Application of Graph Convolutional Networks to Classification of Building Code Requirements

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

[1]Automation and Intelligent Construction (AutoIC) Lab, School of Construction Management Technology, Purdue University, West Lafayette, IN. Email: yang2051@purdue.edu
[2]Automation and Intelligent Construction (AutoIC) Lab, School of Construction Management Technology, Purdue University, West Lafayette, IN; (*corresponding author). ORCID: https://orcid.org/0000-0001-5225-5943. Email: zhan3062@purdue.edu

ABSTRACT

A building must meet various requirements during the design and construction process to ensure the benefits of stakeholders and well-being of construction workers and occupants. These requirements may come from different functional areas such as structure, electricity, and fire protection, and focus on different building materials, such as concrete, steel, and glass. They may overlap or even conflict with each other. In order to identify the sources and focus of building code requirements and further clarify the relationships between them, this paper presents some recent results on using graphic convolutional networks (GCN) to classify building code requirements. One hundred building code provisions were randomly selected from the International Building Code 2015 and labeled into 6 categories manually, and a cutting-edge GCN model was trained to classify them. Experimental results showed an average precision of 91.67% and an average recall of 94.44% when 10% of the data was used for testing, which is comparable to the 84.30% precision and 97.30% recall of the state-of-the-art machine learning-based approaches applied on construction document classification. The effect of the size of training data on testing accuracy was also discussed in this paper.

INTRODUCTION

The construction of a building necessitates strict adherence to numerous requirements that guarantee the safety and well-being of the construction workers and occupants, while simultaneously satisfying the expectations of stakeholders. Building codes and standards establish the minimum requirements for construction projects to achieve the aforementioned objectives. Compliance of a building with building codes and regulations is critical for various parties involved, including contractors, designers, workers, occupants, among other stakeholders. In recent years, automated compliance checking (ACC) has gained popularity as a substitute for traditional manual compliance checking, which is notorious for being a time-consuming and labor-intensive process (Wu et al. 2023; Akanbi and Zhang 2021; Zhang and El-Gohary 2017; Eastman et al. 2009). ACC involves the use of computer technology to automate the process of checking building designs and activities for compliance with building codes and regulations (Yang et al. 2022; Wu and Zhang 2022; Xu and Cai 2020). Compared to manual compliance checking, ACC is more efficient, accurate, reliable, and consistent (Macit İlal and Günaydın 2017). For instance, ACC can identify non-compliant areas in real-time, allowing for early intervention and rectification of potential issues. Additionally, ACC can mitigate the risk of errors and oversights of manual checking, thereby providing consistent and reliable results. The ACC framework can be customized to the unique requirements of a project, guaranteeing that all relevant codes and standards are checked. Given its significant benefits, it is desirable to facilitate the adoption of ACC in the construction industry to ensure compliance with all relevant codes and standards, as well as to enhance the well-being, satisfaction, and productivity of all parties involved, thereby increasing overall industry productivity.

The current ACC approach commonly involves four stages: building code representation, design information representation, checking rule execution and result exportation (Eastman et al. 2009). Among them, building code representation is a crucial stage in ACC as it involves the translation of the legal text of building codes into a form that can be processed by computers. This representation serves as the basis for the design information representation and the checking rule execution stages of ACC. Building code representation is particularly challenging because building codes are typically written in natural language, which is ambiguous and prone to interpretation. Consequently, translating building codes into machine-readable form requires a deep understanding of the legal text, the domain knowledge, and the rules governing building design and construction. Additionally, building codes are frequently updated, which means that the ACC system must be able to adapt to new changes in the codes. To address these challenges, researchers have developed various approaches for building code representations in ACC systems, including semantic web technologies (Dimyadi et al. 2015; Pauwels et al. 2011), rule-based approaches (Zhang and El-Gohary 2016a; Rasdorf and Lakmazaheri 1990; Rasdorf and Wang 1988), and natural language processing techniques (Zhang 2023; Xue and Zhang 2021; Zhang and El-Gohary 2016b; Salama and El-Gohary 2012). These approaches aim to represent building codes in a more structured and machine-readable form that can be easily interpreted by ACC systems.

To ensure the accurate extraction and representation of building code information, building code classification serves as a crucial preliminary step (Zhou and El-Gohary 2016b). As a type of text classification, building code classification involves the automatic categorization of building code requirements into predefined classes or categories based on their content, context, or features. Building code requirements can be sourced from different functional areas, each with its own specific focus and requirements. For instance, requirements for structure, electricity, and fire protection differ significantly, with each area having its own specific regulations. Moreover, building requirements may focus on different building materials such as concrete, steel, and glass. As a result, building codes may overlap, conflict, or even have a cascading effect on one another, making it challenging to extract and represent them accurately. Building code classification techniques simplify the identification of sources of building code requirements and clarify relationships between them. This simplifies the accurate and efficient extraction and categorization of building code information. As a result, a structured and standardized representation of the building code can be created, which is more easily interpreted by computers, enabling more effective automated compliance checking.

Text classification is a classic natural language processing (NLP) task, and researchers have applied machine learning techniques to classify building text or documents in previous studies (Salama and El-Gohary 2016; Zhou and El-Gohary 2016a). However, previous research mainly utilized classic classification models (e.g., support vector machines (SVMs), decision tree (DT), k-nearest neighbors (kNN)). To the best of the authors' knowledge, no research has explored the application of state-of-the-art graphic convolutional networks (GCN) for building code classification, which are expected to outperform those classic models. This paper presents a novel approach of utilizing cutting-edge GCN models to classify building code requirements based on their focus on different building materials. The proposed approach aims to improve the accuracy and efficiency of building code classification, and the results will contribute to the accurate information representation of building code and the development of automated compliance checking systems.

LITERATURE REVIEW

Text classification. Text classification is a fundamental problem in NLP that involves categorizing textual data into pre-defined classes or categories (Vijayan et al. 2017). It plays a critical role in many applications such as document classification (Pappagari et al. 2019; Yoon and Lee 2007), spam filtering (Wu 2009), sentiment analysis (Moraes et al. 2013; Vinodhini and Chandrasekaran 2016), and topic modeling (Nigam et al. 2000; Razavi and Inkpen 2014).

The traditional approach to text classification involves manually selected features such as word frequency, part-of-speech tags, and n-grams. These features are then used to train a machine learning model to classify the text. However, this approach is time-consuming, requires domain expertise, and may not capture the semantic meaning of the text. With the advent of deep learning and the availability of large-scale text data, neural network-based models have shown significant improvements in text classification tasks. Convolutional neural networks (CNN) and recurrent neural networks (RNN), specifically long short-term memory (LSTM) networks, are among the most widely used deep learning models for text classification (Bai 2018; Liu et al. 2016; Wang et al. 2018). These models have the ability to learn representations of words and phrases that capture their semantic meaning and can be used to classify text without the need for handpicked features.

Nowadays, graph convolutional networks (GCN) rose as a powerful approach to text classification, especially for tasks that involve structured data such as social networks, knowledge graphs, and citation networks (Liu et al. 2020; Tang et al. 2020; Tayal et al. 2019). Compared to traditional deep learning models such as CNN and RNN, GCN models can capture the relational structure of a graph and preserve global structure information in graph embeddings, which can be used for node classification (Yao et al. 2019). Building codes, as a set of regulations that govern the construction and maintenance of buildings (Zhang and El-Gohary 2015), can also be represented as a graph (Xue et al. 2022), where nodes represent building elements or requirements, and edges represent the relationships between them. Thus, GCN models are suitable for building code classification as they can effectively capture the interdependencies and hierarchies between building code requirements and enable the incorporation of structured information into the classification process. Moreover, GCN models have been shown to outperform classic ML and deep learning models in different graph-related tasks, such as node and graph classifications, making them a promising approach for building code classification (Lei et al. 2019; Liu et al. 2020; Yao et al. 2019).

Building code classification. Building code classification is a subfield of text classification that focuses specifically on categorizing and organizing the requirements and regulations of building codes. Building code classification is particularly challenging due to the complex and technical nature of building codes, which often involve multiple disciplines and have strict requirements that must be met. Previous research has been conducted to try to address this challenge using different approaches. For instance, Salama and El-Gohary (2016) proposed a hybrid algorithm that combines semantic, syntactic, and machine learning techniques to classify construction contract clauses and subclauses for supporting ACC. Zhou and El-Gohary (2016a) also proposed a machine learning algorithm that classifies clauses of environmental regulatory documents based on the text topic hierarchy. They tested the algorithm's performance using various machine learning techniques and preprocessing methods, achieving an average recall and precision of approximately 97% and 84%, respectively, on the testing data. Furthermore, Zhou and El-Gohary (2016b) proposed an ontology-based text classification algorithm that utilizes the semantic features of the text to improve classification performance. The new algorithm outperforms the previous one consistently due to the advantage of using semantic concepts and relations.

Previous research in building code classification has focused on developing machine learning algorithms and techniques to accurately classify and organize building code requirements. These efforts have shown promising results and have the potential to significantly improve the efficiency and effectiveness of automated compliance checking. However, there is still a need for further research in this field to explore more powerful and suitable machine learning models for building code classification.

METHODOLOGY

To evaluate the suitability of Graph Convolutional Networks (GCN) for building code classification, the authors selected a text GCN model developed by Yao et al. (2019). The text GCN was chosen for its ability to capture global structured information in graph embeddings, which traditional CNN and RNN models may overlook, and its superior performance on multiple benchmark datasets, even without pre-trained word and external knowledge. The authors randomly selected 100 building code requirements from six chapters of the International Building Code 2015 (IBC 2015), specifically from chapters covering building material types such as Glass, Plastic, Soil, Wood, Gypsum, and Masonry. The data was labeled according to the corresponding material and divided randomly into training and testing data, with either 10% or 20% of the data reserved for testing. Prior to analysis, the authors performed necessary code modifications to connect the created dataset and the model. The model automatically preprocessed the data by preparing and cleaning the data, as well as building the graph. The optimized model was then used to classify the testing data and output the results. The workflow of the experiment is illustrated in Figure 1.

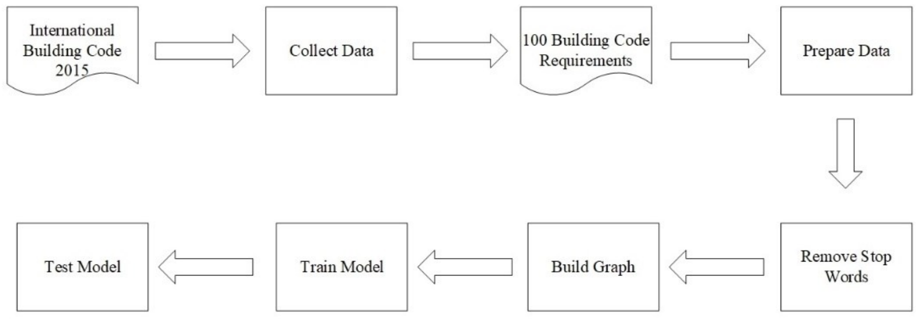


Figure 1. The workflow of the experiment.

The evaluation of the text GCN model included factors such as recall, precision, and accuracy, which were computed using the following well-defined formulas [Eqs. (1)-(3)].

$$Precision = \frac{TP}{TP+FP} \quad (1)$$

$$Recall = \frac{TP}{TP+FN} \quad (2)$$

$$Accuracy = \frac{TP+TN}{TP+FN+TN+FP} \quad (3)$$

Where True Positive (TP) refers to the number of correctly predicted positive instances or samples in a dataset by the model, True Negative (TN) refers to the number of correctly predicted negative instances or samples by the model, False Positive (FP) refers to the number of negative instances that were incorrectly predicted as positive by the model, and False Negative (FN) refers to the number of positive instances that were incorrectly predicted as negative by the model.

Additionally, the experiments examined the effect of the size of training dataset on the accuracy of the model.

EXPERIMENT

Dataset creation. IBC 2015 is a comprehensive document consisting of 35 chapters, each addressing a specific theme related to building design and construction. The authors of this study chose to classify the building code requirements based on the building material used. Specifically, they selected 100 building requirements from six chapters (Glass, Plastic, Soil, Wood, Gypsum, and Masonry), and labeled each requirement according to its respective material. For instance, the following requirement related to masonry was labeled as “Masonry”: “The allowable compressive stress based on gross cross-sectional area of adobe shall not exceed 30 psi (207 kPa)” (International Code Council 2014). The distribution of all selected requirements across the six categories is shown in Table 1. The authors also marked each requirement as either training or testing data and stored the data in an Excel file with four columns: index, label, train_or_test, and sentence. Table 2 presents some examples of the dataset. To assess the impact of training data size, the authors randomly selected 10% or 20% of all data as testing data, while the remainder served as training data. It is important to note that this dataset is unique in that it is stored as an Excel file, which requires some modification to access the data when running the experimental code. This distinction from benchmark datasets, which are often provided online or as text files, highlights the need for flexibility and adaptability in data analysis.

Table 1. The distribution of building code requirements in the six categories.

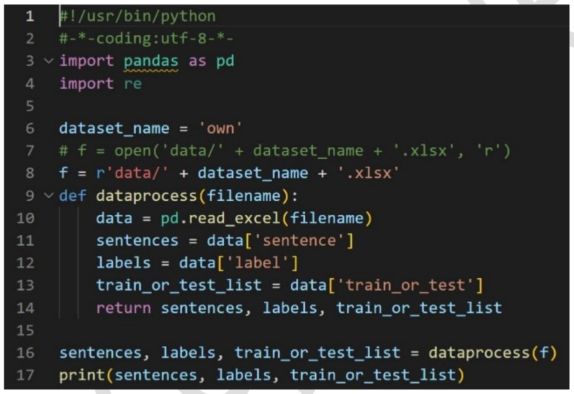
Label	Number of requirements
Masonry	14
Glass	12
Wood	10
Gypsum	6
Plastic	18
Soil	40

Table 2. Some examples of the dataset.

Index	Label	Train_or_Test	sentence
1	Wood	train	“The roof construction shall have rafter and truss ties to the wall below. Resultant uplift loads shall be transferred to

			the foundation using a continuous load path.” (International Code Council 2014)
2	Masonry	train	“The allowable compressive stress based on gross cross-sectional area of adobe shall not exceed 30 psi (207 kPa).” (International Code Council 2014)
3	Soil	test	“Waterproofing shall be applied from the bottom of the wall to not less than 12 inches (305 mm) above the maximum elevation of the ground-water table.” (International Code Council 2014)

Code modification. In order to ensure that the model can effectively work with the newly created dataset, some code modifications were necessary due to the differences between this dataset and the benchmark datasets commonly used in computer science. As shown in Figure 2, the authors performed the necessary code modifications to enable the model to access and comprehend the dataset, laying a foundation for subsequent steps.



```
#!/usr/bin/python
#-*-coding:utf-8-*-
import pandas as pd
import re

dataset_name = 'own'
# f = open('data/' + dataset_name + '.xlsx', 'r')
f = r'data/' + dataset_name + '.xlsx'
def dataprocess(filename):
    data = pd.read_excel(filename)
    sentences = data['sentence']
    labels = data['label']
    train_or_test_list = data['train_or_test']
    return sentences, labels, train_or_test_list

sentences, labels, train_or_test_list = dataprocess(f)
print(sentences, labels, train_or_test_list)
```

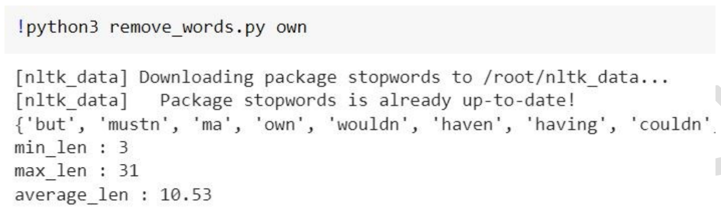
Figure 2. A screenshot of code modification written by the authors.

Data preprocessing. The data preprocessing stage comprises three key sub-steps. Firstly, the “Prepare Data” sub-step aims to provide the model with a basic understanding of the dataset by capturing information such as the number of data points, the meaning of each column, and the attributes (e.g., label, train or test) of each provision. Secondly, the “Remove Stop Words” sub-step involves removing stop words from the corpus as they do not carry significant meaning in a sentence and are not useful for the text classification task. This process helps reduce the dimensionality of the data and improve the efficiency of the model. Furthermore, the length information of the dataset is calculated after the “Remove Stop words” sub-step. Lastly, the clean dataset is converted into a graph by the model, which is represented as matrices. In this graph, the terms are depicted as nodes and their relationships are illustrated as edges.

Training and testing model. After completing the data preprocessing, the data is ready to be used for training. The authors utilized the Adam optimizer to train the model for a maximum of 200 epochs, and training is stopped if the validation loss does not decrease for 10 consecutive epochs. Upon completion of the optimization process, the model is evaluated on the testing dataset, and the results are generated automatically.

RESULTS AND DISCUSSION

The results presented in this study demonstrate the general length information of the created dataset after preprocessing steps, as illustrated in Figure 3. The dataset comprises relatively short sentences, with a minimal, maximal, and average length of provisions of 3, 31, and 10.53 words, respectively.



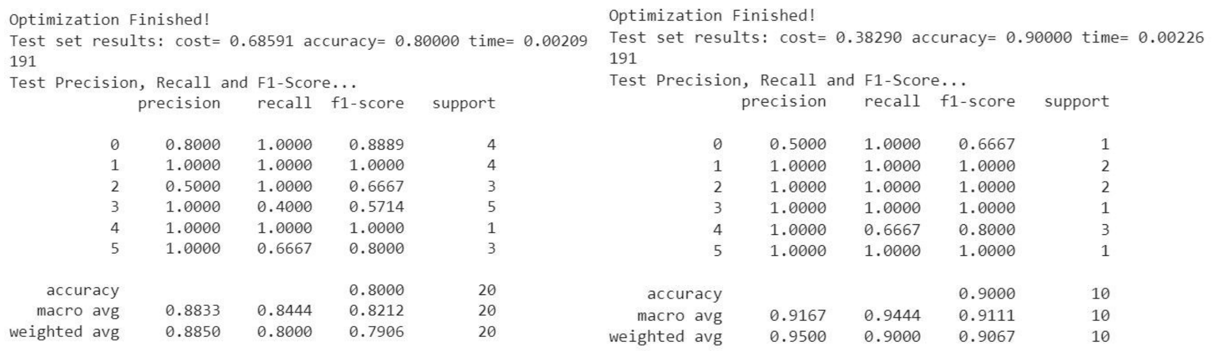
```
!python3 remove_words.py own

[nltk_data] Downloading package stopwords to /root/nltk_data...
[nltk_data]   Package stopwords is already up-to-date!
{'but', 'mustn', 'ma', 'own', 'wouldn', 'haven', 'having', 'couldn'
min_len : 3
max_len : 31
average_len : 10.53
```

Figure 3. The statistics of the sentence length of the created dataset.

For testing purposes, the authors utilized 20% and 10% of the total data and compared their results, as depicted in Figure 4. In text classification, two common techniques used to calculate the overall performance of a classifier across multiple classes are macro-average and weighted average. Macro-average computes the average performance of the classifier across all classes, with equal weight given to each class, making the performance of the classifier on small and large classes equally important. In contrast, weighted-average accounts for the class imbalance in the dataset by giving more weight to the performance of the classifier on larger classes, which contributes more to the overall performance than the smaller classes.

Experimental results reveal that increasing the size of the training data leads to a higher achievable accuracy for the test data. This finding is intuitive as a larger training dataset allows for a better understanding of the data, resulting in higher test accuracy. The best performance achieved for building code classification using 10% of the data for testing was a macro-average precision of 91.67% and a macro-average recall of 94.44%, which is comparable to the state-of-the-art machine learning-based approaches applied in construction document classification. Zhou and El-Gohary (2016a) achieved an average precision of 84.3% and an average recall of 97.30% using other machine learning models.

However, the accuracy of the testing results for the created dataset (90%) was lower than that of the benchmark dataset (average approximately 97%) (Yao et al. 2019). This discrepancy can be attributed to the shorter length of provisions in the created dataset compared to the benchmark datasets. The shorter text in the created dataset lacks linguistic structure, and legal provisions lack contextual links, making it more challenging for the model to build a graph between words and documents (Tayal et al. 2019.). For instance, the R8 dataset, which is a widely used benchmark dataset in text classification, comprises news documents from the Reuters news agency and has a manually categorized eight topics. The dataset contains 7674 news articles, with roughly equal numbers of articles for each category, and has a minimal, maximal, and average length of provisions of 4, 520, and 65.72, respectively. This shows the need and potential of further adapting GCN models to domain-specific texts such as building codes.



```
Optimization Finished!
Test set results: cost= 0.68591 accuracy= 0.80000 time= 0.00209
191
Test Precision, Recall and F1-Score...
              precision    recall  f1-score   support

           0     0.8000    1.0000    0.8889         4
           1     1.0000    1.0000    1.0000         4
           2     0.5000    1.0000    0.6667         3
           3     1.0000    0.4000    0.5714         5
           4     1.0000    1.0000    1.0000         1
           5     1.0000    0.6667    0.8000         3

    accuracy                         0.8000        20
   macro avg     0.8833    0.8444    0.8212        20
weighted avg     0.8850    0.8000    0.7906        20
```

```
Optimization Finished!
Test set results: cost= 0.38290 accuracy= 0.90000 time= 0.00226
191
Test Precision, Recall and F1-Score...
              precision    recall  f1-score   support

           0     0.5000    1.0000    0.6667         1
           1     1.0000    1.0000    1.0000         2
           2     1.0000    1.0000    1.0000         2
           3     1.0000    1.0000    1.0000         1
           4     1.0000    0.6667    0.8000         3
           5     1.0000    1.0000    1.0000         1

    accuracy                         0.9000        10
   macro avg     0.9167    0.9444    0.9111        10
weighted avg     0.9500    0.9000    0.9067        10
```

Figure 4. The performance of the model when 20% (left) or 10% (right) of all data was used for testing.

CONCLUSION

In this paper, the authors propose a novel approach to classify building code requirements using state-of-the-art graphic convolutional networks (GCN). The experimental results on building material categories classification showed that the proposed method achieved high precision (91.67%) and recall (94.44%) rates, which are comparable to the state-of-the-art construction document classification performance. The findings of this study are expected to contribute to the accurate representation of building code information and the development of automated compliance checking systems. The study also highlights the importance of utilizing advanced techniques such as GCN in text classification tasks and provides insights into the effect of training data size on testing accuracy.

To further enhance the accuracy of this model for building code classification, strategies such as increasing the dataset size and tailoring models for short text classification could be beneficial.

ACKNOWLEDGEMENT

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

REFERENCES

Akanbi, T., and J. Zhang. (2021). “Design information extraction from construction specifications to support cost estimation.” *Autom. Constr*., 131(November 2021), 103835.
Bai, X. (2018). “Text classification based on LSTM and attention.” *2018 Thirteenth International Conference on Digital Information Management (ICDIM)*, Berlin, Germany, 29-32.
Dimyadi, J., P. Pauwels, M. Spearpoint, C. Clifton, and R. Amor. (2015). “Querying a regulatory model for compliant building design audit.” *Proc., 32rd international CIB W78 conference*, Conseil International du Bâtiment (CIB), Rotterdam, The Netherlands.
Eastman, C., J. Lee, Y. Jeong, and J. Lee. (2009). “Automatic rule-based checking of building designs.” *Autom. Constr*., 18(8), 1011-1033.

International Code Council. (2014). *International Building Code 2015*. International Code Council, Country Club Hills, IL.

Lei, K., M. Qin, B. Bai, G. Zhang, and M. Yang. (2019). "GCN-GAN: A Non-linear temporal link prediction model for weighted dynamic networks." *IEEE INFOCOM 2019 - IEEE Conference on Computer Communications*, IEEE, New York, NY, 388-396.

Liu, P., X. Qiu, and X. Huang. (2016). "Recurrent neural network for text classification with multi-task learning." <https://arxiv.org/abs/1605.05101> (August 21, 2023)

Liu, X., X. You, X. Zhang, J. Wu, and P. Lv. (2020). "Tensor graph convolutional networks for text classification." *Proc., AAAI Conference on Artificial Intelligence*, 34(05), 8409-8416.

Macit İlal, S., and H. M. Günaydın. (2017). "Computer representation of building codes for automated compliance checking." *Autom. Constr*., 82(October 2017), 43-58.

Moraes, R., J. F. Valiati, and W. P. Gavião Neto. (2013). "Document-level sentiment classification: An empirical comparison between SVM and ANN." *Expert Systems with Applications*, 40(2), 621-633.

Nigam, K., A. K. Mccallum, S. Thrun, and T. Mitchell. (2000). "Text Classification from labeled and unlabeled documents using EM." *Machine Learning*, 39(2), 103-134.

Pappagari, R., P. Zelasko, J. Villalba, Y. Carmiel, and N. Dehak. (2019). "Hierarchical transformers for long document classification." *2019 IEEE Automatic Speech Recognition and Understanding Workshop (ASRU)*, IEEE, New York, NY, 838-844.

Pauwels, P., D. Van Deursen, R. Verstraeten, J. De Roo, R. De Meyer, R. Van de Walle, and J. Van Campenhout. (2011). "A semantic rule checking environment for building performance checking." *Autom. Constr*., 20 (5), 506-518.

Rasdorf, W. J., and S. Lakmazaheri. (1990). "Logic-based approach for modeling organization of design standards." *J. Comput. in Civ. Eng.,* 4(2), 102-123.

Rasdorf, W. J., and T. E. Wang. (1988). "Generic design standards processing in an expert system environment." *J. Comput. in Civ. Eng.,* 2(1), 68-87.

Razavi, A. H., and D. Inkpen. (2014). "Text representation using multi-level latent dirichlet allocation." *Advances in Artificial Intelligence*, Lecture Notes in Computer Science, Springer International Publishing, Cham, Switzerland, 215-226.

Salama, D. M., and N. M. El-Gohary. (2012). "Semantic modeling for automated compliance checking." *Proc., 2011 ASCE International Workshop on Computing in Civil Engineering*, ASCE, Reston, VA, 641-648.

Salama, D. M., and N. M. El-Gohary. (2016). "Semantic text classification for supporting automated compliance checking in construction." *J. Comput. in Civ. Eng.,* 30(1), 04014106.

Tang, H., Y. Mi, F. Xue, and Y. Cao. (2020). "An integration model based on graph convolutional network for text classification." *IEEE Access*, 8, 148865-148876.

Tayal, K., N. Rao, S. Agrawal, and K. Subbian. (2019). "Short Text Classification using graph convolutional network." *NIPS workshop on Graph Representation Learning*.

Vijayan, V. K., K. R. Bindu, and L. Parameswaran. (2017). "A comprehensive study of text classification algorithms." *2017 International Conference on Advances in Computing, Communications and Informatics (ICACCI)*, IEEE, New York, NY, 1109-1113.

Vinodhini, G., and R. M. Chandrasekaran. (2016). "A comparative performance evaluation of neural network based approach for sentiment classification of online reviews." *Journal of King Saud University - Computer and Information Sciences*, 28(1), 2-12.

Wang, S., M. Huang, and Z. Deng. (2018). "Densely connected cnn with multi-scale feature attention for text classification." *Proc., Twenty-Seventh International Joint Conferences on Artificial Intelligence*, IJCAI, CA, 4468-4474.

Wu, C.-H. (2009). "Behavior-based spam detection using a hybrid method of rule-based techniques and neural networks." *Expert Systems with Applications*, 36 (3, Part 1), 4321-4330.

Wu, J., X. Xue, and J. Zhang. (2023). "Invariant signature, logic reasoning, and semantic natural language processing (NLP)-based automated building code compliance checking (I-SNACC) framework." *ITcon*, 28(1), 1-18.

Wu, J., and J. Zhang. (2022). "Model validation using invariant signatures and logic-based inference for automated building code compliance checking." *J. Comput. in Civ. Eng.,* 36(3), 04022002.

Xu, X., and H. Cai. (2020). "Semantic approach to compliance checking of underground utilities." *Autom. Constr*., 109(January 2020), 103006.

Xue, X., and J. Zhang. (2021). "Part-of-speech tagging of building codes empowered by deep learning and transformational rules." *Adv. Eng. Inform.,* 47(January 2021), 101235.

Xue, X., J. Zhang, and N.M. El-Gohary. (2022). "Interactive visual representation of inter-connected requirements in building codes." *Proc., ASCE Construction Research Congress,* ASCE, Reston, VA, 1004-1012.

Yang, F., J. Zhang, Y. Chen, and L. Debs. (2022). "A new schema of logic representation and reasoning for automated building code compliance checking." *Proc., Polytechnic Summit 2022*, Darmstadt University of Applied Sciences, Darmstadt, Germany.

Yao, L., C. Mao, and Y. Luo. (2019). "Graph convolutional networks for text classification." *Proc., AAAI Conference on Artificial Intelligence*, 33(01), 7370-7377.

Yoon, Y., and G. G. Lee. (2007). "Efficient implementation of associative classifiers for document classification." *Information Processing & Management*, 43(2), 393-405.

Zhang, J. (2023). "How can ChatGPT help in automated building code compliance checking?" *Proc. 40th Intl. Symposium on Automation and Robotics in Construction (ISARC 2023), I.A.A.R.C., iaarc.org., 63-70.*

Zhang, J., and N. M. El-Gohary. (2015). "Automated information transformation for automated regulatory compliance checking in construction." *J. Comput. in Civ. Eng.,* 29(4), B4015001.

Zhang, J. and N. M. El-Gohary. (2016a). "Semantic-based logic representation and reasoning for automated regulatory compliance checking." *J. Comput. Civ. Eng.,* 31(1), 04016037.

Zhang, J., and N. M. El-Gohary. (2016b). "Semantic NLP-based information extraction from construction regulatory documents for automated compliance checking." *J. Comput. Civ. Eng.*, 30(2), 04015014.

Zhang, J., and N. M. El-Gohary. (2017). "Integrating semantic NLP and logic reasoning into a unified system for fully-automated code checking." *Autom. Constr*., 73(January 2017), 45-57.

Zhou, P., and N. El-Gohary. (2016a). "Domain-specific hierarchical text classification for supporting automated environmental compliance checking." *J. Comput. Civ. Eng.*, 30(4), 04015057.

Zhou, P., and N. El-Gohary. (2016b). "Ontology-based multilabel text classification of construction regulatory documents." *J. Comput. Civ. Eng.*, 30(4), 04015058.