Automated Classification of Building Objects Using Machine Learning

Nadeem Iftikhar[a], Peter Nørkjær Gade, Kasper Møller Nielsen and Jesper Mellergaard
University College of Northern Denmark, Sofiendalsvej 60, Aalborg, Denmark

Keywords: Building Information Modeling, Machine Learning, Building Object Classification, Digital Tools.

Abstract: In the construction sector, digital technologies are being employed to enable architects, engineers and builders in the creation of digital building models. Although these technologies come equipped with inherent classification systems, they also bring forth certain obstacles. Frequently, these systems categorize building elements at levels that exceed their necessary specificity. To illustrate, these classification systems might allocate values at a broader granularity, such as "*exterior wall*" rather than at a more precise level, like "*exterior glass wall with no columns*". As a result, the manual classification of building elements at a granular level becomes essential. Nonetheless, manual classification frequently results in inaccuracies and erroneous semantic details, while also consuming a significant amount of time. Precise and prompt classification of building objects holds significant importance for activities like cost planning, construction cost management and overall procurement processes. To address this, the current paper suggests an automated classification approach for building objects, focusing on specific types, through the utilization of machine learning. The effectiveness of the proposed system is showcased using real-world data from a prominent architectural firm based in Scandinavia.

1 INTRODUCTION

The construction sector is experiencing a digital revolution in response to heightened demands for sustainability, safety and user specifications. This shift towards digital transformation necessitates the adoption of novel procedures to oversee and harmonize digital workflows. In this context, *Building Information Modeling (BIM)*[1] stands out as a renowned instrument employed by architects, engineers and builders to generate, oversee and distribute digital models. These models encompass a substantial volume of building data, comprising both geometric and informational elements. Hence, it is critical to systematically classify this building information in accordance with industry standards, ensuring a streamlined and proficient process. A classification system operates similar to a universal language, facilitating a clear and unambiguous exchange of digital data across diverse BIM software platforms and among various BIM users. The act of classifying building information serves a dual purpose. On one hand, it enhances communication across all stakeholders engaged in construction projects. Simultaneously, it empowers project collaborators (architects, engineers and builders) to effectively align projects with requirements, schedules and financial allocations. While numerous building design tools possess the capability to automatically recognize and classify building objects, either within general categories like "door" or within specific families like "single flush door," it remains essential to classify these objects at the precise type level conforming to national or international standards before their application in a construction project. Nevertheless, the majority of building design tools lack the inherent capacity to automatically classify building objects at the type level or autonomously assign *assembly codes* to them. These assembly codes comply to established national or international classification standards and remain easily modifiable over the course of the project's life cycle. Additionally, the assembly codes play a pivotal role in dictating the organizational structure of construction information, aligning it with the core building objects. Consequently, the assignment of classifications at the assembly code level is often carried out through manual intervention by architects and engineers. For instance, using the *Cuneco Classification System (CCS)*[2], widely used in Denmark, a product like a "window" can be categorized as "*[L]%QQA90102.01*", with the accompanying description indicating its nature as an "*ex-*

[a] https://orcid.org/0000-0003-4872-8546

[1] https://www.autodesk.com/industry/aec/bim

[2] https://ccs.molio.dk/News/About_CCS

Iftikhar, N., Gade, P., Nielsen, K. and Mellergaard, J.
Automated Classification of Building Objects Using Machine Learning.
DOI: 10.5220/0012197500003598
In *Proceedings of the 15th International Joint Conference on Knowledge Discovery, Knowledge Engineering and Knowledge Management (IC3K 2023) - Volume 1: KDIR*, pages 331-338
ISBN: 978-989-758-671-2; ISSN: 2184-3228

terior multidisciplinary window - type 01". Despite the availability of software applications designed to facilitate automated management of CCS codes, the correct codes still necessitate manual input in the majority of classification systems.

The manual categorization of numerous objects at a specific level within a building model introduces complexities, consumes considerable time and introduces potential ambiguities in the classification process. Additionally, a previous study (Flager and Haymaker, 2007) highlighted that over 50% of the time and resources invested by architects and engineers are allocated to the management of design information, which includes object classification. Moreover, errors during the classification process or inaccuracies in semantic details can result in flawed construction cost estimates, incorrect construction practices, erroneous material choices and related issues. Despite the array of challenges associated with BIM object classification, including the search for relevant international or national BIM standards, establishment and storage of tailored classification tables, collaborative distribution of classification tables among project teams, navigation through multiple classification systems, time-intensive processes and absence of automation. The acceptable automation of BIM object classification at the precise type level remains an open research concern within the construction sector. This paper attempts to tackle this issue by introducing an approach that employs supervised machine learning to automate the classification of objects at the type or assembly code level.

In summary, this paper's key contributions can be outlined as follows:

- Offering comprehensive insights through exploratory data analysis of building information;
- Introducing a comprehensive solution for classifying building elements through the utilization of diverse supervised machine learning approaches;
- Demonstrating practical implementation using actual building data extracted from a prominent construction project executed by a leading Scandinavian architectural firm.

The paper's organization is as follows: In Section 2, a survey of related work is provided. The motivation driving this work is elaborated in Section 3. The exploratory data analysis is detailed in Section 4. The machine learning approach for automated building object classification is presented in Section 5. Section 6 outlines the experimental findings. The paper concludes in Section 7, also highlighting potential directions for future research.

2 RELATED WORK

The focal point of this section centers on prior research attempts related to automated building object classification through the application of semantic enrichment. A state-of-the-art review by (Zabin et al., 2022) indicated that there is a need to integrate machine learning into BIM processes. Similarly a study by (Amor and Dimyadi, 2021) examined evolving approaches for automated compliance checking and pointed out that research in semantic enrichment of BIM models is necessary. In addition, (Zhang and El-Gohary, 2012) suggested a natural language processing approach for automated information extraction from construction regulatory documents. Similarly, an automated compliance checking approach is presented by (Salama and El-Gohary, 2011). A prototype software based on an inference rule engine to semantically enrich the building models is introduced by (Belsky et al., 2016) and its extension is suggested by (Sacks et al., 2021). Further, (Bloch and Sacks, 2019) applied both supervised machine learning and rule-inferencing to correctly classify rooms types in residential apartments. The machine learning approach provided better accuracy than rule-based approach. A data-driven iterative method for automated classification of objects in BIM is presented by (Wu and Zhang, 2019). Moreover, a deep learning framework to utilize both geometric and relational information of BIM objects for classification is proposed by (Luo et al., 2022). Likewise, a deep learning framework for classifying objects based on synthetic data sets created from BIM objects is presented by (Frías et al., 2022). In addition, (Kim et al., 2019) proposed an approach for automating the classification of building element instances within BIM. The approach utilized deep learning based classification technique that uses images of objects as inputs. A machine learning based solution to automatically recognise elements in buildings information models is proposed by (Bassier et al., 2017). The solution can efficiently classify the basic components such as floors, ceilings, roofs, walls and so on. Additionally, (Emunds et al., 2022) proposed a neural network model based on sparse convolutions for the classification of IFC-based geometry and semantic enrichment of BIM models. Concludingly, the study conducted by (Koo et al., 2022) examined the viability of deep and machine learning models in the context of automatically classifying subtypes of door and wall elements.

These previous studies emphasize on various conceptual aspects and recent advancements in semantic enrichment and automated classification in construction industry. It can be concluded from these previous works that the absence of automation in the classification of building objects at specific type level remains an open research issue. To our best understanding, this paper stands among a limited number that concentrate on automated building object classification through machine learning. Additionally, this paper also considers the practical aspects of automated classification to enhance operational effectiveness and efficiency within the construction industry.

3 MOTIVATION

Given that the construction industry employs BIM to create digital presentation of buildings. Hundreds, or even thousands, of objects are incorporated into a model before its completion. In the majority of instances, these objects are required to be manually classified. As previously noted, manual classification within the domain of BIM poses a significant challenge.

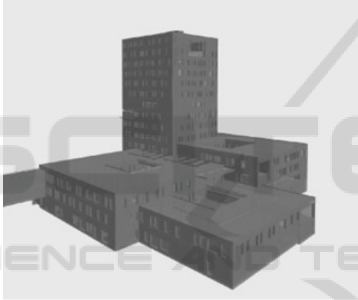

Figure 1: BIM complete 3D model.

In this paper, the focus is on automatic classification of building objects based on their assembly codes. The assembly codes used in this paper are found in *BIM7AA standard*[3], where BIM7AA is a simple encoding method for BIM objects based on Danish building standards. For instance, a typical "*exterior precast concrete wall*" has assembly code 211, and "*interior frame structured wall*" has assembly code 224, and so forth. While building a BIM model, an architect/engineer inserts a standard "*wall*" and edits it into the desired detailed specifications; these often change in the duration of a project. Therefore, the correct assembly code based on Danish building standards is hard to predict. Hence, the architects/engineers have to point-and-click their way through every object in the BIM and classify their assembly codes. Classifying hundreds or thousands of objects manually is time-consuming and erroneous. In addition, ambiguous classification results in additional and/or unwanted expenses.

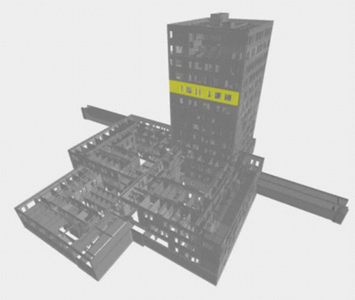

Figure 2: BIM 3D model (walls only).

To provide an illustration, Fig. 1 portrays a comprehensive 3D BIM model featuring walls, roofs, spaces, coverings, stairs and columns. Conversely, Fig. 2 exclusively displays the walls present within the model. The data set encompassing all wall elements within the model comprises a total of 4025 walls, where each of these walls has over 160 features. The features contains a wide range of data types including numerical, alphanumerical, unique identifiers, binary indicators and categorical attributes.

Table 1: Selected set of wall features.

Wall feature	Value	Description
Area	92.09	length * height
Base Constraint	PLAN 10	floor level
Length	26489.99	wall's length
Structural	1	yes 1/no 0
Structural Usage	1	non load-bearing 0/load-bearing 1
Type Id	2147048	wall type
Unconnected Height	4462.99	wall's element height
Volume	49.39	walls's volume
Width	540	wall's width
Assembly code	**211**	exterior precast concrete wall

For the purpose of illustration, only a subset of 10 features out of 160 (for the outer wall distinguished by its highlighted color in Fig. 2) has been chosen. A snapshot of feature data is presented in Table 1. The final row in Table 1, displays the assembly code, which is intended to be automatically assigned or predicted.

4 EXPLORATORY DATA ANALYSIS

In this section, an exploratory data analysis of the classification problem has been carried out. The primary objective of this analysis is to investigate the data set and pinpoint the features that wield an impact on the assembly code. To address the extensive array of features (totaling 160 in this instance), *Prin-*

[3] http://bim7aa.dk/index_UK.html

cipal Component Analysis (PCA)[4] has been employed to reduce their dimensionality. To evaluate the significance of the chosen features, one method involves creating a heatmap that visualizes their correlations. By depicting the correlations between these features, a heatmap offers a graphical depiction of their interrelationships, facilitating the evaluation of their importance. The correlation coefficient is measured on a scale that spans from +1 to 0 to -1. A correlation of -1 indicates a robust negative correlation between two values, while a correlation of +1 indicates strong positive correlation, and a value of 0 implies no correlation between them.

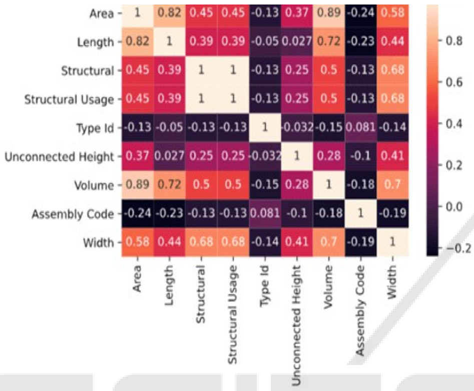

Figure 3: Correlation heatmap.

The correlation heatmap is depicted in Fig. 3. Upon analyzing the heatmap, it becomes evident that *Structural* and *Structural Usage* exhibit positive correlation, suggesting that one of them can safely be removed. The remaining attributes, except for *Type Id*, display cross-correlation and do not raise any concerns regarding their relevance. Another feature to look at is *Type Id* in relation to the *Assembly Code* (the target variable). Both of them display somewhat correlations with the other attributes in a similar fashion, indicating a degree of similarity in behavior. This similarity raises the possibility of data leakage, thus motivating the removal of *Type Id*. Data leakage occurs when the model is trained using an attribute that describes the target variable in some manner. In such cases, the model is prone to making overly optimistic predictions. Furthermore, during exploratory data analysis, it is noted that thirteen different classifications are distributed across 4025 wall instances, as illustrated in Fig. 4.

The depicted figure illustrates the representation of different variations of *Assembly Codes* within the data set. The Y-axis corresponds to the count of rows, while the X-axis corresponds to the various possible classifications. Among these classifications, namely 224, 221, 225, 211 and 226, hold dominance with a combined representation of 3758 instances. Conversely, the remaining eight classifications are notably non-existent, accounting for only 267 examples. This evident imbalance in the data set raises concerns. Imbalanced data set can result in models that exhibit substantial bias, making it extremely difficult or even unattainable for the model to accurately classify the underrepresented types. To address this issue stemming from imbalanced data, over-sampling has been implemented. Over-sampling involves generating multiple additional entries for the minority classes, thereby achieving a balanced representation. An effective approach for this is the the Synthetic Minority Oversampling Technique (SMOTE) (Chawla et al., 2002). Through the application of the SMOTE algorithm, the data set has undergone a substantial expansion, growing from its original 4025 rows to approximately 25000 rows.

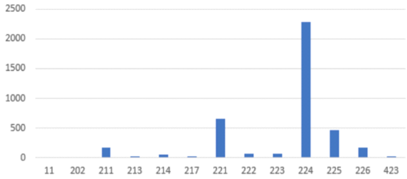

Figure 4: Unequal distribution of assembly codes in the training data set.

5 MACHINE LEARNING APPROACH

In this section, the automated building object classification system utilizing machine learning is introduced. It features a web application for user interaction and an API for data and model management. Supervised learning algorithms and preprocessing techniques are used to train the classifier. The system's performance is evaluated using metrics, and the best model is then deployed for label predictions on new data.

5.1 Problem Formulation

In the realm of construction, a variety of building objects are often encountered, denoted as $O = \{o_1, o_2, \ldots, o_n\}$. Each object o_i is distinguished by a set of features $F_i = \{f_1, f_2, \ldots, f_m\}$ and is assigned an assembly code C_i based on a specific classification standard. The main goal is to devise a classifier $\phi : O \rightarrow C$ that assigns each building object o_i to an

[4] https://www.turing.com/kb/guide-to-principal-component-analysis

assembly code C_i, with the aim of minimizing the prediction error $E = \sum_{i=1}^{n} L(C_i, \phi(o_i))$. Here, $L(C_i, \phi(o_i))$ represents a loss function that quantifies the difference between the actual assembly code C_i and the predicted assembly code $\phi(o_i)$.

5.2 Methodology

To achieve this, supervised machine learning algorithms are proposed for training the classifier ϕ using a labeled data set $D = \{(o_1, C_1), (o_2, C_2), \ldots, (o_n, C_n)\}$. The performance of this classifier is then evaluated using metrics such as accuracy, balanced accuracy and F1-score. To further enhance the performance of the model, PCA is proposed for dimensionality reduction and SMOTE for addressing data imbalance.

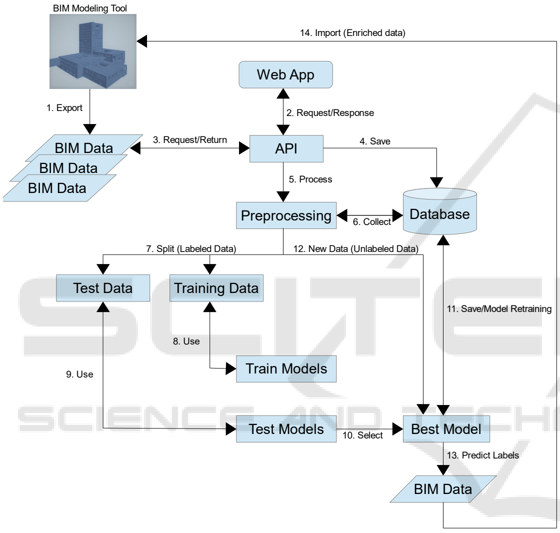

Figure 5: Process flow.

5.3 Process Flow

The process flow of the proposed solution is detailed in Algorithm 1 and visualized in Fig. 5. The solution employs multi-classification algorithms for accurate predictions. Prior to implementation, BIM data must be manually extracted from a BIM software tool such as, *Autodesk Revit*[5] or *Speckel*[6]. Post prediction, the enriched BIM data is manually imported back into the BIM modeling tool. The solution comprises two main components: developing a classifier using labeled data and utilizing this classifier for predicting assembly codes. The classification process, depicted in Fig. 5, unfolds through the following subsections.

```
Result: Enriched CSV file $(O', C_{pred}, P)$
Step 1: Export BIM data $D$ as CSV file(s)
  containing objects $O$ (with features $F$) and labels
  $C$;
Step 2: Load CSV file(s) into web application and
  select features $F$ and target $C$;
Step 3: Preprocess $(F)$ to get preprocessed
  features $(\tilde{F})$;
Step 4: Train classifier $\phi$ using selected machine
  learning algorithms on training subset
  $(\tilde{F}_{train}, C_{train})$ and evaluate performance using
  metrics $L$, where $L = \text{evaluate}(\phi, \tilde{F}_{test}, C_{test})$;
Step 5: Predict labels $C_{pred}$ and confidence scores
  $P$ for new data using best model:
  $(C_{pred}, P) = \phi_{best}(O')$;
```

Algorithm 1: Automated building object classification.

5.3.1 Export BIM Data

1. CSV file(s) are manually exported from the BIM modeling tool(s), containing the required data (D). The CSV file(s) should include the features and labels of the building objects that need to be classified according to a specific classification standard. The features can be numerical, alphanumerical, unique identifiers, binary indicators, or categorical attributes. The labels can be assembly codes or other types of classifications that follow a standard encoding method.

5.3.2 Web Application Usage

2. The web application (W), a user-friendly interface, allows users to interact with the system by selecting features and requesting predictions. It communicates with the API, a software intermediary that connects the web application with the database and the machine learning models.
3. The web application provides a “Find Features” button that prompts the presentation of all potential features extracted from the data. Users can choose the features (F_i) that are relevant for the classification task and the target (C_i) variable that represents the desired output. For example, “Area”, “Length”, “Width”, “Structural Usage”, etc. can be selected as features and “Assembly Code” as target.
4. Data (D) containing the chosen features and target is stored in a database, which is a structured collection of data that can be accessed and manipulated by the system. The database ensures that the data is organized and secure.
   – A classifier (ϕ) is created using selected machine learning algorithms (Random Forest (RF), Gradient Boosting (GB), and K-Nearest

[5] https://www.autodesk.com/products/revit/
[6] https://www.speckel.io/

Neighbors (KNN)). These algorithms are able to learn from labeled data and make predictions for new data. The subsequent preprocessing phase is initiated when the user activates the "Create Classifier" button.

5.3.3 Preprocessing Data

5. The preprocessing phase is automated, eliminating the need for manual involvement. This phase retrieves the data (D) containing the previously chosen features and target from the database.
   - Data (D) is cleaned by handling missing or duplicate values, outliers and inconsistencies. During this phase appropriate methods are also applied to handle any errors or anomalies in the data. For example, missing or duplicate values can be removed or imputed, outliers can be detected or treated and inconsistencies can be resolved or corrected.
   - Data (D) is scaled or normalized using appropriate methods to transform the values of the features into a common range. This helps to reduce the effect of different units or magnitudes on the performance of the machine learning models. For example, scaling or normalization methods can include min-max scaling, standardization or log transformation.
   - Categorical data (D) is encoded using appropriate methods to convert categorical features into numerical values that can be used by the machine learning models. Categorical features are those that have a finite number of possible values that represent categories or classes. For example, encoding methods can include label encoding, one-hot encoding or ordinal encoding.
   - Dimensionality of data (D) is reduced using PCA ($PCA(D)$), which is a technique that transforms a large number of correlated features into a smaller number of uncorrelated components that capture most of the variance in the data. This helps to reduce noise and redundancy in the data and improve computational efficiency and performance of the machine learning models.
   - Imbalance of data (D) is addressed using SMOTE ($SMOTE(D)$), which is a technique that generates synthetic samples for the minority classes in the data set to balance their representation with the majority classes. This helps to reduce bias and improve accuracy and generalization of the machine learning models.
6. Processed data (D) is reinserted into the database for further use by the system.

5.3.4 Training and Testing Models

7. The system splits data (D) into training and testing subsets (D_{train} and D_{test}). The training subset is used to train the classifier using the selected machine learning algorithms. The testing subset is used to test the classifier and evaluate its performance using evaluation metrics.
8. The classifier (ϕ) is trained using selected machine learning algorithms (RF, GB, and KNN) and the training subset (D_{train}). These algorithms learn from labeled data and make predictions for new data. The training subset is used to fit the parameters of the algorithms and optimize the classifier.
9. The classifier (ϕ) is tested using the testing subset (D_{test}) and performance is evaluated using metrics (accuracy, balanced accuracy, and F1-score). These metrics measure how well the classifier can predict the correct labels for new data.
10. The best model is selected based on highest metric scores among the three algorithms. The best model is the one that can achieve the highest accuracy, balanced accuracy and F1-score on the testing subset.
11. The best model is saved in the database for future use and potential retraining.

5.3.5 Label Prediction

12. New unlabeled data (D') is loaded from a BIM modeling tool. The "Predict Labels" button can be used to utilize the classifier for predicting labels for new data. The new data is a set of building objects that need to be classified according to a specific classification standard. The new data is loaded from a BIM modeling tool as a CSV file.
    - The new data (D') is preprocessed using the same methods as before. The system preprocesses the new data ensuring that it is compatible with the classifier and has the same format and structure as the training and testing data.
    - The best model is loaded from the database. This model was selected based on the highest metric scores in the previous step.
13. Labels (C') for new data (D') are predicted using the best model. The labels are assembly codes or other types of classifications that follow a specific classification standard. Confidence scores for each prediction are also generated, indicating how confident the model is about its prediction. The enhanced data set is showcased in Table 2.

Table 2: Snapshot of the predicted assembly codes with probabilities.

Area	Base constraint	Length	Structural Usage	Unconnected Height	Volume	Width	**Assembly Code**	**Probability**
..	..	..	..	..	..	..	**..**	**..**
19.6495	PLAN 03	4975	0	4533	2.7954	145	**224**	**1**
1.4679	PLAN 12	1472.4999	0	1110	0.2128	145	**225**	**1**
4.6728	PLAN 11	2010	0	2360	0.3738	80	**221**	**1**
11.9191	PLAN 01	2992.5000	1	4533	3.5757	300	**221**	**0.8466**
0.4799	ANIMAL STABLE	3199.9999	0	150	0.036	75	**223**	**1**
2.4478	PLAN 03	270	1	4533	1.3212	540	**221**	**0.7125**
13.7785	PLAN 01	3600	0	4538	1.9978	145	**224**	**1**
3.81	PLAN 02	2050	0	2000	0.3619	95	**225**	**0.6966**
13.8608	PLAN 03	3552	0	4533	2.0098	145	**224**	**1**
11.6124	PLAN 05	2727	0	4333	1.5386	132	**224**	**1**
22.8693	ROOF	6564.9997	0	3421	5.4886	240	**217**	**0.6333**
5.015	PLAN 12	2055	0	2360	0.3009	60	**221**	**1**
2.3101	PLAN 04	3650	1	650	0.4435	192	**214**	**0.98**
23.3638	PLAN 03	7115.2569	0	4533	3.3188	145	**212**	**1**
..	..	..	..	..	..	..	**..**	**..**

– An enriched CSV file containing new data (D'), predicted labels (C') and confidence scores (P) is generated. This file can be used to update or enrich the BIM model with accurate and consistent classifications.

14. The enriched CSV file is manually imported into a BIM modeling tool to complete the classification process.

By adhering to this process flow, the suggested solution facilitates the establishment and utilization of a classifier, allowing automated prediction of assembly codes within building objects.

6 EXPERIMENTS

Within this section, an evaluation of the models employed to classify building objects is conducted. The experiments are carried out using real-world data derived from a building project.

6.1 Setup

For conducting multi-class classification, three supervised machine learning algorithms (Random Forest (RF), Gradient Boosting (GB) and K-Nearest Neighbors (KNN)) were employed for experimentation. RF, GB and KNN were chosen for their unique strengths in handling building object classification. RF's robustness to overfitting and ability to handle high-dimensional data make it suitable for large data sets. GB is known for its high accuracy and provides feature importance, crucial for understanding influential characteristics in building object classification. KNN offers a simple yet effective approach, making no assumptions about the data distribution, which is beneficial when classifying objects with similar features. Collectively, these algorithms provide a comprehensive and robust approach to the classification task.

The experiments were conducted on the extended data set comprises 25000 wall objects, each with a list of over 160 features. The algorithms have run on a single-node hardware platform with a 8th Generation Intel Core i7-8565U 1.8 GHz processor, 32GB DDR4 RAM and 1TB SSD. The reported outcomes were derived from running each algorithm 20 times, and the results were averaged over the best 5 executions.

6.2 Test Results

Table 3 displays the classification model test results. It is emphasized that accuracy may not be reliable for imbalanced data sets, hence balanced accuracy and F1-score are also considered. Accuracy is the ratio of correct predictions to the total predictions made. Balanced accuracy represents average recall per class, while F1-score is the harmonic mean of precision and recall, with all having an optimal value of 1 and a minimum value of 0.

Table 3: Evaluation metrics for classification models.

Model	Accuracy	Balanced accuracy	F1-score
RF	0.93	0.87	0.94
GB	0.90	0.83	0.91
KNN	0.82	0.71	0.81

Based on the evaluation metrics presented, the RF model stands out as the most suitable choice, demonstrating generally high scores (above 85%), which are deemed acceptable in building object classification contexts.

7 CONCLUSIONS AND FUTURE WORKS

In the construction sector, seamless collaboration between stakeholders, such as architects, engineers, and builders, is essential to avoid miscommunications arising from different terminologies. Classification systems address this by offering a standardized language throughout the project life-cycle, from conception to maintenance. This process involves assigning unique codes to each object within a BIM model, thus facilitating accurate quantity evaluations, cost estimations and comprehensive project planning. In this paper, an automated approach for classifying building objects at a specific type level has been presented, utilizing machine learning algorithms such as Random Forest, Gradient Boosting, and K-Nearest Neighbors. The effectiveness of this classification technique was verified with a real-world data set, showing encouraging results. The proposed system, although promising, has limitations including data quality dependency and possible inaccuracies due to algorithm assumptions. Its scalability and adaptability to other projects or classification schemes are yet to be confirmed, with its current evaluation limited to a specific project.

Future research should focus on improving data quality and feature selection, experimenting with various machine learning algorithms, optimizing system scalability and conducting assessments across a range of projects and classification schemes.

ACKNOWLEDGEMENTS

We sincerely thank Jan Buthke and the team at LINK Arkitektur for their significant contribution to this paper, providing real data set that greatly enriched the quality and relevance of our research.

REFERENCES

Amor, R. and Dimyadi, J. (2021). The promise of automated compliance checking. *Developments in the built environment*, 5:100039.

Bassier, M., Vergauwen, N., and Genechten, B. V. (2017). Automated classification of heritage buildings for asbuilt bim using machine learning techniques. In *ISPRS Annals of the photogrammetry, remote sensing and spatial information sciences*, pages 25–30.

Belsky, M., Sacks, R., and Brilakis, I. (2016). Semantic enrichment for building information modeling. *Computer-Aided Civil and Infrastructure Engineering*, 31(4):261–274.

Bloch, T. and Sacks, R. (2019). Comparing machine learning and rule-based inferencing for semantic enrichment of bim models. *Automation in Construction*, 91:256–272.

Chawla, N. V., Bowyer, K. W., Hall, L. O., and Kegelmeyer, W. P. (2002). Smote: synthetic minority oversampling technique. *Journal of artificial intelligence research*, 16:321–357.

Emunds, C., Pauen, N., Richter, V., Frisch, J., and van Treeck, C. (2022). Sparse-bim: classification of ifc-based geometry via sparse convolutional neural networks. *Advanced Engineering Informatics*, 53:101641.

Flager, F. and Haymaker, J. (2007). A comparison of multidisciplinary design, analysis and optimization processes in the building construction and aerospace industries. In *24th International Conference on Information Technology in Construction*, pages 625–630.

Frías, E., Pinto, J., Sousa, R., Lorenzo, H., and Díaz-Vilariño, L. (2022). Exploiting bim objects for synthetic data generation toward indoor point cloud classification using deep learning. *Journal of Computing in Civil Engineering*, 36(6):04022032.

Kim, J., Song, J., and Lee, J. K. (2019). Recognizing and classifying unknown object in bim using 2d cnn. In *18th International Conference on Computer-Aided Architectural Design Futures*, pages 47–57. Springer.

Koo, B., Jung, R., , and Yu, Y. (2022). Automatic classification of wall and door bim element subtypes using 3d geometric deep neural networks. *Advanced Engineering Informatics*, 47:101200.

Luo, H., Gao, G., Huang, H., Ke, Z., Peng, C., and Gu, M. (2022). A geometric-relational deep learning framework for bim object classification. In *Computer Vision–ECCV 2022 Workshops*, pages 349–365. Springer.

Sacks, R., Ma, L., Yosef, R., Borrmann, A., Daum, S., and Kattel, U. (2021). Semantic enrichment for building information modeling: Procedure for compiling inference rules and operators for complex geometry. *Journal of Computing in Civil Engineering*, 31(6):04017062.

Salama, D. M. and El-Gohary, N. M. (2011). Semantic modeling for automated compliance checking. In *International Conference on Computing in Civil Engineering*, pages 641–648. ASCE Library.

Wu, J. and Zhang, J. (2019). New automated bim object classification method to support bim interoperability. *Journal of Computing in Civil Engineering*, 33(5):04019033.

Zabin, A., González, V. A., Zou, Y., and Amor, R. (2022). Applications of machine learning to bim: A systematic literature review. *Advanced Engineering Informatics*, 51:101474.

Zhang, J. and El-Gohary, N. (2012). Extraction of construction regulatory requirements from textual documents using natural language processing techniques. In *International Conference on Computing in Civil Engineering*, pages 453–460. ASCE Library.