applied sciences

MDPI

Article

Automated Extraction and Time-Cost Prediction of Contractual Reporting Requirements in Construction Using Natural Language Processing and Simulation

Parinaz Jafari, Malak Al Hattab, Emad Mohamed and Simaan AbouRizk *

5-080 NREF, Department of Civil and Environmental Engineering, University of Alberta, Edmonton, AB T6G 2W2, Canada; parinaz@ualberta.ca (P.J.); elhattab@ualberta.ca (M.A.H.); ehmohame@ualberta.ca (E.M.)

* Correspondence: abourizk@ualberta.ca

Featured Application: The approach rapidly extracts reporting requirements from construction contracts and predicts overhead costs and durations associated with report preparation. Application of the approach is anticipated to provide the insights necessary to enhance contract negotiations, reporting workflow processes, and submittal procedures between clients and contractors.

Abstract: Due to a lack of suitable methods, extraction of reporting requirements from lengthy construction contracts is often completed manually. Because of this, the time and costs associated with completing reporting requirements are often informally approximated, resulting in underestimations. Without a clear understanding of requirements, contractors are prevented from implementing improvements to reporting workflows prior to project execution. This study developed an automated reporting requirement identification and time–cost prediction framework to overcome this challenge. Reporting requirements are extracted using Natural Language Processing (NLP) and Machine Learning (ML), and stochastic simulations are used to predict overhead costs and durations associated with report preparation. Functionality and validity of the framework were demonstrated using real contracts, and an accuracy of over 95% was observed. This framework provides a tool to rapidly and efficiently retrieve requirements and quantify the time and costs associated with reporting, in turn providing necessary insights to streamline reporting workflows.

Keywords: construction reports; construction contracts; natural language processing; machine learning; simulation modeling

Citation: Jafari, P.; Al Hattab, M.; Mohamed, E.; AbouRizk, S. Automated Extraction and Time-Cost Prediction of Contractual Reporting Requirements in Construction Using Natural Language Processing and Simulation. *Appl. Sci.* **2021**, *11*, 6188. https://doi.org/10.3390/app11136188

Academic Editor: Mariusz Szóstak

Received: 17 June 2021
Accepted: 1 July 2021
Published: 3 July 2021

1. Introduction

Work within the construction industry is allocated through construction contracts [1], which include information such as instructions, definitions, supporting statements, and contractual requirements that detail the standards and project specifications of the client [2]. A core component of construction contracts is reporting and information requirements, which require contractors to periodically submit various reports detailing different aspects of project progress to the client [3]. As construction projects and contracts are becoming increasingly complex, clients are demanding that contractors provide more information and reports on different project aspects [4,5]. Reporting has quickly become a laborious procedure, with construction personnel spending as much as 40% of their time gathering field data, organizing and analyzing data, preparing reports, and verifying report accuracy [6].

Although a large administrative burden for contractors [7], the time and resources needed to complete reporting requirements—as well as the precise reporting requirements themselves—are often unknown and unaccounted for during the preliminary planning stages of a project. An integral component of project success, preliminary planning involves, among other activities, the selection of the project management team and the creation of the

project documentation system. Consideration of specific reporting requirements during the preliminary planning stage ensures that (1) an adequate number of personnel is available to complete reporting requirements on time and within budget, (2) efficient project reporting systems are implemented, and (3) redundant and/or overlapping reporting requirements are addressed prior to the execution phase of a project. Contract documentation, however, remains an immature area of practice, with the identification of reporting requirements involving the manual reading, interpretation, and analysis of hundreds of unstructured textual contract pages to differentiate between statements related to requirements and other unimportant texts, such as instructions and definitions. Due to the time and effort involved, the manual extraction of reporting requirements is often not completed during the preliminary planning stages of a project, with project managers informally approximating reporting costs and resource requirements. Indeed, it has been reported that office-related processes, such as project reporting, continue to suffer from low reliability, where planned durations are often underestimated [8].

The poor estimation of project reporting costs and resource requirements during the preliminary planning stages of construction can result in a number of challenges for contractors [4,9–11]. For example, an inadequate number of available project management personnel may result in project reports that are submitted late or with errors. In the case of certain types of contracts (e.g., cost-plus), reporting costs that exceed the preliminary estimate can result in disputes between the contractor and client. Furthermore, a lack of understanding of the reporting requirements in the preliminary stages of a project may prevent contractors from increasing reporting efficiency during the construction phase through the consolidation of redundant reporting requirements or by optimizing the composition of the project management team. As such, the ability to quickly, accurately, and automatically extract reporting requirements and predict associated costs is expected to have a notable impact on project performance [12]. Although text-mining techniques, such as information extraction, text classification, and other predictive analytics, have been used by researchers to develop requirement extraction models [4,5,13], existing models are not fully automated, do not provide high requirement extraction accuracy, and lack a cost-and-time prediction component. Methods capable of automatically extracting contractual reporting requirements and predicting the time and costs associated with report preparation, therefore, remain relatively unexplored.

To address this challenge, this study has developed a framework capable of (1) automating the identification and extraction of reporting requirements and (2) predicting and analyzing the overhead costs and durations associated with report preparation. The framework employs Natural Language Processing (NLP) and Machine Learning (ML) techniques to automate the extraction of reporting requirements, and uses stochastic simulation to predict the durations and costs associated with report preparation using historical project data. Real contractual documents from an actual case study were used to (1) develop and refine the reporting requirement extraction module and (2) demonstrate the functionality and validity of the complete framework. This framework provides practitioners and researchers with an automated tool to more efficiently identify reporting requirements and quantify the time and costs associated with report preparation. Practical application of this approach is anticipated to provide decision makers with the insights necessary to enhance contract negotiations, reporting workflow processes, and submittal procedures between clients and contractors, in turn increasing value for all project stakeholders.

2. Research Background

2.1. Construction Reporting

Many problems in the construction industry involve communication and reporting procedures, with ineffective reporting systems leading to poor project management [11,14,15]. Construction reports, therefore, are often required by clients as a means of monitoring project progress, estimating production rates, and resolving disputes and claims [6]. Project reporting involves the collection and structuring of large volumes of site data from nu-

merous field management activities by many site personnel on a frequent—even daily—basis [16,17]. Given the amount of preparation work required together with the frequency of submittals, reporting has become a time- and effort-intensive procedure that can result in notable increases in overhead costs of the project.

Various construction field management tools have been developed to establish project reporting systems tailored to the needs of contractors, while ensuring the reporting requirements of projects are met [2,6,16]. For example, Russell [18] developed a daily construction project management system that rapidly reports and shares site information and project progress status between project participants. Similarly, Shiau and Wang [19] developed a construction management information system consisting of daily reports as well as cost management and design-change management modules to compile daily site management information. El-Omari and Moselhi [3] proposed a model to facilitate automated data acquisition from construction sites by deploying an information technology platform. Their goal was to integrate automated data acquisition technologies to collect required data for progress measurement purposes to support efficient time–cost tracking and control of construction projects [3]. Following the same line of work, Lee et al. [16] proposed an approach to automatically generate daily reports from text messages exchanged through a commonly used text messaging system.

It is important to note that the aforementioned models were primarily focused on effective data acquisition, information flow, and communication to facilitate the monitoring of site work, incurred costs, and potential challenges [3,20]. Although these studies have addressed the downstream aspect of reporting, they have been developed with the assumption that reporting requirements are already defined and known in advance. In practice, however, reporting requirements for complex types of construction, such as oil and gas or infrastructure projects, often differ between projects and from contract-to-contract, making the time, resources, and costs associated with report preparation difficult to approximate. Methods for automating the extraction of contractual reporting requirements or the estimation of time and cost implications associated with reporting, however, remain relatively unexplored.

Importantly, the lack of research literature in the area of contract documentation is not indicative of the practical importance of this issue. Discussions with experienced professionals at a construction company in Alberta, Canada, revealed that contractors are very interested in techniques that can support the extraction, management, and time–cost prediction of reporting requirements. Once considered an obligatory and static activity, contractors are beginning to explore methods capable of enhancing the planning, and therefore efficiency and cost, of project reporting—particularly for complex types of construction where contracts are often specific to each individual project.

2.2. NLP Applications in Construction

To avoid unnecessary changes, rework, and potential claims, contractors must thoroughly analyze construction contracts and specifications to ensure that client requirements are identified, managed, and fulfilled [13]. The traditional approach of identifying reporting requirements involves the manual reading, interpretation, and analysis of hundreds of unstructured textual contract pages to differentiate between statements related to requirements and other extraneous text (e.g., instructions and definitions). Techniques capable of accelerating the reporting requirement extraction process, therefore, represent a key prerequisite for the development of an automated time–cost prediction model.

Natural Language Processing (NLP) is an area of Artificial Intelligence (AI) that focuses on the development of techniques to analyze, process, and extract information from natural human language. Applications include machine translation, speech recognition, and automated content analysis [21]. In construction, a large number of project documents are generated in text format [22]. The use of NLP techniques to organize and improve access to information contained in these types of documents is becoming ever more essential for effective construction management [7], with NLP techniques being increasingly applied

in construction research [22–24]. Caldas et al. [7], for instance, employed NLP techniques to automate the classification of construction documents to improve organization of and access to information within interorganizational systems. Al Qady and Kandil [25] also developed an automated classification system of construction documents according to their semantic relationships. Fan and Li [23] used NLP for the automatic retrieval of similar cases from an electronic case repository of construction accidents.

Text classification, a subfield of NLP, is an automated process for classifying text into categories [26,27]. Text classification is divided into rule-based and Machine Learning (ML)-based methods [26]. Rule-based text classification categorizes text using a manually defined pattern to create rules; in contrast, under ML-based text classification, a machine learns how to classify text on its own using data. A variety of text classification models have been developed for the construction domain. For example, Salama and El-Gohary [28] developed a hybrid semantic, multilabel ML-based text classification algorithm for classifying clauses and subclauses of general conditions to support automated compliance checking. Lee et al. [5] proposed a rule-based model to automatically detect risk-related sentences of contracts to support contract risk management for construction contractors. Zhong et al. [29] combined NLP and convolutional neural networks to develop a classification model capable of automatically classifying accident narratives to support safety management on site. Zhou and El-Gohary [30] proposed an ontology-based, multilabel text classification approach for classifying environmental regulatory clauses to support automated compliance checking in construction, and Hassan and Le [4] proposed a domain-specific classification model to identify client requirements from construction contracts. It is important to note that the implementation of existing text classifiers to different applications remains limited, as text classification models, text features, and performance requirements vary greatly across domains and applications [28]. Designed for a specific domain, the aforementioned text classifiers and are not well-suited for applications that require alternate classification structures.

2.3. Research Gap

Despite these advancements, research focused on enhancing the management of contractual reporting requirements remains relatively unexplored and fragmented. Most of the studies in the area of construction reporting have focused on the development of systems that improve data acquisition, information flow, and communication. While other studies, such as those mentioned previously, have deployed NLP and AI to automate information retrieval and extraction from construction contracts, the text approaches used to develop these requirement extraction models are limited by a lack of full automation, low extraction accuracy, and the absence of a cost–time prediction component [4,5,13]. Indeed, a review of construction literature could not identify any established study capable of automatically extracting reporting requirement statements from hundreds of pages of contractual and project specification documents.

3. Framework Overview

To address the gap existing in literature, this study developed a novel, NLP-based framework for the automated extraction and time–cost prediction of contractual reporting requirements in construction. The framework consists of two modules, namely the (1) automatic extraction of reporting requirements module and (2) prediction of reporting time and cost module that are linked as illustrated in Figure 1.

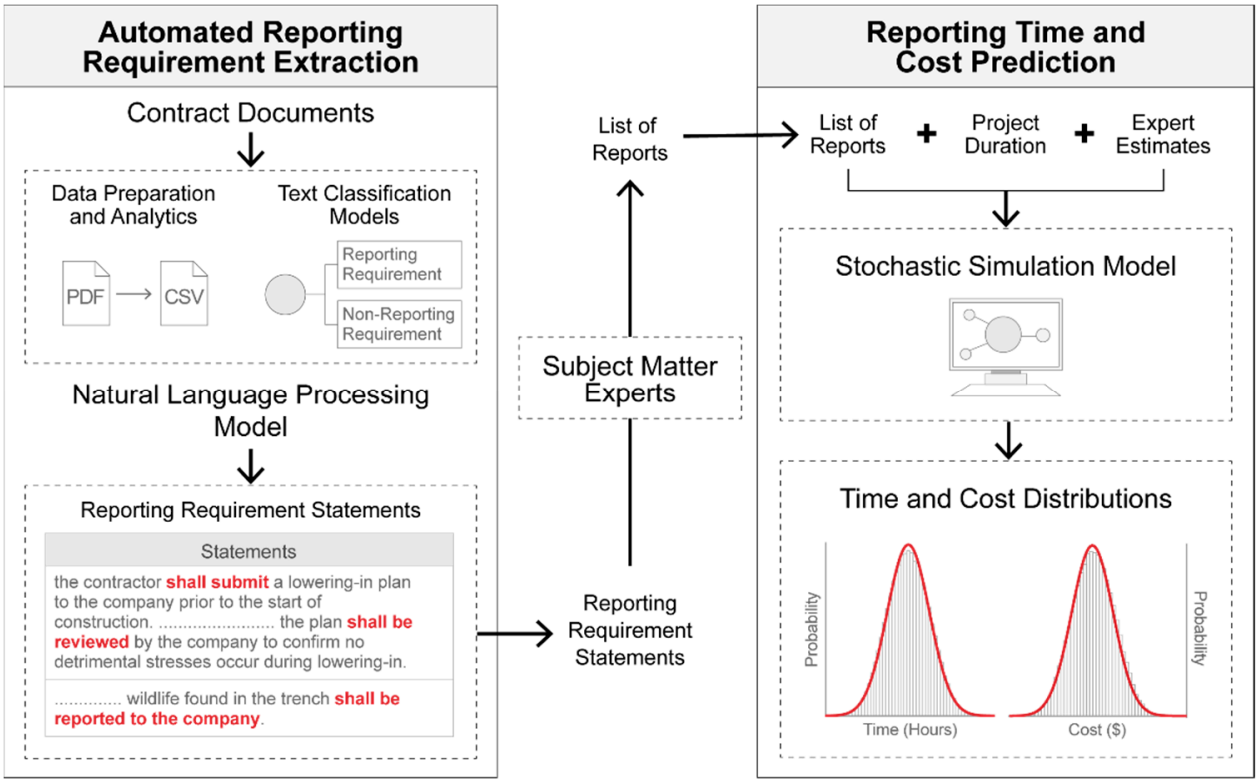

Figure 1. Proposed Framework.

The first module, hereafter referred to as the extraction module, is responsible for identifying statements in contract or project specification documents that prescribe reporting requirements. Contract documents and project specifications are input into the NLP model. Relevant text is extracted from the documents and is transformed into a format that is compatible with the text classification models.

Text classification algorithms are then used to classify contractual and project specification statements as either a (1) reporting requirement or (2) non-reporting statement. Both rule-based and ML-based text classification methods can be used to classify statements; application of either method will depend on the specific requirements of the user. While rule-based text classification is more time-consuming than ML-based classification due to the involvement of manual rule development, rule-based classification commonly results in higher precision and recall [26]. ML, on the other hand, makes it possible to automatically classify text, provided that sufficient learning opportunities are available [27].

The second module, hereafter referred to as the prediction module, is responsible for generating relevant time–cost predictions. Reporting requirements output from the extraction module are used by practitioners to prepare a list of required reports and their associated submittal frequencies that are then input into the prediction module. Estimates of the time required to prepare a specific report are used as inputs. Then, a Monte Carlo simulation model, which uses random sampling to obtain numerical results or a probability distribution [31], is used to predict the cost associated with report preparation based on project duration and historical data.

Although distinct, the practical functionality of these two modules increases considerably when used together. Manual extraction is very tedious and time-consuming, and contractors do not have enough time during the bidding stage to identify the contract requirements and plan accordingly. Because of the ability of the extraction module to quickly extract reporting requirements, the prediction module can now be applied in a more impactful stage of the project life-cycle (i.e., pre-construction bidding and planning stages). Specifically, the outputs of the prediction module can be used to (1) ensure that

sufficient cost and time contingencies for report preparation are included in bid estimates, (2) engage in negotiations to reduce redundant reporting requirements before finalizing contracts, and (3) improve resource allocation.

4. Extraction Module

Development of the extraction module was completed in three main steps, namely (1) data preparation and pre-processing, (2) development and training of text classification models, and (3) evaluation of model performance. Ten contractual and specification documents of an oil-and-gas project were supplied by a private Canadian construction contracting firm and were used to develop the extraction module. Python [32], an open-source programming language, was used to automate module development steps.

4.1. Data Preparation and Preprocessing

Data preparation and pre-processing transformed raw data into a labeled dataset that was used to develop and train the text classification models. This involved the (1) extraction of textual data from documents, (2) manual assignment of labels to extracted statements, and (3) cleaning of labeled data. An overview of the data preparation process is illustrated in Figure 2.

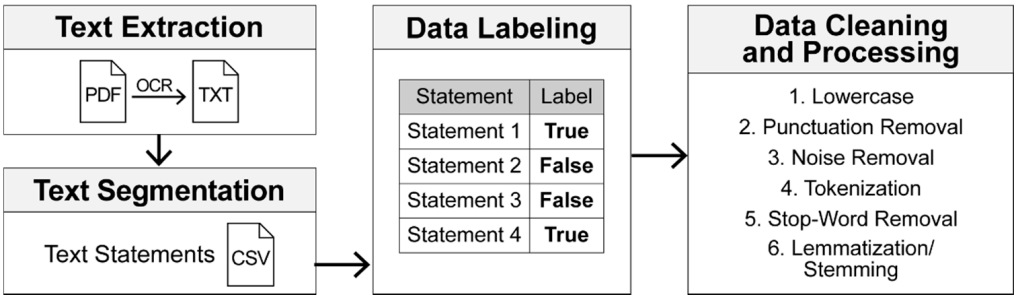

Figure 2. Data preparation and pre-processing during extraction module development.

Documents, which were provided in an imaged portable document format (i.e., .pdf), were first converted into a standard, processable text format (i.e., .txt) using Optical Character Recognition (OCR). Next, text documents were automatically segmented into individual text statements using document formatting. A total of 8943 text segments were extracted from 10 contractual and project specification documents. Since pre-labeled textual construction datasets are not as readily available as other domain applications, such as movie reviews or twitter messages [33], extracted statements were labeled manually. A label of "true" was manually applied to statements prescribing a reporting requirement, while a label of "false" was applied to non-reporting statements. A sample of the labeled dataset is presented in Figure 3.

Data were then structured into a single, comma-separated values (.csv) file, with text statements stored in the first column and the document name and associated page stored in subsequent columns. The last column contained the pair label (i.e., type: true/false) of each statement. The labeled dataset was validated by domain experts to ensure accuracy of the manual labels. The final dataset used in the study included 340 reporting requirements and 8603 non-reporting statements.

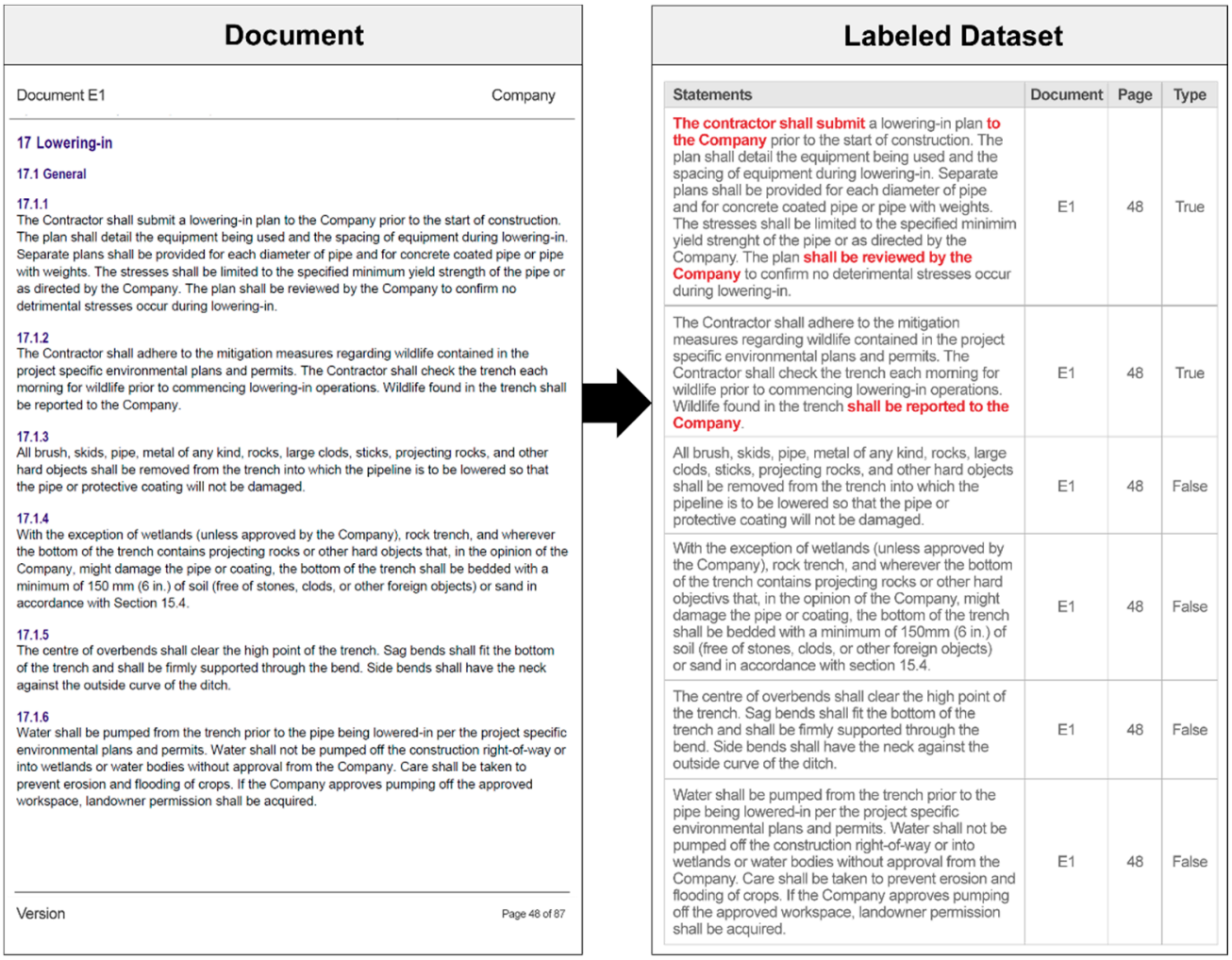

Figure 3. Sample contract document (left) and labeled dataset (right).

The final dataset was then cleaned to reduce data noise and enhance the quality of data used to train the text classification models. First, text was converted to a lowercase form to ensure identical words were treated as like terms (e.g., "Submit" and "SUBMIT"). Then, punctuation was removed from the text. In addition to text in the main body of the document, OCR extracted text from footers, page numbers, headers, annotations, and footnotes. This text acted as data noise for the text classification algorithms and was, therefore, removed. Then, tokenization was used to divide text statements into words (i.e., tokens) and to convert text into a feature vector form to prepare text for feature engineering and further analysis [34]. Stop-words, which are frequent words such as conjunctions, prepositions, and pronouns (e.g., the, for, so, is, of, and a) that do not carry relevant information for text classification, were removed using a standard English stop-word list [22]. Lemmatization and stemming were applied to reduce the number of features through word grouping. While word stemming groups words by removing prefixes and/or suffixes to conflate words to their original root [35], lemmatization groups words subsequent to a full morphological analysis. Once data cleaning was completed, the dataset was randomly split into training (80%) and testing (20%) sets, which were used to train the text classifiers and to evaluate classifier performance, respectively.

4.2. Rule-Based Classification

In the rule-based classification approach, a set of hand-coded "IF-THEN" rules that define the label assignment criteria for a certain category were prepared [5]. These rules

were iteratively constructed and refined to improve accuracy of the classifier. The process used to develop the rule-based classification model is depicted in Figure 4.

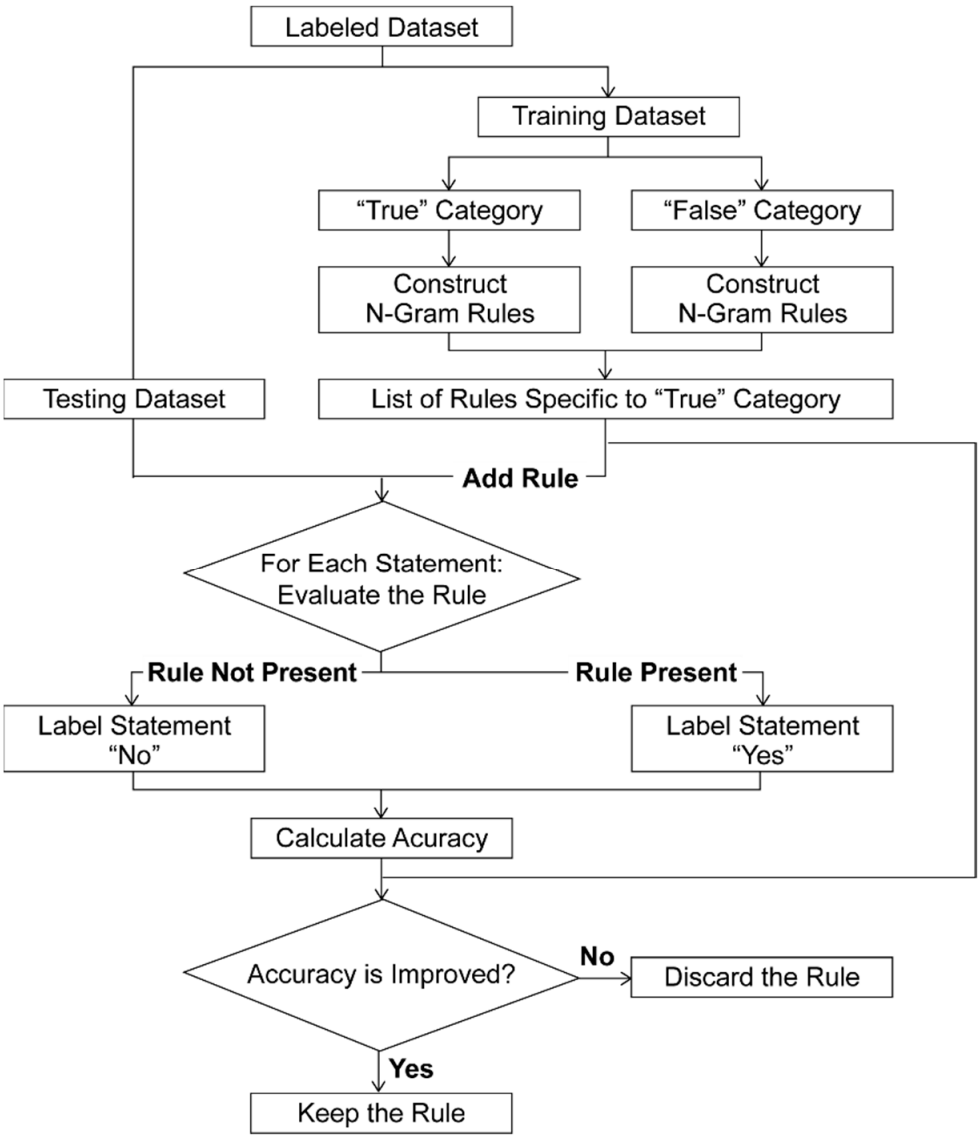

Figure 4. Rule-based classification.

Accurate filtering of reporting requirements from contractual documents requires the development of robust and comprehensive rules. Although keywords, such as "report" and "submit," may be helpful in identifying certain reporting requirements, construction contracts also contain key phrases, such as "shall be reported/submitted," which indicate that a report or deliverable must be provided contractually. It is important to note that keywords alone cannot distinguish a reporting requirement from any other contractual requirement. As such, critical phrases were extracted using text analytics. Using the training dataset, the rule-based model was used to extract n-grams (i.e., a sequence of co-occurring words as a single token) from the textual statements. The most common n-grams (i.e., phrases) appearing in the reporting requirement statements are summarized in Figure 5.

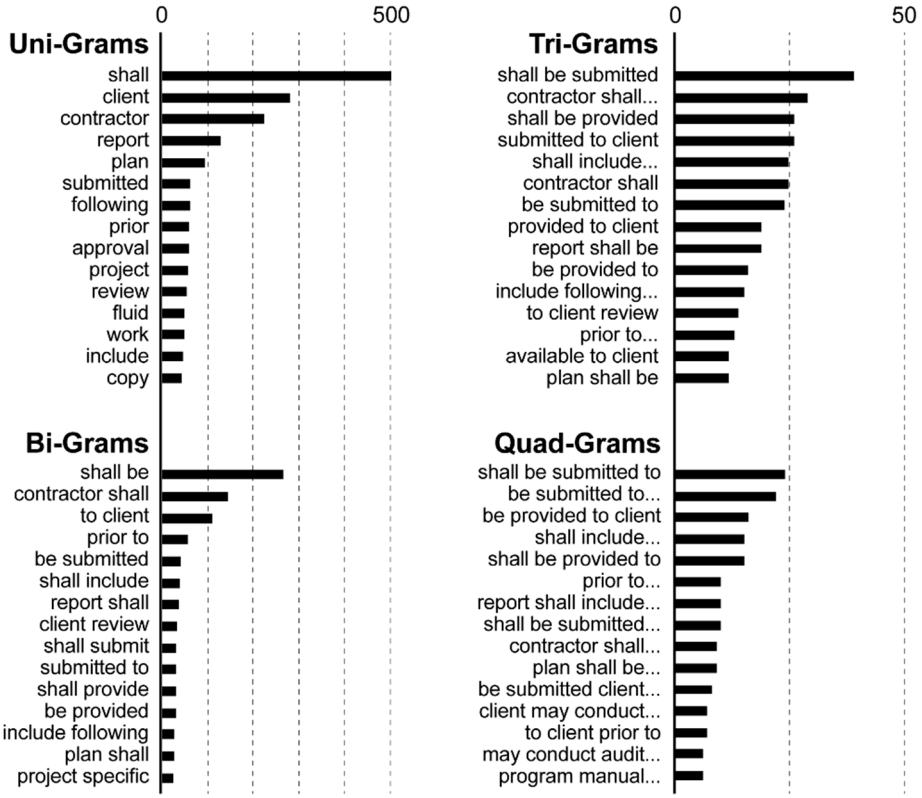

Figure 5. Most common n-grams in reporting requirement statements.

To avoid errors, a list of n-grams specific only to the "true" category was prepared. N-grams common to both the "true" and "false" categories were removed. The final rules consisted of four different sets of n-grams capable of discerning between "true" and "false" statements, namely uni-grams, bi-grams, tri-grams, and quad-grams representing single-, two-, three-, or four-word phrases, respectively. These four sets of n-grams were developed to evaluate the effect of each n-gram set on the performance of the rule-based text classifier. N-grams were flagged as rules, with each rule consisting of a pattern and a predicted category. N-grams in each N-gram set were closely monitored, and rules for each statement were evaluated. Each rule was added, one-by-one, to the text classification model. Predicted labels were then compared to actual labels, and classifier performance was calculated. If the performance (i.e., accuracy) increased with the addition of the rule, the rule was retained. If not, the rule was removed. This was repeated for each rule of each n-gram until a final list was created.

4.2.1. Performance of Rule-Based Classification Models

Performance of the text classification models were evaluated using accuracy, precision, recall, and F1-score, which were calculated using Equation (1) through Equation (4), respectively.

$$\text{Accuracy} = \frac{TP + TN}{TP + FP + TN + FN} \tag{1}$$

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

$$\text{Precision} = \frac{TP}{TP + FP} \tag{3}$$

$$\text{F1} - \text{Score} = \frac{2 \times Precision \times Recall}{Precision + Recall} \tag{4}$$

where *TP* are true positives (i.e., statements correctly labeled "true"), *FP* are false positives (i.e., incorrectly labeled "true"), *TN* are true negatives (i.e., correctly labeled "false"), and *FN* are false negatives (i.e., incorrectly labeled "false").

Accuracy (Equation (1)) is defined as the percentage of correctly classified statements over the total number of statements in the testing set. Recall (Equation (2)), is defined as the percentage of true positives identified by the model. Precision (Equation (3)) is defined as the percentage of positives that are correctly labeled [36]. Finally, the F1-score (Equation (4)), combines precision and recall to provide an overall assessment of model effectiveness.

Recall is considered to be the most critical performance metric in the context of requirement extraction, where the extraction of all reporting requirements is the primary objective. For instance, a model may have low performance accuracy because it results in a larger number of false positives (i.e., non-reporting statements labeled as requirements). However, the model may have high recall results (i.e., 100%) if it is able to correctly label all reporting requirements as "true".

Specific rules for text processing were developed and applied to improve results of the rule-based classification model. Initial tests were conducted on different n-gram sets. The testing approach was conducted in an iterative manner, and results from 24 different combinations of n-grams and text pre-processing techniques (e.g., stop-word removal, lemmatization, etc.) were compared. The four sets of n-grams extracted from the training set are summarized in Table 1. The total number of rules generated increased with the number of n-grams (Table 1). Using the process summarized in Figure 4, the number of rules maintained for each n-gram was considerably reduced for all n-gram sets (Table 1). For example, of the 2363 rules generated for the bi-grams set, only 38 rules were retained. Accuracy was increased from 97%, using uni-grams, to 99%, using bi-grams, yet was decreased to 98% and 97% using tri- and quad-grams, respectively. Although differences between n-gram sets were minimal, optimal accuracy was achieved using the retained bi-gram rules. The impact of adding the first 10 and the last bi-gram rule on model accuracy is visualized in Figure 6.

Table 1. Number of generated and retained bi-gram rules and associated accuracy.

N-Gram Set	Number of Generated Rules	Number of Retained Rules	Maximum Accuracy (%)
Uni-Grams	118	8	97
Bi-Grams	2363	38	99
Tri-Grams	3762	38	98
Quad-Grams	4268	34	97

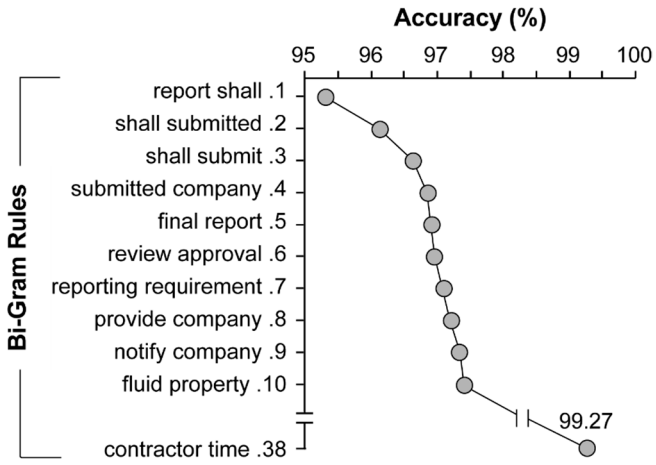

Figure 6. Impact of adding bi-gram rules on the accuracy of the rule-based text classifier.

The first bi-gram rule, "report shall", resulted in an accuracy of 95.3%. The third bi-gram, "shall submit", further increased the accuracy of the classifier to 96.7%. The 37 rules added after "shall submit" collectively increased performance by 3.97% to 99.3%.

The impact of stop-word removal, lemmatization, and stemming on the performance of text classification models is known to differ based on the textual context and application. As such, the impact of stop-word removal and lemmatization/stemming were evaluated. Various experimental scenarios examining the impact of n-gram sets, stop-word removal, and lemmatization on model performance are summarized in Table 2.

Table 2. Effect of n-gram sets and data pre-processing on performance of rule-based classification for two experimental scenarios.

		Scenario 1: without Lemmatization Stop-Words Retained			Scenario 2: with Lemmatization Stop-Words Removed		
N-Gram Set	**Class Label**	**Precision (%)**	**Recall (%)**	**F1-Score (%)**	**Precision (%)**	**Recall (%)**	**F1-Score (%)**
Uni	True	91	56	69	93	55	69
	False	98	100	99	98	100	99
Bi	True	100	86	92	96	88	92
	False	99	100	100	99	100	100
Tri	True	99	86	92	98	71	83
	False	99	100	100	98	100	99
Quad	True	100	74	85	100	49	49
	False	99	100	99	97	100	99

Uni-grams had the lowest performance for both experimental scenarios (Table 2), and bi-grams demonstrated the highest performance in all three metrics in the base condition (Scenario 1).

When stop-words were removed and lemmatization was applied, bi-grams had the highest recall and F1-score, with precision only differing marginally from other n-grams. Interestingly, lemmatization and stop-word removal resulted in a 2% increase in the recall of the bi-gram classifier, while the recall of the other n-gram sets decreased (Table 2). Notably, the F1-score of tri-grams (without lemmatization and with stop-words retained) was equal to the F1-score of bi-grams (with lemmatization and with stop-words removed). This result is expected as, in some cases, removing stop-words from tri-grams transforms them into bi-grams. For example, when the stop-word "be" is removed from the tri-gram "shall be submitted", the bi-gram "shall submitted" results. Given the importance of the recall measurement when extracting reporting requirements, and based on the findings that bi-grams resulted in the highest model accuracy (Table 1) and recall (Table 2), bi-grams are selected as the optimal classifier for rule-based text classification.

4.3. Machine Learning-Based Classification

In contrast to rule-based classification, the alternate classification approach used in the present study was supervised ML models, with the learning process driven by previous knowledge of the data [28]. In ML algorithms, a general inductive process automatically builds a classification model for each class by observing the characteristics of a set of manually classified statements. The ML-based text classification approach is summarized in Figure 7.

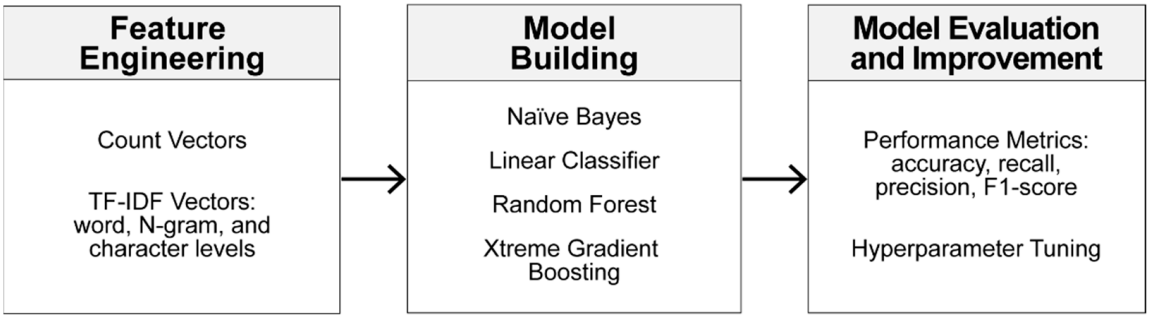

Figure 7. Machine learning-based text classification method for extraction module development.

To ensure compatibility with computer processors, words were first converted into a numeric format using feature engineering. Here, raw text data were transformed into feature vectors, and new features were created using the dataset. Different methods were used to create relevant dataset features prior to input into the text classification algorithm [37].

Two approaches for constructing representation vectors, namely count vectors and term frequency-inverse document frequency (TF-IDF) vectors, were implemented. Count vectors are a matrix representation of the dataset, where every row represents a statement, every column represents a word, and every cell represents the frequency count of a particular word (i.e., either zero or a real number) in a particular statement [38]. Words that appear in many textual statements are considered less meaningful and, therefore, each vector component (i.e., a word) can be weighed based on the number of statements in which the word appears. Another approach for constructing representation vectors is TF-IDF, which is a technique designed to identify important terms in a dataset by weighing a term's frequency (TF) together with its inverse document frequency (IDF), which weighs down high-frequency domain-specific terms while scaling up rare terms [38]. In TF-IDF vectors, terms can be extended to include characters and n-gram-level models, such as uni-gram (i.e., words), bi-grams (i.e., pairs of words), as well as tri and quad-grams (i.e., phrases). The TF-IDF of terms are calculated using Equation (5) [38],

$$\text{TF} - \text{IDF} = \frac{tf_i}{T} \times \left(1 + log\left(\frac{N}{N_i}\right)\right) \quad (5)$$

where tf_i is the frequency of term i in the statement, T is the total number of words in the statement, N is the total number of statements, and N_i is the number of statements containing term i.

For the proposed method to be feasible in practice, retraining and prediction [39] must be completed within a relatively short period of time. As such, models that required longer than an hour to be fine-tuned and retrained (e.g., deep learning algorithms) were excluded from this study to ensure applicability of this research. Based on this criterion, four popular supervised ML algorithms, which have been shown to perform differently depending on the application and domain [4,28], were implemented to build the ML-based text classification model. Characteristics of the ML algorithms are summarized as follows:

Naïve Bayes (NB) is a simple algorithm, based on the Naïve Bayes Theorem, that is used for solving practical domain problems including text classification [40]. Because it assumes that every feature is conditionally independent of other features for a given class label, the computational cost of applying the NB algorithm is comparatively low.

Logistic Regression (LR) is a linear statistical ML algorithm that correlates discrete categorical dependent features with a set of target variables [40]. It is a complex form of linear regression that can predict data probability for predefined categories.

Random Forest (RF) is a supervised ML method based on ensemble learning that involves the construction of multiple decision trees during training. Outputs are classes that are averaged or voted the most by individual trees [41]. Decision Tree algorithms,

such as the RF classifier, are often used to combat imbalanced classes, such as the scenario described here, where the number of non-reporting statements considerably exceeds the number of reporting requirements.

Extreme Gradient Boosting (XGBOOST) is a scalable ML system based on gradient boosting [42]. It generates a strong classifier by iteratively updating parameters of the former classifier to decrease the gradient of loss function. XGBOOST has superior performance in supervised ML, with high accuracy and low risk of overfitting.

4.3.1. Performance of ML-Based Classification Models

The final step in the development of the extraction module was the evaluation of the various ML-based text classification models. The performance of ML-based text classification algorithms is highly dependent on feature selection (i.e., domain dependent), type of ML techniques, and training datasets [28]. Therefore, all possible combinations of text pre-processing, feature engineering, and ML algorithms—resulting in 160 exhaustive combinations—were evaluated. Various conditions of text pre-processing approaches, such as stop-word retention or removal with or without the implementation of stemming and/or lemmatization, were tested to evaluate the effect on model performance. While the methodology was conducted in an iterative manner to allow for the detailed comparison of results, only a subset of the results is presented to maintain brevity.

The effect of using lemmatization or stemming is illustrated in Figure 8. Stemming improved classification accuracy of LR and XGBOOST algorithms, while lemmatization marginally improved the accuracy of NB and RF algorithms. Notably, the difference in classification performance accuracy between the two text pre-processing techniques was negligible, ranging from 0.03% to 0.4% (Figure 8). The XGBOOST algorithm with stemming applied resulted in the highest accuracy (98.4%).

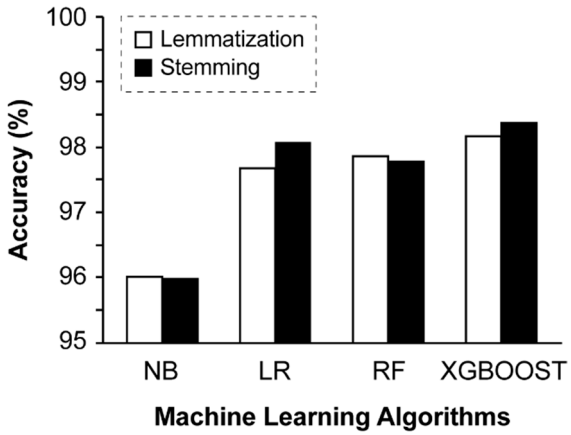

Figure 8. Impact of lemmatization and stemming on performance of Naïve Bayes (NB), Logistic Regression (LR), Random Forest (RF), and Extreme Gradient Boosting (XGBOOST)-based machine learning algorithms.

The recall, precision, and F1-score of the different ML algorithms were evaluated (Figure 9). Given that XGBOOST was found to have the highest accuracy with stemming, stemming was applied to all ML techniques prior to performance metric evaluation. The ML algorithms exhibited relatively similar recall values of over 98% for non-reporting statements (i.e., "false").

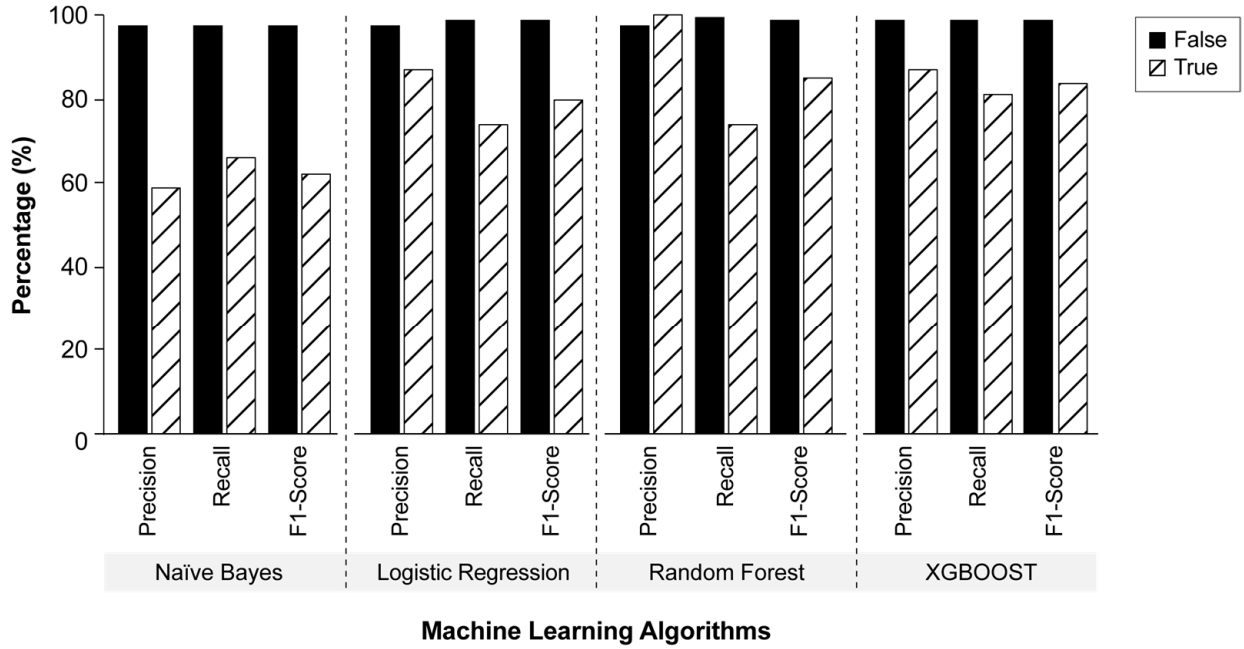

Figure 9. Performance measures of machine learning algorithms using uni-gram text classifications and word stemming.

In contrast, recall values for reporting requirements (i.e., "true") varied considerably amongst the various ML algorithms. The NB algorithm resulted in the lowest "true" recall value (66%), whereas LR, RF, and XGBOOST algorithms resulted in "true" recall values of 74%, 74%, and 81%, respectively. The lower recall values and increased variability observed for the "true" class is likely due to the imbalanced distribution of statements in the contractual documents used (340 "true" requirements versus 8603 "false" statements). In terms of precision, RF, LR, and XGBOOST resulted in "true" precision results of 100%, 87%, and 87%, respectively. The XGBOOST algorithm exhibited the highest recall (Figure 9) and accuracy (Figure 8) results for both the "true" and "false" classes and the second-highest F1-score and precision measurements.

Variations in recall when using different n-gram sets for both the "true" and "false" class were evaluated and illustrated in Figure 10. As discussed previously, higher recall values were observed for non-reporting statements (i.e., "false" class). Uni-grams resulted in higher "true" recall values compared to bi-grams for all classification algorithms except the NB algorithm. The combined use of uni-grams and bi-grams with the LR and XGBOOST classification models yielded the highest "true" recall performance, with values of 77% and 87%, respectively.

It is important to note, however, that a number of factors, such as dataset size, can influence the performance of ML models. The hyperparameters of the ML models, therefore, must be tuned to specific data [43]. Here, hyperparameters were objectively changed, one-by-one, to mitigate overfitting and improve classifier performance. After identifying optimal hyperparameters (i.e., a single set of well-performing hyperparameters), the model was retrained with the full training dataset, and the testing dataset was re-evaluated.

The two models that were examined were the RF and XGBOOST algorithms, as they have many parameters, and the impact of their hyperparameters is significant. Table 3 summarizes the values of the four performance metrics of these two classifiers before and after fine-tuning of their hyperparameters. Fine-tuning parameters improved recall, precision, and F1-score measurements for both classifiers under both classes. The largest improvement for both the RF and XGBOOST models was observed for the "true" class. XGBOOST achieved the highest recall and F1-scores after fine-tuning for both the "true" and "false" classes at 89% and 100% for recall and 92% and 99% for F1-score, respectively. The results

demonstrated that fine-tuning hyperparameters to optimize parameter value by analyzing their impact, in terms of over- and underfitting, results in increased model robustness.

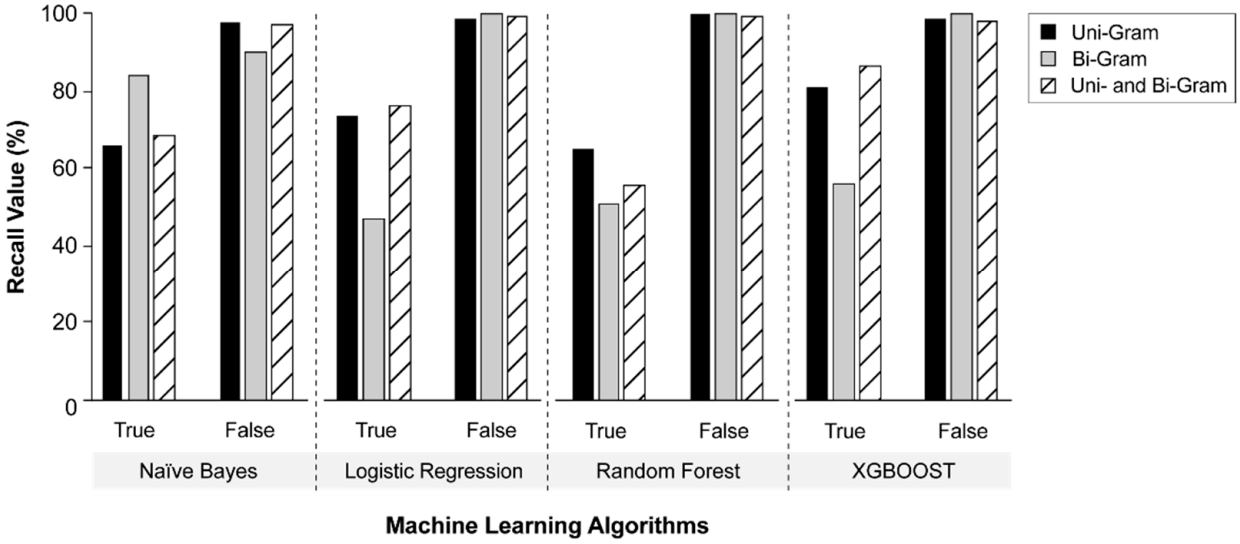

Figure 10. Impact of n-gram sets on recall performance of machine learning models.

Table 3. Effect of fine-tuning hyperparameters on performance metrics of Random Forest (RF) and Extreme Gradient Boosting (XGBOOST)-based algorithms.

Metrics	Class Label	RF		XGBOOST	
		Before	After	Before	After
Accuracy	-	97.8	98	98.4	98
Precision	True	99	100	87	96
	False	98	98	99	99
Recall	True	59	74	81	89
	False	100	100	99	100
F1-Score	True	73	85	84	92
	False	99	99	99	99

Altogether, ML-based performance measurements revealed that the XGBOOST model outperformed the other ML algorithms in terms of accuracy (Figure 8) and recall (Figure 10) performance. Accordingly, the XGBOOST model is selected as the optimal classifier for ML-based text classification.

4.4. Comparison of Classification Models

Performance results achieved by the best-performing rule-based and ML-based classifiers are summarized in Table 4. Under the rule-based classifier, application of the bi-gram rule set with lemmatization and stop-word removal resulted in accuracy and "true" recall values of 99% and 88%, respectively (Table 4). Comparatively, application of the XGBOOST-based machine learning algorithm resulted in accuracy and "true" recall values of 98% and 89%, respectively.

Table 4. Performance of rule-based versus machine learning-based text classification models.

Metrics	Class Label	Rule-Based	ML-Based
Accuracy	-	99	98
Precision	True	96	96
	False	99	99
Recall	True	88	89
	False	100	100
F1-Score	True	92	92
	False	100	99

The patterns used to construct the rules in the rule-based model were manually defined, requiring more effort in terms of rule preparation. It is also important to note that the results of the rule-based model are quite sensitive to input rules: adding or removing a specific rule may have a considerable impact on classifier performance. Alternatively, the ML-based text classification model learns the classification process by using training data. In this regard, the results of the ML model are sensitive to the number of training sets, performing best in the presence of more data. Given the results provided in Table 4, the choice of classification model depends on the availability of training data for the ML- based model or the level of effort able to be invested for rule construction in the rule-based model.

5. Prediction Module

The prediction module is used to estimate the time, resources, and cost needed to fulfill the reporting requirements. Module inputs include (1) the list of reports prepared by subject-matter experts using outputs of the rule-based or ML-based extraction module that describe the types and submittal frequencies of the reporting requirements, (2) the resources, time, and hourly rate associated with each reporting requirement, and (3) estimated project duration. To account for underlying uncertainties in model inputs and outputs, a Monte Carlo simulation model is employed, with uncertain parameters (e.g., report preparation time and project duration) input as probabilistic distributions derived from historical data. If sufficient historical data are unavailable, probabilistic distributions, such as a triangular distribution with minimum, most likely, and maximum values reported by experts, can be input into the model instead [44]. The Monte Carlo simulation is then run for multiple iterations, with each iteration randomly selecting a value from each parameter's distribution. Outputs of the model include the predicted (1) time, (2) cost, and (3) distribution among the various personnel types to complete the reporting requirements.

6. Case Study

An oil-and-gas project led by a private Canadian construction contracting firm was used to demonstrate the proposed framework. The project was considered a small-size project by the contractor and was awarded by the client to the contractor through a cost-plus contract type. This project was completed before the initiation of this research study. Actual durations of report preparation were not recorded by the contracting firm.

6.1. Data Collection

While manual extraction of reporting requirements is not required for future construction contracts, manual extraction was required, here, for initial development of the extraction module. As such, and for this case study only, the list of manually extracted reporting requirements from the set of contract documents detailed in Section 4.1 were input into the model. Notably, outputs of the rule-based or ML-based extraction module can be used to prepare a list of required reports and their submittal frequencies for input into the simulation model for resource prediction of future contracts.

Since report preparation times were not recorded by the project team, the minimum, most likely, and maximum values for the preparation time of each report were provided by

company experts based on prior experience. Individual labor rates for each personnel type were not provided by the industrial partner; therefore, an average labor rate of 60 CAD per hour was input into the model. A subset of the data is summarized in Table 5.

Table 5. Sample of report preparation-associated input data.

Report Name	Frequency	Resources	Time (Minutes) Min, Most Likely, Max
Daily Update: work plan and estimated progress	Daily	1 Safety Coordinator 1 Project Manager 4 Superintendents 1 Quality Controller	45, 60, 75
Equipment Log	Bi-Weekly	1 Project Coordinator	90, 120, 150
Installation Work Package Report	Bi-Weekly	1 Project Controller 1 Scheduler 1 Project Manager	210, 240, 270
3-Week Look-Ahead Schedule	Bi-Weekly	1 Project Control 1 Scheduler 1 Project Manager	360, 420, 480

6.2. Results and Discussion

6.2.1. Extraction Module

A sample of the extracted reporting requirements is illustrated in Figure 3. As detailed in Section 6.1, 340 individual reporting requirements and their submittal frequencies were identified from over 500 contract pages. Although quite high (88%, Table 4), the recall of the current extraction module is not 100%. Sufficient for planning purposes, the extraction module should not be used as the only means of requirement extraction during the execution phase of a project. A manual review during the execution phase should continue to be performed until the ability of the framework to consistently extract 100% of reporting requirements is achieved. Failing to determine the exhaustive list of submittals and information deliverables required by the client can result in claims and litigations, subjecting both parties to disputes and conflicts that could have been prevented. Nevertheless, manual extraction is also prone to error, and the use of the extraction module as an adjunct tool during the execution phase of a project is strongly recommended. The list of reporting requirements extracted manually and by the automated extraction module should be compared to identify requirements that may be missing from the manually extracted list.

The 340 reporting requirements output by the extraction module were reviewed by the project team. It was determined that some of the reporting requirements were repetitive, requiring submission of the same report. A total of 70 distinct reports and information submittals was identified. A list of these reports was prepared and input into the prediction module.

6.2.2. Prediction Module

The Monte Carlo simulation model was run for 100,000 iterations, as increasing the number iterations beyond 100,000 slowed the execution speed without resulting in a notable impact on output results. The total duration and cost associated with the requirement reporting process was calculated using the probability distributions defined for each report. In each iteration, random numbers were sampled from the preparation time distributions of each report type, and a total reporting duration (or total cost) was achieved as the cumulative time (or cost) of all reports for that iteration. Total reporting duration (or cost) values of each iteration were then used to form a distribution of total reporting time, as shown in Figure 11.

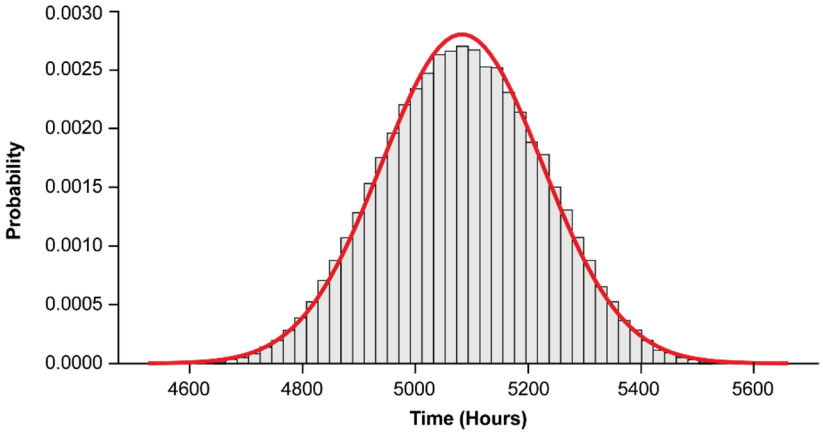

Figure 11. Predicted cumulative report preparation time as a distribution.

The mean value for the cumulative report preparation time was 5083 labor-hours (σ = 142) for the project life cycle. The simulation was then run again using an average rate of 60 CAD per hour; here, the mean value of the total cost associated with the reporting process was calculated to be $304,939 ($\sigma$ = $8538), as shown in the predicted cost distribution in Figure 12. Notably, including individual labor rates for each personnel type will increase the accuracy of the framework's results.

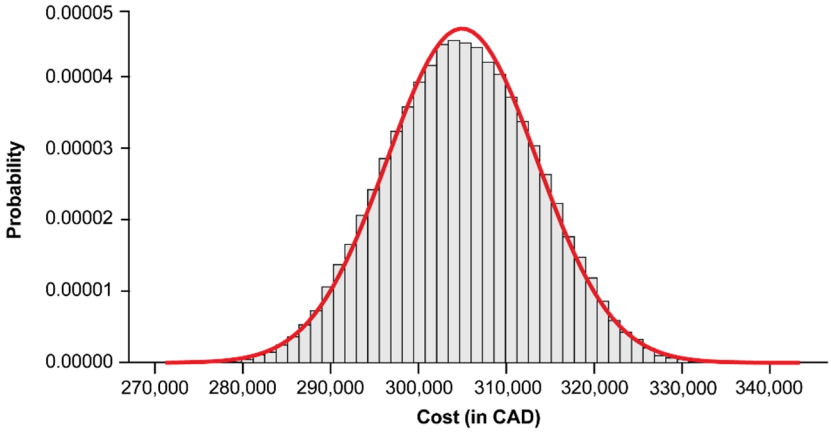

Figure 12. Predicted cumulative cost of reporting as a distribution.

The distribution of report preparation time between various personnel types is summarized in Figure 13. The plurality of the cumulative report preparation time (31%) was associated with the project manager, who must review and approve most reports. Based on the mean cumulative duration of 5083 h (Figure 11), the project manager is expected to spend an estimated 1576 h (or, assuming a 9-h work day, 175 days) completing reporting requirements. With a provided project duration of 4400 h, the project manager is estimated to be performing reporting activities 36% of the time. Similarly, two other highly utilized resources were the project control team (28%) and scheduler (28%), who are responsible for collecting, merging, and overseeing the preparation of various report types. Together, these two resources will spend an estimated 2846 h (or 316 days) completing reporting requirements—equal to 32% of their time.

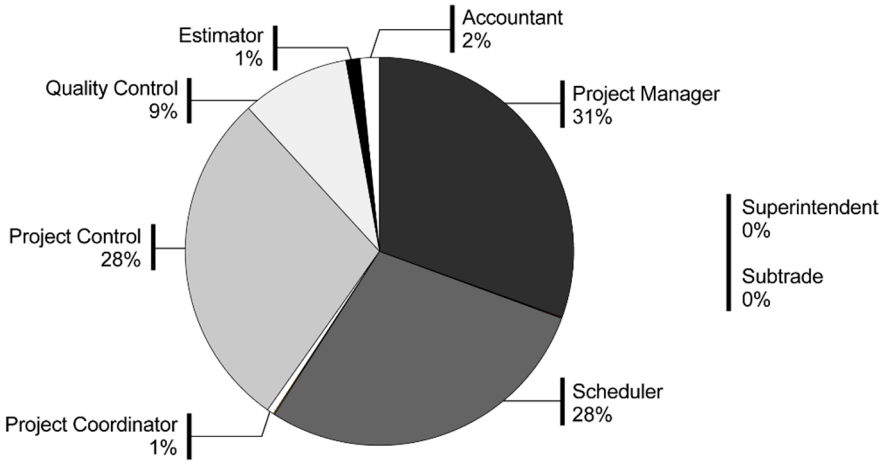

Figure 13. Distribution of report preparation time between personnel types.

6.3. Framework Validation

6.3.1. Validation of Extraction Module

The extraction module underwent extensive validation testing. Here, reporting requirements were manually extracted and compared to the list of reporting requirements identified using the extraction module. Then, a subset of real project data (different from those used for model development and training) was used to evaluate the performance of the rule-based and machine-learning-based classification models. A discussion of the validation process is detailed in Sections 4.2.1 and 4.3.1, respectively. A comparison of the models is summarized in Section 4.4.

6.3.2. Validation of Prediction Module

In contrast, the prediction module was evaluated using face validation. Since actual report preparation durations were not recorded by the contractor, face validation was performed by subject-matter experts to evaluate whether or not the simulation model results (i.e., prediction module outcomes) were accurately representing the current reporting process. Simulation results (Figures 11–13) were presented to the project management team responsible for executing the case study project. The experts confirmed that the simulated results were acceptable and were consistent with the outcomes of the actual project. Overall, face validation by the subject-matter experts confirmed that the prediction module was representative, comprehensive, and easy to use.

7. Discussion

Having a list of reporting requirements during the planning phase of a project will provide the project management team with the opportunity to enhance the reporting process, resulting in a reduction in reporting-associated costs. For example, similar or redundant reports can be consolidated, specialized data collection systems and report templates can be developed and implemented prior to project execution, and the allocation of reporting requirements to specific personnel types can be optimized.

The probability distributions output by the proposed framework allow decision makers to more accurately estimate the probability of achieving project targets, while gaining insight on potential best- and worst-case scenarios. More accurate time preparation estimates will allow contractors to ensure that a sufficient number of personnel are available to complete reporting requirements on time. Moreover, by more accurately estimating the overhead costs associated with reporting requirements for each particular project, contractors are able to enhance bid preparation to improve competitiveness, or provide more realistic direct–indirect cost ratios to avoid potential disputes for cost-plus contracts. Furthermore, these outputs can be used to optimize the composition of project management teams based on the specific requirements of each contract. Together with the list of

requirements output from the extraction module, the personnel distribution results can be used to examine and potentially reallocate reporting duties to lower-wage personnel, where appropriate, thereby reducing report preparation costs.

The level of benefit achieved by considering reporting-associated costs in the planning phase of construction depends on the construction type. Repetitive types of construction, such as residential building construction, are typically associated with contracts that remain similar between projects. Due to a lack of variability in reporting requirements, project teams are able to accurately approximate the time, cost, and resources required without the need to extract reporting requirements for each contract. However, due to the increased level of risk, complexity of the work, and large project scale, contract documents for complex projects, such as those in the oil and gas industry, are typically longer, more intricate, and more variable from project to project. With these types of construction, clients tend to request additional information and detailed reporting submittals from contractors, which substantially increases overhead costs. The benefits of applying the proposed framework, therefore, are expected to expand as project complexity is increased.

8. Limitations and Future Work

An automated approach for rapidly extracting reporting requirements from contractual documents and predicting the time and cost required to complete reporting activities was developed. Although the functionality of the proposed framework was demonstrated using real contractual documents from an actual case study, the following points should be considered.

First, the extraction module was developed using a labeled dataset obtained from one set of contract and specification documents for an oil and gas project. While the extraction module is expected to be applicable—in its current form—to all construction contracts with similar characteristics (e.g., terminology, document structure, and/or report structure), the development methodology described may need to be reapplied for other contract types. Moreover, the classification models were trained using a limited amount of training data. The comparatively low performance of the classification models for the "true" class may be due to the size of the "true" dataset (i.e., an imbalanced data problem). Future work should examine the impact of increasing the training dataset through the incorporation of additional contract documents to enhance the performance of the classification models. With sufficient training data, the extraction module is anticipated to achieve the desired performance of 100% recall for the "true" class (i.e., identification of all reporting requirements).

Second, the success of the prediction module is highly dependent on accurately modeling the inputs. One of the difficulties in analyzing probabilistic processes inherent to construction is defining the probability distributions that best reflect the uncertainties associated with each variable. The more accurate the model of the inputs, the more closely the simulation model mimics real-life behavior. A primary constraint for any simulation model, therefore, is the time and effort required to collect pertinent and correct information, as well as processing it for input into the model. While the resources and time required from construction sites and administration offices to complete reporting requirements are not commonly recorded, efforts to improve data collection related to project reporting process are expected to improve model results.

Third, contract documentation remains an immature area of practice, and more reliable and efficient approaches to better and more rapidly understand contract requirements are needed. Future work should focus on providing a holistic solution to this problem, such as writing contracts using a structured-database approach. While this would provide seamless integration between clients and contractors (thereby alleviating the need for rule-based/ML-based model (re)training), achieving this ideal will require a tremendous amount of input, effort, and collaboration among all of the stakeholders involved in a project. Additionally, methods for dealing with modifications or alternate arrangements

will need to be researched and developed. Consequently, the framework proposed here provides a much-needed interim solution as these more holistic solutions are pursued.

9. Conclusions

Automating the reporting requirement extraction process and estimating its associated time–cost implications are expected to reduce the effort, time, and overhead costs expended by the multiple personnel involved. To overcome the shortcomings of the traditional manual approach, this study developed a framework for reporting requirement extraction based on NLP—a domain-specific and application-oriented text classification process—that is capable of automatically identifying reporting requirements from contractual documents to considerably reduce the time and effort required to extract reporting requirements. To account for project uncertainties due to variation or unforeseeable events that may occur during execution, a Monte Carlo simulation was used to predict the time and cost needed to complete reporting requirements.

The model begins by collecting textual data, in this case the sentences and terms in contractual documents, which describe the reporting requirements mandated by the client. Rule-based and ML-based classification methods were developed, and their performances were evaluated. The performance of rule-based classification using different sets of n-grams was assessed, with an accuracy of 99.27% achieved using bi-grams as rules. Application of lemmatization to and removal of stop-words from the bi-gram rules resulted in a recall and F1-score of 88% and 92% for the "true" category, respectively. Four ML algorithms were also implemented, and their performance was assessed under different pre-processing settings and feature engineering techniques. All of the ML classification models achieved promising accuracies of over 95%; notably, XGBOOST achieved the highest recall value of 89% after parameter tuning. Then, numerical data regarding report preparation times and associated resources (based on prior experience of experts) were provided by an industrial partner and were used to predict the time and cost required to complete the reporting requirements detailed in the contractual documents. Input of these data into the Monte Carlo simulation model resulted in a mean cumulative reporting duration and cost of 5083 h and 304,939 CAD, respectively.

During the bidding and contract negotiation phase of a project, decision makers can now use the proposed framework to automatically review reporting requirements prior to accepting the contractual agreement. Not feasible using time-consuming, traditional extraction methods, the extraction speed of the framework allows decision makers to identify and subsequently negotiate difficult and/or inefficient reporting requirements prior to signing. If contract conditions are unfavorable to the contractor in terms of project reporting cost, a revision of contract conditions may be requested or a contract may be abandoned by contractors to prevent further loss. With a thorough and realistic understanding of contract reporting requirements, contractors can focus on establishing the best means, methods, pricing, and schedules for completing the proposed project.

Author Contributions: Conceptualization, P.J. and S.A.; data curation, P.J., M.A.H. and E.M.; formal analysis, P.J.; funding acquisition, S.A.; investigation, P.J.; methodology, P.J.; project administration, S.A.; software, P.J.; supervision, S.A.; validation, P.J.; visualization, P.J. and M.A.H.; writing—original draft, P.J.; writing—review and editing, M.A.H., E.M. and S.A. All authors have read and agreed to the published version of the manuscript.

Funding: This research was funded by the Natural Sciences and Engineering Research Council of Canada through a Collaborative Research and Development Grant (CRDPJ 492657).

Institutional Review Board Statement: Not Applicable.

Informed Consent Statement: Not Applicable.

Data Availability Statement: All data used in the study were provided by a third party. Direct requests for these materials may be made to the provider indicated in the Acknowledgments. Models

and code that support the findings of this study are available from the corresponding author upon request and with permission from the partner indicated in the Acknowledgments.

Acknowledgments: The authors would like to thank Graham Industrial Services LP for their support and for providing contractual documents and report preparation-associated data. The authors also would like to acknowledge Catherine Pretzlaw for her assistance with manuscript editing and composition.

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

References

1. Shash, A.A.; Habash, S.I. Construction Contract Conversion: An Approach to Resolve Disputes. *J. Eng. Proj. Prod. Manag.* **2020**, *10*, 162–169. [CrossRef]
2. Barlow, G.; Dew, C.; Woolley, P.; Dempsey, H. *Effective Reporting for Construction Projects: Increasing the Likelihood of Project Success*; Project Advisory Leadership Series; KPMG New Zealand: Auckland, New Zealand, 2014.
3. El-Omari, S.; Moselhi, O. Integrating automated data acquisition technologies for progress reporting of construction projects. *Autom. Constr.* **2011**, *20*, 699–705. [CrossRef]
4. Hassan, F.U.; Le, T. Automated Requirements Identification from Construction Contract Documents Using Natural Language Processing. *J. Leg. Aff. Disput. Resolut. Eng. Constr.* **2020**, *12*, 04520009. [CrossRef]
5. Lee, J.; Yi, J.-S.; Son, J. Development of Automatic-Extraction Model of Poisonous Clauses in International Construction Contracts Using Rule-Based NLP. *J. Comput. Civ. Eng.* **2019**, *33*, 04019003. [CrossRef]
6. Jeong, H.D.; Gransberg, D.; Shrestha, K.J. *Framework for Advanced Daily Work Report System*; Institute for Transportation, Iowa State University: Ames, IA, USA, 2015.
7. Caldas, C.H.; Soibelman, L.; Han, J. Automated Classification of Construction Project Documents. *J. Comput. Civ. Eng.* **2002**, *16*, 234–243. [CrossRef]
8. Pestana, C.; Alves, T.; Barbosa, A. Application of Lean Construction Concepts to Manage the Submittal Process in AEC Projects. *J. Manag. Eng.* **2014**, *30*, 05014006. [CrossRef]
9. Levin, P. *Construction Contract Claims, Changes & Dispute Resolution*; American Society of Civil Engineers (ASCE): Reston, VA, USA, 1998.
10. Jeon, K.; Lee, G.; Jeong, H.D. *Classification of the Requirement Sentences of the US DOT Standard Specification Using Deep Learning Algorithms*; Springer: Berlin/Heidelberg, Germany, 2020; pp. 89–97.
11. El Gindi, M. *User Friendly Progress Reporting System for Construction Projects*; The American University in Cairo: Cairo, Egypt, 2017.
12. El Sawy, I.; Hosny, H.; Razek, M.A. A Neural Network Model for Construction Projects Site Overhead Cost Estimating in Egypt. *Int. J. Comput. Sci.* **2011**, *8*, 273–283.
13. Jallow, A.K.; Demian, P.; Anumba, C.J.; Baldwin, A.N. An enterprise architecture framework for electronic requirements information management. *Int. J. Inf. Manag.* **2017**, *37*, 455–472. [CrossRef]
14. Morgan, A. *Does Poor Project Governance Cause Delays?* Pricewaterhouse Coopers LLP: London, UK, 2010.
15. Jafari, P.; Mohamed, E.; Lee, S.; Abourizk, S. Social network analysis of change management processes for communication assessment. *Autom. Constr.* **2020**, *118*, 103292. [CrossRef]
16. Lee, G.; Cho, J.; Song, T.; Roh, H.; Jung, J.; Chung, J.; Yong, G.; Jeong, D. *Construction Field Management Using a Popular Text Messenger*; Springer: Berlin/Heidelberg, Germany, 2020; pp. 971–979.
17. Shrestha, K.J.; Jeong, H.D. Computational algorithm to automate as-built schedule development using digital daily work reports. *Autom. Constr.* **2017**, *84*, 315–322. [CrossRef]
18. Russell, A.D. Computerized Daily Site Reporting. *J. Constr. Eng. Manag.* **1993**, *119*, 385–402. [CrossRef]
19. Shiau, Y.-C.; Wang, W.-C. Daily Report Module for Construction Management Information System. In Proceedings of the 20th International Symposium on Automation and Robotics in Construction ISARC 2003—The Future Site, Budapest, The Netherlands, 21–24 September 2003; Maas, G., Van Gassel, F., Eds.; International Association for Automation and Robotics in Construction (IAARC): Eindhoven, UK, 2003; pp. 603–609.
20. Omar, T.; Nehdi, M.L. Data acquisition technologies for construction progress tracking. *Autom. Constr.* **2016**, *70*, 143–155. [CrossRef]
21. Manning, C.; Schutze, H. *Foundations of Statistical Natural Language Processing*; MIT Press: Cambridge, MA, USA, 1999; ISBN 0-262-30379-5.
22. Tixier, A.J.-P.; Hallowell, M.R.; Rajagopalan, B.; Bowman, D. Automated content analysis for construction safety: A natural language processing system to extract precursors and outcomes from unstructured injury reports. *Autom. Constr.* **2016**, *62*, 45–56. [CrossRef]
23. Fan, H.; Li, H. Retrieving similar cases for alternative dispute resolution in construction accidents using text mining techniques. *Autom. Constr.* **2013**, *34*, 85–91. [CrossRef]
24. Zhang, J.; Zi, L.; Hou, Y.; Deng, D.; Jiang, W.; Wang, M. A C-BiLSTM Approach to Classify Construction Accident Reports. *Appl. Sci.* **2020**, *10*, 5754. [CrossRef]

25. Al Qady, M.; Kandil, A. Automatic Classification of Project Documents on the Basis of Text Content. *J. Comput. Civ. Eng.* **2015**, *29*, 04014043. [CrossRef]
26. Manning, C.D.; Raghavan, P.; Schutze, H. *Introduction to Information Retrieval*; Cambridge University Press: Cambridge, UK, 2008; Volume 19, pp. 1041–4347.
27. Russell, S.; Norvig, P. *Artificial Intelligence: A Modern Approach*; Pearson: London, UK, 2020; ISBN 0-13-461099-7.
28. Salama, D.M.; El-Gohary, N.M. Semantic Text Classification for Supporting Automated Compliance Checking in Construction. *J. Comput. Civ. Eng.* **2016**, *30*, 04014106. [CrossRef]
29. Zhong, B.; Pan, X.; Love, P.E.; Ding, L.; Fang, W. Deep learning and network analysis: Classifying and visualizing accident narratives in construction. *Autom. Constr.* **2020**, *113*, 103089. [CrossRef]
30. Zhou, P.; El-Gohary, N. Ontology-Based Multilabel Text Classification of Construction Regulatory Documents. *J. Comput. Civ. Eng.* **2016**, *30*, 04015058. [CrossRef]
31. Hastings, W.K. Monte Carlo sampling methods using Markov chains and their applications. *Biomolecules* **1970**, *57*, 97–109. [CrossRef]
32. Python. *Python 3.7.0*; Python Software Foundation: Beaverton, OR, USA, 2018.
33. Priyanka, H.; Ramya, B.; Ashok, K. Classification Model to Determine the Polarity of Movie Review Using Logistic Regression. *Int. Res. J. Comput. Sci.* **2019**, *6*, 87–91. [CrossRef]
34. Grefenstette, G.; Tapanainen, P. What Is a Word, What Is a Sentence? Problems of Tokenisation. In Proceedings of the 3rd International Conference on Computational Lexicography, Budapest, Hungary, 7–10 July 1994; Research Institute for Linguistics, Hungarian Academy of Sciences: Budapest, Hungary, 1994; pp. 79–87.
35. Porter, M.F. An algorithm for suffix stripping. *Program* **1980**, *14*, 130–137. [CrossRef]
36. Buckland, M.; Gey, F. The Relationship between Recall and Precision. *J. Am. Soc. Inf. Sci.* **1994**, *45*, 12–19. [CrossRef]
37. Forman, G. An Extensive Empirical Study of Feature Selection Metrics for Text Classification. *J. Mach. Learn. Res.* **2003**, *3*, 1289–1305.
38. Sebastiani, F. Machine learning in automated text categorization. *ACM Comput. Surv.* **2002**, *34*, 1–47. [CrossRef]
39. Valieva, I.; Voitenko, I.; Björkman, M.; Åkerberg, J.; Ekström, M. Multiple Machine Learning Algorithms Comparison for Modulation Type Classification Based on Instantaneous Values of the Time Domain Signal and Time Series Statistics Derived from Wavelet Transform. *Adv. Sci. Technol. Eng. Syst.* **2021**, *6*, 658–671. [CrossRef]
40. Witten, I.; Frank, E.; Mark, A. *Hall Data Mining: Practical Machine Learning*; Elsevier: Amsterdam, The Netherlands, 2011; ISBN 9780123748560.
41. Breiman, L. Random Forests. *Mach. Learn.* **2001**, *45*, 5–32. [CrossRef]
42. Chen, T.; Guestrin, C. Xgboost: A Scalable Tree Boosting System. In Proceedings of the 22nd ACM SIGKDD International Conference on Knowledge Discovery and Data Mining, New York, NY, USA, 13–17 August 2016; pp. 785–794.
43. Bergstra, J.; Bengio, Y. Random Search for Hyper-Parameter Optimization. *J. Mach. Learn. Res.* **2012**, *13*, 281–305.
44. Abourizk, S.M.; Halpin, D.W. Statistical Properties of Construction Duration Data. *J. Constr. Eng. Manag.* **1992**, *118*, 525–544. [CrossRef]