applied sciences

MDPI

Article

Filtering of Irrelevant Clashes Detected by BIM Software Using a Hybrid Method of Rule-Based Reasoning and Supervised Machine Learning

Will Y. Lin [1,*] and Ying-Hua Huang [2]

1 Department of Civil Engineering, Feng Chia University, Taichung 407, Taiwan
2 Department of Civil and Construction Engineering, National Yunlin University of Science and Technology, Yunlin 640, Taiwan; huangyh@yuntech.edu.tw
* Correspondence: weiylin@mail.fcu.edu.tw; Tel.: +886-973-531-289

Received: 9 October 2019; Accepted: 2 December 2019; Published: 6 December 2019

Abstract: Construction projects are usually designed by different professional teams, where design clashes may inevitably occur. With the clash detection tools provided by Building Information Modeling (BIM) software, these clashes can be discovered at an early stage. However, the number of clashes detected by BIM software is often huge. The literature states that the majority of those clashes are found to be irrelevant, i.e., harmless to the building and its construction. How to filter out these irrelevant clashes from the detection report is one of the issues to be resolved urgently in the construction industry. This study develops a method that automatically screens for irrelevant clashes by combining the two techniques of rule-based reasoning and supervised machine learning. First, we acquire experts' knowledge through interviews to compile rules for the preliminary classification of clash types. Subsequently, the results of the initial classification inferred by the rules are added into the training dataset to improve the predictive performance of the classifiers implemented by supervised machine learning. The average predictive performance obtained by using the hybrid method is up to 0.96, which has been improved from the traditional machine learning process only using individual or ensemble learning classifiers by 6%–17%.

Keywords: clash detection; supervised machine learning; building information modeling (BIM)

1. Introduction

Design conflicts refer to the errors in which building components overlap with each other spatially when compiling various types of engineering drawings. Since engineering drawings are generally formed after compiling the designs by engineers of different professions, design conflicts between different system components are often common [1,2]. Minor design conflicts often result in rework and increase the project costs. In severe cases, design changes may be required, resulting in cost overruns, delay in progress, and compromising the structural safety. As pointed out by previous studies, unresolved design conflicts will hugely impact on the project success [3].

In recent years, the emergence of BIM software has enabled the easy detection of design conflicts; conflict checking has become one of the important functions of BIM software. Since resolving design conflicts is critical to the success of a project, many countries have mandated all public projects to execute clash detection. In the United Kingdom, for example, the design team must perform clash detection once every week or every two weeks to ensure that the engineering design receives complete coordination and is free of conflicts, thereby reducing the probability of change orders [4]. However, the clash detection algorithms of most BIM software are simple; as long as two building components are spatially overlapping, in contact, or within a given distance, they will be identified as a conflict/clash and listed in the detection report. Therefore, even for small projects, the number of clashes detected using BIM software can be enormous [5–7]. As many studies discovered, 50% or more of the clashes detected from BIM software are found to be "irrelevant clashes"; that is, these conflicts will not have a substantial impact on the projects, or they can be directly resolved by site engineers during the construction phase [7]. However, the clash detection report of BIM software does not identify these irrelevant clashes. Ideally, every single clash in a detection report should be evaluated by engineers to decide whether the resolution is needed. This is an extremely time-consuming job. According to our interviews with senior project managers, many projects in Taiwan cannot afford the high incurred costs. Thus, their BIM managers merely selectively review a few clashes or even neglect the entire report. It is why many studies have pointed out that unless filtering those irrelevant clashes is automated, the clash detection report with an overwhelming number of clashes will become trivial and meaningless [6–8].

Scholars have proposed methods to resolve this issue from three aspects, namely: clash avoidance, clash detection improvement, and clash filtering [6,7]. Clash avoidance begins with the modeling method with the emphasis on adopting collaboration or strengthened coordination to reduce the occurrence of clashes. Undoubtedly, this method will increase the burden on the design staff [7]. For those project participants without direct contractual relationships, collaboration is also difficult to implement [6,9]. By contrast, other scholars consider that the algorithm to detect clashes in BIM software can be improved by increasing the accuracy of its detection, thereby reducing the number of irrelevant clashes [5]. However, as some studies pointed out, the refinement of the clash detection algorithm cannot effectively prevent irrelevant clashes caused by human errors from happening [7,10]. Recent studies suggest that an alternative is to directly identify those irrelevant clashes and filter them out from the clash detection report generated by BIM software. This method is broadly divided into two approaches. One is to apply rules to identify the dependency relationship of conflicting/clashing components, thereby filtering out irrelevant clashes [7]. However, the constructing dependency relationships between components and query algorithms is often time-consuming and labor-intensive when acquiring and maintaining the rules. Jiang et al. proposed a rule-based knowledge system to automate the code-checking process for green construction [11]. They found that the domain knowledge is usually dispersed and fragmented, so rule acquisition requires human experts from many professional fields. Therefore, the process of knowledge representation and acquisition is often a complex and time-consuming task [12]. The other is the use of machine learning methods that train classifiers of machine learning through historical data to filter out irrelevant clashes [6]. However, researchers using machine learning on complex problems usually observe that a favorable classification performance often requires a larger training dataset that allows a more complex model with more features [13]. Nevertheless, identifying and labeling a large number of clashes requires tremendous and expensive manpower; therefore, the prediction accuracy of machine learning is often insufficient before a sufficiently large number of cases are collected [7]. The industry is in urgent need of more cost-effective solutions on this issue.

In the field of machine learning, many studies applied a combination of two or more sophisticated methods on specific domains and obtained better results than using individual methods. A hybrid method was developed for discretizing continuous attributes to enhance the accuracy of the naïve Bayesian classifier [14]. An algorithm based on support vector machine (SVM), 2D fast Fourier transform (FFT), and hybrid fuzzy c-mean techniques was proposed to recognize and visualize the cracking incurred in the structure [15]. The hybrid ML algorithm performs better to recognize cracks with higher accuracy than the traditional SVM. Another hybrid computational model based on genetic algorithm (GA) and support vector regression (SVR) was developed to predict bridge scour depth near piers and abutments [16]. The proposed hybrid model achieved 80% more accurate error rates than those obtained using other methods, such as regression tree, chi-squared automatic interaction detector, artificial neural network, and ensemble models. These studies provide examples that demonstrate the effect of using a hybrid method.

This study attempted to combine the techniques of rule-based reasoning and supervised machine learning to develop an algorithm that can automatically filter out irrelevant clashes from the BIM-generated clash detection reports. The main purpose of this study is to explore whether the hybrid method we proposed can enhance the predictive accuracy by machine learning algorithms without a large number of training cases as well as without increasing the development manpower. Unlike most rule-based systems that require an exhaustive knowledge acquisition process, the rule-based reasoning in this study is not intended to obtain accurate results, because this often requires a great amount of efforts regarding knowledge acquisition. Instead, we intend to first obtain a preliminary classification of clashes by applying a simple rule set acquired from the same experts of labeling for the subsequent machine learning process and incorporate these preliminary results in the machine learning process to see if they can help improve the prediction accuracy.

2. Related Work

In most construction projects, structural, mechanical, electrical, and plumbing (MEP) engineers develop their designs based on the architectural model. This base model is often constantly updated along with the progress of the design work. Without the adequate synchronization of all the updates among these design teams, there will be so-called "design clashes". It refers to a conflict of building components overlapping each other spatially when various types of engineering drawings are compiled. As pointed out by some scholars, if a collaboration environment exists between design teams, most clashes can be avoided [7]. However, in the participatory action research of the United Kingdom, researchers introduced a collaboration environment in a multi-floor large-scale construction project, where the engineers were assisted by software to avoid design clashes. However, there were still more than 400 clashes found between the structural model and the MEP model [4]. Their study pointed out that collaboration can indeed reduce design conflicts, but clash detection is still a necessary operation.

In the era of 2D drawings, design conflicts were not easily detected at the design phase, but remained until the construction, leading to reworks or even change orders. Clashes have been regarded as one of the major factors causing cost overruns and project delays. The emergence of BIM software enables easy design conflict/clash detection; conflict checking has been one of the essential functions of BIM software. However, Helm et al. [5] found that those clash detection algorithms in most BIM software are relatively simple: as long as two building components are spatially overlapping, touching, or within a given distance, they are recognized as conflicts and are listed in the detection report. Identifying and resolving those detected clashes is a time-consuming and laborious task [4,6,7]. The clashes detected by BIM software can be roughly divided into three categories: (1) errors, which are the clashes that will affect the project and must be resolved, such as structural components penetrated by pipes; (2) deliberate clashes, which includes intentional clashes originating from the designer, such as the pipes and conduits penetrating through the slabs; (3) pseudo clashes, which are permissible clashes appearing to be errors. As Wang and Leite [17] discovered, the proportions of deliberate and pseudo clashes, which are also known as "irrelevant clashes", in a particular project were up to 50% [10]. Among the cases considered in our study, this proportion was even higher. Scholars termed these conflicts that do not have substantial impacts on the project as "irrelevant clashes" [6,7]. These irrelevant clashes can be discovered in subsequent project stages and easily handled by the site engineers themselves; therefore, there is no need to resolve them during the clash detection. However, the clash detection report of BIM software does not disclose these irrelevant clashes, which means they must be manually identified by BIM managers instead. As pointed out by Hu et al. [7], in practice, many projects can have millions of clashes, so automating the filtering of irrelevant clashes is an important and urgently needed function [4,6,7].

Existing methods of reducing irrelevant clashes can be roughly divided into three aspects: clash avoidance, clash detection improvement, and clash filtering [7]. Clash avoidance begins with the modeling method during the design phase, emphasizing the collaboration and coordination among the design teams to avoid the occurrence of clashes from the beginning. Mehrbod et al. [18] established taxonomy to classify the causes of clashes into three categories, namely, process-based, model-based, and physical design [12]. They aimed to understand the causes of design conflicts and the consideration factors for conflict/clash resolution. With the aim of reducing the occurrence of clashes through automated coordination, Wang and Leite [17] constructed a sematic schema for MEP coordination that was used to present and acquire the experience and knowledge hidden behind the coordination issues. Both Hartmann [19] and Gijezen [20] re-examined the BIM model via a more organized work breakdown structure (WBS) to reduce the number of irrelevant clashes. However, scholars believe that this approach undoubtedly increases the burden on the design teams [7]. Collaboration would be difficult to implement in many projects because project participants may not have a mutual contractual relationship [6,9].

Meanwhile, some scholars consider that improving the clashes detection algorithms in BIM software can increase the accuracy of its detection, thereby reducing the number of irrelevant clashes [5]. These methods include the sphere-trees method [21], approximate polyhedra with spheres and bounding volume hierarchy [22,23], oriented bounding boxes or OBB-trees method [24], and ray–triangle intersection algorithm [5]. These algorithms are continually improved to increase the accuracy of clash detection. Yet, as pointed out by scholars, the refined clash detection algorithms still cannot effectively reduce irrelevant clashes [10], especially those caused by human errors [7].

The third method is to directly identify and filter out irrelevant clashes in the clash detection report of BIM software. One of the popular methods of identification or diagnosis on a certain domain is rule-based systems [25]. Rule-based systems, also known as rule-based expert systems, have been commonly used in many fields such as medical, engineering, manufacturing, education, etc. since the 1980s and have been proved to be effective in pattern recognition, diagnosis, decision-making, control, planning, and so on due to the transparency of knowledge reasoning and consistency of reasoning results [12]. However, despite their advantages, rule-based systems require a considerable amount of time to acquire the knowledge that is needed for reasoning. A rule-based system was proposed to automate the code checking process for green construction. Still, the researchers found that the domain knowledge is dispersed and fragmented, and rule acquisition requires human experts from many professional fields [11]. Hu et al. [7] applied the rules to identify the dependency relationship of conflicting/clashing components, thereby constructing a component-dependent network. This network can be used to identify the central components of clashes, group those repetitive clashes, and finally filter out irrelevant clashes. However, the number of irrelevant clashes being filtered out depends on the components' dependency relationships and their query algorithms, which are similar to a rule-based knowledge base, which requires a lot of effort to capture and maintain those rules. Besides, the rules developed by their study may not necessarily fit other projects.

Another method that also is popular for complex problems and does not require too many efforts on knowledge acquisition is machine learning. Machine learning algorithms use computational methods to predict results directly from historical data without relying on predetermined rules or equations on domain knowledge. Besides, the algorithms adaptively improve their performance as the number of training cases increases [13,15]. Despite its ease of identifying trends and patterns without human intervention, researchers often argued that machine learning requires a sufficiently large training dataset that allows a more complex model in order to obtain favorable results [7,15]. In the field of identifying design clashes, Hu and Castro-Lacouture [6] used a historical dataset of 204 clashes from a three-story building and implemented six different machine learning classifiers including J48-based decision trees, random forest, Jrip, binary logistic regression, naïve Bayesian, and Bayesian network to filter out irrelevant clashes. The features selected for machine learning process considered three aspects: (1) the information uncertainty level; (2) problem complexity, such as clashing objects' size, priority, materials, type, and clashing volume, and (3) contextual flexibility, such as the location, spatial relationship, and available space [6]. However, their method based on machine learning obtained an average prediction accuracy of 80%, but it required a great amount of labor to preprocess the training data. Some researchers then argued that before a sufficient number of training cases are collected, the prediction accuracy is insufficient [7].

In summary, both rule-based reasoning and machine learning have their own pros and cons to provide a solution to domain problems. The former produces a favorable result no matter how big the training data size is but often requires a great number of efforts to acquire knowledge from human experts. On the contrary, the latter does not require human efforts to prepare and formulate the domain knowledge to produce results. Still, it requires sufficient training data in order to obtain a favorable result. In the field of machine learning, many studies have proved that applying the hybrid method that combines two or more sophisticated algorithms on certain domains can obtain better results than using individual methods [14–16]. However, most studies combined two or more machine learning algorithms as their hybrid methods, but few have taken advantage of machine learning and rule-based systems from the perspectives of minimal development efforts and maximal predictive performances.

Considering the nature of clash detection, in which a large training dataset is not easy and cost-effective to collect and prepare, this study made the best use of expertise from human experts hired by the research team to both prepare the training dataset for machine learning and to be interviewed to acquire their heuristic know-how for rule-based reasoning. Based on the perspective of clash filtering, this study first used rule-based reasoning to preliminarily determine the type of clashes; subsequently, the results of the preliminary classification are added into the dataset of machine learning for training and the testing of classifiers in order to improve the prediction accuracy under a small training dataset.

3. Methodology

The knowledge acquisition of a rule-based system that takes into consideration spatial relations is not easy, and machine learning requires numerous training cases in order to obtain a reasonable accuracy. Therefore, this study proposes a hybrid method by first developing a simple rule-based system that merely takes into consideration clash attributes; then, the results of the rule-based reasoning is merged to the dataset of machine learning. This study develops the research methodology shown in Figure 1 to validate the effectiveness of this method. First, a real architectural project is selected, BIM software is used for clash detection, and then the (clash) detection report is submitted to two experts for labeling clash types. The labeling results with identical labels from the experts are selected and further adjusted to form training dataset #1; this dataset is subjected to a supervised machine learning process to obtain the "pure machine learning results". At the same time, the same experts are interviewed to acquire their knowledge of determining the types of clashes and incorporate this knowledge into rules. These rules only consider attributes of clashing components and do not delve into other deeper spatial relations with others. After implementing the rules, training dataset #1 is used in the same manner for rule-based reasoning to obtain the "pure rule-based results". Next, these results are regarded as a field of training data inserted to training dataset #1 and form training dataset #2; this dataset is again processed by the same supervised machine learning to obtain the "hybrid results". Finally, the accuracy of the three prediction results is evaluated and compared. In the following paragraphs, Section 3.1 first describes the data collection process; Section 3.2 describes the process of hiring experts to label the type of case clashes for the clash detection report; and Section 3.3 describes the selection and adjustment of the labeling results. The development of the rule-based system and its prediction results are introduced in Section 4, followed by Section 5 describing the prediction results of two similar machine learning processes.

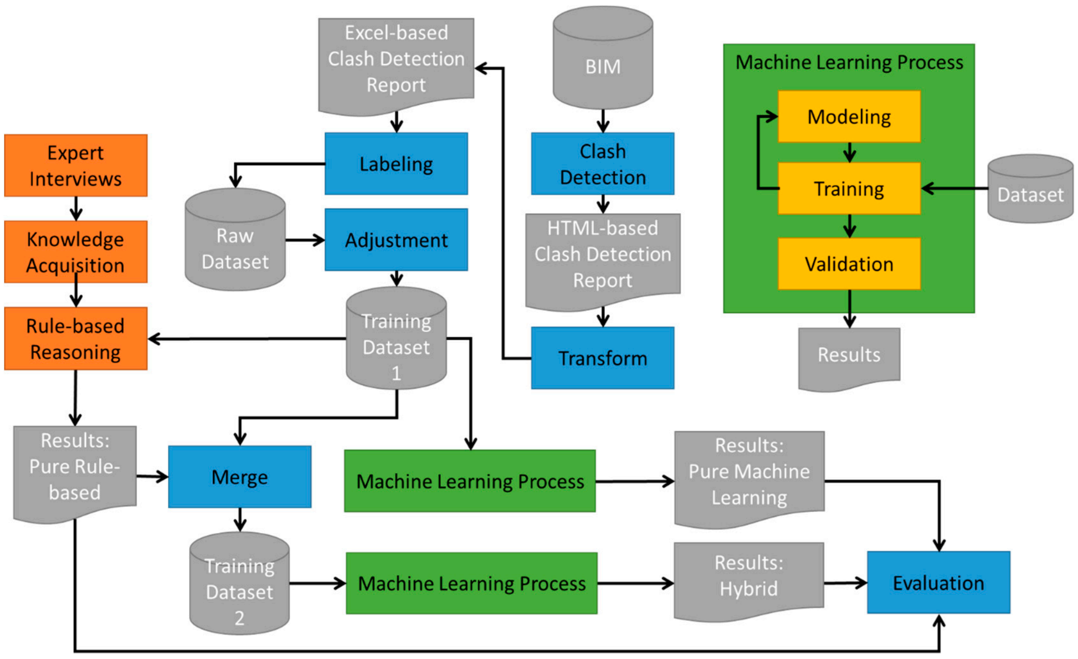


Figure 1. Process of research methodology.

3.1. Data Collection

This study used a large shopping mall with nine floors above the ground and four floors underground as the testing case. This building was considered mainly because it consisted of a large number of complicated pipes and conduits, as shown in Figure 2. The structural and MEP model of the building were extracted for clash detection through Autodesk Navisworks Manage 2017. Clashes detected by most BIM software can be hard clashes or soft clashes. Hard clashes exist when building elements have physical overlaps, whereas soft clashes occur when an element is not given the spatial tolerances it requires. Similar to most of the studies discussing clash detection mentioned in Section 2, this study only considers "hard clashes" because they have a universal definition, and therefore research results can be compared. In order to control the number of total clashes within an acceptable range for labeling work by the hired experts, this study merely selects the water supply pipes and fittings from the MEP model for clash detection against the structural model. Figure 3 shows a part of the HTML-based clash detection report. The summary table on the top records the total number of clashes in the report; the table below lists the detailed information of each conflict/clash. The detailed information includes the grid location of the conflict/clash, clash point, and distance, as well as the information of two clashing objects, including their IDs, floors, names, and types. Moreover, snapshots of two clashing objects are attached. Clicking on the thumbnails in the first column of the table allows viewing the enlarged snapshots, as shown in Figure 4. Table 1 presents the complete statistics of the clash detection report; there are a total of 415 clashes between four structural components (beams, columns, slabs, walls) and two MEP components (pipes and fittings).

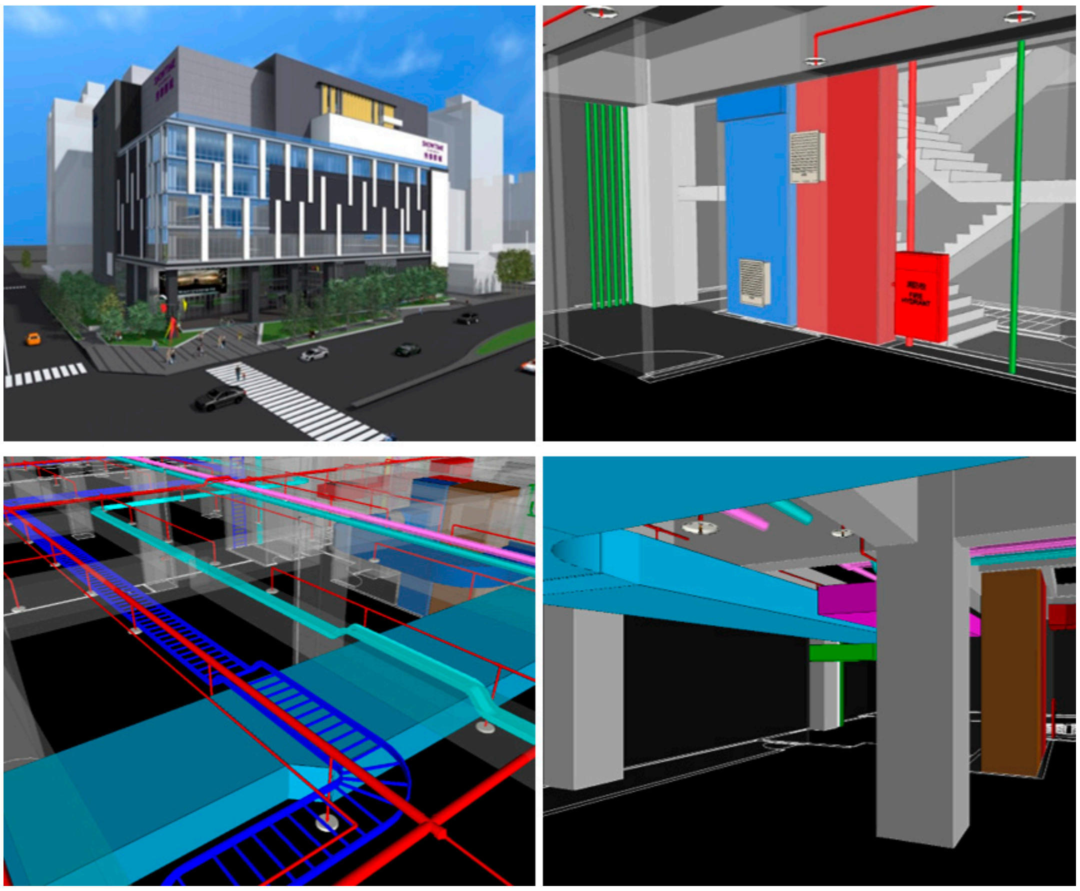
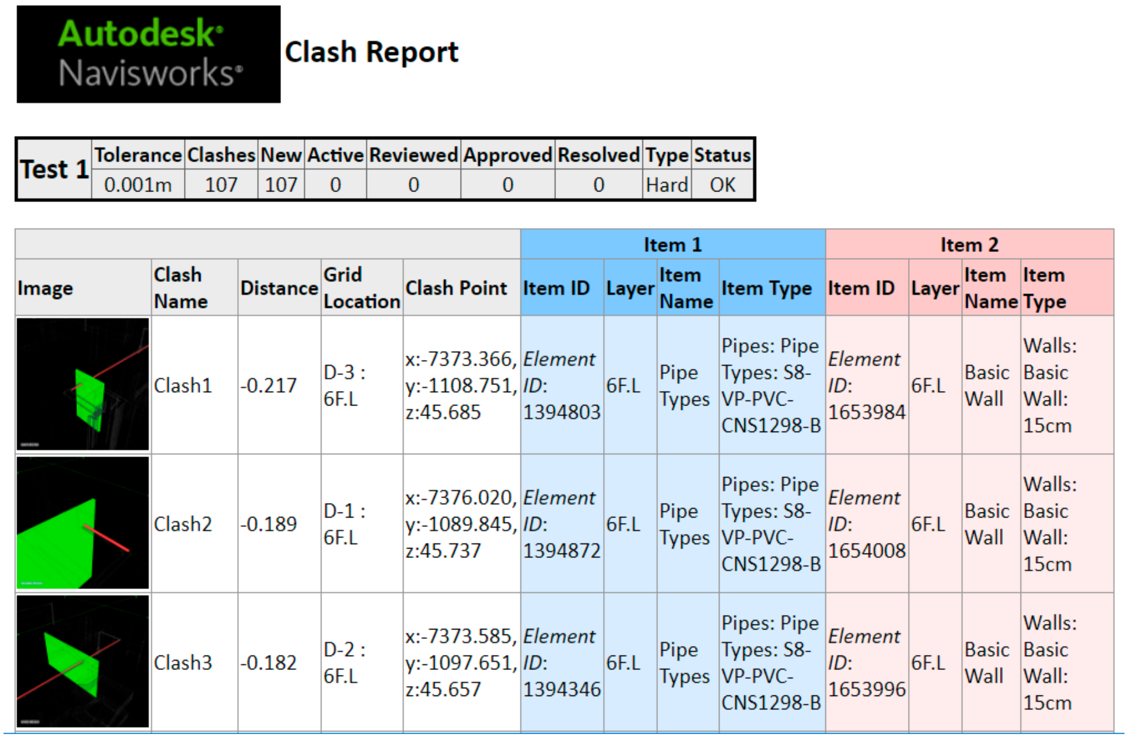


Figure 2. Case study building: exterior and internal pipeline configuration of the building object.



Autodesk Navisworks Clash Report

Test 1	Tolerance	Clashes	New	Active	Reviewed	Approved	Resolved	Type	Status
	0.001m	107	107	0	0	0	0	Hard	OK

					Item 1				Item 2			
Image	Clash Name	Distance	Grid Location	Clash Point	Item ID	Layer	Item Name	Item Type	Item ID	Layer	Item Name	Item Type
	Clash1	-0.217	D-3 : 6F.L	x:-7373.366, y:-1108.751, z:45.685	*Element ID*: 1394803	6F.L	Pipe Types	Pipes: Pipe Types: S8-VP-PVC-CNS1298-B	*Element ID*: 1653984	6F.L	Basic Wall	Walls: Basic Wall: 15cm
	Clash2	-0.189	D-1 : 6F.L	x:-7376.020, y:-1089.845, z:45.737	*Element ID*: 1394872	6F.L	Pipe Types	Pipes: Pipe Types: S8-VP-PVC-CNS1298-B	*Element ID*: 1654008	6F.L	Basic Wall	Walls: Basic Wall: 15cm
	Clash3	-0.182	D-2 : 6F.L	x:-7373.585, y:-1097.651, z:45.657	*Element ID*: 1394346	6F.L	Pipe Types	Pipes: Pipe Types: S8-VP-PVC-CNS1298-B	*Element ID*: 1653996	6F.L	Basic Wall	Walls: Basic Wall: 15cm

Figure 3. Illustration of the clash detection report produced by Autodesk Navisworks Manage 2017.

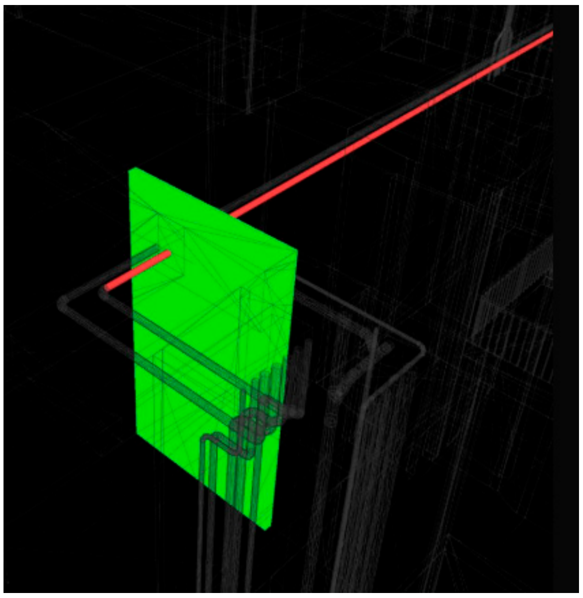


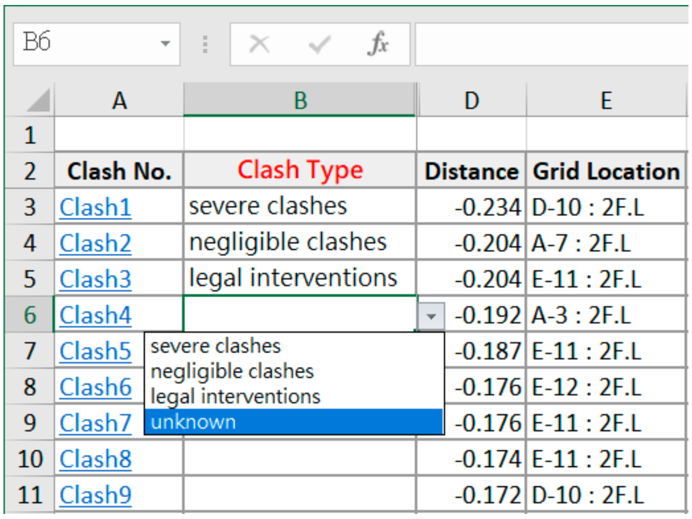
Figure 4. Snapshot of a clash produced by Autodesk Navisworks Manage 2017.

Table 1. Statistics of the clash detection report.

Clashing Item Types	Pipes	Fittings	Total
Framings	153	25	178
Walls	155	68	223
Columns	1	1	2
Slabs	12	0	12
Total	321	94	415

3.2. Labeling the Clash Types

Human experts still inevitably have different subjective judgments on the same cases, and therefore the research team hired two experts who both have more than five years of experience on clash coordination and resolution to label the clash types from the same clash detection report. Those clashes with different labels by the two experts will be excluded from the training process during the machine learning phase. Once obtaining the HTML-based clash detection report, we transform it into a spreadsheet, as shown in Figure 5, for human experts to label the clash types. Besides all the information on the clash detection report shown in Figure 3, the spreadsheet also contains a column "Clash Type" with a drop-down list to facilitate labeling clash types by the experts. The drop-down list consists of four options: serious clashes, negligible clashes, legal interventions, and unknown. These four options use a more intuitive vocabulary; at the time of subsequent analysis, these options will correspond to the four categories suggested by the literature as errors, pseudo clashes, deliberate clashes, and unknown, respectively. Serious clashes or errors are those relevant and crucial clashes that need to be carefully examined and resolved if necessary. Except for the spreadsheet containing information derived from the clash detection report, the researcher did not provide the labeling experts with other information about the building, such as CAD drawings or 3D building models. The only information for them to determine the clash types is the clash detection report mentioned in Section 3.1.



	A	B	D	E
1				
2	Clash No.	Clash Type	Distance	Grid Location
3	Clash1	severe clashes	-0.234	D-10 : 2F.L
4	Clash2	negligible clashes	-0.204	A-7 : 2F.L
5	Clash3	legal interventions	-0.204	E-11 : 2F.L
6	Clash4		-0.192	A-3 : 2F.L
7	Clash5		-0.187	E-11 : 2F.L
8	Clash6		-0.176	E-12 : 2F.L
9	Clash7		-0.176	E-11 : 2F.L
10	Clash8		-0.174	E-11 : 2F.L
11	Clash9		-0.172	D-10 : 2F.L

Figure 5. Spreadsheet-based clash detection report for labeling of clash type by experts.

3.3. Label Adjustment

Table 2 shows a summary of the labeling results by two experts. After comparing the details, 89 clashes were found with different labels by the two experts and excluded, which means only the remaining 326 cases were used for the processing. Moreover, there was no negligible clash among the clash types labeled by both experts. An investigation reveals that most negligible clashes may occur when pipes penetrate a beam. According to the specification of reinforced concrete structures [26] published by the Chinese Society of Structural Engineers in Taiwan for the positions of legal pipes penetrating through beams, if the position of clash falls on the grid area in Figure 6, the structural behavior of the beam will not be affected. In other words, this clash can be classified as a negligible clash, or a pseudo clash. Applying this specification requires the precise dimensions of the clashing objects and the clashing position. The experts were not able to determine whether those clashes were negligible or not by merely viewing the snapshots with their naked eye. Therefore, to be on the conservative side, the experts mostly labeled those clashes with pipes penetrating through beams as "serious clashes"; a few cases were labeled as unknown.

In order to increase the granularity of the training data and improve the filtering rate of irrelevant clashes later on, researchers decided to further apply the specification as mentioned above toward the original labeling results. The research team used the Model Builder embedded by Environmental Systems Research Institute (ESRI) ArcScene to implement the specification, as shown in Figure 7, and obtain the adjusted labeling result, as shown in the rightmost column of Table 2. The original 58 serious clashes and five unknowns were adjusted as negligible clashes. After adjustment, the numbers of labeling for four clash types turn to be 100 errors, 63 pseudo clashes, 127 deliberate clashes, and 36 unknowns.

Table 2. Statistics of training and testing data.

Clash Type		Expert #1	Expert #2	Common	Adjusted
Serious clashes	Errors	159	208	158	100
Negligible clashes	Pseudo clashes	0	0	0	63
Legal intervenes	Deliberate clashes	174	148	127	127
Unknown	Unknown	82	59	41	36
Total		415	415	326	326

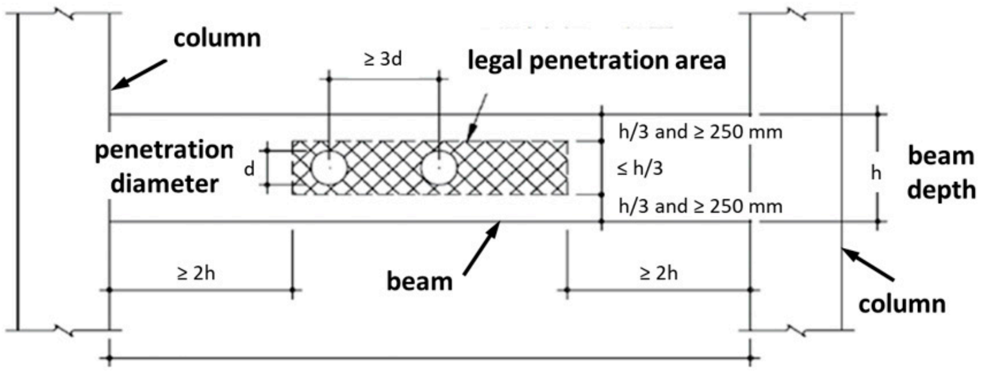


Figure 6. Legal penetrating area for structural beams [26].

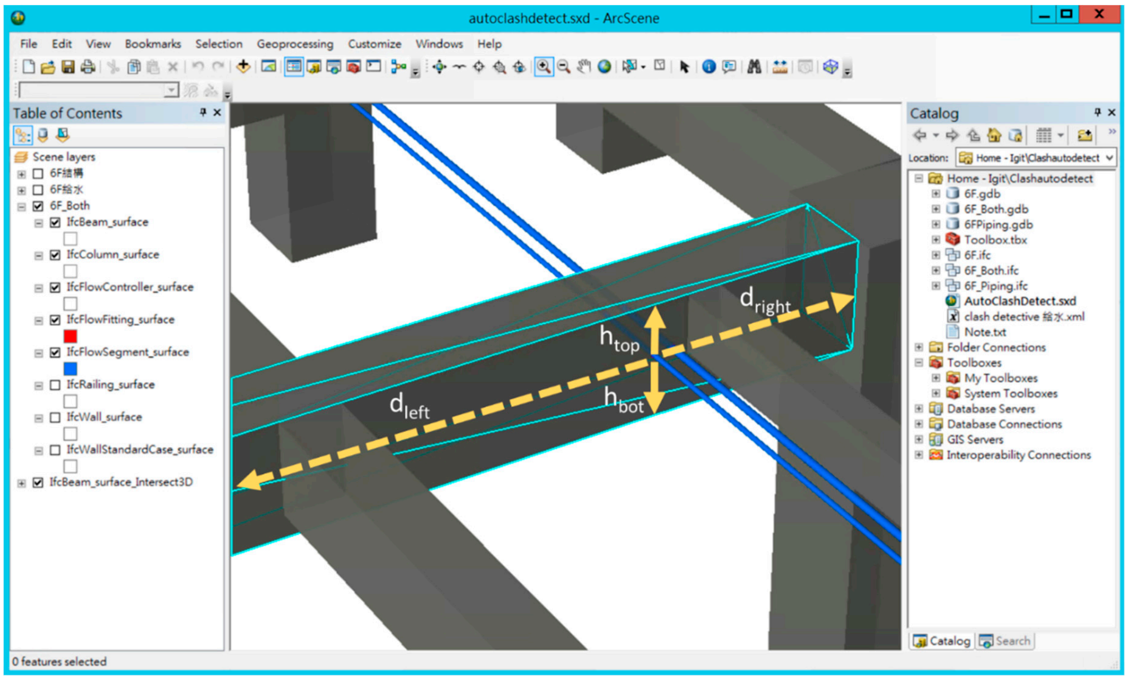


Figure 7. Automated adjustment on labeling of clash types by using ESRI (Environmental Systems Research Institute) ArcScene.

4. Rule-Based Reasoning

While directly obtaining the labeling results from the experts, we also interviewed the two hired experts to acquire their knowledge used to classify the clash types. Different from most rule-based reasoning systems [17,27] acquiring as many rules as possible, we only focus on those rules of thumb requiring facts that can be found in the clash detection report. The reason for this is that the rule-based reasoning in this study is not meant to serve as a robust method for classifying clash types; instead, it is used to serve as the catalyst to improve the prediction performance of the supervised machine learning. In addition, the clash detection report is the only reference for the experts to do their jobs. The following six rules are directly acquired after interviewing the experts:

1. Beam Rule: If the type of clashing object from the structural model is a beam, the clash type will be an "error".
2. Extended Beam Rule: If the type of clashing object from the structural model is a beam and the clash position falls within the legal area defined by the specification, the clash type will be a "pseudo clash". This rule is based on the specification we used to adjust the original labeling result mentioned in Section 3.3.

3. Column Rule: If the type of clashing object from the structural model is a column, the clash type will be an "error".
4. Slab Rule: If the type of clashing object from the structural model is a slab, the clash type will be a "deliberate clash".
5. Wall Rule: If the type of clashing object from the structural model is a wall, the clash type can be a "deliberate clash" or an "error".
6. Bearing Wall Rule: If the type of clashing object from the structural model is a wall and is not a bearing wall, the clash type will be a "deliberate clash"; otherwise, it is an "error".

Among these rules, Rule #6 requires other supporting information that the clash detection report does not provide to ensure whether the clashing wall is a bearing wall or not. Therefore, it is excluded from our final rule repository. Instead of classifying those clashing walls as "errors", the researchers simply revised Rule #5 as follows:

7. Simplified Wall Rule: If the type of clashing structural component is a wall, the clash type will be unknown.

Then, the research team applied the Rules #1–4 and #7 stated above to perform the rule-based reasoning and obtained the clash classification result, as presented in Table 3. The average accuracy rate is approximately 60% (194/326). The columns in Table 3 represent the numbers of clash types predicted using the rules, whereas the rows represent the actual clash types specified by the two experts. For example, among the 100 true errors, the rule-based reasoning correctly predicts 98 of them, and the remaining two errors are determined as unknown.

Table 3. Results of rule-based reasoning using simplified rules.

Clash Type		**Predicted Labels**				**Total**
		Errors	**Deliberate**	**Pseudo**	**Unknown**	
True labels	Errors	98	-	-	2	100
	Deliberate	-	12	-	115	127
	Pseudo	-	-	63	-	63
	Unknown	15	-	-	21	36
Total		113	12	63	138	326
Accuracy		98/113 (0.88)	12/12 (1.00)	63/63 (1.00)	21/138 (0.15)	194/326 (0.60)

As mentioned earlier, the aim of rule-based reasoning in this study is only to improve the outcomes of machine learning under a small training dataset. Therefore, the rules we applied did not consider deep and complex relationships among those clashing objects. As a result, the prediction accuracy (60%) tended to be low. The prediction results by rule-based reasoning here will be further included as a feature for the machine learning process that is introduced in Section 5.5.

5. Machine Learning Process

5.1. Feature Selection and Manipulation

The first task of the supervised machine learning process is to select those features of the dataset (i.e., attributes) that may contribute to problem-solving. Table 4 presents the results of the feature selection in this study. In the table, the features from the clash detection report include numeric ones, such as "Distance", "Floor-1", and "Floor-2". The values of these features are left unchanged without manipulation. The "clash point" is the coordinates with a mix of text and numbers, so the values are extracted separately and form three independent numeric features, namely, "Clash Point_x", "Clash Point_y", and "Clash Point_z". Furthermore, the features whose data types are nominal/text, such as "ItemType-1" and "ItemType-2", additionally require "one-hot encoding" to be transformed into numeric features. Therefore, the original six features found in the clash detection report finally result in the first 12 deriving features, which are shown in the last column of Table 4. These features will be used for the machine learning process to be introduced in Section 5.3 and 5.4.

Table 4. Summary of feature selection and preprocessing.

Original Feature	Description	Data Type	Example Value	Revised Feature
Distance	Length of clash point away from the component edge	numeric	−0.234	unchanged
Floor-1	The floor where the clashing component 1 is located	numeric	2	unchanged
Floor-2	The floor where the clashing component 2 is located	numeric	2	unchanged
Clash Point	Clash point coordinates	text	x: −7370, y: −1171, z: 24	Clash Point_x, Clash Point_y, Clash Point_z
ItemType-1	Component type of clashing component 1	nominal	Pipes | Fittings	Pipes; Fittings
ItemType-2	Component type of clashing component 2	nominal	Framings | Walls | Slabs | Columns	Framings, Walls, Slabs, Columns
Rule-Tag	Clash type predicted by rule-based reasoning	nominal	Errors | Pseudo clashes | Deliberate clashes | Unknown	Rule_errors, Rule_pseudo, Rule_deliberate, Rule unknown

In order to demonstrate whether rule-based reasoning can improve the predicting accuracy of machine learning, the results of rule-based reasoning mentioned in Section 4 are also added as a feature "Rule-Tag" for the machine learning process, which is addressed in Section 5.5. Since the data type of this feature is also nominal/text, "one-hot encoding" is also applied in the same manner. That makes a total of 16 features that are present in the training dataset for the machine learning process of Section 5.5.

The subsequent machine learning process is divided into two experiments. The first experiment uses the first 12 features shown in the last column of Table 4 for the training and testing of classifiers, i.e., the dataset does not contain the results of rule-based reasoning; this dataset is denoted as training dataset #1, as shown in Figure 1. Section 5.3 and 5.4 will explain this process and the results in detail. The second experiment uses all the features in the last column of Table 4; the dataset is denoted as training dataset #2. These two experiments are then compared to evaluate the impacts of rule-based reasoning on the prediction accuracy of machine learning. Section 5.5 will detail the results of this experiment.

5.2. Classification Algorithms and Parameters

This study adopted the open-source machine learning library, Scikit-Learn, as a development tool, which provides a rich set of tools and various algorithms required for classification and regression problems. As suggested by many machine learning studies [28,29], no single classifier can work best across all scenarios. The best practice of choosing a classification algorithm is to compare the performance of several learning algorithms and select the best model for the problem to be solved. Even so, the best model under this consideration may still vary regarding the nature of the training dataset we collect, the number of training cases, and the features we select for the learning process. Besides, instead of determining the best model for a particular problem, the main purpose of this study is to demonstrate the effect of the hybrid method we propose on the predictive performance. Therefore, this study implemented several classifiers using common classification algorithms including decision tree (DT) [30,31], support vector machine (SVM) [32–34], and k-nearest neighbor (k-NN) [28,35], as well as three classifiers applying ensemble learning strategies [28], and then conducted the training process using these classifiers upon two training datasets, one with the feedback from rule-based reasoning and the other without it, and compared both predictive performances. The decision tree is chosen because it works well with both numerical and categorical features and provides interpretability for decision-making [29], while SVM can generate robust results for complex classification problems [15]. k-NN, an easy-to-implement classifier that works well with a small dataset but suffers from low efficiency when the dataset grows, is used as a benchmark for DT and SVM classifiers [29]. The reason why those ensemble learning classifiers are also included in this study is that they combine multiple classifiers to have a better performance than individual classifiers alone. However, as mentioned above, we are not intending to decide which classifier best fits the domain problem, so we did not dive deep into tuning those hyper-parameters of each classifier. The results of the comparison among them will be addressed later in Section 6.

Table 5 summarizes the manipulation of training classifiers implemented by this study. To test and evaluate the prediction accuracy of different classifiers, this study randomly selected 30% of 326 cases, i.e., 98 sets, as the testing dataset; the remaining 70% was the training dataset, i.e., 228 cases. For the training of all types of classifiers, the training dataset was subjected to k-fold cross-validation (k = 5) with a 4:1 split ratio. When implementing classifiers such as kNN, SVM, and Voting, the dataset was subjected to standardization before proceeding to the learning process. Tables 6 and 7 list the modeling parameters of individual classifiers and ensemble learning classifiers, respectively.

Table 5. Summary of machine learning manipulation. K-NN: k-nearest neighbor, SVM: support vector machine.

Measures	Description
Data splitting	7:3 (228 cases for training; 98 cases for testing)
Performance metric	Error matrix (also known as Confusion matrix)
Classifiers	Decision trees, SVM, k-NN, Voting, Bagging, Random forest
Evaluation	k-fold cross-validation (k = 5)

Table 6. Modeling parameters of individual classifiers.

Classifiers	Parameters
DecisionTree	criterion = “gini”, max_depth = 6
KNeighborsClassifier	n_neighbors = 3
SVMClassifier	c = 1.0, kernel = “linear” & “rbf”

Table 7. Modeling parameters of ensemble learning classifiers.

Ensemble Learning Strategy	Estimators	Parameters

VotingClassifier	DecisionTree (max_depth = 6)
	KNeighbors (n_neighbors = 3)
	SVM (kernel = “linear”)
		voting = “soft”,
		weights = [5, 1, 1]

BaggingClassifier	DecisionTree (max_depth = 6)	n_estimators = 100
RandomForestClassifier	DecisionTree (max_depth = 6)	n_estimators = 100, criterion = “gini”

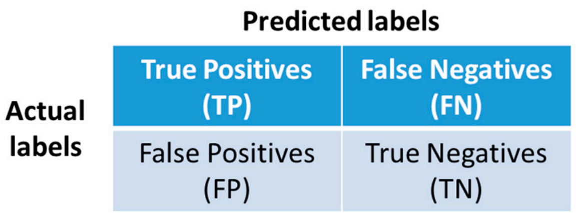
Furthermore, the error matrix, also known as the confusion matrix, was adopted for the evaluation, which reports the counts of the true positive, true negative, false positive, and false negative predictions of a classifier, as shown in Figure 8 [28]. Three indicators derived from the error matrix were recorded to evaluate the performance metrics against all the training classifiers, including precision, recall, and f1-score, which are defined according to Equations (1)–(3), respectively. Precision is the number of true positive results divided by the number of all the positive results returned by the classifier, and recall is the number of true positive results divided by the number of all the samples that should have been identified as positive. The f1-score is the harmonic mean of the precision and recall, with its best value at 1 and worst value at 0.

$$Precision = \frac{TP}{TP + FP} \tag{1}$$

$$Recall = \frac{TP}{TP + FN} \tag{2}$$

$$f1 - score = 2 * \frac{Precision * Recall}{Precision + Recall} \tag{3}$$

Sections 5.3 and 5.4 explain the machine learning processes using training dataset #1 for individual classifiers and ensemble learning classifiers, respectively; Section 5.5 explains the results obtained using training dataset #2 along with rule-based reasoning results for the same machine learning process.



	Predicted labels	
Actual labels	True Positives (TP)	False Negatives (FN)
	False Positives (FP)	True Negatives (TN)

Figure 8. Error matrix used to evaluate the performance of the trained classifiers.

5.3. *Individual Classifiers of the Linear Model*

1. Decision Trees (DTs) Classifier

First, the research team implemented a classifier using the DT, which can work with both numerical and categorical features and also provides interpretability for decision-making [29]. The algorithm starts at the root of the feature tree and splits the dataset on the tree node, which results in the largest information gain, IG, as shown in Equation (4) [28]. The objective function of a decision tree to optimize the training is to maximize the information gain at each node [29].

$$IG\left(D_p,\ f\right) = I\left(D_p\right) - \sum_{j=1}^{m} \frac{N_j}{N_p} I\left(D_j\right) \tag{4}$$

In Equation (4), f is the feature to perform the split, D_p and D_j are the datasets of the parent and jth child node, I is the measure of data applying the splitting criteria, N_p is the total number of samples at the parent node, and N_j is the number of samples in the jth child node. As stated by this equation, the information gain is the difference between the measures of splitting data of the parent node and the sum of the child node impurities. The splitting criteria we used in this study is Gini impurity (I_G), which is defined in Equation (5) [28]:

$$I_G(t) = \sum_{i=1}^{c} p(i|t)(1 - p(i|t)) = 1 - \sum_{i=1}^{c} p\left(i|t\right)^2 \tag{5}$$

Here, $p(i|t)$ is the proportion of the cases that belong to class i for a particular node t. Table 8 and Figure 9 shows one of the prediction results of the DT classifier that we implemented and trained using training dataset #1 depicted in Figure 1, or the dataset without the feedback from rule-based reasoning. Among 98 testing cases, 81 cases were correctly classified, which made an overall precision of 0.85, recall of 0.83, and f1-score of 0.83. In order to avoid bias from one test, we repeated the above training and testing process 30 times, where different training and testing cases were randomly selected for each test. The average precision, recall, and f1-score of 30 tests are 0.88, 0.87, and 0.87, respectively.

Table 8. One of the predictive results of the DT classifier using training dataset #1.

Clash Types	Precision	Recall	F1-Score	Cases
Errors	0.57	0.81	0.67	21
Pseudo clashes	0.89	0.70	0.78	23
Deliberate clashes	1.00	1.00	1.00	42
Unknown	0.75	0.50	0.60	12
Average/Total	0.85	0.83	0.83	98

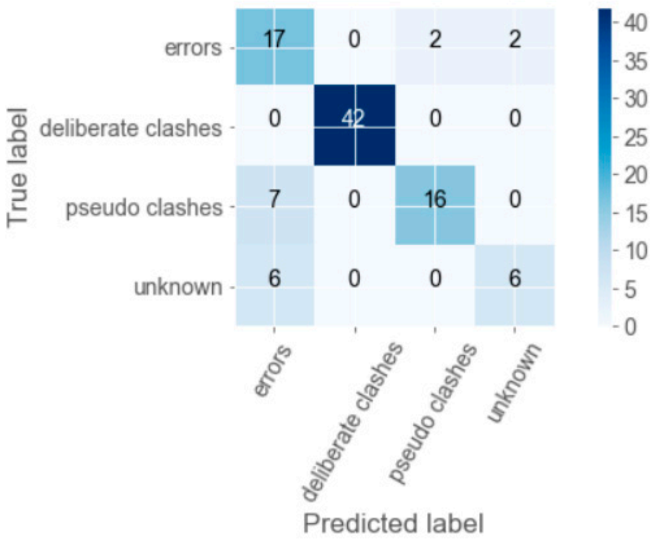


Figure 9. The error matrix of the test as shown in Table 8.

2. Support Vector Machine (SVM) Classifier

SVM is also a common machine learning classifier that is widely used in practical and theoretically domains [29]. This study implemented a classifier using the SVM algorithm because they can generate accurate and robust results for classification problems, even when training data are nonlinearly separable [29,33]. They can also be easily extended to solve nonlinear classification problems using a nonlinear "kernel" [32,34]. The optimization objective of SVM classifiers is to maximize the distance between the separating decision boundary and the training samples that are closest to this boundary [29]. A penalty can also be applied for misclassification [28]. Using training dataset #1 and repeating 30 train-then-test cycles the same as stated previously, the SVM classifier implemented by this study used a linear kernel and obtained an average f1-score of 0.79 among 30 tests and an average f1-score of 0.74 when using a radius basis function kernel.

3. K-Nearest Neighbors (k-NN) Classifier

Next, a classifier using the k-NN algorithm was implemented. k-NN was selected because it is a robust classifier that is often used as a benchmark for more complex classifiers, such as SVM and decision trees [35]. Besides, since both the number of features and size of our training dataset are not very large, this classifier does not suffer from the "curse of dimensionality" and low computation efficiency [28]. The average precision, recall, and f1-score of 30 tests for the k-NN classifier with three neighbors and a uniform weight are 0.79, 0.78, and 0.78, respectively, which is close to the results of the SVM classifier and lower than the DT classifier we implemented. This implies that the performances of both SVM and DT classifiers implemented by this study are considerably robust and trustworthy.

5.4. Multiple Classifiers by Ensemble Learning

The three individual classifiers mentioned previously could obtain a predictive precision ranging from 0.79 to 0.88. The research team further considered constructing a group of individual classifiers that can often receive a better predictive precision than any of its members, as suggested by the so-called ensemble learning [28]. The benefit of the ensemble method is that it combines different classifiers into a multiple classifier and thus has a better prediction performance than individual classifiers alone. The three ensemble learning classifiers implemented in this study are based on different strategies, including majority voting, bagging, and random forest.

1. Majority Voting Classifier

As stated earlier, during the ensemble learning, a group of individual classifiers will be involved in the prediction process. Majority voting means that we combine different types of classifiers for prediction and select the answer that is predicted by the majority of classifiers [28]. The majority voting classifier implemented by this study comprises all three individual classifiers introduced in Section 5.3. Since the DT classifier has the best predictive performance among the three, the voting weights of the DT, SVM, and k-NN classifier is assigned as 5:1:1, respectively, indicating that the prediction made by the DT classifier has five times more weight than the predictions by other two classifiers.

Table 9 and Figure 10 present one of the predictions made by the majority voting classifier that we implemented using training dataset #1, which was the one without the feedback from rule-based reasoning. Its overall predictive precision, recall, and f1-score are 0.86, 0.84, and 0.84. Again, we repeated the train-then-test cycle 30 times and obtained an average precision of 0.89, recall of 0.88, and f1-score of 0.88. The predictive performance of the majority voting classifier is close to the individual DT classifier, indicating that the ensemble learning effect is not significant. Although this result can be further improved by choosing a new set of individual classifiers or tuning the hyper-parameters such as weights, we still stopped here and moved forward to implement other ensemble learning classifiers.

Table 9. One of the predictive results of the majority voting classifier using training dataset #1.

Clash Types	Precision	Recall	F1-Score	Cases
Errors	0.61	0.81	0.69	21
Pseudo clashes	0.90	0.78	0.84	23
Deliberate clashes	1.00	0.95	0.98	42
Unknown	0.70	0.58	0.64	12
Average/Total	0.86	0.84	0.84	98

2. Bagging

Bagging is another ensemble strategy that is similar to majority voting that includes multiple classifiers and casts votes to the final decision. Instead of using different types of individual classifiers and the same training dataset to fit the classifiers, a bagging classifier only employs the same type of individual classifier and draws random samples from the training set to train the classifiers. It is a useful technique to reduce the model variance [28]. The bagging classifier we implemented employed 100 DT classifiers. Using training dataset #1 and repeating 30 train-then-test cycles, the bagging classifier obtains an average predictive performance of 0.91. Comparing with the previous individual classifiers mentioned in Section 5.3, this result moderately demonstrates the effect of ensemble learning.

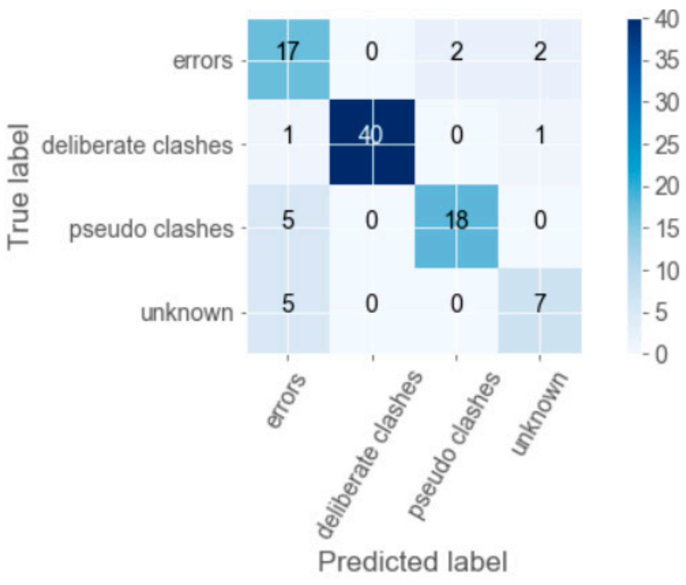


Figure 10. The error matrix of the test as shown in Table 9.

3. Random Forest

The last ensemble learning classifier we implemented is the random forest, which is a special case of bagging. Similar to bagging that only employs one type of individual classifier, which is DT, and draws random samples from the training set, the random forest also randomly selects feature subsets to fit the individual decision trees [28]. The random forest classifier we implemented also employs 100 DT classifiers and obtains a very close prediction performance of 0.89 compared with the bagging classifier, which also reflects the effect of ensemble learning.

Table 10 summarizes the average f1-scores of 30 tests using training dataset #1 for both the individual classifiers and ensemble learning classifiers implemented by this study. Table 10 also presents the average f1-scores of each clash type predicted by all classifiers.

Table 10. Summary of average f1-scores of 30 test for classifiers using training dataset #1.

Clash Types	**Individual Classifiers**			**Ensemble Learning Classifiers**		
	DT	**k-NN**	**SVM**	**Voting**	**Bagging**	**Random Forest**
Errors	0.804	0.698	0.713	0.829	0.871	0.838
Deliberate clashes	0.969	0.952	0.964	0.967	0.969	0.976
Pseudo clashes	0.773	0.681	0.649	0.789	0.841	0.809
Unknown	0.817	0.531	0.56	0.816	0.861	0.769
Average	0.868	0.775	0.785	0.876	0.905	0.881

5.5. Machine Learning with Feedback from Rule-Based ReasoningFormatting of Mathematical Components

According to the research process of Figure 1, we used training dataset #2 to perform the same machine learning process as described in Section 5.3 and 5.4 once we obtained the predictive results from rule-based reasoning introduced in Section 4. A feature called "Rule-tag" whose values are the corresponding clash types predicted by rule-based reasoning is inserted into training dataset #1, forming training dataset #2. Table 11 summarizes the average f1-scores of 30 tests using training dataset #2 for both the individual classifiers and ensemble learning classifiers implemented by this study. As shown in Table 11, the average f1-scores of individual classifiers using the training dataset with the feedback from rule-based reasoning range from 0.91 (both k-NN and SVM classifier) to 0.94 (DT classifier), whereas the average f1-score of ensemble learning ranges from 0.94 (random forest classifier) to 0.96 (bagging classifier). More evaluation and discussion between the results of machine learning using training dataset #1 and #2 will be detailed in Section 6.

Table 11. Summary of average f1-scores of 30 tests for classifiers using training dataset #2.

Clash Types	Individual Classifiers			Ensemble Learning Classifiers		
	DT	k-NN	SVM	Voting	Bagging	Random Forest
Errors	0.932	0.889	0.914	0.94	0.955	0.958
Deliberate clashes	0.970	0.959	0.958	0.97	0.972	0.966
Pseudo clashes	1.000	1.000	1.000	1.000	1.000	1.000
Unknown	0.766	0.577	0.515	0.823	0.844	0.746
Average	0.942	0.906	0.905	0.95	0.957	0.944

6. Results and Evaluation

6.1. Results

In summary, we first implemented a basic rule-based reasoning system, which was introduced in Section 4, for predicting the clash types according to the clash detection report generated by BIM software, and obtained the preliminary predictive accuracy rate of 60% (194/326). Next, we implemented six common classifiers (three individual classifiers and three ensemble learning classifiers) and conducted the same machine learning process twice using two training datasets: training dataset #1, derived from the clash detection report with clash types labeled by two human experts, and training dataset #2, derived from merging training dataset #1 and a feature of clash types predicted by the rule-based reasoning.

For the experiment with training dataset #1, the best predictive performance from 30 tests among three individual classifiers is obtained by the DT classifier (f1-score is 0.87), whereas the best ensemble learning classifier is the one using the strategy of bagging (f1-score is 0.91). Among the three individual classifiers, the k-NN classifier obtained the lowest f1-score, indicating that the results of both SVM and DT classifiers are considerably robust and trustworthy. The DT classifier outperforms the SVM classifiers by 10% probably because the dataset is relatively small, and the number of labels in this study is four rather than two or three, where SVM could function better [28]. The information gain of DT classifiers usually tends to be larger for the label with more cases in training data, so it may benefit from the nature of the training data we selected.

Among the three ensemble learning classifiers, there is no significant difference in their performance. Still, all of them outperform the three individual classifiers, proving the findings suggested by previous studies of machine learning [28], especially compared with the SVM and k-NN classifiers.

The experiment results with the training dataset #1 are also compared with the previous work by Hu and Castro-Lacouture [6], where six machine learning classifiers were implemented to filter out irrelevant clashes. Their best predictive performance in terms of f1-score, obtained by both random forest classifier and a rule-based classifier, Jrip, was 0.74, lower than the average f1-score of the random forest classifier in our study (0.881). This outperformance does not necessarily indicate that our classifiers are better because their classifiers may suffer from underfitting due to the smaller training dataset (204 versus 326 cases) and the larger number of features (10 versus 6 features). The other reason could also be the difference in the nature of training data, as we mentioned in the comparison between DT and SVM classifiers.

For the experiments with training dataset #2, the predictive performances of all three individual classifiers reach a favorable level of 0.91, while the level is 0.94 or higher for ensemble learning classifiers. The best performances for both individual and ensemble learning classifiers remain to be DT and bagging, respectively. The predictive performances of all the classifiers achieved improvements in terms of average f1-score, as shown in Figure 11. Taking a closer look at the comparisons, the average f1-scores of the SVM and k-NN classifiers significantly increased by approximately 15%–17% while the prediction improvement by the DT and three ensemble learning classifiers also increased by 6%–8.5%. Among those clash types predicted by machine learning classifiers, errors and pseudo clashes gained the greatest increase in two experiments, as shown in Figure 12.

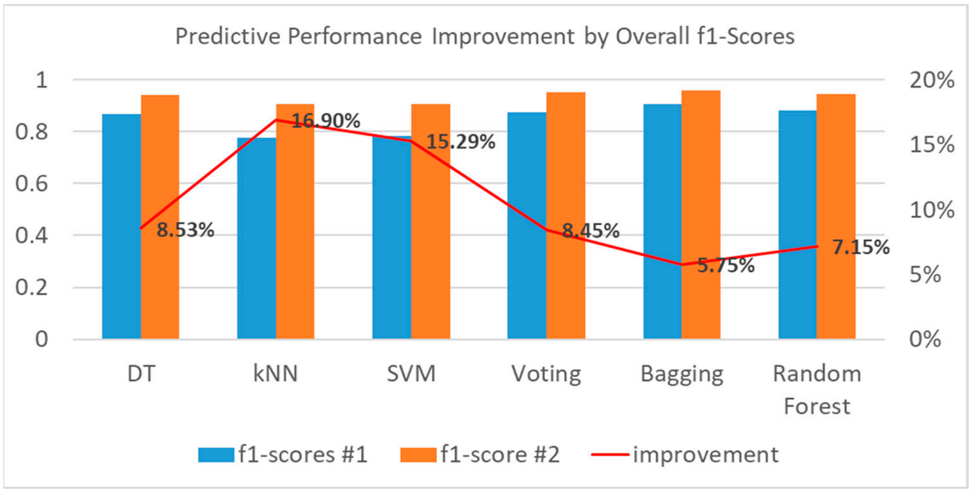


Figure 11. Predictive performance improvement by overall f1-scores.

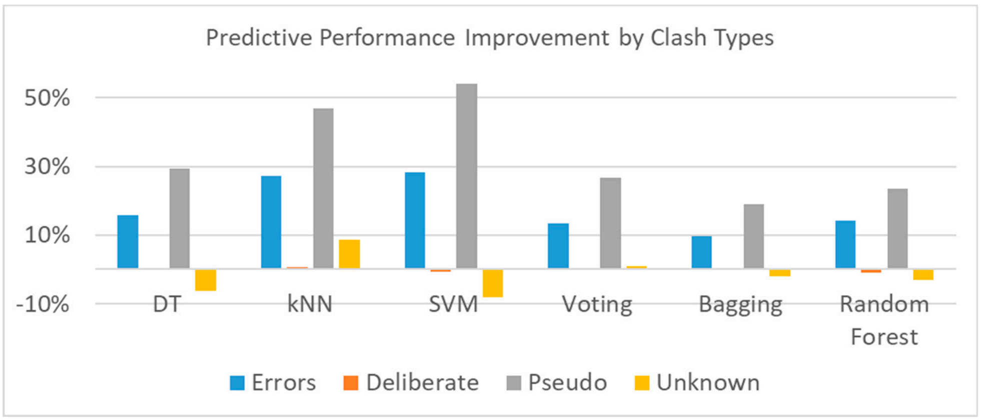


Figure 12. Predictive performance improvement by clash types.

From the results of these two experiments, we could preliminarily prove that under the condition of a small training dataset, the hybrid method proposed by this study combining rule-based reasoning and supervised machine learning can improve the predictive performance compared with using machine learning approach alone. A more in-depth evaluation will be discussed next.

6.2. Evaluation

The effect of the hybrid method on predictive performances obviously comes from the contribution of preliminary results by rule-based reasoning. Without this feedback, though it is not very accurate, the machine learning algorithms require more cases to reach a better performance. The improvement itself is not beyond our expectation, but what amazed us was that such a tiny rule base with five simple rules, possessing a relatively low predictive accuracy of 0.6, can still make contributions to those classifiers with a much higher predictive accuracy of 0.8.

Figures 13–15 illustrate the feature importance of the decision tree, bagging, and random forest classifiers. The feature importance reveals the contribution of each feature in deriving the prediction results. The left histogram of these figures shows the results using training dataset #1, whereas the right histogram uses training dataset #2. As shown in the left histograms of Figures 13–15, the most discriminative feature processed by training dataset #1 is "Framing", which implies that if the type of clashing item is a beam, it can contribute nearly 40% of correct prediction. However, among those features processed by training data with the feedback of rule-based reasoning, or training dataset #2, the rule-based tag with "errors" becomes the most contributing feature to the prediction results. The shift of the most discriminative feature between the two experiments responds and explains the above-mentioned prediction improvement. According to Table 2, the proportion of actual cases that are "errors" is around 30%, representing the second-largest label cluster in the training data and reflecting the character of DT classifiers that tend to make a prediction to those labels with more cases. The feature importance of both "framing" and "rule_errors" in our experiments may reflect the effect of the beam rule, which was introduced in Section 4. Besides, the feature "rule_pseudo" also outperforms other features. These two features could be the reason why adding the feedback of rule-based reasoning can improve the prediction performance of machine learning. On the contrary, the label with the largest cluster, "deliberate clashes", does not appear in feature important ranking list, which needs more investigation in the future.

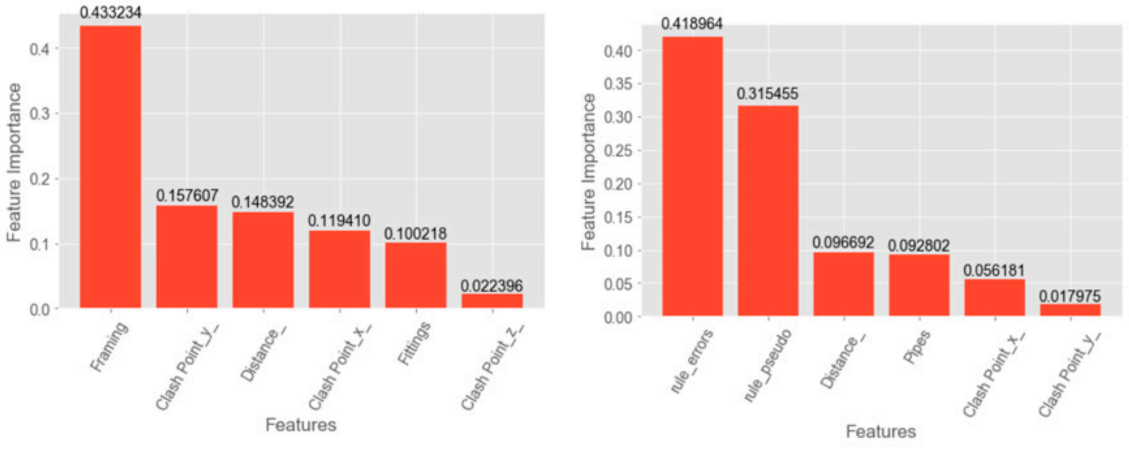


Figure 13. Feature importance suggested by the DT classifier.

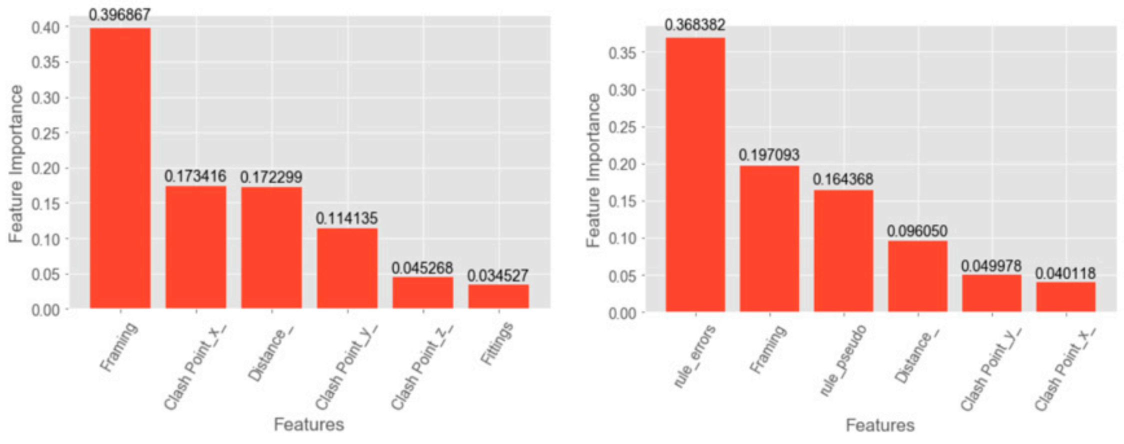


Figure 14. Feature importance suggested by the bagging classifier.

Figures 16 and 17 show the learning curves of six classifiers in the two experiments. As shown in Figure 16, the training curves and validation curves converge with a large gap, indicating that the models processed by training dataset #1 may suffer from a small degree of overfitting. This problem is significantly improved in the second experiment when the feedback of rule-based reasoning is added in the training data. According to Figure 17, all the curves converge within a small gap with an f1-score higher than 0.90.

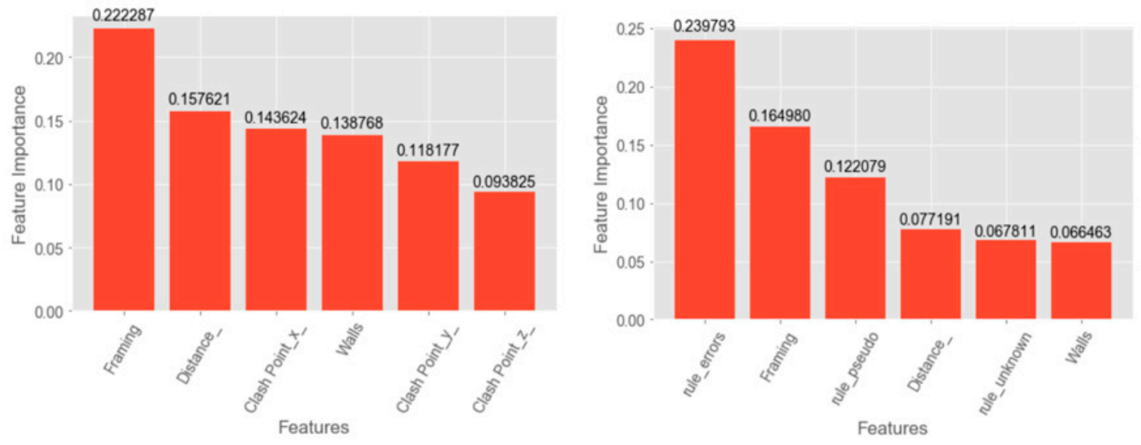


Figure 15. Feature importance suggested by the random forest classifier.

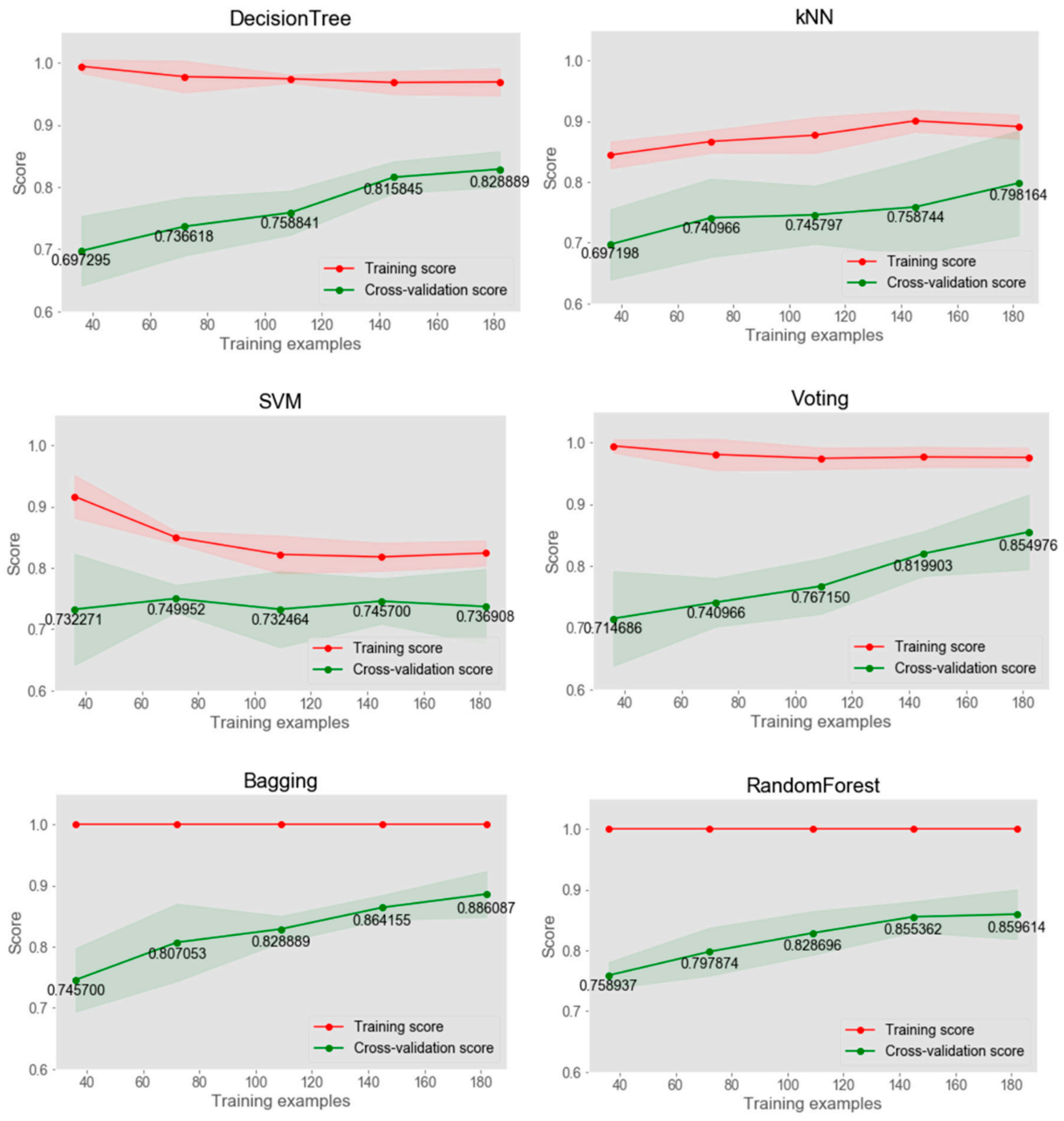


Figure 16. Learning curves of classifiers using training dataset #1.

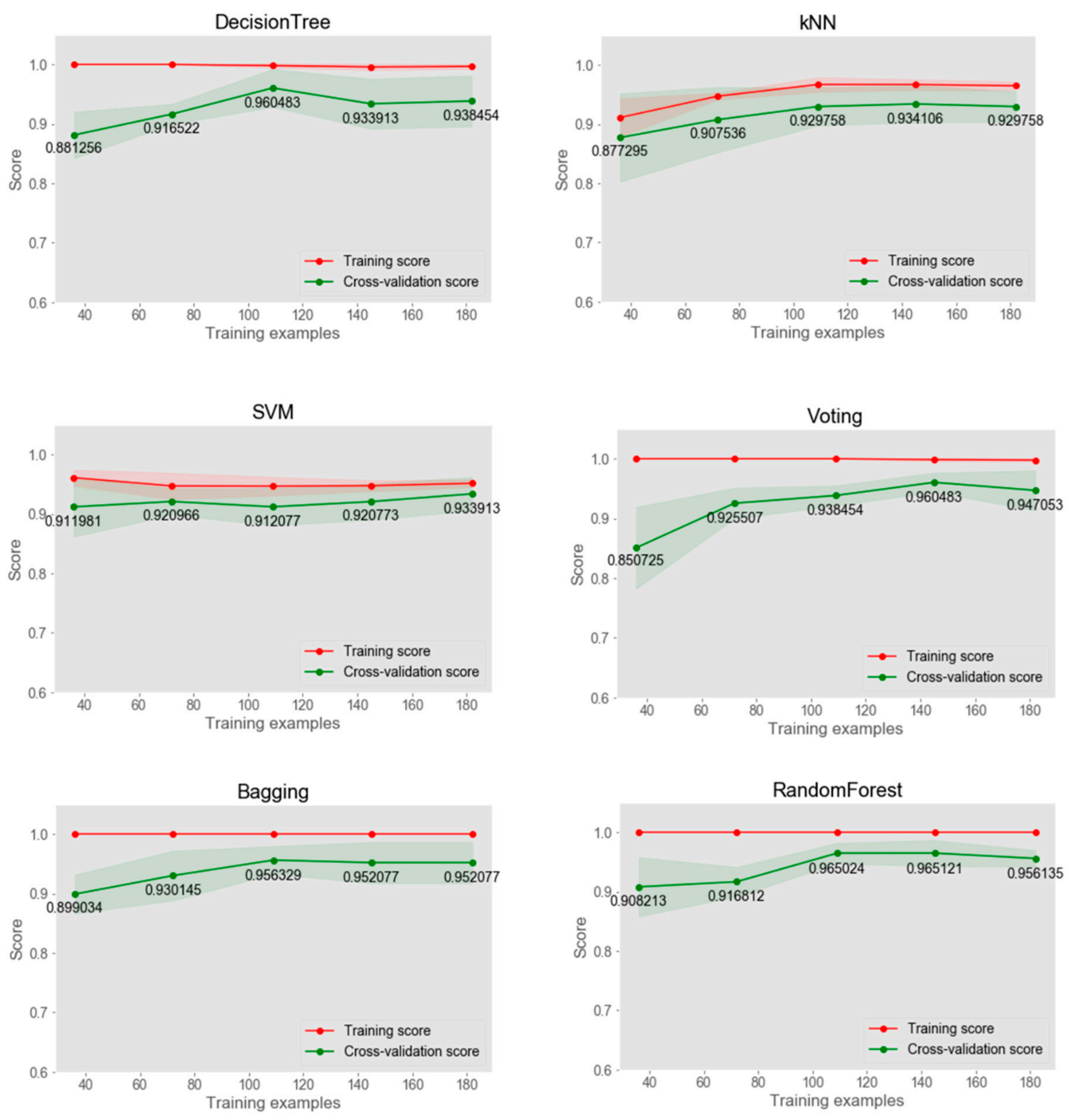


Figure 17. Learning curves of classifiers using training dataset #2.

Simply looking at the numbers, it may indicate that the models processed by training dataset #2 can achieve a satisfactory f1-score with a low bias and variance. Nevertheless, the predictive performance may be still under the influence of the nature of the training dataset we collect. For instance, among those clashes in our training data, "errors" accounted for nearly 50% of all the 326 cases. Histograms of feature importance shown in Figures 13–15 reveal the influence of this distribution on final prediction. It can be reasonably expected that the predictive performance of the same models may vary from one training dataset to another. More experiments with different training data from similar architectural projects are still needed.

Nonetheless, in light of the average f1-score improvement shown in Figure 11, the hybrid method combining the feedback from rule-based reasoning and machine learning still can be regarded as a favorable approach to enhance machine learning under a small training dataset.

Considering the ultimate goal of this study that BIM managers can benefit from the automatic clash classification of clash detection reports, both a high filtering rate of irrelevant clashes and a low misclassification rate, especially for errors, are expected. The filtering rate of irrelevant clashes we are referring to is the proportion of pseudo and deliberate clashes correctly identified among all 98 testing cases, while the error misclassification rate is the proportion that the actual errors are wrongly classified as pseudo or deliberate clashes. From the 30 tests of 98 randomly selected testing cases, we obtained an average filtering rate of 50%–60%. For the experiment of processing training dataset #1, both individual and ensemble learning classifiers wrongly classified 9.5% of errors into pseudo clashes or deliberate clashes. This number decreased to 0%–4.8% in the second experiment of processing training dataset #2. It shows that the hybrid method also had a positive effect on reducing the misclassification rate of errors.

7. Conclusions

Previous studies stated that the clash detection reports produced by most BIM software are prone to present a huge number of clashes, many of which belong to irrelevant or ignorable clashes. Manually filtering out serious clashes from the long list in the clash detection report is both time and cost consuming; thus, automatic filtering out those irrelevant clashes by algorithms is a crucial need for the current industry.

This study proposed a hybrid method that combines simple rule-based reasoning and supervised machine learning to automatically filter out irrelevant clashes from those conflicts detected by BIM software. The experiment results showed that the hybrid method can obtain a rise in the prediction accuracy of machine learning by 15%–17% for individual classifiers and 6%–8.5% for both DT and ensemble learning classifiers. The proposed method conquered the difficulties of purely developing a rule-based system considering complex relationships or merely implementing a machine learning to filter out irrelevant clashes. The former requires lots of effort for knowledge acquisition, while the number of cases collected often limits the latter's performance. It indicated that when the predictive accuracy of the conventional supervised machine learning for design clash classification is unfavorable and more training cases cannot be collected shortly, adding a feature of prediction results obtained by rule-based reasoning to the original training dataset provides an alternative to improve the prediction performance. However, the extent of improvement may depend on how well the rule-based reasoning performs. In other words, there exists a trade-off between the accuracy improvement and efforts to acquire domain knowledge when implementing the rule-based reasoning system.

The ultimate goal of identifying clash types is to resolve those errors or serious clashes before the construction phase to avoid delays and costs incurred. The resolution of serious clashes is the most important task worthy of time and effort. Even though the average predictive accuracy we obtained by the hybrid method is as high as 95%, we still conducted an analysis of the misclassification of serious clashes by our method, where actual serious clashes are wrongly classified as pseudo or deliberate clashes. One of 30 tests with 98 cases from testing dataset #2, the serious clashes misclassification rate by the bagging classifier is up to 11% (out of actual serious clashes in testing cases). How to reduce and avoid the misclassification of serious clashes remains one of the important issues in future work.

Since the training data used to conduct the machine learning process contains only clashes between structural and piping components, the current models and their predictive results can only be applied to those clash detection reports with a similar setting. The training data needs to include more MEP components, such as ducts, conduits, fire alarm devices, or lighting devices, to extend the practical value of this study. Several individual classifiers also have overfitting issues. More training data are required for more experiments in the future. Another issue of this study is that labeling the dataset highly relied on manual work. Future studies can consider applying unsupervised machine learning based on those labeled training data to build up larger training data.

Author Contributions: Funding acquisition (W.Y.L.); Conceptualization (W.Y.L.); Methodology (W.Y.L.); Data Collection (Y.-H.H.); Labeling (Y.-H.H.); Label Adjustment (W.Y.L.); Knowledge Acquisition (Y.-H.H.); Rule-based reasoning (W.Y.L.); Machine Learning Process (W.Y.L.); Validation (Y.-H.H.); Visualization (W.Y.L.); Writing—Original draft (W.Y.L.); Writing—Review and editing (Y.-H.H.).

Funding: This research was funded by the Ministry of Science and Technology, Taiwan, under grant MOST 107-2221-E-035-039.

Acknowledgments: The authors would like to express their sincere gratitude to the editor and the anonymous reviewers who significantly enhanced the contents of the study with their insightful comments. We also would like to thank Uni-edit (www.uni-edit.net) for editing and proofreading this manuscript.

Conflicts of Interest: The authors declare no conflict of interest. The funders had no role in the design of the study; in the collection, analyses, or interpretation of data; in the writing of the manuscript, or in the decision to publish the results.

References

1. Lopez, R.; Love, P.E.; Edwards, D.J.; Davis, P.R. Design error classification, causation, and prevention in construction engineering. *J. Perform. Constr. Facil.* **2010**, *24*, 399–408. [CrossRef]
2. Han, S.; Lee, S.; Pena-Mora, F. Identification and quantification of non-value-adding effort from errors and changes in design and construction projects. *J. Constr. Eng. Manag.* **2011**, *138*, 98–109. [CrossRef]
3. Won, J.; Lee, G. How to tell if a BIM project is successful: A goal-driven approach. *Autom. Constr.* **2016**, *69*, 34–43. [CrossRef]
4. Pärn, E.A.; Edwards, D.J.; Sing, M.C. Origins and probabilities of MEP and structural design clashes within a federated BIM model. *Autom. Constr.* **2018**, *85*, 209–219. [CrossRef]
5. Van den Helm, P.; Böhms, M.; van Berlo, L. IFC-based clash detection for the open-source BIMserver. In Proceedings of the International Conference on Computing in Civil and Building Engineering, Nottingham, UK, 30 June–2 July 2010; Nottingham University Press: Nottingham, UK, 2010; p. 30. Available online: http://www.engineering.nottingham.ac.uk/icccbe/proceedings/pdf/pf91.pdf (accessed on 1 May 2019).
6. Hu, Y.; Castro-Lacouture, D. Clash relevance prediction based on machine learning. *J. Comput. Civ. Eng.* **2018**, *33*, 04018060. [CrossRef]
7. Hu, Y.; Castro-Lacouture, D.; Eastman, C.M. Holistic clash detection improvement using a component dependent network in BIM projects. *Autom. Constr.* **2019**, *105*, 102832. [CrossRef]
8. Leite, F.; Akinci, B.; Garrett, J., Jr. Identification of data items needed for automatic clash detection in MEP design coordination. In Proceedings of the Construction Research Congress 2005: Building a Sustainable Future, Seattle, WA, USA, 5–7 April 2009; pp. 416–425.
9. Ciribini, A.L.C.; Ventura, S.M.; Paneroni, M. Automation in construction implementation of an interoperable process to optimise design and construction phases of a residential building: A BIM pilot project. *Autom. Constr.* **2016**, *71*, 62–73. [CrossRef]
10. Akponeware, A.O.; Adamu, Z.A. Clash detection or clash avoidance? An investigation into coordination problems in 3D BIM. *Buildings* **2017**, *7*, 75. [CrossRef]
11. Jiang, S.; Wu, Z.; Zhang, B.; Cha, H.S. Combined MvdXML and semantic technologies for green construction code checking. *Appl. Sci.* **2019**, *9*, 1463. [CrossRef]
12. Fernandez-Millan, R.; Medina-Merodio, J.A.; Plata, R.; Martinez-Herraiz, J.J.; Gutierrez-Martinez, J.M. A laboratory test expert system for clinical diagnosis support in primary health care. *Appl. Sci.* **2015**, *5*, 222–240. [CrossRef]
13. Ziolkowski, P.; Demczynski, S.; Niedostatkiewicz, M. Assessment of failure occurrence rate for concrete machine foundations used in gas and oil industry by machine learning. *Appl. Sci.* **2019**, *9*, 3267. [CrossRef]
14. Wong, T.T. A hybrid discretization method for naïve Bayesian classifiers. *Pattern Recognit.* **2012**, *45*, 2321–2325. [CrossRef]
15. Hoshyar, A.N.; Rashidi, M.; Liyanapathirana, R.; Samali, B. Algorithm development for the non-destructive testing of structural damage. *Appl. Sci.* **2019**, *9*, 2810. [CrossRef]
16. Chou, J.S.; Pham, A.D. Hybrid computational model for predicting bridge scour depth near piers and abutments. *Autom. Constr.* **2014**, *48*, 88–96. [CrossRef]

17. Wang, L.; Leite, F. Formalized knowledge representation for spatial conflict coordination of mechanical, electrical and plumbing (MEP) systems in new building projects. *Autom. Constr.* **2016**, *64*, 20–26. [CrossRef]
18. Mehrbod, S.; Staub-French, S.; Mahyar, N.; Tory, M. Beyond the clash: Investigating BIM-based building design coordination issue representation and resolution. *J. Inf. Technol. Constr.* **2019**, *24*, 33–57. Available online: http://www.itcon.org/2019_03-ITcon-Mehrbod.pdf (accessed on 12 June 2019).
19. Hartmann, T. Detecting design conflicts using building information models: A comparative lab experiment. In Proceedings of the CIB W78 2010: 27th International Conference, Cairo, Egypt, 16–18 November 2010; pp. 16–18. Available online: http://itc.scix.net/pdfs/w78-2010-57.pdf (accessed on 20 April 2019).
20. Gijezen, S. *Organizing 3D Building Information Models with the Help of Work Breakdown Structures to Improve the Clash Detection Process*; VISICO Center, Univ. of Twente: Enschede, The Netherlands, 2010.
21. Palmer, I.J.; Grimsdale, R.L. Collision Detection for Animation Using Sphere-trees. *Comput. Graph. Forum* **1995**, *14*, 105–116. [CrossRef]
22. Hubbard, P.M. Approximating polyhedra with spheres for time-critical collision detection. *ACM Trans. Graph.* **1996**, *15*, 179–210. [CrossRef]
23. Klosowski, J.T.; Held, M.; Mitchell, J.S.B.; Sowizral, H.; Zikan, K. Efficient collision detection using bounding volume hierarchies of k-DOPs. *IEEE Trans. Vis. Comput. Graph.* **1998**, *4*, 21–36. [CrossRef]
24. Gottschalk, S.; Lin, M.C.; Manocha, D.; Hill, C. Obbtree: A Hierarchical Structure for Rapid Interference Detection. Available online: http://gamma.cs.unc.edu/SSV/obb.pdf (accessed on 28 July 2019).
25. Leo Kumar, S.P. Knowledge-based expert system in manufacturing planning: State-of-the-art review. *Int. J. Prod. Res.* **2019**, *57*, 4766–4790. [CrossRef]
26. Chinese Society of Structural Engineers. *The Manual for Structural Reinforcement in Reinforced Concrete Buildings*; Technology Books Co., Ltd.: Taipei, Taiwan, 2011; ISBN 9789576554964.
27. Solihin, W.; Eastman, C. Classification of rules for automated BIM rule checking development. *Autom. Constr.* **2015**, *53*, 69–82. [CrossRef]
28. Raschka, S. *Python Machine Learning*; Packt Publishing: Birmingham, UK, 2015.
29. Bilal, M.; Oyedele, L.O.; Qadir, J.; Munir, K.; Ajayi, S.O.; Akinade, O.O.; Owolabi, H.A.; Alaka, H.A.; Pasha, M. Big Data in the construction industry: A review of present status, opportunities, and future trends. *Adv. Eng. Inform.* **2016**, *30*, 500–521. [CrossRef]
30. Pietrzyk, K. A systemic approach to moisture problems in buildings for mould safety modelling. *Build. Environ.* **2015**, *86*, 50–60. [CrossRef]
31. Desai, V.S.; Joshi, S. *Application of Decision Tree Technique to Analyze Construction Project Data, in: Information Systems, Technology and Management*; Springer: Berlin/Heidelberg, Germany, 2010; pp. 304–313.
32. Liu, H.B.; Jiao, Y.B. Application of genetic algorithm-support vector machine (ga-svm) for damage identification of bridge. *Int. J. Comput. Intell. Appl.* **2011**, *10*, 383–397. [CrossRef]
33. Mahfouz, T.; Jones, J.; Kandil, A. A machine learning approach for automated document classification: A comparison between SVM and LSA performances. *Int. J. Eng. Res. Innov.* **2010**, *2*, 53–62.
34. Dehestani, D.; Eftekhari, F.; Guo, Y.; Ling, S.; Su, S.; Nguyen, H. Online support vector machine application for model based fault detection and isolation of HVAC system. *Int. J. Mach. Learn. Comput.* **2011**, *1*, 66. [CrossRef]
35. Ur-Rahman, N.; Harding, J.A. Textual data mining for industrial knowledge management and text classification: A business oriented approach. *Expert Syst. Appl.* **2012**, *39*, 4729–4739. [CrossRef]