Advanced Engineering Informatics 41 (2019) 100917

Contents lists available at ScienceDirect

Advanced Engineering Informatics

journal homepage: www.elsevier.com/locate/aei

Full length article

Automated classification of building information modeling (BIM) case studies by BIM use based on natural language processing (NLP) and unsupervised learning

Namcheol Jung, Ghang Lee*

Department of Architecture and Architectural Engineering, Yonsei University, South Korea

A R T I C L E   I N F O

Keywords:
Building information modeling (BIM)
BIM use
Latent semantic analysis (LSA)
Latent Dirichlet allocation (LDA)
Text mining
Unsupervised learning
Natural language processing (NLP)
Automated document classification (ADC)

A B S T R A C T

This paper comparatively analyzes a method to automatically classify case studies of building information modeling (BIM) in construction projects by BIM use. It generally takes a minimum of thirty minutes to hours of collection and review and an average of four information sources to identify a project that has used BIM in a manner that is of interest. To automate and expedite the analysis tasks, this study deployed natural language processing (NLP) and commonly used unsupervised learning for text classification, namely latent semantic analysis (LSA) and latent Dirichlet allocation (LDA). The results were validated against one of representative supervised learning methods for text classification—support vector machine (SVM). When LSA and LDA detected phrases in a BIM case study that had higher similarity values to the definition of each BIM use than the threshold values, the system determined that the project had deployed BIM in the detected approach. For the classification of BIM use, the BIM uses specified by Pennsylvania State University were utilized. The approach was validated using 240 BIM case studies (512,892 features). When BIM uses were employed in a project, the project was labeled as "1"; when they were not, the project was labeled as "0." The performance was analyzed by changing parameters: namely, *document segmentation*, *feature weighting*, dimensionality reduction coefficient (*k-value*), *the number of topics*, and *the number of iterations*. LDA yielded the highest F1 score, 80.75% on average. LDA and LSA yielded high recall and low precision in most cases. Conversely, SVM yielded high precision and low recall in most cases and fluctuations in F1 scores.

1. Introduction

The number of building information modeling (BIM) projects increases rapidly [1]. Nevertheless, obtaining information about BIM projects of interest is still challenging and often takes several hours or more to search, filter, review, analyze, and extract information. For example, if someone is interested in finding projects that used BIM for "design coordination" or "clash detection" and Google them, Google returns 62,000 results for "BIM coordination" and 33,200 results for "BIM clash detection." Reviewing all these web pages and relevant documents is clearly time- consuming. Many of these sources can be scanned through once; however, based on our experience of building the Global BIM Dashboard, a knowledge-based system for BIM projects, reviewing one BIM-related document in detail takes approximately 1–6 h [2]. Contrarily, some specific information of interest is so rare that it is very challenging to find even one information source related to what has been searched for. Moreover, different information sources can describe the same project from different perspectives. Consequently, a single information source rarely provides complete information of interest. Our experience with the development of the Global BIM Dashboard database required an average of four information sources to acquire the one set of information we sought.

Quite a few studies have been conducted to automate the classification of construction documents. Caldas et al. [3] conducted a study using supervised learning algorithms to automatically categorize 4,000 construction project documents into 13 manually defined areas. Fan et al. [4] defined a local dictionary for construction to improve the performance of text classification by the construction of keywords using supervised learning. Salama and El-Gohary [5] conducted a study that performed automated compliance checking using supervised learning algorithms. Nevertheless, supervised learning has a critical disadvantage in that it takes a lot of time to manually label a training set [6].

To overcome this limitation, Yalcinkaya and Singh [7] and Lee et al.

* Corresponding author.
E-mail addresses: namcheoljung@naver.com (N. Jung), glee@yonsei.ac.kr (G. Lee).

https://doi.org/10.1016/j.aei.2019.04.007
Received 28 September 2018; Received in revised form 4 March 2019; Accepted 13 April 2019
Available online 30 April 2019

[8] deployed unsupervised learning, which did not require a manual preparation for a training set. Yalcinkaya and Singh used latent semantic analysis (LSA) to explore major topics in BIM studies [7]. Lee et al. used latent Dirichlet allocation (LDA) to explore legal disputes related to issues of construction [8].

In order to expedite the review process of BIM case studies, this study utilized natural language processing (NLP) and unsupervised learning—particularly LSA and LDA—to automatically analyze how BIM was used in a project. NLP is a field of study that automatically analyzes, understands, and generates sentences in natural language [9,10]. In NLP, the process of automatically analyzing the contents of documents and categorizing the documents according to the results of the analysis is referred to as automated document classification (ADC) [11]. Detailed reviews of ADC and other processes are provided in Section 2.

This paper is organized into six sections. The next section describes machine learning-based ADC methods and the application of ADC in construction. The third section describes the research method. One of the key challenges in deploying machine learning is to determine how to set up parameters and preprocess data to yield the best performance. The Research Method section describes the parameters and data preprocess in detail. The fourth section reports the performance of two unsupervised learning methods, i.e., the LSA- and LDA-based approaches, and validates the results against those of the support vector machine (SVM)—one of the representative supervised learning methods for text classification [12–15]. The fifth section discusses the results and concludes the paper with contributions and limitations.

2. Background

2.1. Previous studies on ADC

In 1986, Salton and McGill [11] introduced an ADC system to improve the performance of document searching. Since then, many studies have deployed ADC to retrieve information from documents. In the early 2000s, rapid improvement in computing power made it possible to apply ADC to filter spam mails, categorize topics, analyze product trends, and conduct a sentiment analysis [16,17]. Since the start of text classification, various algorithms such as Rocchio's algorithm, Naive Bayes classifier, k-nearest neighbor, decision trees, SVM, rule-based classifier, regression- based classifier, neural network, etc. are deployed [18,19].

Naive Bayes classifier categorize documents using conditional probability of appearance of features [20]. Although the Naive Bayes classifier showed worse performance than the other algorithms [14,15], it is easy-to-use algorithm [18,20], and continuously being updated [20]. K-nearest neighbors finds the most similar k documents in corpus and decide the classification result based on vote of reference documents. It is hard to decide value k when using k-nearest neighbors [18]. Decision trees is a rule-based algorithm that is suitable for structured documents [21,22]. Random forest is a combination of decision trees that considers various IF-THEN condition [23]. SVM and neural network are commonly used supervised learning methods for text classification. They can analyze documents with high dimensional features [24,25], and showed high performance in text classification [15]. The hybrid techniques that deploy more than two algorithms were used to improve the performance [18,19]. However, as discussed in the Introduction section, supervised learning has a major drawback. It is very time-consuming to generate training data for supervised learning [5,6].

On the other hand, unsupervised learning does not require a training process and thus has an advantage over supervised learning in that it can reduce the time and effort required to develop rules or generate training data. The two commonly used unsupervised learning methods in ADC are LSA and LDA. LSA represents the correlation of word-to- document, word-to-word, and document-to-document in quantitative values using singular value decomposition [26]. LDA is a clustering method that explores the topics of documents based on a generative model [27]. This study uses the LDA and LSA methods to take advantage of unsupervised learning and comparatively analyzes the performance of these two approaches against those of SVM, one of the most commonly used supervised learning methods in ADC.

Table 1
Previous studies that employed ADC in AEC.

Authors	Applied algorithms	Machine learning type	Purpose of the study
Caldas et al. [3]	Naive Bayes k-nearest neighbors Rocchio's algorithm SVM	Supervised learning	Categorize documents into pre- defined categories
Fan et al. [4]	Naive Bayes k-nearest neighbors Decision tree	Supervised learning	Text classification by construction keywords
Salama and El-Gohary [5]	Naive Bayes Maximum entropy SVM	Supervised learning	Automated compliance checking of legal issues
Yalcinkaya and Singh [7]	Latent semantic analysis	Unsupervised learning	Explore major topics of BIM studies
Lee et al. [8]	Latent Dirichlet allocation	Unsupervised learning	Explore legal dispute issues

2.2. Limitations of ADC in construction

Several studies in architecture, engineering, and construction (AEC) deployed both supervised and unsupervised learning for document classification (Table 1). Caldas et al. [3] compared the accuracy of four algorithms, Naive Bayes classifier, k-nearest neighbors (kNN), Rocchio's algorithm, and SVM. Among them, SVM yielded the highest performance. Categorizing documents by 13 topics using a relatively small number of documents (4000 construction project documents) was regarded as a significant challenge. Fan et al. [4] developed a project-specific dictionary and deployed it in supervised learning using Naive Bayes, kNN, and decision tree. This demonstrated that use of a project-specific dictionary could help improve the performance of text classification. Salama and El-Gohary [5] used Naive Bayes, maximum entropy, and SVM to automate the classification of clauses in legal documents for design-compliance checking. That study reported that the SVM-based method yielded the best performance—100% and 96% recall and precision, respectively— when it was used with the porter stemmer algorithm, odds ratio scoring function, term frequency weighting function, and 20 best features, at a 26% threshold. Yalcinkaya and Singh [7] used LSA—an unsupervised learning method—to explore patterns and trends in BIM research. Lee et al. deployed LDA—another type of unsupervised learning method—to detect potential legal disputes in construction [8].

All of the above studies demonstrated that it is challenging to collect a large number of documents for a specific topic, particularly in construction, and to manually analyze them. Another challenge is that the outcome of a document classifier may greatly vary depending on how parameters are set. Yang et al. [50] tested the performance of five major text classifiers—SVM, neural network, kNN, the Linear Least Squares Fit, and Naive Bayes—and found that the performance of a significance test was affected by the choice of performance measure, the sensitivity of the test, and the training-set frequency of categories being tested. This study was conducted carefully considering all these parameters and finding the optimal settings to achieve the best outcomes.

2.3. A general ADC process

This section briefly reviews the general ADC process, terms, and parameters. The ADC generally comprises three steps: NLP (text preprocessing, feature weighting, and feature selection), machine learning, and result analysis [19]. Text preprocessing includes tokenization, stopword removal, stemming, and the generation of a domain-specific dictionary, which is referred to as a local dictionary [10]. Tokenization is a process that segments a document into features, which are the smallest units of data for machine learning. In general, one feature corresponds to one word in a document for text classification. Stopwords are words that are unnecessary for information retrieval, such as am, are, is, and, etc. They act as noise during machine learning [28]. Thus, stopwords must be removed during text preprocessing. Further, stemming is a process that converts features into their root forms. Lovins stemmer, Porter stemmer, Paice/Husk stemmer, and Dawson stemmer are representative stemming methods. Among these, Porter stemmer is the most popular [29]. After the stemming process, a series of words that appears repeatedly must be defined as a multi-word feature in a local dictionary. For example, “building information modeling” is registered as “BIM” in a local dictionary. When multi-word features are appropriately identified, the amount of noise can be greatly reduced [30].

Feature weighting is a method that approximately measures the importance of a feature in document classification. The most popular feature weighting method is term-frequency-inverse-document-frequency (TF-IDF), which involves the multiplication of term-frequency (TF) and inverse-document-frequency (IDF) [11]. Documents are transformed into a numerical form, called the feature vector, using the vector space model (VSM) [31]. The feature vector is the TF-IDF value [10].

Feature selection reduces noise by selecting a subset of features that are relevant to pre-defined topics [32]. Previous studies have applied numerous feature selection methods. Among these, the most notable are information gain, chi- squared statistics, and the Gini index [18]. These methods measure the importance of each feature and exclude unimportant features from machine learning. Document representation is a summary table of features, feature weights, and manual classification results (labels). The document representation data comprises the dataset required for machine learning. Topics of a document are labeled as “true” if the document includes these topics, and “false” if it does not.

The performance of supervised learning is evaluated by measurements such as precision, recall, and F1 score [33]. True positive is the number of documents that are categorized as “true” when their labels are true. False negative is the number of documents that are categorized as “false” when their labels are true. False positive is the number of documents that are categorized as “true” when their labels are false. True negative is the number of documents that are categorized as “false” when their labels are false. Precision is the number of true positives divided by the number of true and false positives. Recall is the number of true positives divided by the number of true positives and false negatives. The F1 score is a multiplication of precision and recall times two divided by the sum of prevision and recall.

As briefly mentioned in the introduction section, machine learning is mainly distinguished into two groups: supervised learning and unsupervised learning [6]. Moreover, many variations of machine learning exist. Text classification using supervised learning requires training data that are set according to pre-defined topics [10]. Among various supervised learning algorithms, the SVM is representative of text classification [18].

The SVM identifies the most appropriate hyperplane that divides feature vectors labeled as true and false [14]. As the SVM is independent of the number of features and generates a linear separator, which is suitable for NLP, it is commonly used for text classification [24]. In previous studies that compared the performance of text classification methods, the SVM showed a higher F1 score than other algorithms. Yang and Liu [14] compared the performance of text classification of five supervised learning methods: SVM, k-nearest neighbor (kNN) classifier, neural network (NN), linear least-squares fit (LLSF), and naive Bayes (NB) classifier. They reported that SVM, kNN, and LLSF outperformed NN and NB [14]. In Yu and Xu’s research [15], SVM and the relevance vector machine (RVM) were superior to NN and NB when filtering spam mails. Colas and Brazdil [12] and Thota et al. [13] also showed that SVM outperformed other supervised learning methods [12,13].

Unsupervised learning methods are commonly used for text clustering. LSA and LDA are notable unsupervised learning algorithms for text clustering [26,27]. LSA is a method that conducts a quantitative analysis of the similarity among words in a corpus (a collection of words) and among documents. Further, LSA converts the VSM—the numerical representation of a corpus—into a three-component matrix using singular value decomposition: $U[m \times m]$, $\Sigma[m \times n]$, and $V^{T}[n \times n](m > n)$ [26]. LSA reduces noise and improves performance by reducing the dimensions of the matrix. After dimensionality reduction, the component matrix is converted into $U'[m \times k]$, $\Sigma'[k \times k]$, and $V^{T'}[k \times n](n > k)$, which preserve the more important and compact information of the corpus [26]. The cosine similarity of each row of $U' \times \Sigma'$ implies similarity between words, and the cosine similarity of each column of $\Sigma' \times V^{T'}$ implies similarity between documents [34]. The similarity values between words are used for searching synonyms, and the similarity values between documents are used to find similar documents. In addition, the factor analysis for text clustering deploys the factor loading values in $\Sigma \cdot V^{T}$ [7].

LDA is a topic modeling algorithm that generates probability distribution based on the frequency of words. In LDA, it is assumed that documents are written following the generative process. The generative process defines topics for a corpus, selects topic Z for a document, and finally represents the document using the most relative word W [27]. The topic distribution of corpus θ and the word distribution for topic φ follow the Dirichlet distribution: θ_m $\mathrm{Dir}(\alpha)$, φ_k $\mathrm{Dir}(\beta)$, where α and β are user-defined parameters. The LDA explores topic Z, which maximizes the likelihood function $L(W|Z) = P(Z|W = \mathrm{observedsample})$, by iterating Gibbs sampling [35]. The probability distribution of topics (θ_m) and words (φ_k), which are derived from LDA, are used for text clustering and extracting representative words for each topic [27]. LDA is similar to LSA in that it estimates the similarity among documents. The difference is that LDA measures similarity using the Hellinger distance (Eq. (1)) [36].

$$\text{Hellinger distance } (V_P, V_Q) = \frac{1}{\sqrt{2}} \cdot \sqrt{\textstyle\sum_{1}^{T} (\sqrt{p_i} - \sqrt{q_i})}$$

where V_P and V_Q are probability distributions of $P(p_1, \cdots, p_T)$ and $Q(q_1, \cdots, q_T)$

(1)

3. Research method

This study was conducted in the following sequence (Fig. 1). First, we collected and labeled BIM case studies based on the BIM use classification defined by Pennsylvania State University [37]. The labeling method is described in greater detail in the following section. Then, the collected documents were converted into a VSM through NLP—text preprocessing, feature weighting, and feature selection were completed to improve classification performance by removing stopwords and defining a local dictionary. In this study, the NLTK English stopwords list was used to remove stopwords [38,39]. Then, the preprocessed texts were analyzed using LSA and LDA to detect whether a case study contained a certain BIM use. For validation of the effectiveness of the proposed method, the SVM was employed to validate the performance of the proposed method. Each step was conducted using Python 3.5 [40] and additional libraries: NLTK 3.2.2 [41], Numpy 1.13.0 [42], scikit-learn 0.18.1 [43], and lda 1.0.5 [44].

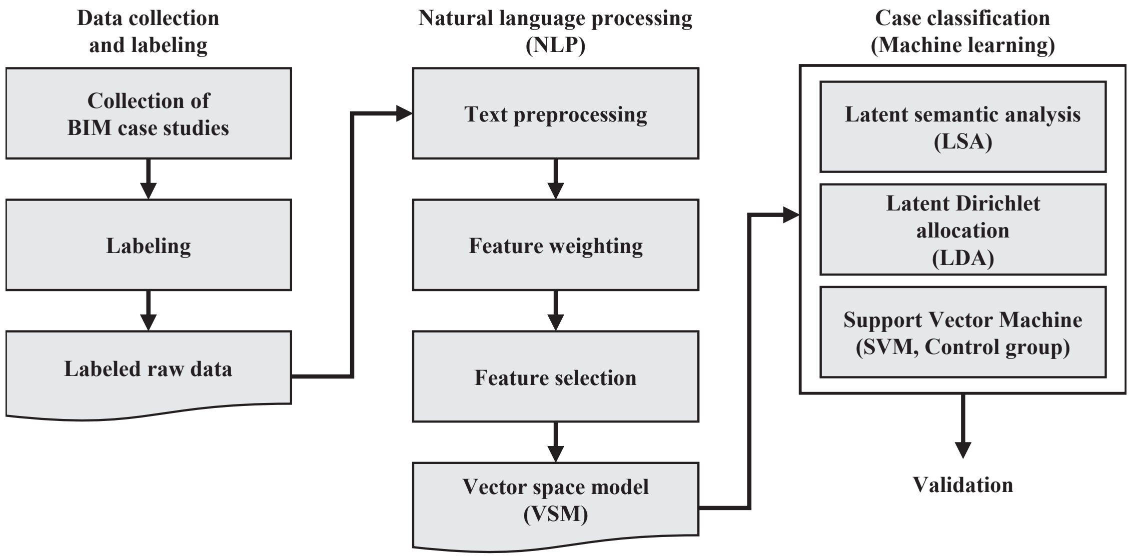
Data collection and labeling
Collection of BIM case studies
Labeling
Labeled raw data
Natural language processing (NLP)
Text preprocessing
Feature weighting
Feature selection
Vector space model (VSM)
Case classification (Machine learning)
Latent semantic analysis (LSA)
Latent Dirichlet allocation (LDA)
Support Vector Machine (SVM, Control group)
Validation

Fig. 1. Research process.

3.1. Data collection and labeling

A total of 240 BIM case studies were collected from major BIM websites, such as The BIM Hub and BIM+, and literature, such as the BIM Handbook, "The BIM" magazine, Archdaily, Tekla references, and other BIM case-study reports [45–48]. The collected BIM case studies were converted into plain text files. Non-text components, such as images, tags, hyperlinks, etc., were removed from the documents. Then, the documents were labeled according to the 25 BIM uses defined in the BIM Project Execution Planning Guide v2.1 [37].

The 25 BIM uses were re-categorized into a total of 21 BIM uses after merging and removing those uses that had a small number of cases. The four items—structural analysis, lighting analysis, mechanical analysis, and other engineering analysis—were combined into "engineering analysis" and "disaster planning," the cases for which were very rare, and were removed. Each BIM use was labeled as 1 if a document included a description of the BIM use, and as 0 if it did not. Fig. 2 depicts the number of documents for each BIM use. Among the 21 BIM uses, only 10 had over 17 cases. These 10 BIM uses included existing conditions modeling (V01), cost estimation (V02), phase planning (V03), design reviews (V04), design authoring (V05), engineering analysis (V06), 3D coordination (V07), digital fabrication (V08), record model (V09), and maintenance scheduling (V10). The labeled data were used for validation of the unsupervised learning approach as well as for training and validation data for the SVM for cross-validation.

3.2. Text preprocessing

Labeled raw texts were tokenized using space as a delimiter. Next, punctuations in the corpus were removed, and texts were stemmed using the Porter stemmer. Multi-word features that appeared over five times were registered to a local dictionary. The alphanumeric terms that represented a certain meaning (e.g., 3D and 4D) and abbreviations (e.g., BIM) were also registered to the local dictionary as "three dimensional" and "building information modeling." Then, the alphanumeric words and abbreviations in the corpus were replaced with matching terms based on the local dictionary. For example, "3D" was replaced with "three_dimensional", and "building information modeling" and "building information model" were replaced with "BIM."

3.3. Feature weighting and selection

The preprocessed texts were converted to the VSM through feature weighting and feature selection. TF and TF- IDF were used as feature weighting methods. Feature selection was conducted only for the SVM. Information gain, chi- squared statistics, and the Gini index were used as ranking methods for feature selection. The number of features per document was 313 on average, a minimum of 105, and a maximum of 1793. The total number of analyzed features was 512,892.

3.4. Parameter optimization

LSA- and LDA-based BIM-case classification algorithms were used for document categorization. As explained earlier, the BIM-use detection algorithm proposed in this study was designed to detect phrases similar to the definition of each BIM use in BIM case studies that employed LSA and LDA. When the similarity value between a phrase and the definition of BIM use exceeded a threshold value, the case study was determined to deploy BIM use. For cross-validation, the results of the LSA- and LDA-based algorithms were compared with those of the SVM.

By iterating through different settings of parameter values that affect the performance of machine learning methods, the optimal settings for detecting BIM uses in case studies were determined. The parameters used in different machine learning methods—namely, LSA, LDA, and the SVM—are depicted in Fig. 3. The value of *feature weighting* was either TF or TF-IDF. The value of *feature selection* was information gain (IG), chi-squared statistics (Chi), and the Gini index (Gini). The value of *number of iterations* was set to 20%, 40%, 60%, 80%, and 100% of the total number of features in the corpus. The *number of features* for *feature selection* was defined as a ratio rather than an absolute number, because the total *number of features* is subject to change when documents are added to or removed from the corpus. *Feature weighting, feature selection*, and the *number of features* were the parameters required to generate a VSM, particularly for the SVM; *feature weighting* was also used in LSA.

The remaining parameters were required for LSA and LDA. Some sentences spanned the length of a paragraph, while others were very short. To reduce the impact of the length of a sentence on the LSA and LDA results, a document was divided into several segments (phrases) depending on whether a sentence exceeded a certain threshold value. *Document segmentation* represents the threshold value that divides a sentence. The *document segmentation* values were set at 70, 140, 210, 280, 350, and all features (no division). The baseline was set to 70

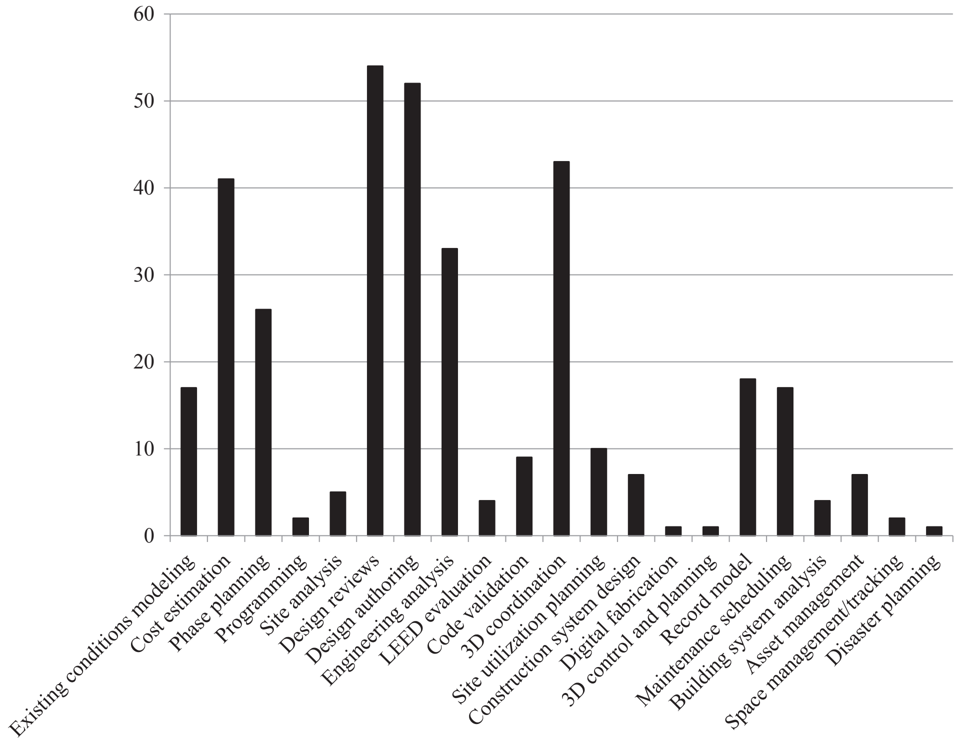

Fig. 2. The number of documents for each BIM use.

features, as the length of the typical paragraph was 100–200 words; and when stopwords were excluded from a paragraph, the *number of features* became approximately 70. *Threshold* represents the threshold value for determining the similarity among sentences in LSA and LDA. The *threshold* values ranged from −0.9 to 0.9 at an interval of 0.1. Further, *k-value* represents the dimensionality reduction value used in LSA, which was set to 10, 30, 50, and 70. The value of *number of topics* was set to 7, 14, and 21. The *number of iterations* of Gibbs sampling was set to 200, 600, and 1000 times in LDA.

Fig. 3 represents the overall process of NLP and machine learning with the parameters used. LSA was performed 9120 times using six types of *document segmentation*, two types of *feature weighting*, four *k-values*, and 19 *threshold* values as variables. LDA was performed 10,260 times with the variables *document segmentation*, *number of topics*, *number of iterations*, and *threshold*. The SVM categorized documents and validated the result 300 times using *feature weighting*, *feature selection*, and *number of features* as variables. The optimized variable values for the best performance of the proposed methods are presented in Sections 4.1–4.3.

- LSA:
$$(6\ \ document\ \ segmentation) \times (2\ \ feature\ \ weighting) \times (4k - value) \times (10\ \ BIM\ \ use) \times (19\ \ threshold) = 9120\ \ \text{processes done}$$

- LDA:
$$(6\ \ document\ \ segmentation) \times (3\ \ number\ \ of\ \ topics) \times (3\ \ number\ \ of\ \ iterations) \times (10\ \ BIM\ \ use) \times (19\ \ threshold) = 10,260\ \ \text{Process done}$$

- SVM $$(2\ \ feature\ \ weighting) \times (3\ \ feature\ \ selection) \times (5\ \ number\ \ of\ \ features) \times (10\ \ BIM\ \ use) = 300\ \ \text{Processes done.}$$

The main concept of the methods proposed in this study is analyzing documents to classify them by pre-defined categories without training data. Unsupervised learning was deployed to throw off the yoke of training data. To detect texts that describe a specific BIM use, we estimated similarity value of the definition of BIM use and a part of document. When the similarity value, which was calculated by unsupervised learning, was higher than *threshold*, the document was judged that contents about the specific BIM use were contained.

LSA measures the similarity values between phrases using the cosine similarity. To acquire cosine similarity values, a document was divided on the basis of the *document segmentation* value first, and weight was added to each feature using TF and TF-IDF. Since the total *number of features* was 5283, and the total number of documents was 240, matrix A had a dimension of $[240 \times 5283]$. Matrix A was converted to $U[5283 \times 5283]$, $\Sigma[5283 \times 240]$, and $V^T[240 \times 240]$ through the SVD. Then, U, Σ, and V^T were reduced to $U'[5283 \times kv]$, $\Sigma'[kv \times kv]$, and $V^{T'}[kv \times 240]$ by *k-value*. The cosine similarity of each column of the matrix multiplication of Σ' and $V^{T'}$ was the value that represented the similarity among phrases.

LDA measures the similarity values between documents using the Hellinger distance (Eq. (1)). As in LSA, documents were divided according to *document segmentation*. LDA explores the optimized result by changing the *number of topics* and *number of iterations*. Since the Hellinger distance ranges from 0 to 1 and is close to 0 when documents are similar, we used the H-similarity for the validation of the performance to use the same threshold values (*threshold*) as those of LSA (Eq. (2)).

$$\mathrm{H-Similarity}(V_P,\ \ V_Q) = 1 - (2 \times \mathrm{Hellinger\ \ distance}),$$
$$\mathrm{where}\ \ V_P,\ \ V_Q\ \ \mathrm{are\ \ probability\ \ distributions\ \ of}\ \ P(p_1,\cdots,p_T),\ \ Q(q_1,\cdots,q_T) \tag{2}$$

3.5. Validation methods

The performance of the proposed methods was analyzed from two perspectives: time and the confusion matrix. The main motivation for this study was to use ADC to reduce the time taken to analyze BIM uses in case studies. It is predictable that ADC is much faster than manual classification. Nevertheless, the time required to automatically analyze

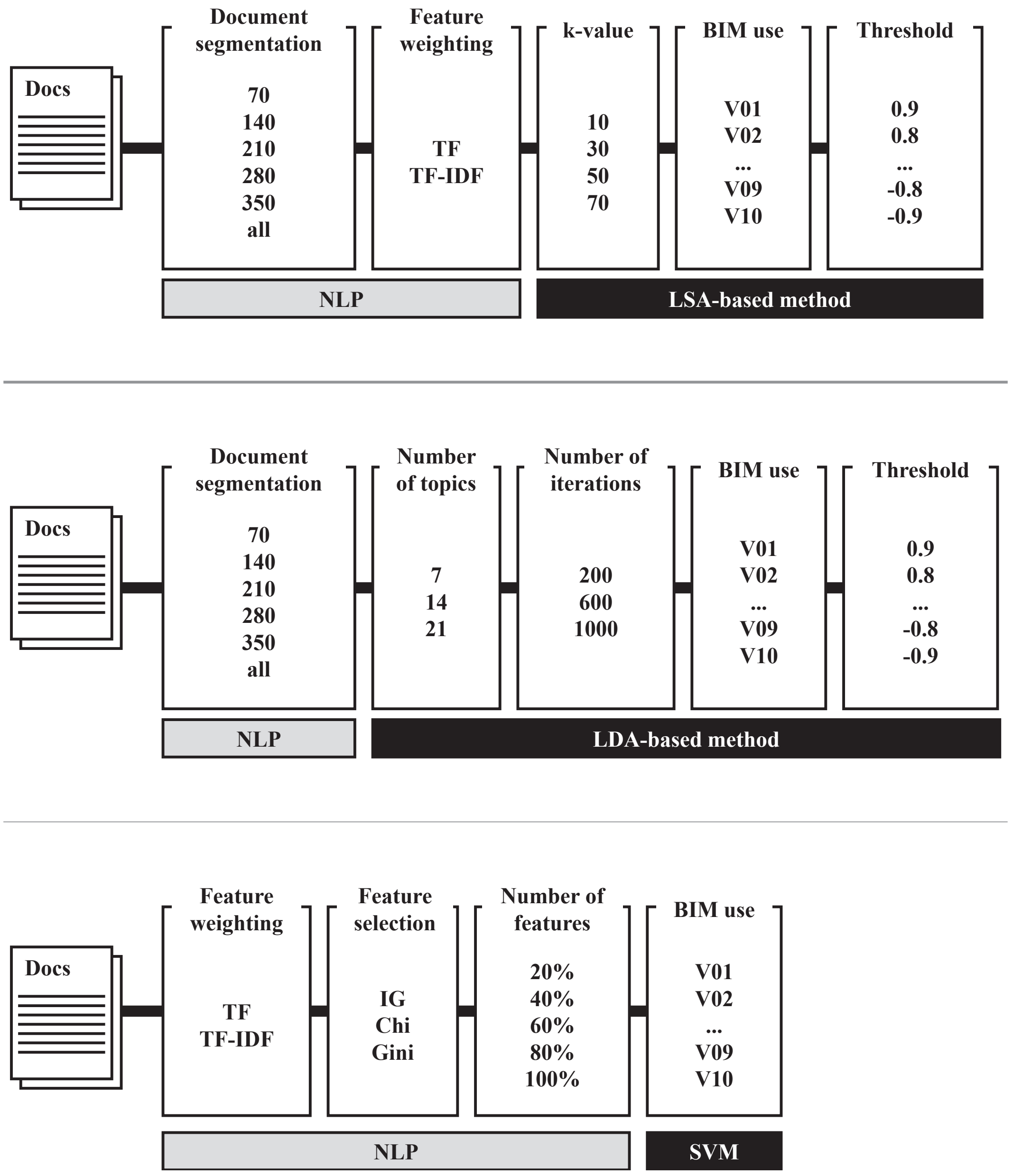

Fig. 3. Parameters used in natural language processing and machine learning methods.

Table 2
Optimized variables and performance of the LSA.

Topics	Parameters				Performance		
	Document segmentation	*Feature weighting*	*k-value*	*Thresholds*	Precision	Recall	F1 score
V01	350	TF	10	0.3	74.86%	89.12%	81.37%
V02	280	TF	30	0.2	62.07%	97.96%	75.99%
V03	210	TF	70	0.2	67.50%	91.84%	77.81%
V04	140	TF	10	0.4	66.19%	94.56%	77.87%
V05	140	TF	10	0.2	61.76%	100.00%	76.36%
V06	70	TF	10	0.5	66.51%	94.56%	78.09%
V07	210	TF	30	0.3	73.94%	94.56%	82.99%
V08	70	TF	70	0.1	67.63%	95.24%	79.10%
V09	140	TF	10	0.4	71.74%	89.80%	79.76%
V10	All	TF	10	0.2	67.15%	94.56%	78.53%

a BIM case was measured to understand the exact magnitude of time reduction.

Even if the analysis time is reduced, if the proposed method cannot produce accurate analysis results, the method is not useful. BIM uses in the 240 case studies utilized in this study were manually analyzed by researchers with over two years of experience in BIM research. By using the manually analyzed BIM uses as the ground truth, the performance of the LSA-, LDA-, and SVM-based methods were compared using the confusion matrix (precision, recall, and the F1 score). The statistical significance among the results was tested using Fisher's exact test. If this test yielded no statistical significance, the simple matching coefficient test—a qualitative analysis method—was used.

4. Results and validation

This section discusses the performance of the proposed methods. In terms of analysis time, as expected, ADC was much faster than manual document classification. The results of the confusion matrix analysis and the optimized variable values are described in detail in the following subsections.

4.1. The performance of proposed classification methods

4.1.1. LSA-based classification method

The optimized variables and performance of the LSA are represented in Table 2. The average F1 score was 78.79%, and the standard deviation was 2.14%. Among the two *feature weightings*, TF yielded better performance. Further, *threshold* yielded the best results when its value ranged from 0.1 to 0.5. The optimized values of *document segmentation* and *k-value* varied according to BIM use. The LSA-based approach performed best when documents were segmented by a certain length.

4.1.2. LDA-based classification method

The optimized variables and performance of the LDA are represented in Table 3. The average F1 score was 80.75%, and the standard deviation was 4.47%. The optimized values of *document segmentation*, *number of topics*, and *number of iterations* for the best results varied according to BIM use. The results showed that the documents that included a small *number of topics*—such as V01, V02, V03, V05, and V07—yielded better results than the documents with a higher *number of topics*, such as V04, V06, V07, V08, V09, and V10. Naturally, the *number of iterations* for the documents with a large number of topics was much higher than that of the documents with a small number of topics. The optimized values of *threshold* ranged from −0.8 to −0.3.

4.2. The performance of SVM

The optimized variables and performance of the SVM are represented in Table 4. Among the two *feature weighting* methods, TF-IDF

Table 4
Optimized variables and performance of the SVM.

Topics	Parameters			Performance		
	Feature weighting	*Feature selection*	*Number of features*	Precision	Recall	F1 score
V01	TF-IDF	IG	20%	100.00%	14.29%	25.00%
V02	TF-IDF	IG	40%	100.00%	31.97%	48.45%
V03	TF-IDF	Chi	20%	100.00%	24.49%	39.34%
V04	TF-IDF	IG	60%	100.00%	39.46%	56.59%
V05	TF-IDF	IG	60%	100.00%	40.82%	57.97%
V06	TF-IDF	IG	40%	100.00%	26.53%	41.94%
V07	TF-IDF	Gini	20%	100.00%	30.61%	46.87%
V08	TF-IDF	Gini	20%	100.00%	24.49%	39.34%
V09	TF-IDF	IG	40%	100.00%	22.45%	36.67%
V10	TF-IDF	Chi	20%	100.00%	14.97%	26.04%

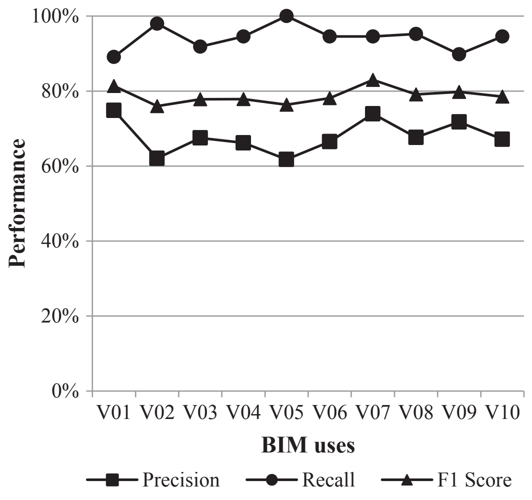

Fig. 4. Performance of LSA-based classification method.

showed better performance than TF. In most situations, the SVM yielded high precision and low recall. The average precision was 100%; average recall was 27.01%; and average F1 score was 41.82%. Further, the standard deviation of the F1 score was 11.15%. In summary, the SVM had a poorer performance than LSA- and LDA-based approaches; however, the SVM, which is more sensitive to the number of samples, might perform better with a larger sample size [5].

4.3. Comparison of the LSA-, LDA-, and SVM-based approaches

In summary, the overall performance of the LSA- and LDA-based algorithms was better than that of the SVM in terms of precision, recall, and F1 score. Figs. 4–7 depicts the performance of the LSA-, LDA-, and

Table 3
Optimized variables and performance of the LDA.

Topics	Parameters				Performance		
	Document segmentation	*Feature weighting*	*k-value*	*Threshold*	Precision	Recall	F1 score
V01	140	7	200	−0.3	80.39%	89.12%	84.52%
V02	70	7	200	−0.4	78.86%	93.88%	85.71%
V03	All	7	200	−0.7	61.51%	100.00%	76.17%
V04	140	21	1000	−0.6	77.22%	94.56%	85.02%
V05	70	7	200	−0.3	80.12%	90.47%	84.98%
V06	210	21	1000	−0.6	77.40%	93.20%	84.57%
V07	350	7	600	−0.8	61.51%	100.00%	76.17%
V08	70	21	1000	−0.7	61.51%	100.00%	76.17%
V09	210	21	600	−0.5	68.39%	89.80%	77.65%
V10	140	14	600	−0.6	62.03%	100.00%	76.56%

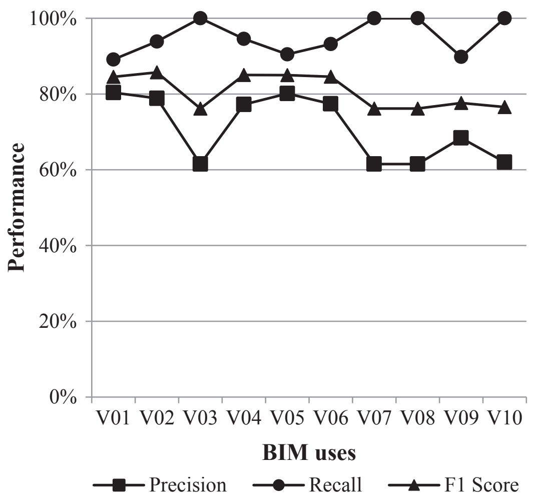

Fig. 5. Performance of LDA-based classification method.

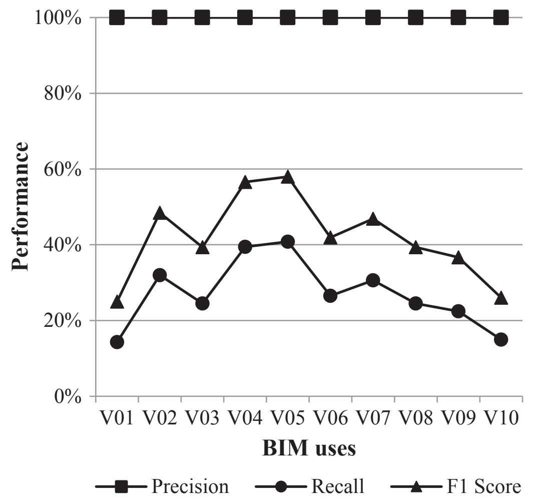

Fig. 6. Performance of SVM.

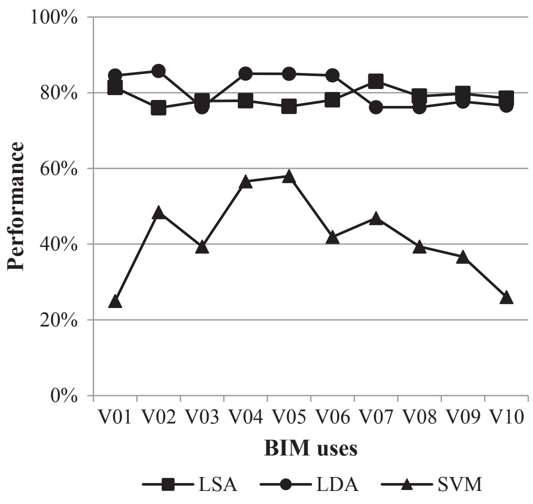

Fig. 7. Comparison of performance of classification method.

SVM-based approaches. Although the precision of the SVM was 100%, the SVM failed to categorize BIM case studies by BIM uses because the method labeled almost every topic of the documents as 0. On the other hand, unsupervised learning-based methods derived relatively high recall and low precision. Future research could study how to reduce the number of false positives. There were considerable differences in the performance of LSA and LDA, depending on the combination of variable values. Nevertheless, no obvious pattern was observed, which makes one method superior to the others by changing the combination of variables.

To further investigate whether there were significant differences between the categorization results of LSA and LDA by a combination of variables—*document segmentation*, *feature weighting*, *k-value*, *number of topics*, and *number of iteration*—using Fisher's exact test and the simple matching coefficient analysis, it was determined that the variables proposed in the study affected the results of categorization. Although, as evident in Figs. 4–6, the categorization algorithm based on LSA and LDA showed better performance than the SVM, it is important to select appropriate values for variables to improve the performance of the algorithms. Moreover, the optimized combination of variables must be updated as data are added.

4.3.1. Differences due to document segmentation methods

Categorizing documents by V01 using LSA without *document segmentation* showed better performance than other values of *document segmentation*. On the other hand, categorizing documents using LSA by V06 showed the best performance, using 70 as a value for *document segmentation*. For V02, V03, V06, V08, V09, and V10, categorizing documents using LSA showed significant differences, but it was difficult to find a common pattern for each categorization. There were no significant differences among results according to the *document segmentation* for V05 and V07. In LDA, categorizing documents by V01, V02, V04, V05, V06, and V09 showed significant differences according to the *document segmentation*, but not by V03, V07, V08, and V10.

4.3.2. Differences due to feature weighting

The *feature weighting* methods, TF and TF-IDF, caused significant differences between categorization results when LSA was conducted. For categorizing documents by V02, V03, V04, V06, V07, and V10, TF performed better than TF-IDF; however, for categorizing documents by V01, V05, V08, and V09, both TF and TF-IDF had the same performance.

4.3.3. Differences due to the dimensionality reduction coefficient (k-value)

In LSA, the dimensionality reduction coefficient, *k-value*, affected the categorization performance. The optimized value for V02 was 30, while that for V06 was 10. The performance for V09 lagged when 70 was selected as a value of *k-value*. The F1 scores for V03 and V04 by each *k-value* were the same, but there were significant differences in the distribution of labels. There were no significant differences among the results according to the *k-value* for V01, V05, V07, and V08.

4.3.4. Differences due to the number of topics

In LDA, the value of the *number of topics* for V01 and V05 that showed the best performance was 7, while that for V04 and V09 was 21. There were no significant differences among the results according to the *number of topics* for V02, V03, V06, V07, and V10.

4.3.5. Differences due to the number of iterations

In LDA, the value of the *number of iterations* for V02 and V05 that showed the best performance was 200, while that for V04 and V06 was 1000. The categorization performance for V06 showed the best performance when the *number of iterations* was 600. There were no significant differences among the results according to the *number of topics* for V03, V07, V08, and V10.

5. Conclusion

It could take hours to find a BIM case study that deployed BIM in the manner in which one is interested. To reduce classification time, this study proposed an automated BIM case study categorization method by BIM use based on NLP and unsupervised learning—LSA and LDA. The proposed methods measured the similarity values between the definition of each BIM use and phrases in case study documents. In terms of the classification time, the proposed methods took an average of 7.5 s to detect a certain BIM use from documents comprising 105 to 1793 features (i.e., preprocessed terms). In terms of performance, the LDA-based method yielded the highest F1 score—64.53%, 96.19%, and 77.18% precision, recall, and F1 score— when LDA was used with 70 features for *document segmentation*, 600 for the *number of iterations*, 7 for number of topics, and −0.6 of threshold. The performance of the proposed methods was better than that of the SVM, which is a representative supervised learning method. This implies that the proposed methods can also reduce the time and effort required to obtain training data. The precision, recall, and F1 score yielded by the SVM for 10 BIM uses were 100%, 27.01%, and 41.82%, respectively. Despite the very high precision, fluctuation in recall resulted in a standard deviation of 11.15% in the F1 score.

The main contribution of this study is that it opens up the possibility of automated detection of BIM use from BIM case studies that employ unsupervised learning, using a method that does not require manual data training and labeling. Machine learning is not a mere "black-box" operation. It requires a significant creative thinking process to design a research method and trials-and-errors to identify a way to preprocess data to improve performance. This study proposes a method to automatically categorize BIM case studies by measuring and comparing the similarities between the definitions of BIM uses and phrases using unsupervised learning methods. This method, including the detailed preprocessing methods proposed in this study, may be improved by future studies. This study's second contribution is that it clearly demonstrated several advantages of the proposed method over the traditional manual BIM project classification. The automated classification of BIM case studies will also guarantee the objectivity, consistency, and repeatability of the categorization of BIM case studies. Moreover, the proposed methods have an advantage in that they employ a very simple approach for comparing the definition of an instance of BIM use against phrases in BIM case studies to detect BIM use in a project. All these attributes can contribute to facilitating the development of a BIM knowledge base from scattered information sources and eventually enable a rapid search of BIM information. The third contribution of this study is that it identified the variables and their optimal settings for the best performance of the proposed LSA- and LDA-based methods, although there is room for improvement. This can help future researchers reduce the time required to investigate the optimal range of variable values. In addition, the local dictionary created for this study could be re-used for other BIM-related studies.

This study has certain limitations. Although it took over two years to collect and analyze 240 BIM cases, and the analyses were based on 512,892 features in 240 cases, machine learning can always benefit from a larger set of data, particularly when the number of dependent variables (BIM uses in this study) is large. For example, the deployment of certain BIM uses defined by Pennsylvania State University, such as disaster planning, have rarely been reported. Such rarely deployed BIM uses had to be excluded from the analysis in this study due to the lack of data. Further, performance could be improved by strengthening the definition of each BIM use, the local dictionary, and a thesaurus. For example, the current algorithm removes words that denote a negative meaning (not, no) during the stopword removal phase. However, such terms may affect the accurate identification of BIM uses [33,49]. The addition of a thesaurus that includes synonyms may also help to improve performance. Another limitation is that the optimal settings for variables change as a corpus changes when new documents are added to the database. All these aspects require further research.

Acknowledgments

This research was supported by a grant (19AUDP-B127891-03) from the Architecture & Urban Development Research Program funded by the Ministry of Land, Infrastructure and Transport of the Korean government.

Appendix A. Supplementary material

Supplementary data to this article can be found online at https://doi.org/10.1016/j.aei.2019.04.007.

References

[1] R. Sacks, C. Eastman, G. Lee, P. Teicholz, The future: building with BIM, BIM Handbook: A Guide to Building Information Modeling for Owners, Designers, Engineers, Contractors, and Facility Managers, Wiley, Hoboken, New Jersey, 2018 9781119287568.
[2] G. Lee, K. Lee, W. Jung, K. Hwang, S. Han, N. Jung, H. Lee, M. Kim, Introduction to the global BIM dashboard, Proc. 16th Int. C. Comp. Civ. Build. Eng., Osaka, Japan, (2016).
[3] C.H. Caldas, L. Soibelman, J. Han, Automated classification of construction project documents, J. Comp. Civ. Eng. 16 (4) (2002) 234–243.
[4] H. Fan, F. Xue, H. Li, Project-based as-needed information retrieval from unstructured AEC documents, J. Manage. Eng. 31 (1) (2015) A4014012.
[5] D.M. Salama, N.M. El-Gohary, Semantic text classification for supporting automated compliance checking in construction, J. Comp. Civ. Eng. 30 (1) (2016) 04014106.
[6] O. Chapelle, B. Scholkopf, A. Zien, Semi-Supervised Learning, The MIT Press, Cambridge, Massachusetts, United States, 2006 9780262033589.
[7] M. Yalcinkaya, V. Singh, Patterns and trends in Building Information Modeling (BIM) research: a latent semantic analysis, Automat. Constr. 59 (2015) (2015) 68–80.
[8] J.-H. Lee, J.-S. Yi, J. Son, Unstructured construction data analytics using R programming – focused on overseas construction adjudication cases, J. Arch. Inst. Korea Struct. Constr. 32 (5) (2016) 37–44.
[9] C.D. Manning, P. Raghavan, H. Schütze, Introduction to Information Retrieval, Cambridge University Press, Cambridge, United Kingdom, 2008 9780521865715.
[10] S.M. Weiss, N. Indurkhya, T. Zhang, Fundamentals of Predictive Text Mining, second ed., Springer, London, United Kingdom, 2015 9781447167501.
[11] G. Salton, M.J. McGill, Introduction to Modern Information Retrieval, McGraw-Hill, New York City, New York, United States, 1986 9780070544840.
[12] F. Colas, P. Brazdil, Comparison of SVM and some older classification algorithms in text classification tasks, Proc. IFIP Int. C. Artif. Intell., Santiago, Chile, (2006).
[13] H. Thota, R.N. Miriyala, S.P. Akula, K.M. Rao, C.S. Vellanki, A.A. Rao, S. Gedela, Performance comparative in classification algorithms using real datasets, J. Comp. Sci. Syst. Biol. 2 (1) (2009) 97–100.
[14] Y. Yang, X. Liu, A re-examination of text categorization methods, Proc. 22nd Ann. ACM SIGIR C. Res. Dev. Info. Retrieval, Berkeley, California, Unite States, (1999).
[15] B. Yu, Z.-B. Xu, A comparative study for content-based dynamic spam classification using four machine learning algorithms, Knowl.-Based Syst. 21 (4) (2008) 355–362.
[16] C. Lipizzi, D.G. Dessavre, L. Iandoli, J.E. Ramirez Marquez, Towards computational discourse analysis: a methodology for mining Twitter backchanneling conversations, Comp. Hum. Behav. 64 (2016) (2016) 782–792.
[17] A.N. Srivastava, M. Sahami, Text Mining: Classification, Clustering, and Applications, CRC Press, Boca Raton, Florida, United States, 2009 9781420059403.
[18] A. Khan, B. Baharudin, L.H. Lee, K. Khan, A review of machine learning algorithms for text-documents classification, J. Adv. Info. Tech. 1 (1) (2010) 4–20.
[19] M.M. Mirończuk, J. Protasiewicz, A recent overview of the state-of-the-art elements of text classification, Expert Sys. Appl. 106 (2018) (2018) 36–54.
[20] J.D. Rennie, L. Shih, J. Teevan, D.R. Karger, Tackling the poor assumptions of naive bayes text classifiers, 20th Int. C. Mach. Learn. Washington D.C., United States, 3 2003, pp. 616–623.
[21] C. Apté, F. Damerau, S.M. Weiss, Towards language independent automated learning of text categorization models, Proc. 17h Ann. Int. ACM-SIGIR C. Res. Dev. Inform. Retrieval, Dublin, Ireland, (1994).
[22] M. Uhm, G. Lee, Y. Park, S. Kim, J. Jung, J. Lee, Requirements for computational rule checking of requests for proposals (RFPs) for building designs in South Korea, Adv. Eng. Inf. 29 (3) (2015) 602–615.
[23] L. Breiman, Random forests, Mach. Learn. 45 (1) (2001) 5–32.
[24] T. Joachims, Text categorization with support vector machines: learning with many relevant features, Proc. 10th Eur. C. Mach. Learn., Chemnitz, Germany, (1998).
[25] P. Myllymaki, H. Tirri, Bayesian case-based reasoning with neural networks, Proc. IEEE International Conference on Neural Networks, San Francisco, United States, (1993).
[26] T.K. Landauer, P.W. Foltz, D. Laham, An introduction to latent semantic analysis, Discourse Process. 25 (2–3) (1998) 259–284.
[27] D.M. Blei, A.Y. Ng, M.I. Jordan, Latent Dirichlet allocation, J. Mach. Learn. Res. 3 (Jan) (2003) 993–1022.
[28] S. Vijayarani, M.J. Ilamathi, M. Nithya, Preprocessing techniques for text mining – an overview, Int. J. Comp. Sci. Comm. Net. 5 (1) (2015) 7–16.
[29] A.G. Jivani, A comparative study of stemming algorithms, Int. J. Comp. Tech. Appl.

2 (6) (2011) 1930–1938.

[30] A.J.P. Tixier, M.R. Hallowell, B. Rajagopalan, D. Bowman, Automated content analysis for construction safety: a natural language processing system to extract precursors and outcomes from unstructured injury reports, Automat. Constr. 62 (2016) (2016) 45–56.

[31] G. Salton, A. Wong, C.-S. Yang, A vector space model for automatic indexing, Commun. ACM 18 (11) (1975) 613–620.

[32] G. Chandrashekar, F. Sahin, A survey on feature selection methods, Comp. Elec. Eng. 40 (1) (2014) 16–28.

[33] C.J. van Rijsbergen, Information Retrieval, second ed., Butterworth-Heinemann, Oxford, United Kingdom, 1979 9780408709293.

[34] C.D. Manning, H. Schütze, Foundations of Statistical Natural Language Processing, The MIT Press, Cambridge, Massachusetts, United States, 1999 9780262133609.

[35] T.L. Griffiths, M. Steyvers, Finding scientific topics, Natl. Acad. Sci. 101 (suppl 1) (2004) 5228–5235.

[36] V. Rus, N. Niraula, R. Banjade, Similarity measures based on latent Dirichlet allocation, Proc. 14th Int. C. Intell. Text Process. Comp. Ling., Samos, Greece, (2013).

[37] J. Messner, C. Anumba, C. Dubler, S. Goodman, C. Kasprzak, R. Kreider, R. Leicht, C. Saluja, N. Zikic, BIM Project Execution Planning Guide, The Computer Integrated Construction Research Program (Pennsylvania State University), State College, Pennsylvania, United States, 2010.

[38] S. Bird, E. Loper, NLTK: the natural language toolkit, Proc. ACL 2004 Interact. Poster Demonstrat. Session, Barcelona, Spain, (2004).

[39] NLTK Project, Accessing text corpora and lexical resources, http://www.nltk.org/book/ch02.html (last access: August 22, 2017).

[40] Python Software Foundation, Welcome to Python.org, https://www.python.org/ (last access: August 22, 2017).

[41] NLTK Project, Natural language toolkit – NLTK 3.2.4 documentation, http://www.nltk.org/ (last access: August 22, 2017).

[42] NumPy developers, NumPy – NumPy, http://www.numpy.org/ (last access: August 22, 2017).

[43] scikit-learn developers, scikit-learn: machine learning in Python – scikit-learn 0.19.0 documentation, http://scikit-learn.org/stable/ (last access: August 22, 2017).

[44] Python Software Foundation, lda·PyPI, https://pypi.org/project/lda/ (last access: August 22, 2017).

[45] Allplan GmbH, bim+ a service by Allplan | the open BIM platform, https://www.bimplus.net/ (last access: January 4, 2017).

[46] buildingSMART Korea, The BIM, Issue 1–15, buildingSMART Korea, Seoul, Korea, 2008–2016.

[47] C. Eastman, P. Teicholz, R. Sacks, K. Liston, B.I.M. Handbook, A Guide to Building Information Modeling for Owners, Managers, Designers, Engineers, and Contractors, second ed., John Wiley & Sons, New York City, New York, United States, 2011 9780470541371.

[48] The BIM Hub Ltd, A trusted environment for BIM Professionals | The BIM Hub, https://thebimhub.com/ (last access: January 4, 2017).

[49] H. Saif, M. Fernández, Y. He, H. Alani, On stopwords, filtering and data sparsity for sentiment analysis of twitter, Proc. 19th Int. C. Lang. Res. Eval., Reykjavik, Iceland, (2014).

[50] Y. Yang, X. Liu, A re-examination of text categorization methods, in: Proc. 22nd Ann. ACM SIGIR C. Res. Dev. Info. Retrieval, Berkeley, California, United States, 1999. http://people.csail.mit.edu/jim/temp/yang.pdf.