected concepts from the International Building Code 2015. Comparing to a manually developed gold standard, 99.8% precision and 99.6% recall were achieved. This demonstrates

[1] Automation and Intelligent Construction (AutoIC) Lab, School of Construction Management Technology, Purdue University, West Lafayette, IN, 47907, PH (301) 275-1272; email: wu1275@purdue.edu.

[2] Automation and Intelligent Construction (AutoIC) Lab, School of Construction Management Technology, Purdue University, West Lafayette, IN, 47907, PH (765) 494-1574; FAX (765) 496-2246. (*corresponding author) email: zhan3062@purdue.edu.

that the proposed method is promising in supporting information matching between BIMs and building code concepts for ACC purpose.

Introduction

Building information modeling (BIM) is very promising in the architecture, engineering, and construction (AEC) domain (Azhar 2011). It helps information conversion and communication in a more effective way. Comparing to the traditional method of manual building code compliance checking, automated compliance checking (ACC) that utilizes BIM technologies is expected to be more efficient in time, cost, and tends to generate fewer errors (Nguyen and Kim 2011). However, current BIM technology, such as industry foundation classes (IFC) only supports limited information coverage related to building codes (Zhang and El-Gohary 2016a). The seamless linkage between the IFC standard and the building codes is missing. More specifically, the concepts in building codes mostly do not have a direct mapping to the entities defined in IFC standards, thus cannot be directly matched with IFC models in checking the model compliance. To allow the full automation of the model compliance checking process, several efforts have been made. For example, Tan et al (2010) proposed a framework of automated code compliance checking for building envelope design. Yang and Xu (2004) proposed a software implementation of knowledge modeling for building code compliance checking. Zhang and El-Gohary (2016b) proposed a method to extend the IFC schema with extracted regulatory concepts. Xu and Cai (2020) proposed to convert BIM data to an ontology for matching/checking with spatial constraints in utilities regulatory documents. In spite of the advancement made by these existing efforts, there is still the lack of a systematic investigation of the feasibility of automation in matching BIMs with building code concepts, at the instance level and from the practical point of view. To address this gap, the authors propose a new method to conduct BIMs model validation

by automatically identifying building code concepts from the models using invariant signatures and logic-based inference. In this new method, the explicit information that could be directed extracted from BIMs is further used to infer more types of concepts (i.e., information extension) with the help of minimal information input from the users, for arriving at a set of computable building design data that can be directly matched to building code concepts. To represent the extended information, logic facts were generated for their storage. The proposed method does not need to modify or extend the existing schema of the IFC standard in use. Instead, it creates a new information set using invariant signatures that can: (1) validate all the explicit information from the IFC-based BIMs, (2) check for missing information, (3) infer target information, and (4) solicit un-inferable information from users, for mapping to corresponding concepts from building codes. For the missing information, only minimal input from users are required, through the help of inference making of intermediate data that brings BIMs data closer to target concepts. The resulting logic facts from the proposed method also ease the integration into building code compliance checking systems.

Background

Domain knowledge representations of building codes

To enable automated building code compliance checking, one of the main steps is to convert the building codes written in natural/human language into computing language. In natural language's discussion, syntax refers to the word sequences, semantics refers to the sense and meaning, and pragmatics refers to different interpretations in different contexts (Nawari 2018). To enable machines to process such syntax, semantics, and pragmatics, natural language processing (NLP) (Nadkarni et al. 2011) methods were developed and used. NLP approach has been shown to be promising in converting building codes rules and infrastructure regulations written in plain text

into computable representations (Zhang and El-Gohary 2016b; Xu and Cai 2020). Regarding computable representations of building codes, a lot of research has been done in representing the building code requirements in various computable formats. For example, Khemlani (2011) proposed to use predicate logic to represent building codes in FORNAX (i.e., a C++ library for IFC data editing). Ding et al. (2006) proposed to use rule-based language to represent accessibility requirements in building codes. Martins and Monteiro (2013) proposed to use XML-based language to represent building codes of water systems. Tan et al. (2010) proposed to use XML-based language to represent building codes for checking building envelope using the Jboss rule engine. In addition, Jeong and Lee (2009) proposed to use direct hard coding to check building fire resistance requirements automatically. Nawari (2012) proposed to use object-oriented representation for encoding a knowledge domain. Zhang and El-Gohary (2016c) proposed to use first-order logic (FOL) for encoding building codes. FOL consists of objects, relations, and functions. In recent research, domain-specific ontology was also used in constructing a set of semantic and syntactic structures for building codes (Zhang and El-Gohary 2016c). Independent from such computable representations, however, is the need of matching BIMs to related building code concepts.

Automated code compliance checking

The first successful effort for code compliance checking can be traced back to the 1960s when Fenves (1966) designed an if-then system to represent American Institute of Steel Construction (AISC) standard specifications. Later, many research efforts followed and made advancement in automated code compliance checking (Lopez and Wright, 1985; Lopez et al., 1989, Garrett and Fenves, 1987). In recent studies (Holzer 2015, Sionov et al. 2015, Volk et al. 2014, Zou et al. 2017, and Sacks 2018), researchers showed great interests in using building information

modeling (BIM) to support code compliance checking, which is expected to be interoperable among platforms for designers, contractors, clients, vendors, and others. An important problem each and every BIM-based ACC system must address, is how to match building design information from BIMs to building code concepts that are essential building blocks of regulatory rules in building codes. This was reflected in the four-stage rule checking framework that Eastman et al. (2009) summarized which included "(1) rule interpretation and logical structuring of rules for their application; (2) building model preparation, where the necessary information required for checking is prepared; (3) the rule execution phase, which carries out the checking, and (4) the reporting of the checking results." Both the first two stages are intended to prepare input (i.e., building design and building codes, respectively) for the third phase – rule execution. In order for the building code rules to execute over the building design input, there lies the matching between the two inputs. In tackling this problem, recent ACC efforts mainly endeavored at the rule level (i.e., regulatory requirements from building codes). For example, Kasim et al. (2013) proposed a reusable solution for sustainability compliance checking using the requirement-applicabilities-selection-exception (RASE) methodology developed by Hjelseth and Nisbet (2011). Their method can extract sustainability requirements and convert them into compiled rules for use with a rule engine. It has the potential to support dynamic checking during the building design stage. Solihin and Eastman (2015) proposed to classify the rules in building codes into six categories, including: (1) rules for checking the well-formedness of a building model, (2) rules for building regulatory code checking, (3) rules for constructability/other contractor requirements, (4) rules for safety/other rules with corrective actions, (5) rules for warranty approvals, and (6) rules for checking BIM data completeness. Zhang and El-Gohary (2017) proposed a semantic NLP and logic reasoning-based method to support fully automated

code compliance checking. Their prototype achieved 98.7% recall & 87.6% precision in noncompliance detection in Chapter 19 of the International Building Code 2009 which was based on representing each regulatory requirement as a logic rule. Haubler et al. (2021) proposed a rule-based method to implement the code compliance checking of railways, with 12 categories of rules (e.g., component definitions, directional definitions). Their approach was shown to be able to automatically examine 37%-75% of the 943 rules.

BIM has a great potential in supporting automated code compliance checking, especially with the support of IFC standard, which is open and platform-neutral. However, despite the existing efforts and progresses achieved, practical obstacles remain, in information extraction and matching from both BIMs and specific building codes, at the regulatory concept level. For example, BIMs are not expected to contain explicit depictions of "egress", which is an important concept in building codes.

Information exchange in building information models (BIMs)

Information exchange has been frequently studied (Sarawagi 2008, and Yang et al. 2019), especially using the ontology-based approach in recent research (Fernández et al. 2011, Yehia et al. 2019, Paliouras et al. 2011, and Wimalasuriya and Dou 2010). For information exchange in BIMs, in a design science research by Ding et al. (2017), a real-time quality checking system was accomplished using Industrial Foundation Classes-based Inspection Process Model (IFC-IPM). Their system demonstrated the efficiency in quality information exchange and could be used in inspection activities. Kim et al. (2016) developed a new approach to extract and process material information in BIMs, to explore energy-saving options early in the design phase. The major limitation of such an approach was the inaccuracies during simplifications in

135 construction/material data. Their new system consisted of three parts, namely, information
136 extraction, material property matching, and file writing. The system improved the efficiency of
137 energy modeling by eliminating the manual information input and increased the accuracy using
138 the developed matching algorithms.

139 For information exchange of IFC models in automated code compliance checking, Zhang and El-
140 Gohary (2019) proposed a machine learning-based approach to map building code concepts and
141 relations to IFC elements and relations. Their method achieved 77% matching accuracy in IFC
142 elements and 78% accuracy in IFC relations, which is representing the state of the art. For
143 practical building code compliance checking automation, performance improvement will be
144 needed. In addition, many concepts in building codes do not have a one-to-one mapping to IFC
145 entities. The bridge between building codes concepts and IFC entities still need to be built with
146 additional ways.

147 ***Identification of building components***

148 There is also no lack of research in automated identification of building components. For example,
149 Quintana et al. (2018) proposed a method for door detection from 3D colored point clouds data.
150 Their approach could detect open, semi-open, and closed doors from the laser scanned data of an
151 indoor environment by integrating geometry, colors, and other characteristics into the detection
152 analysis. As a result, they were able to detect doors with close to 100% precision and higher than
153 90% recall. Adán et al. (2018) proposed a method for detecting secondary building components
154 (e.g., MEP components) from laser scanners. They proposed a 6D approach (XYZ+RGB) to
155 recognize small objects such as switches and signs by inferring the objects following a consensus
156 procedure, to create an as-is semantically rich 3D model. Puente et al. (2014) proposed a method

for detecting road tunnel luminaires using mobile Light Detection and Ranging (LiDAR) technology. Hamledari et al. (2017) proposed a method to detect components of under-construction indoor partitions using computer vision-based technologies. It was found that the methods for detecting and constructing as-is building components have been studied extensively with practical results. However, reasoning about semantic building concepts (e.g., egress) from as-design models was under-researched to produce reliable and practical results, as discussed in the information exchange section. The detection of building code concepts is also different from detecting objects, because the building code concepts may involve multiple objects, relationships between them, and their combinations. As a result, new methods need to be developed to help identify these building code concepts, which the existing methods did not cover.

Proposed Method for ACC-Oriented BIMs Model Validation

The authors propose an iterative method to extend the information from BIMs to better support information mapping to building codes, for use in ACC systems. Different from previous approaches that directly map the building code concepts to IFC entities, which still do not provide enough information in compliance checking, or approaches that extend the IFC schema, which requires deep knowledge and understanding of IFC, the authors propose to use invariant signatures of AEC objects (Wu et al. 2021) as intermediate results to connect building code concepts and IFC entities, and extend the information from IFC models to infer more concepts based on the invariant signatures, to better map the information in IFC-based BIMs to building code concepts (Fig. 1).

The authors propose an iterative approach to develop an intermediate information set with extended information added to BIMs, to provide information mapping between BIMs and

building code concepts, which in turn better supports automated compliance checking compared to the state of the art (Fig. 2). In this proposed method, there are four main steps: (1) construct invariant signatures for AEC objects from IFC models, (2) identify and classify target concepts from logic rules that represent regulatory information from building codes, (3) develop algorithms for extending model information to match target concepts, and (4) generate logic facts to store the extended information. The algorithms are developed following a data-driven approach so that the algorithms can be continuously developed to cover more concepts and model representations. New rules and algorithms can be added under the condition that they do not interfere with previous results, to ensure compatibility and robustness. With more training data, the system will be more robust. It is not our goal in this paper, however, to build the ultimate all-in-one set of algorithms that cover each and every possible concept in building codes. Our goal in this paper is to introduce the method and framework to enable that.

Step 1 - Construct invariant signatures for AEC objects from IFC models

To allow the information matching in later steps, this step extracts information from the BIMs. During this processing of the BIMs data, invariant signatures (Wu et al. 2021) are used to convert the IFC models into value entries of a set of pre-defined features, a data format that is easier to process and use. Invariant signatures are "a set of intrinsic properties of the object that distinguish it from others and that do not change with data schema, software implementation, modeling decisions, and/or language/cultural contexts." (Wu et al. 2021). They capture the geometric nature of the IFC model and can fully represent the AEC objects in the IFC models. In this proposed method, invariant signatures are extracted from BIMs using the IFC2x3 standard, which is commonly used in commercial projects. While invariant signatures are platform neutral and independent from specific data schema, during their use, however, they still need to be extracted

from practical models that follow certain data standards. For example, consistent invariant signatures must be generated from different representations of the same shape, e.g., boundary representations (Brep) and extruded solid representations, two of the most commonly used 3D representations in IFC. In addition to these commonly used representations, a data-driven approach is used to develop algorithms for extracting information from the models that follow uncommon representations. In summary, the state-of-the-art invariant signatures extraction algorithms by Wu and Zhang (2019a, b) and Wu et al. (2021) are extended to allow full support for information extraction and information exchange for BIMs model validation.

Experimental testing requires different BIMs as training data and testing data. The performance in terms of precision and recall is reported using element-level assessment, and the results are manually verified. During the training phase, the results shall achieve 100% in information extraction from the training models, and then the trained algorithms are tested on the testing models to assess their performance.

Step 2 - Identify and classify target concepts

The target concepts are extracted from logic rules that are in turn generated from building codes (regulations) using the state-of-the-art regulatory information extraction and transformation algorithms (Zhang and El-Gohary 2015; 2016b). These algorithms can generate logic rules based on the building codes fully automatically. As the state-of-the-art algorithms are not 100% accurate yet, the generated logic rules are further manually verified and modified as necessary, to form gold standards. This is still much more efficient comparing to generating the logic rules solely manually from scratch. In an experimental trial to generate the logic rules by comparing not using and using the machine-generated results, the average time for generating one logic rule

was reduced from 321.2 seconds/rule to 84.2 seconds/rule (i.e., 73.8% time saving) by using the machine-generated results. In the gold standard logic rules generated from the building codes, the logic clause in the form of "*entity(Entity)"* (where the predicate name has the same string with the argument but in lowercase) indicates a declaration of a concept instance. An observation of the building codes and those logic clauses showed that these predicates contain all the concepts in the corresponding sections of building codes, which are the target concepts to be mapped. On the other hand, because almost all nouns are generated in this form of instance declaration, the generated logic rules contain multiple types of building code concepts. For example, building, door opening, group h, and section 10.4 are all concepts generated from the building codes, which are of completely different types of concepts. To allow seamless information mapping, it is critical to classify these concepts, and then develop rule-based identification algorithms based on the characteristics of each type.

In the proposed method, the authors classify the concepts into four types, which are explicit concept, inferable concept, user-assisted concept, and system default, respectively. These target concepts are classified by developers and end-users are not required to participate in this classification process.

Explicit concepts are the concepts that are directly generable from the BIMs, i.e., the concept has a one-to-one mapping to IFC entities and can be directly identified. For example, a wall concept has corresponding AEC objects in IFC entities: *IfcWall* and *IfcWallStandardCase*.

Non-explicit concepts are more challenging in their detection comparing to the explicit concepts because there is no direct one-to-one mapping between the concepts and IFC entities. However, some non-explicit concepts can be inferred based on related information in the IFC model.

246 Accordingly, there are two sub-categories, inferable and user-assisted. If a consistent inference
247 rule can be found, then that concept is considered inferable, i.e., the inferable concepts are the
248 concepts that can be heuristically inferred from the explicit information in the model with
249 consistency; if not, then that concept is considered to require user judgement and therefore
250 classified as user-assisted. For example, egress is considered an inferable concept whereas
251 bathroom is classified as a user-assisted concept. The identification of inferable concepts is
252 performed by recognizing related IFC entities and then selecting those that satisfy additional
253 constraints based on heuristic rules. The additional constraints leverage the geometry and relative
254 locations of these IFC entities comparing to other IFC entities in the same model. For each
255 inferable concept, one heuristic rule-based algorithm is developed. The algorithms are developed
256 by analyzing the corresponding human recognition process to summarize needed heuristics, and
257 then digitalizing and formalizing these heuristics into rules. The rules are then verified on the
258 training data to validate the correctness of the identification algorithms during the training
259 process. One of the most representative inferable concepts is egress. For identifying an egress,
260 the main heuristic was to identify the exit doors of the building. Therefore, the constraints used
261 in the egress identification algorithm were to find doors that locate at the boundary of the building
262 that connect interior and exterior of the building. Based on the geometric and locational
263 information, interior doors can be excluded from the egress. During this process, all the reasoning
264 can be performed automatically without human judgment, with the developed heuristic rules.
265 While it is not difficult for a human to judge if a door is an egress, the algorithm for identifying
266 these requirements still needs to be developed with exact steps, to ensure the algorithm can
267 function well to generate the expected results robustly.

In contrast, user-assisted concepts are the concepts that can be semi-inferable with additional user input and judgement. The concepts that rely solely on user input could be straightforwardly handled by taking user inputs and therefore are outside of the scope of this paper. While logically it is straightforward to demonstrate if a concept is inferable (i.e., by finding a way to infer it), it is close to impossible to prove if a concept is not inferable. The boundary between inferable and user-assisted concepts therefore depends on the current implementation and reasoning capability. To differentiate from inferable concepts, user-assisted concepts can be inferred to a certain extent, but not 100% inferred. For example, while it might be tempting to computationally infer bathrooms from their size (i.e., bathrooms tend to be smaller than other rooms), this cannot guarantee a consistent result, as storage rooms can also be small whereas public restaurants can have quite large bathrooms. As such, additional manual input is needed to decide if a room is a bathroom. As a result, it is hard to quantify the size of the bathrooms, and relative size does not work either. Furthermore, room functions may change during the planning phase, so additional inputs from users are required for these concepts' identification. By allowing certain extent of user judgment, the proposed method also incorporates some flexibility and error-tolerance in the building design. An important consideration is that the additional user input shall be minimal with inferred information computed to the largest extent possible, and the input should be collected preferably by multiple-choice or yes/no type of questions, with the least amount of domain knowledge and manual efforts needed.

The system defaults are the concepts that are not representing actual objects from a building, such as tables in the regulation, equations for calculations, references to other sections, or other concepts that are not directly related to BIMs, which as a result will not be the focus of this

290 research. However, they are still needed in order for the downstream logic-based reasoning to
291 execute successfully.

292 To illustrate the idea, Fig. 3 shows the relations between the four categories of target concepts
293 based on the above-mentioned IFC models and the IBC 2015. It also reveals that direct mapping
294 from building codes to BIMs is not feasible for all concepts, and IFC models also contain
295 information not used in the building code compliance checking. With the extended information,
296 such gaps in matching BIMs with building code concepts can be addressed.

297 ***Step 3 - Develop algorithms for extending model information to match target concepts***
298 In this step, information extension and matching algorithms are developed to match the invariant
299 signatures generated from Step 1 to the target concepts identified in Step 2. Heuristic rule-based
300 algorithms are developed iteratively until consistent results are obtained in the training data, i.e.,
301 the target concepts matching need to achieve 100% precision and recall in the training data.

302 For explicit concepts, the algorithms are straightforward in that each object will generate one
303 instance of the corresponding target concept. For example, a door object (IfcDoor) from the IFC
304 model can be used to directly generate an instance of the door concept. The attributes of the
305 instance should also be extracted, such as the length, width, etc. These attribute values can be
306 directly obtained from the invariant signatures of the object.

307 For inferable concepts, heuristic rule-based algorithms are developed to extend the explicit
308 information to the target information by inference. Each concept has its own identification sub-
309 algorithm. For example, the egress concept (i.e., target concept) does not have a one-to-one
310 mapping to existing IFC entities. The identification of egresses therefore needs to be conducted
311 through inference. To develop the algorithm, heuristic rules are developed first. For example, one

possible heuristic rule for egress is that doors at boundaries of the building that connect interior and exterior of the building can be recognized as egresses. An algorithm can therefore be developed to identify a door's shape, position, level, and relative locations to decide if the door can be classified as an egress, using this heuristic rule. Then all inferred results are added to the extended information set (i.e., the information originated from the BIMs), so the extended information set now contains all the inferred information in addition to the original explicit information. All information is stored for later use.

For user-assisted concepts, a light user interface (UI) is developed to allow users to input the missing information. Inference algorithms are still developed to infer intermediate results, so that the questions presented to the user only ask for minimal input, and preferably using multiple-choice questions and/or binary yes/no questions. In addition to soliciting user inputs, the algorithms also combine the user input with existing explicit and inferred information, to enrich the extended information set.

At this step, all system default concepts are identified to support further development. No further processing is needed from the model validation perspective, as the information does not directly map to the BIMs per se. For example, section_1003_3 is such a concept, which refers to Chapter 10, Section 1003.3 of a selected building code. While such concepts do not have a direct mapping in the extended information, they are important in the later automated reasoning stage of code compliance checking.

Step 4 - Generate logic facts to store extended information set

With the extended information set produced from the previous step, logic facts are generated to store this information. Instead of higher-order logic (Gallin 2011) and defeasible logic (Naeem

2014), the logic facts store all the instances of target concepts in first-order logic (FOL) format, which is widely used and successfully tested for storing objects and relations to conduct deductive reasoning effectively and expressively (Zhang and El-Gohary 2015). It is machine-readable so that they can be directly used for logic-based automated code compliance reasoning. The advantage of this logic fact format is that it is very easy to read, edit, store, and manage. The logic facts can be directly fed into a logic-based building code compliance checking system for automated reasoning, with no further processing needed on the model information side.

The logic facts follow a uniform representation of concepts and properties. For each instance of the target concept, a numbered instance is declared (e.g., *egress(egress32)*). Then properties of the concept are linked to the instance in the form of logic relations (e.g., *has(egress32, height117*) indicates the height property of egress instance number 32 is represented by height instance number 117). Different concepts may have their unique properties, e.g., the room type property applies to a room but does not apply to a window. For algorithm development of generating logic facts, each target concept uses its own designated sub-algorithm. All generated logic facts are then manually checked for evaluation.

Experiment

For experimental testing, the authors collected five real commercial projects and a residential project, namely, a convenience store, a warehouse, two restaurants (a fast-food restaurant and an Italian restaurant), a hotel, and a duplex apartment. The first five models were provided by our industry partner and all of them are located in Texas, and the last one is a sample residential building model from the open-source common BIM files site (WBDG 2021). All the six models, however, were designed by different architects. For each project, the authors used a

corresponding building model in IFC format. The authors followed a 5:5 ratio for splitting the projects into training and testing models. As a result, the convenience store, warehouse, and the Italian restaurant were used as training data, whereas the fast-food restaurant, the hotel, and the duplex apartment models were used as testing data. Fig. 4 shows the training projects, and Fig. 5 shows the testing projects. The hotel project contains a four-story main building and two one-story side buildings, which has the largest number of building elements, and was also the largest in footage. The duplex apartment project contains a two-story building. The remaining four commercial buildings are all one-story buildings without side buildings. All models are well-functioned complete building models with multiple rooms, walls, doors, etc. For example, the Italian restaurant, the convenience store, the warehouse, the fast-food restaurant, the hotel, and the duplex contained 14, 12, 1, 1, 200, and 11 rooms, 34, 37, 18, 54, 673, and 57 walls, 13, 11, 34, 11, 317, and 14 doors, respectively. In summary, two of the testing models are similar to the three training models, which is one or two-story but with different layouts and different number of rooms, while the other testing model (the hotel project) is less similar in that it contains four stories, and has significantly more rooms, which adds more complications in the validation process.

Step 1 - Construct invariant signatures from IFC models

For the model validation process, the first step is to preprocess the models to extract invariant signatures-based information from the IFC models.

In constructing the invariant signatures, the state-of-the-art algorithms (Wu et al. 2021) were used to extract values from the IFC models. For each AEC object, 36 features were generated as invariant signatures, including geometric, locational, and metadata signatures. The distribution

of the invariant signatures-based features is shown in Table 1. The invariant signatures were able to fully represent the information about building objects of a model. This simplified the process of further conversion, as the invariant signatures were ready to be used in developing the heuristic rule-based classification algorithms. Later algorithms development follows an iterative approach, i.e., for any information found missing in the later development phase, a refinement/extension of the invariant signatures was conducted.

Step 2 - Identify and classify target concepts

With the invariant signatures-based processing of IFC models completed, the building codes were then processed, using the state-of-the-art information extraction and transformation algorithm (Zhang and El-Gohary 2015; 2016b) to generate logic rules from International Building Code 2015 (IBC 2015). To illustrate the idea, the authors chose Chapter 10 for demonstration. The automatically generated logic rules were then manually refined to error-freely represent the regulatory requirements in Chapter 10 of IBC 2015.

To extract the target concepts from the logic rules, the authors developed a target concept identifier that can identify a target concept by recognizing the single-variable conjunct pattern of a logic clause (i.e., conjunct of the form "*entity(Entity)*"). For example, a "*door(Door)"* conjunct indicates a declaration of a door object variable. For each target concept, the logic rules always contain this type of declaration. Therefore, the target concept identification algorithm could identify all the related building code concepts from the logic rules.

With this target concept identification algorithm, the authors were able to identify 1,408 target concepts. These concepts were then classified into the four types, as shown in Table 2 and Table 3. For the concept coverage of models, every model covered most of the 1408 concepts from

Chapter 10 of the IBC 2015. The exceptions were fence, stair, and mezzanine. Only the Italian restaurant contained fences. Only the hotel model and the duplex apartment model contained stairs. No model contained mezzanines.

Step 3 - Develop algorithms for extending model information to match target concepts

Explicit concepts

Explicit concepts were straightforward in their identification algorithm development, as the name suggested. Each explicit concept had a one-to-one mapping to the IFC entities, which was directly used for identifying them in the IFC models. For example, doors were identified through *IfcDoor* and rooms were identified through *IfcSpace*.

The validation of these concepts' identification was conducted by directly checking the number and properties of the corresponding IFC entities. Results showed all the needed explicit concepts were identified with 100% precision and 100% recall in the training data (Table 4).

Inferable concepts

For inferable concepts, one of the most representative concepts is egress. The authors conducted an experiment following the heuristic described in the method. The detailed implementation was as follows:

Step 1: Identify the boundary of the building. The boundary is found by taking the maximum and minimum coordinates of walls and slabs on the first floor of the building. Four line segments need to be identified for each building model.

Step 2: Select doors that are at the boundary of the building. This is achieved by checking the position of the doors. The doors selected are the ones that are close (i.e., within a predefined threshold) to the boundary of the building.

Step 3: Verify the orientation of each door selected from Step 2 and keep the ones that connect the interior and exterior of a building. This is achieved by checking if the door's orientation is in parallel to the corresponding boundary.

The visualization of the egress identification results on the Italian restaurant model is shown in Fig. 6.

As illustrated in Fig. 6, Doors 1, 2, 3, 4, 6, and 8 were not identified as egress because they were not at the boundary of the building (Step 2 criterion was not satisfied). Doors 5 and 7 were further eliminated because they failed to satisfy the orientation criterion at Step 3, i.e., the direction was not in parallel with the corresponding boundary. Finally, Doors 9, 10, 11, 12, and 13 were identified as egresses because they met all the three criteria in the algorithm.

In addition to egress, the authors also developed algorithms for the rest of the inferable concepts (Table 5). For example, for the identification of a building, the main heuristic used was to identify the boundary of the building based on a range for all involved building objects. For identifying floor levels, clustering algorithm was used to cluster slab objects together based on their elevation. For identifying a mezzanine, the main heuristic used was to recognize the slab objects between floor levels. The results of the algorithm development for inferable concepts are shown in Table 6.

User-assisted concepts

For the other subtype of non-explicit concepts, the user-assisted concepts, the information from the IFC models was not sufficient to automatically identify those concepts even with inference. However, the inferences were able to be conducted to an extent to narrow down the range of selection. For example, based on the observation of the models, there was not enough information

for differentiating bathrooms from other rooms. For the target concept of a bathroom, however, it can be inferred that it must be a room, which could help limit the range. Therefore, to identify bathrooms, the algorithm will display questions to the user to help filter through the narrowed range, e.g., "Is room2 a bathroom? Enter 1 for Yes, enter 2 for No." Similarly, algorithms for processing other user-assisted concepts were developed with interactive questions. Table 7 shows the additional information needed from users for successful identification of the user-assisted concepts. Table 8 shows the results of identifying user-assisted concepts.

System defaults

As described above, system defaults were not representing actual objects from a building design and therefore do not have (or need to have) any mapping in the extended information set. As a result, during the model validation process, these system defaults were identified, but no further action was conducted on them. They can be used to support the later compliance checking stage by instantiating logic rules for automated reasoning execution. For example, an equation concept referred to an equation from the building codes. A fully automated compliance checking system would need to use the equation for calculations in the checking, the process of which should also be automated. In line with such an approach, all 1,382 system defaults concepts were identified for future use.

Step 4 - Generate logic facts to store extended information set

With all the algorithms for extending the information set, identified target concepts were stored in logic facts. The logic facts can be directly used for later logic-based automated code compliance reasoning.

To generate logic facts, each instance of the target concept from the extended information was generated into connected B-Prolog (a Prolog system implementation with extensions for programming concurrency, constraints, and interactive graphics) clauses (Zhou 2014) including its declaration and properties, where Prolog is a FOL-based logic programming implementation. For example, the identified egress entity *1qPjXNL6r8rRF28rPA_nZx* (Door 10 in Fig. 6) was stored in "*egress(egress2). height(height32). has(egress2, height32). has_value(height32, 7.33). has_unit(height32, foot). …*" "*egress2*" was used because it was the second identified egress among all the identified egresses. The height of the egress was 7.33 feet. To represent such information, a height instance was declared as "height32", because there were already 31 height instances declared beforehand. The "*has_value(height32, 7.33).*" *and "has_unit(height32, foot).*" defined the value and unit of the height property, respectively. Additional information was also represented in the same format, such as its length, width, position, orientation direction, etc. Table 9 shows the results of the logic clauses generated from the extended information set.

Results and Analysis

To evaluate the robustness of the proposed method and developed algorithms, the authors tested them on the three testing models, namely, the fast-food restaurant model, the hotel model, and the duplex apartment model (Fig. 5). The results are shown in Tables 10-13. Table 10 shows the results on explicit concepts. Note that the hotel model had four roofs which were not directly observable in Fig 5, so the authors showed the roofs again in Fig. 7.

Table 11 shows the results on inferable concepts. The precision was 96.0% and the recall was 93.9%. As an example illustration, the egresses identified are shown in Fig. 8 and Fig. 9, for the fast-food restaurant model and the hotel model, respectively. For fast-food restaurant model, both

egresses were successfully identified. For the hotel model (Fig. 9), six egressed were successfully identified whereas five egresses were missed. This error occurred because there were three separate buildings in the hotel model, whereas the algorithm only identified one building - the main building. A further analysis showed that this was because in all three training models there was only one building for each model. Following the data-driven approach, the assumption of one building for each model was established from the training models, whereas it does not hold in the hotel testing model. As a result, the trained algorithms in our experiment failed to identify the other two buildings and the corresponding five egresses. In addition, one assembly area was also incorrectly identified because of this error.

Table 12 shows the results of user-assisted concepts. The developed algorithms produced simple binary (Yes/No) questions based upon which perfect precision and recall could be achieved. Table 13 shows the generated logic facts for the whole model.

As shown in Table 13, the algorithms achieved an overall 99.8% precision and 99.7% recall which is very promising. However, for the inferable concept category, the precision was only 96.8% and the recall was only 93.9% in the generated logic facts. The logic clause-level recall was different from the concept-level recall shown in Table 11, because one egress concept needs to be represented by multiple logic clauses.

The error in the inferable concept category occurred because the developed system did not identify all three buildings in the hotel model, which can be seen in Fig. 7 and Fig. 9. A further inspection showed that the developed algorithm used a relatively strong assumption that a building is close to a rectangular shape (with some error margin) and there is only one building in the model. This assumption was valid during the training phase because all training models

were single-building models with a rectangular-shaped boundary. However, this assumption was not valid on the testing hotel model because there were two side buildings attached to the main building. Therefore this hotel model did not have a one-to-one mapping between buildings and *IfcBuilding* entities (i.e., there was only one *IfcBuilding* entity but three connected buildings). This unseen multi-building model resulted in an error in the developed system and caused a 4% drop in precision and a 6.1% drop in recall on the logic clauses of the inferable concept category. To resolve it, the authors did a follow-up experiment to modify the algorithm to allow polyline shape as building boundaries and allow multiple buildings to exist in one project model. This modification enabled the algorithm to then correctly identify all buildings and egresses. Based on the nature of rule-based algorithms, the same level of performance may not be readily achieved on unseen patterns. However, with further development on more training data to cover more concepts and patterns for building code and building design models, respectively, the expectation is that the accumulated set of algorithms will continuously be expanded and therefore enhanced its robustness, to asymptotically approach the ultimate superset of algorithms.

In Chapter 10, 1408 regulatory concepts can be extracted, which is sufficient to demonstrate the robustness of the method. This method is expected to generate comparable results on other chapters of the IBC 2015. For other chapters in the IBC 2015, there are many concepts that were in common with Chapter 10, such as building, wall, ceiling, etc. For new concepts, the proposed method can also be used to categorize them and develop new identification algorithms accordingly.

In summary, the proposed method achieved an overall 99.8% precision and 99.6% recall in the resulting logic clauses.

Contributions to the Body of Knowledge

The contributions to the body of knowledge are four-fold. First, the authors investigated the feasibility of automated inference of non-explicit information in BIMs, to match with building code concepts. It was found that this feasibility depends on the type of non-explicit information to be derived. For non-explicit information that is not directly inferable, the authors still leveraged inference to reduce the needed manual input from users. Second, the authors presented a new categorization of the concepts from building codes in four categories: (1) explicit, (2) inferable, (3) user-assisted, and (4) system defaults, in the context of serving as target concepts of validating/matching with BIMs, to facilitate the analysis of BIMs for ACC in a divide-and-conquer manner. Third, in the identification of concepts from BIMs, the authors developed invariant signatures-based algorithms for matching with each of the four categories of building code concepts, with high recall and precision. Last but not least, the proposed method was able to extend the information from BIMs and check for missing information in the context of ACC. Combined with logic rule-based algorithms, the resulting information set (i.e., the collected information) was demonstrated to be significantly extended to facilitate the matching of BIMs with building codes. In summary, the authors made a solid advancement in filling the gap of information mapping from BIMs to building codes for ACC, by introducing an extended information set using invariant signatures-based inferences. This method supports model validation in ACC and therefore indirectly and positively affects interoperability of BIM.

Conclusions

The authors proposed an iterative method for BIMs model validation using invariant signatures and logic inference to support automated building code compliance checking. The proposed method can extend the information extracted from BIMs with non-explicit concepts, to support

mapping from BIMs to target concepts in building codes. The proposed method uses invariant signatures of AEC objects as intermediate results to extract information from IFC entities and extended the extracted information by inferring more concepts based on heuristic rule-based algorithms and user inputs. An experiment on International Building Code 2015 and five commercial building models showed that the proposed method achieved 99.7% precision and 99.5% recall in generating the extended information set. The proposed method was shown to function with consistent results between testing data and training data, in mapping the extended information set from BIMs to building code concepts. This helps address the existing research gap of BIMs validation to support fully automated building code compliance checking.

Limitations and Future Work

The authors acknowledged the following limitations of this paper: (1) the proposed method was tested in only one chapter (i.e., Chapter 10) of the IBC 2015, how it will perform in other chapters and other building codes remain to be tested. However, the authors would expect comparable results on other chapters and other building codes using the same method. (2) In order for the processing of inferable concepts to be error-free, the assumption used in the heuristic rules needs to be broadly applicable in any foreseeable BIMs. (3) Due to the nature of rule-based algorithms, it is time-consuming and labor intensive to grow the number of concepts and patterns covered, whereas error is always possible before the number of concepts and patterns saturate. However, the authors believe the number of concepts and patterns in this context is enumerable, and a compatible set of rules are feasible with careful implementation, such as by performing a comprehensive check after each new rule is added. In addition, while the theoretical optimal rule set may be hard to achieve, it is always feasible to arrive at practically "comprehensive" solutions

for a known scope (e.g., concepts in International Fire Code) using data-driven approaches. Lastly, the method was validated only on commercial and residential buildings. Other types of built environment, such as civil infrastructure (e.g., railway station) remain to be tested. In their future research, the authors plan to test their method on more building codes and explore ways to ensure the broad applicability and compatibility of heuristic rules. Furthermore, more models will be used to accumulatively and continuously expand the set of algorithms and building code concepts covered.

Data Availability Statement

All data that support the findings of this study are available from the corresponding author upon reasonable request.

Acknowledgement

The authors would like to thank the National Science Foundation (NSF). This material is based on work supported by the NSF under Grant No. 1827733. Any opinions, findings, and conclusions or recommendations expressed in this material are those of the authors and do not necessarily reflect the views of the NSF.

Reference

Adán, A., Quintana, B., Prieto, S.A., and Bosché, F. (2018). "Scan-to-BIM for 'secondary' building components." *Advanced Engineering Informatics,* 37, 119-138.

Azhar, S. (2011). "Building Information Modeling (BIM): Trends, Benefits, Risks, and Challenges for the AEC Industry." *Leadership and Management in Engineering*, 11(3), 241-252.

Whole Building Design Guide (WBDG). (2021). "Common building information model files and tools." < https://www.wbdg.org/bim/cobie/common-bim-files> (Jul 20, 2021).

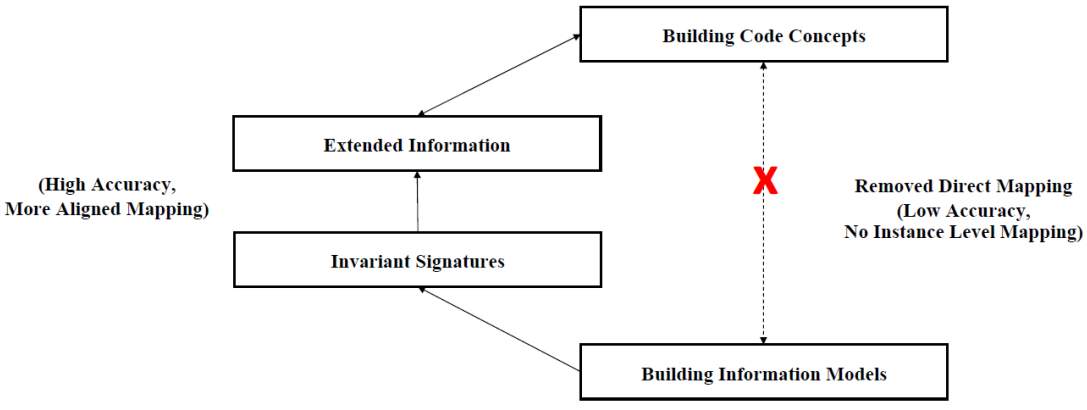
605 Ding, L. Drogemuller, R., Rosenman, M., Marchant, D., and Gero. J. (2006). "Automating code
606 checking for building designs: DesignCheck." *Clients Driving Innovation: Moving Ideas into*
607 *Practice*, 1-16.
608
609 Ding, L., Li, K., Zhou, Y., and Love, P.E.D. (2017). "An IFC-inspection process model for
610 infrastructure projects: Enabling real-time quality monitoring and control". *Automation in*
611 *Construction*, 84, 96-110.
612
613 Eastman, C., Lee, J., Jeong, Y., and Lee, J. (2009). "Automatic rule-based checking of building
614 designs." *Automation in Construction*, 18(8), 1011-1033.
615
616 Fernández, M., Cantador, I., López, V., Vallet, D., Castells, P., and Motta, E. (2011) "Semantically
617 enhanced information retrieval: an ontology-based approach." *Web Semantics*, 9(4), 434-452.
618
619 Fenves, S. J. (1966). "Tabular decision logic for structural design." *Journal of the Structural*
620 *Division*, 6(92), 473-490.
621
622 Gallin, D. (2011). "Intensional and higher-order modal logic With applications to Montague
623 semantics." Burlington : Elsevier Science, Amsterdam.
624
625 Garrett, J. H., Jr. and Fenves, S. J. (1987). "A knowledge-based standard processor for structural
626 component design." *Engineering with Computers*, 2(4), 219-238.
627
628 Hamledari, H.,McCabe, B., and Davari. S. (2017). "Automated computer vision-based detection
629 of components of under-construction indoor partitions." *Automation in Construction,* 74, 78-94.
630
631 Haubler, M., Esser, S., and Borrmann, A (2021). "Code compliance checking of railway designs
632 by integrating BIM, BPMN and DMN." *Automation in Construction,* 1, 121.
633
634 Holzer, D. (2015). "BIM manager's handbook. Best practice BIM. EPart 1." Wiley, West Sussex,
635 England.
636
637 Hjelseth, E. and N. Nisbet (2011). "Capturing normative constraints by use of the semantic mark-
638 up (RASE) methodology." *Proc., CIB W78 2011 - 28th International Conference-Applications of*
639 *IT in the AEC Industry*, Conseil International du Bâtiment (CIB), Rotterdam, Netherlands.
640
641 Jeong, J. and Lee, G. (2009). "Requirements for automated code checking for fire resistance and
642 egress rule using BIM." *International Conference on Construction Engineering and Project*
643 *Management*, 316-322.
644
645 Kasim, T., Li. H., Rezgui, Y., and Beach, T. (2013). "Automated sustainability compliance
646 checking process: proof of concept." *CONVR 2013*, <http://itc.scix.net/paper/convr-2013-1>
647 (accessed 2/1/2021).
648

649 Khemlani, L. (2011). "CORENET e-PlanCheck: Singapore's automated code checking system.
650 AECbytes." <http://www.aecbytes.com/feature/2005/CORENETePlanCheck.html> (accessed
651 2/1/2021).
652
653 Kim, H., Shen, Z., Kim, I., Kim, K., Stumpf, A., and Yu, J. (2016). "BIM IFC information mapping
654 to building energy analysis (BEA) model with manually extended material information."
655 *Automation in Construction*, 68, 183-193.
656
657 Lopez, L. A. and Wright, R. N. (1985). "Mapping Principles for the Standards Interface for
658 Computer Aided Design." National Bureau of Standards, Gaithersburg, MD.
659
660 Lopez, L., Elam, S., and Reed, K. (1989). "Software concept for checking engineering designs for
661 conformance with codes and standards." *Engineering with Computers*, 5(2), 63-78.
662
663 Martins, J.P. and Monteiro, A. (2013). "LicA: A BIM based automated code-checking application
664 for water distribution systems." *Automation in Construction*, 29, 12-23.
665
666 Naeem, K.J. (2014). "A defeasible logic programming-based framework to support argumentation
667 in semantic web applications." Springer, New York.
668
669 Nadkarni, P.M., Ohno-Machado, L., Chapman, W.W. (2011). "Natural language processing: an
670 introduction." *Journal of the American Medical Informatics Association*, 18(5), 544-551.
671
672 Nawari, O. "Automating Codes Conformance." *Journal of Architectural Engineering*, 18(4), 315-
673 323.
674
675 Nawari O. (2018). "Building information modeling: automated code checking and compliance
676 processes." CRC Press, Boca Raton.
677
678 Nguyen, T., and Kim, J. (2011). "Building code compliance checking using BIM technology."
679 *Proc., 2011 Winter Simulation Conference (WSC),* IEEE, New York, 3395-3400.
680
681 Paliouras, G., Spyropoulos, C.D., and Tsatsaronis, G. (2011). "Knowledge-driven multimedia
682 information extraction and ontology evolution: bridging the semantic gap." Springer, Berlin,
683 Heidelberg.
684
685 Puente, I., González-Jorge, H., Martínez-Sánchez, J., and Arias. P. (2014). "Automatic detection
686 of road tunnel luminaires using a mobile LiDAR system." *Measurement: Journal of the*
687 *International Measurement Confederation,* 47(1), 569-575.
688
689 Quintana, B., Prieto, S.A., Adán, A., and Bosché, F. (2018). "Door detection in 3D coloured point
690 clouds of indoor environments." *Automation in Construction*, 85, 146-166.
691
692 Sarawagi, S. (2008). "Information Extraction." *Foundations and Trends in Databases*, 3(3), 261-
693 377.
694

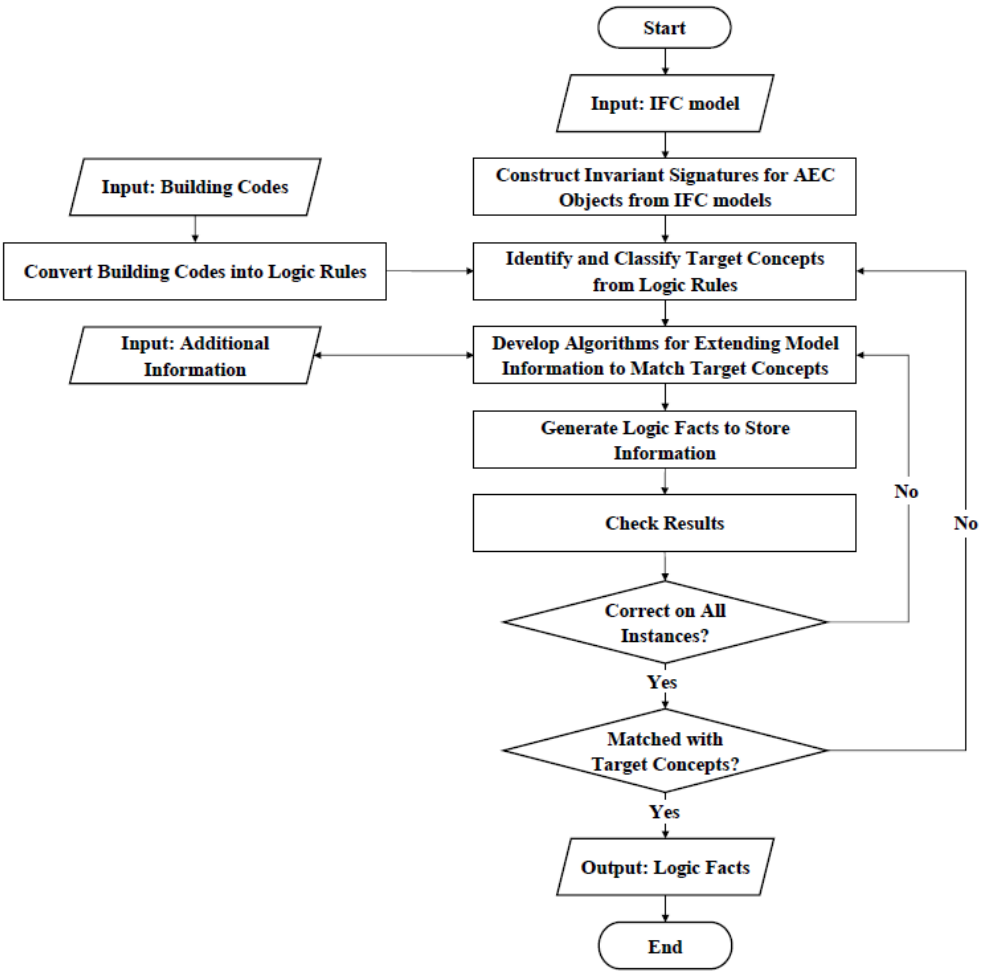
695 Tan, X., Hammad, A., and Fazio, P. (2010) "Automated code compliance checking for building
696 envelope design." *Journal of Computing in Civil Engineering*, 24(2), 203-211.
697
698 Sacks, R. (2018). "BIM handbook: a guide to building information modeling for owners, designers,
699 engineers, contractors and facility managers." Hoboken, New Jersey Wiley.
700
701 Sionov, R.V., Vlahopoulos, S.A., and Granot, Z. (2015). "Regulation of BIM in health and disease."
702 *Oncotarget*, 6(27), 23058-23134.
703
704 Solihin, W., and Eastman, C. (2015). "Classification of rules for automated BIM rule checking
705 development." *Automation in Construction*, 53, 69-82.
706
707 Volk, R., Stengel, J., and Schultmann, F. (2014). "Building Information Modeling (BIM) for
708 existing buildings - Literature review and future needs." *Automation in Construction*, 38, 109-127
709
710 Wimalasuriya, D.C., and Dou, D. (2010). "Ontology-based information extraction: An
711 introduction and a survey of current approaches." *Journal of Information Science*, 36(3), 306-323.
712
713 Wu, J., Sadraddin, H.L., Ren, R., Zhang, J., and Shao, X. (2021). "Invariant signatures of
714 architecture, engineering, and construction objects to support BIM interoperability between
715 architectural design and structural analysis." *Journal of Construction Engineering and*
716 *Management*, 147(1).
717
718 Wu, J., and Zhang, J. (2019a). "Introducing geometric signatures of architecture, engineering, and
719 construction objects and a new BIM dataset." *Proc., 2019 ASCE International Conference on*
720 *Computing in Civil Engineering*, ASCE, Reston, VA, 264-271.
721
722 Wu, J., and Zhang, J. (2019b). "New automated BIM object classification method to support BIM
723 interoperability." *Journal of Computing in Civil Engineering*, 33(5), 04019033.
724
725 Xu, X., and Cai, H. (2020). "Semantic approach to compliance checking of underground utilities."
726 *Automation in Construction*, 109, 103006.
727
728 Yang, Q.Z, and Xu, X. (2004). "Design knowledge modeling and software implementation for
729 building code compliance checking." *Building and Environment*, 39(6), 689-698.
730
731 Yang, Z., Yu, H., Tang, J., and Liu, H. (2019). "Toward keyword extraction in constrained
732 information retrieval in vehicle social network." *IEEE Transactions on Vehicular Technology*,
733 68(5), 4285-4294.
734
735 Yehia, E., Boshnak, E., AbdelGaber, S., Abdo, A., Elzanfaly, D.S. (2019). "Ontology-based
736 clinical information extraction from physician's free-text notes." *Journal of Biomedical*
737 *Informatics*, 98, 103276-103276.
738

739 Zhang, J., and El-Gohary, N. (2015). "Automated information transformation for automated
740 regulatory compliance checking in construction." *Journal of Computing in Civil Engineering*,
741 29(4).
742
743 Zhang, J., and El-Gohary, N. (2016a). "Extending building information models semi-automatically
744 using natural language processing techniques." *Journal of Computing in Civil Engineering, 30*(5).
745
746 Zhang, J., and El-Gohary, N. (2016b). "Semantic NLP-based information extraction from
747 construction regulatory documents for automated compliance checking." *Journal of Computing in*
748 *Civil Engineering*, ASCE, 1943-5487.
749
750 Zhang, J. and El-Gohary, N. (2016c). "Semantic-based logic representation and reasoning for
751 automated regulatory compliance checking." *Journal of Computing in Civil Engineering,* 31(1),
752 4016037.
753
754 Zhang, J., and El-Gohary, N. (2017). "Integrating semantic NLP and logic reasoning into a unified
755 system for fully-automated code checking." *Automation in Construction*, 73, 45-57.
756
757 Zhang, R., and El-Gohary, N. (2019). "A machine-learning approach for semantic matching of
758 building codes and building information models (BIMs) for supporting automated code checking."
759 *Proc. GeoMEast 2019 International Congress and Exhibition*, Cairo, Egypt.
760
761 Zhou, N. (2014). "B-Prolog user's manual (version 8.1): Prolog, agent, and constraint
762 programming." ⟨http://www.picat-lang.org/bprolog/download/manual.pdf⟩ (Apr. 1, 2021).
763
764 Zou, Y., Kiviniemi, A., and Jones, S.W. (2017). "A review of risk management through BIM and
765 BIM-related technologies." *Safety Science*, 97, 88-98.
766

767 **List of Figures**

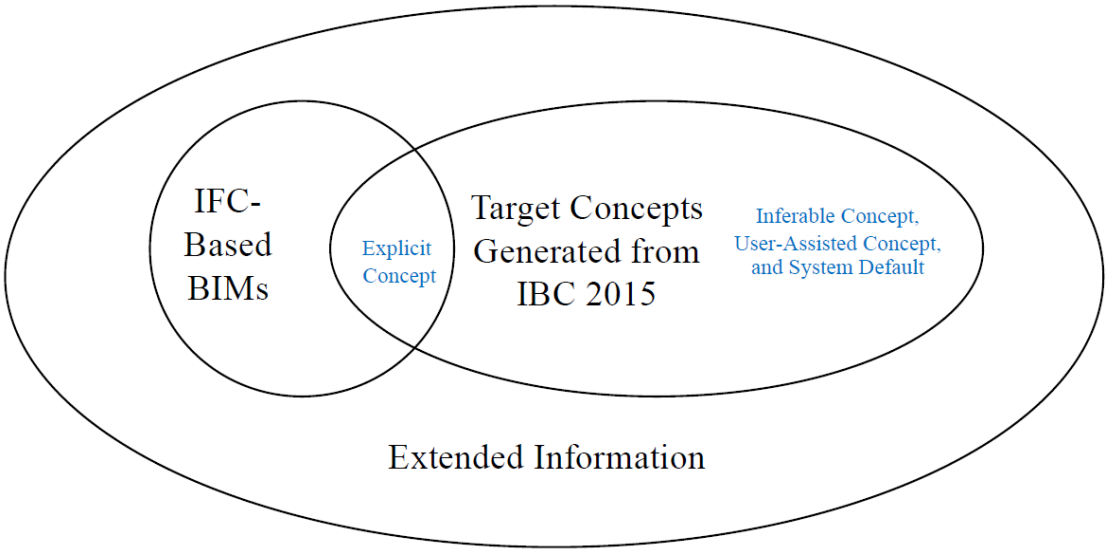


768
769 **Fig. 1.** Idea illustration of the proposed method.



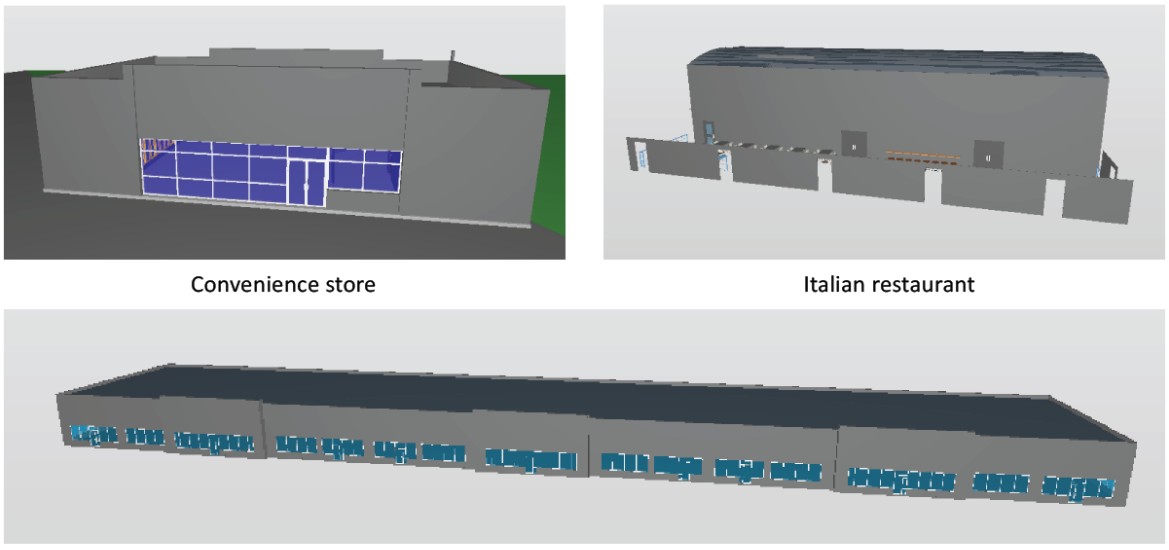
770

771 **Fig. 2.** Workflow of the proposed method.



772

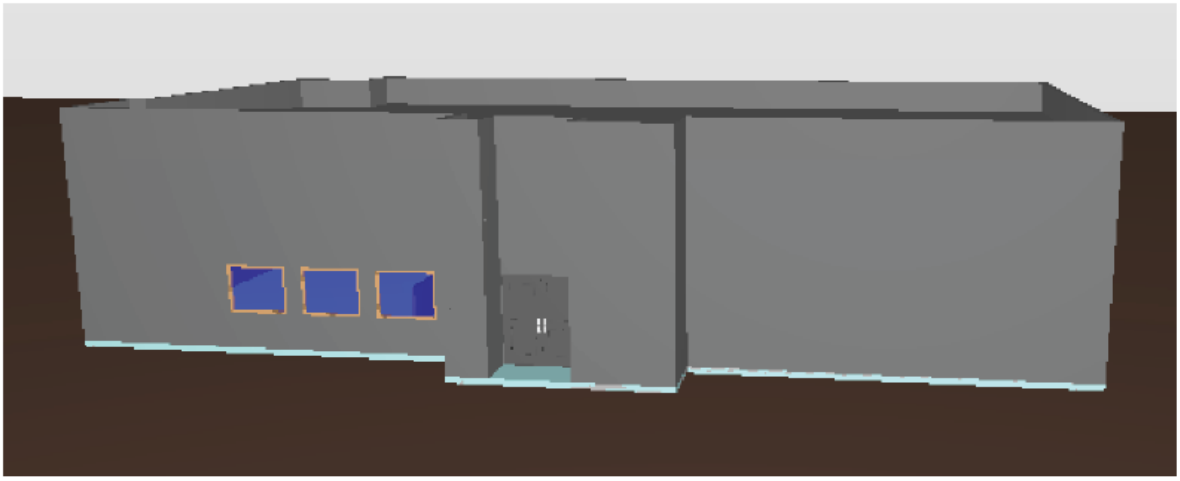
773 **Fig. 3.** Venn Diagram of concepts in building codes and IFC models.



774

775 **Fig. 4.** Visualizations of training project models: the convenience store, the Italian restaurant,

776 and the warehouse.



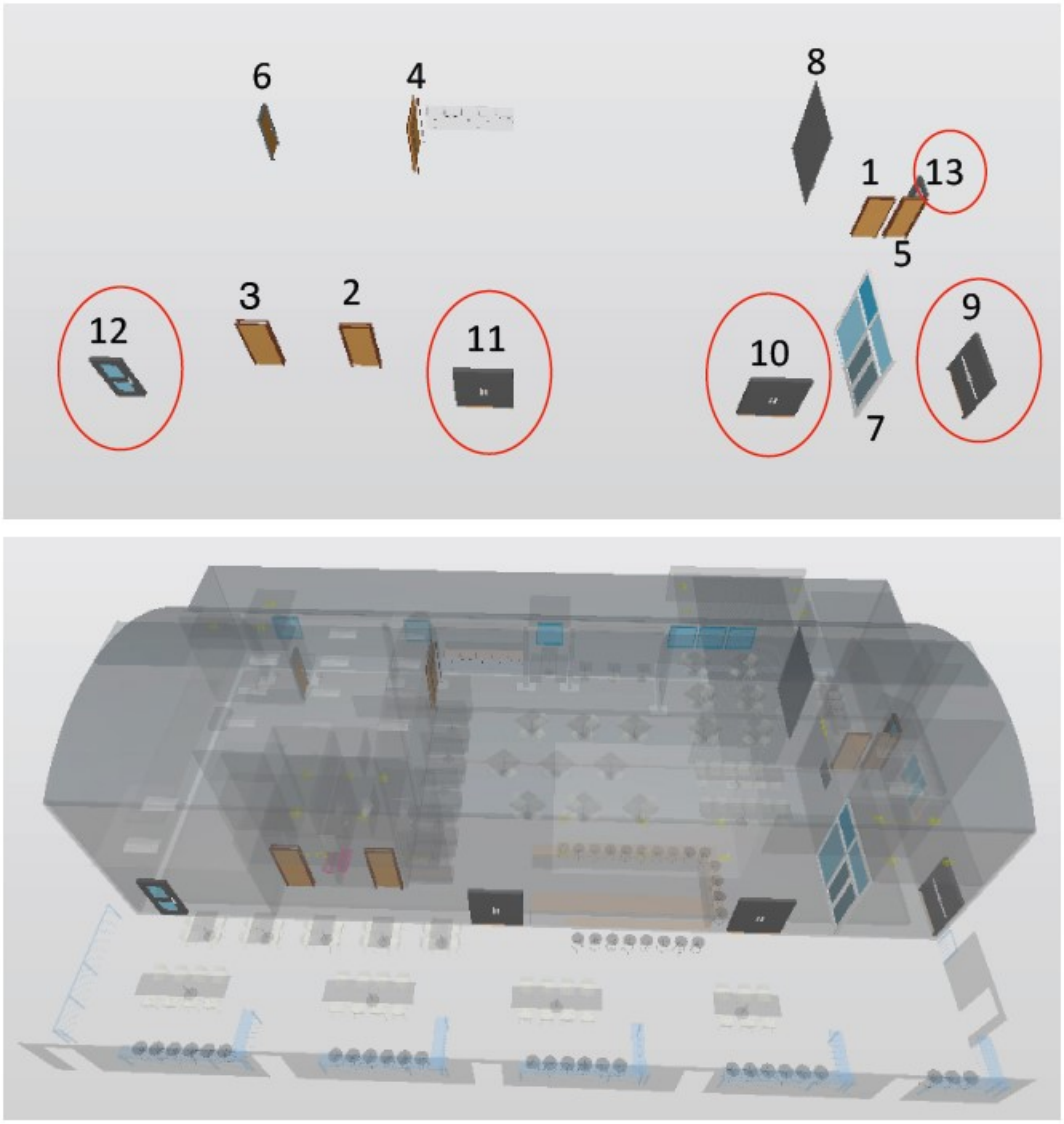
Fast-food restaurant

Hotel

777

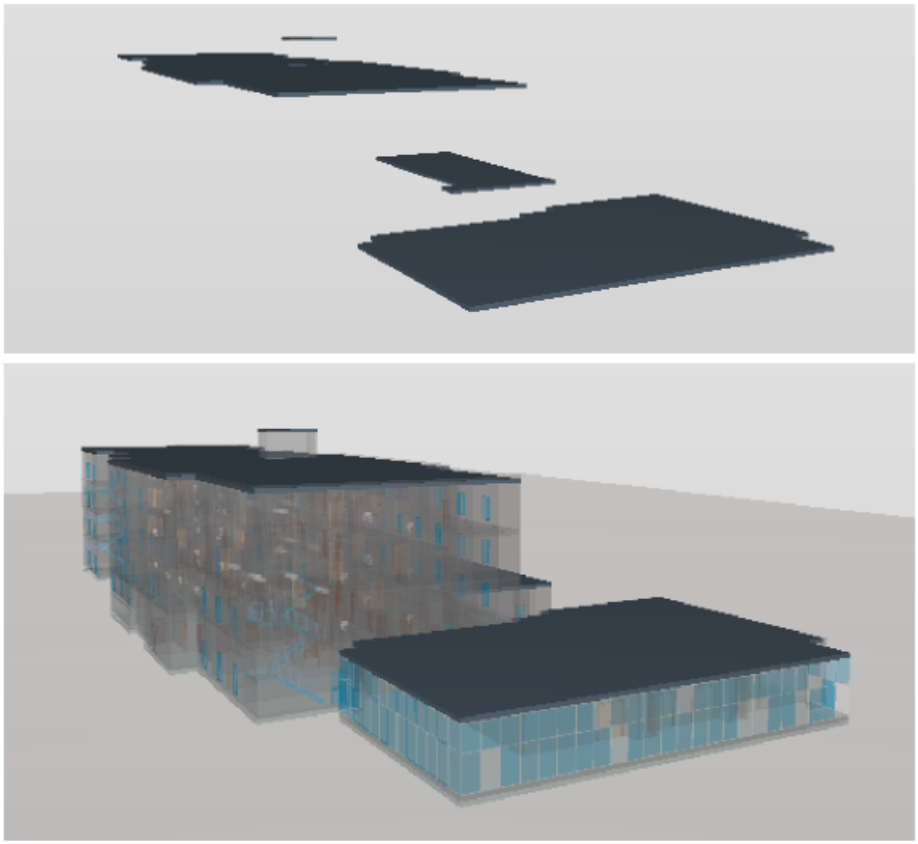
778 **Fig. 5.** Visualizations of testing project models: the fast-food restaurant, the hotel, and the duplex

779 apartment.



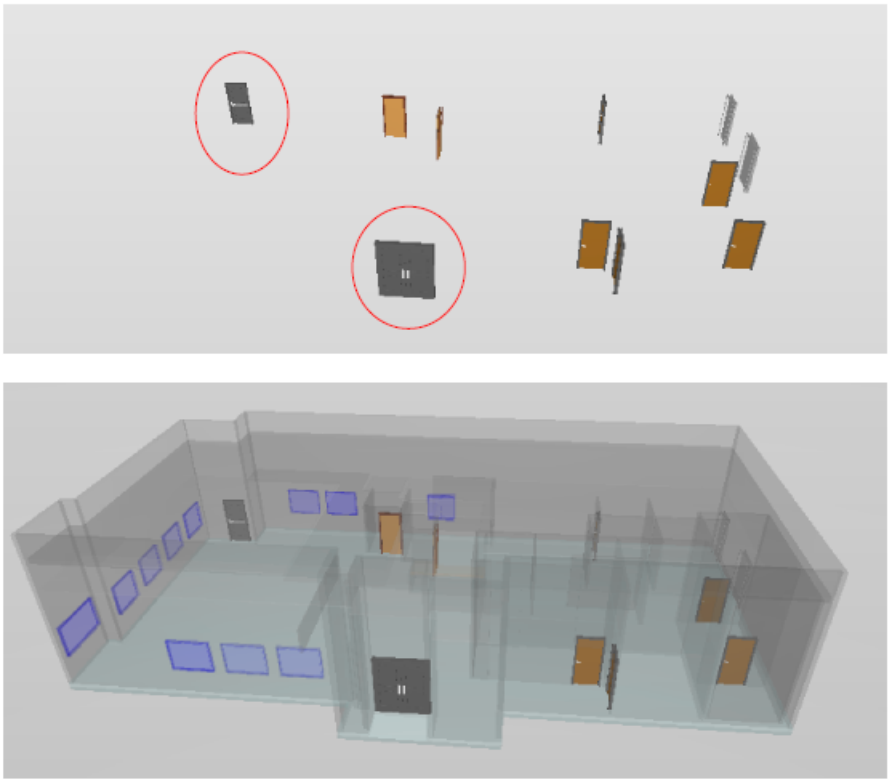
780

781 **Fig. 6.** Visualization of all the egresses identified for Italian restaurant.



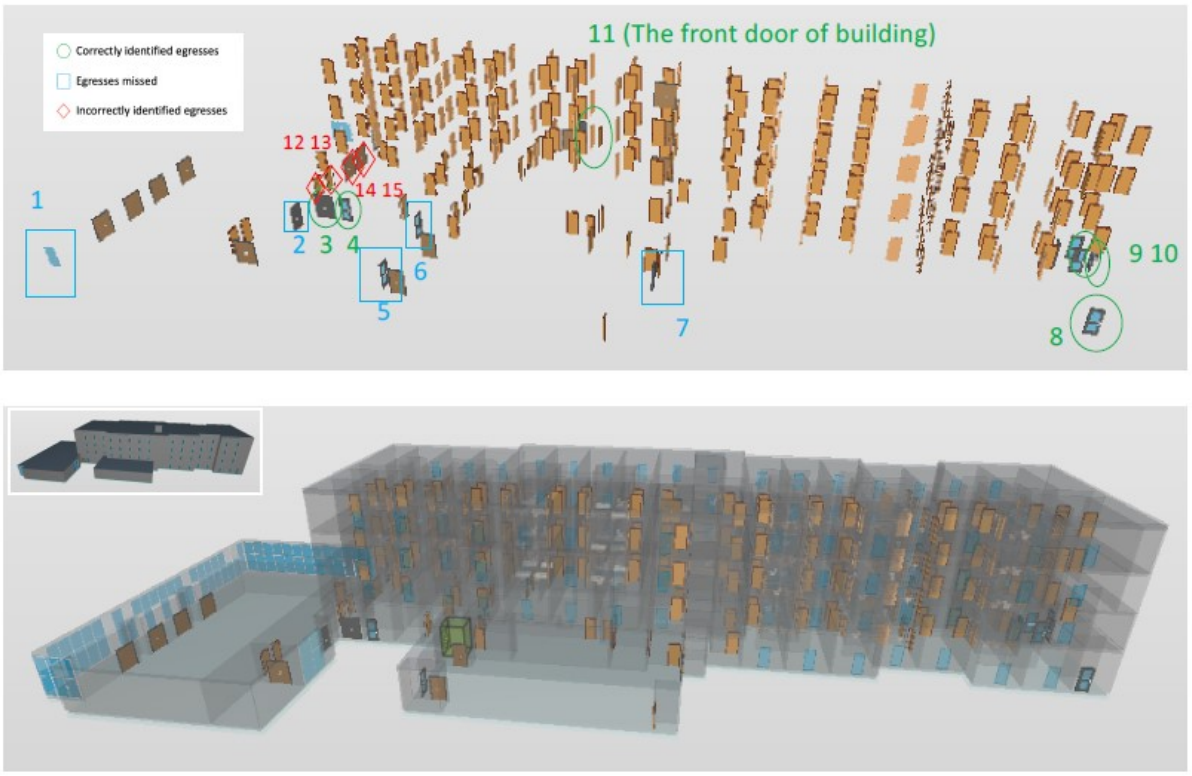
782

783 **Fig. 7.** Visualization of the four roofs identified for the hotel model.



784

785 **Fig. 8.** Visualization of the two egresses identified for the fast-food restaurant model.



786

787 **Fig. 9.** Visualization of all egresses identified for the hotel model (viewed from the backside of

788 the building). Egresses 3, 4, 8, 9, 10, 11 were correctly identified, whereas egresses 1, 2, 5, 6,

789 and 7 were missed, and egresses 12, 13, 14, 15 were incorrectly identified.

790

791

792

793

794

795

List of Tables

Table 1. Properties of invariant signatures-based features.

Signature Type	Example Feature	Feature Count
Geometric	Length, Radius	21
Locational	Position, Orientation Direction	7
Metadata	# of Faces, Average # of Vertices of Faces	8

Table 2. Count of concepts for each type.

Concept Type	Example Concept	Concept Count
Explicit	Wall, Fence, Window	8
Inferable	Egress, Story	11
User-Assisted	Bathroom, Occupied_roof	7
System Default	Equation 20, Message, Group_H	1,382

Table 3. All four types of concepts.

Explicit Concept	Inferable Concept	User-Assisted Concept	System Default
Floor	Building	Stairway_doors	Group_H
Wall	Building_height	Machinery_rooms	Means
Window	Floor_level	Storage_rooms	Barrier
Door	Floor_surface	Lobbies	Message
Room	Egress	Bathroom	1,378 more concepts
Ceiling	Assembly_areas	Exit_discharge_doors	
Fence	Means_of_egress_system	Occupied_roof	
Stair	Exit		
	Mezzanine		
	Room_elevations		
	Story		

Table 4. Explicit concepts identification result on training models.

Concept	IFC Map	No. Identified	No. Correctly Identified	Gold Standard	Precision	Recall
Floor	IfcSlab	24	24	24	100%	100%
Wall	IfcWall/IfcWallStandardCase	99	99	99	100%	100%
Window	IfcWindow	20	20	20	100%	100%
Door	IfcDoor	58	58	58	100%	100%
Room	IfcSpace	27	27	27	100%	100%
Ceiling	IfcRoof	5	5	5	100%	100%
Fence	IfcRailing	12	12	12	100%	100%
Stair	IfcStair	0	0	0	-	-
Total	-	245	245	245	100%	100%

Table 5. Heuristics used in inferable concepts identification algorithms.

Concept	Heuristics Description
Building	Identify the boundary.
Building_height	One of the properties of the building concept.
Floor_level	Using clustering algorithm to identify levels.

Floor_surface	One of the properties of the floor concept.
Egress	Door entity at the boundary that connects interior and exterior.
Assembly_areas	One of the properties of the building concept.
Means_of_egress_system	Same as egress.
Exit	Same as egress.
Mezzanine	Floor objects between floor levels.
Room_elevations	One of the properties of the room concept.
Story	Same as floor level.

Table 6. Inferable concepts identification results on training models.

Concept	No. Identified	No. Correctly Identified	Gold Standard	Precision	Recall
Building	3	3	3	100%	100%
Building_height	3	3	3	100%	100%
Floor_level	3	3	3	100%	100%
Floor_surface	3	3	3	100%	100%
Egress	42	42	42	100%	100%
Assembly_areas	3	3	3	100%	100%
Means_of_egress_system	42	42	42	100%	100%
Exit	42	42	42	100%	100%
Mezzine	0	0	0	-	-
Room_elevations	29	29	29	100%	100%
Story	3	3	3	100%	100%
Total	173	173	173	100%	100%

Table 7. Additional information needed for user-assisted concepts.

Concept	Inferred Concept	Additional Information	Input Type
Machinery_rooms	Room	Room type.	Binary (Y/N)
Storage_rooms	Room	Room type.	Binary
Lobbies	Room	Room type.	Binary
Bathroom	Room	Room type.	Binary
Exit_discharge_doors	Egress	Egress type.	Binary
Occupied_roof	Roof	If the roof is occupied.	Binary
Stairway_doors	Door	Door types.	Binary

Table 8. User-assisted concepts identification result on training models.

Concept	No. Questions	No. Correct Questions	Gold Standard	Precision	Recall
Machinery_rooms	27	27	27	100%	100%
Storage_rooms	27	27	27	100%	100%
Lobbies	27	27	27	100%	100%
Bathroom	27	27	27	100%	100%
Exit_discharge_doors	12	12	12	100%	100%
Occupied_roof	5	5	5	100%	100%
Stairway_doors	58	58	58	100%	100%
Total	183	183	183	100%	100%

Table 9. Number of logic clauses in the generated logic facts on training models.

Concept Type	No. Logic Clauses	No. Correct Logic Clauses	Gold Standard	Precision	Recall
Explicit	3,458	3,458	3,458	100%	100%
Inferable	699	699	699	100%	100%
User-assisted	80	80	80	100%	100%
System defaults	4,146	4,146	4,146	100%	100%
Total	8,333	8,333	8,333	100%	100%

Table 10. Explicit concepts identification results on testing models.

Concept	IFC Map	No. Identified	No. Correctly Identified	Gold Standard	Precision	Recall
Floor	IfcSlab	39	39	39	100%	100%
Wall	IfcWall/IfcWallStandardCase	795	795	795	100%	100%
Window	IfcWindow	145	145	145	100%	100%
Door	IfcDoor	344	344	344	100%	100%
Room	IfcSpace	212	212	212	100%	100%
Ceiling	IfcRoof	6	6	6	100%	100%
Fence	IfcRailing	0	0	0	-	-
Stair	IfcStair	8	8	8	100%	100%
Total	-	1549	1549	1549	100%	100%

Table 11. Inferable concepts identification result on testing models.

Concept	No. Identified	No. Correctly Identified	Gold Standard	Precision	Recall
Building	3	3	5	100%	60.0%
Building_height	3	3	5	100%	60.0%
Floor_level	7	7	7	100%	100%
Floor_surface	39	39	39	100%	100%
Egress	16	12	17	75%	70.6%
Assembly_areas	3	2	3	66.7%	66.7%
Means_of_egress_system	16	12	17	75%	70.6%
Exit	16	12	17	75%	70.6%
Mezzanine	0	0	0	-	-
Room_elevations	212	212	212	100%	100%
Story	7	7	7	100%	100%
Total	322	309	329	96.0%	93.9%

Table 12. User-assisted concepts identification results on testing models.

Concept	No. Questions	No. Correct Questions	Gold Standard	Precision	Recall
Machinery_rooms	212	212	212	100%	100%
Storage_rooms	212	212	212	100%	100%
Lobbies	212	212	212	100%	100%
Bathroom	212	212	212	100%	100%
Exit_discharge_doors	344	344	344	100%	100%
Occupied_roof	6	6	6	100%	100%
Stairway_doors	344	344	344	100%	100%
Total	1542	1542	1542	100%	100%

Table 13. Number of logic clauses in the generated logic facts on testing models.

Concept Type	No. Logic Clauses	No. Correct Logic Clauses	Gold Standard	Precision	Recall
Explicit	22,316	22,316	22,316	100%	100%
Inferable	1,617	1,565	1,667	96.8%	93.9%
User-assisted	84	84	84	100%	100%
System Defaults	4,146	4,146	4,146	100%	100%
Total	28,163	28,111	28,213	99.8%	99.6%