Unsupervised Machine Learning for Augmented Data Analytics of Building Codes

Ruichuan Zhang, M.S., S.ASCE[1] and Nora El-Gohary, Ph.D., M.ASCE[2]

[1]Department of Civil and Environmental Engineering, University of Illinois at Urbana-Champaign, 205 North Mathews Avenue, Urbana, IL, 61801; e-mail: rzhang65@illinois.edu
[2] Department of Civil and Environmental Engineering, University of Illinois at Urbana-Champaign, 205 North Mathews Avenue, Urbana, IL, 61801; e-mail: gohary@illinois.edu

ABSTRACT

Existing automated code checking methods/tools are unable to automatically analyze and represent all types of requirements (e.g., requirements that are too complex or that require human judgement). Recent efforts in the area of augmented data analytics have proposed the use of templates to facilitate the analysis of text. However, most of these efforts have constructed such templates manually, which is labor-intensive. More importantly, it is difficult for manually-developed templates to capture the linguistic variations in building codes. More research is, thus, needed to automate the generation of templates to support the tagging and extraction of information from building codes. To address this need, this paper proposes an unsupervised machine-learning based method to extract sentence templates that describe syntactic and semantic features and patterns from building codes. The proposed method is composed of four main steps: (1) data preprocessing; (2) identifying the different groups of sentence fragments using clustering; (3) identifying the fixed parts and the slots in the templates based on the syntactic and semantic patterns of the sentence fragment groups; and (4) evaluating the extracted templates. The proposed method was implemented and tested on a corpus of text from the International Building Code. An accuracy of 0.76 was achieved.

INTRODUCTION

Various automatic and semi-automatic code checking methods have been developed, including methods for automated text analysis (to capture the requirements in the textual codes) and automated rule formalization (to formalize these requirements in the form of computable rules). Although these methods have achieved different levels of automation and accuracy, they have one common limitation, which is the inability to represent all types of building code requirements, especially those requirements that have complex syntactic and semantic structures, such as nested clauses and multiple proposition (verb-argument) units. Therefore, efforts must be made to enhance the analysis of building code requirement sentences before any automatic or semi-automatic compliance checking can be improved.

Recent efforts in the area of augmented data analytics have proposed the use of document and sentence templates to facilitate the analysis of natural language data. A template is usually composed of several fixed parts and slots. The fixed parts consist of words and/or phrases frequently appearing in the corpus, and the slots are labeled by the syntactic and/or semantic roles that the corresponding sentence fragments play in the sentence. Templates have been used to facilitate semantic annotation and information extraction for various applications such as document management (Arantes and Falbo 2010) and intelligent contracting (Clack et al. 2016). A limited number of template-based approaches have also been used in the construction domain for the analysis of contract specifications. For example, Ryoo et al. (2010) used premade templates to facilitate the writing and editing of construction specifications in a web-based system (Ryoo et al. 2010). Commercial software, such as e-Specs (Avitru 2018) and BIMdrive (Digicon 2018), also used templates based on the National Master Specifications for developing BIM-compatible specifications.

However, the template-based approach usually assumes that the templates are pre-defined or manually constructed by domain experts, which is labor-intensive. Such manually-developed templates are also typically rigid and static, lacking the flexibility and dynamism to capture the linguistic variations across different types of chapters/documents. Manually-developed templates are, thus, difficult to adapt from on chapter/document to another, especially when dealing with complex, technical, and/or domain-specific documents like specifications and building codes. Existing research to automate the generation of templates from text data includes automatic template extraction from human-written weather reports for summarization (Das et al. 2008), from news reports for information extraction (Chambers and Jurafsky 2011), and from emails for email auto-reply (Proskurnia et al. 2017). Generally, automatic or semi-automatic template generation approaches are composed of two primary steps: text/sentence clustering and template induction from the clusters of textual data, with the latter step consisting of two substeps to deal with the fixed parts and the slots, respectively. The objects to be clustered and the similarity measures for clustering vary from one effort to another [e.g., verbs and WordNet similarity (Das et al. 2008, Chambers and Jurafsky 2011), sentences and ROUGH similarity (Das et al. 2008), and term frequencies and Euclidean distance (Proskurnia et al. 2017)] depending on the type of text, domain/application characteristics and requirements, and template generation approach taken. Thus, there is a need to develop a domain-specific automatic template generation method to support the tagging and extraction of information from building codes.

To address this need, this paper proposes an unsupervised learning-based approach for automatic generation of templates from building code sentences. The proposed approach consists of four steps: data preparation and preprocessing; clustering of sentence fragments; induction of templates from the clusters; and evaluation of the generated templates.

BACKGROUND

Constituency parsing and shallow semantic labeling. Constituency parsing aims to organize words in a sentence into nested constituents based on phrase structures (Jurafsky 2000). The result of constituency parsing is often represented in the form of a tree, where the nodes are phrase structure categories, and the leaves are part-of-speech (POS) tags and words. Shallow sematic role labeling aims to extract the proposition units, which consist of target verbs and other constituents, where each fills a specific semantic role of the verb (Carreras and Màrquez 2005). The semantic roles include agent, patient, and also adjuncts such as location, manner, etc. The results of both techniques are usually used to gain a better understanding of the syntactic and semantic structure of a sentence, which can function as features in machine learning problems such as clustering to further analyze the natural language data.

Agglomerative hierarchical clustering. Clustering is an unsupervised learning problem that aims to find groups of similar objects in the data (Aggarwal and Zhai 2012). Clustering algorithms can be classified into several categories: agglomerative, partitioning, and probabilistic model-based algorithms. Agglomerative hierarchical clustering aims to successively merge groups of data in a pairwise manner based on the pairwise distance/similarity, until all the data are within one single group (Aggarwal and Zhai 2012).

PROPOSED UNSUPERVISED APPROACH FOR AUGMENTED DATA ANALYTICS OF BUILDING CODES

The proposed unsupervised machine learning-based approach for automatic generation of templates for supporting building code analytics consists of four main steps, as per Figure 1.

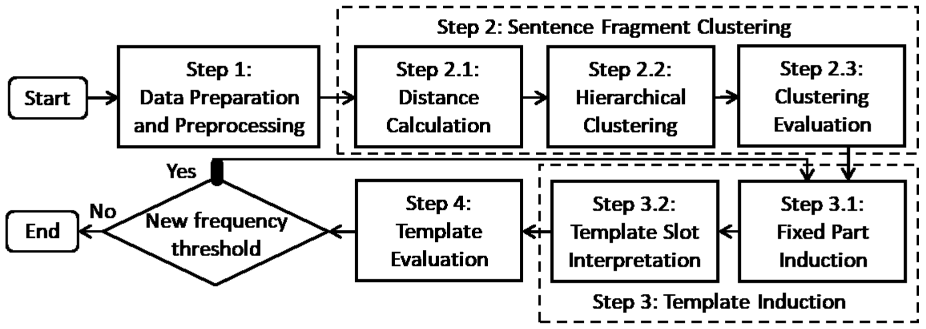

Step 2: Sentence Fragment Clustering
Start
Step 1: Data Preparation and Preprocessing
Step 2.1: Distance Calculation
Step 2.2: Hierarchical Clustering
Step 2.3: Clustering Evaluation
Yes
No
End
New frequency threshold
Step 4: Template Evaluation
Step 3.2: Template Slot Interpretation
Step 3.1: Fixed Part Induction
Step 3: Template Induction

Figure 1. Proposed unsupervised learning-based approach for augmented data analytics of building codes.

Step 1: Data Preparation and Preprocessing. Around 1,000 sentences were collected from Chapters 11 to 16 of the International Building Code (IBC) 2009 (ICC 2009). Three steps of data preprocessing were conducted: tokenization, lowercasing, and stemming. All sentences were then tagged and parsed using the Stanford CoreNLP constituency parser (Manning et al. 2014). The results of constituency parsing were used for slot interpretation when inducing the templates (Step 3.2). The sentences were also labeled using a shallow semantic role labeler, which segments a sentence into several proposition units. Each proposition unit has a verb-argument structure that is composed of: (1) one verb (V); (2) arguments (A): arguments are usually noun phrases that define the verb-specific semantic roles [e.g., the agent (A0), the patient or theme (A1), and a general argument with no specific semantic meaning (A2)]; (3) adjuncts (AM): adjuncts are adverbs, adverb phrases, and preposition phrases that describe and/or modify the verb [e.g., location (LOC), modal verb (MOD)]; and (4) references (R). The results of shallow semantic role labeling were used as features for sentence fragment clustering (Step 2) and slot interpretation (Step 3.2). Figure 2 shows an example of the semantic role labeling results.

Step 2: Sentence Fragment Clustering.
Pairwise distance calculation. Edit distance was used to describe the dissimilarity between each pair of proposition units generated by shallow semantic role labeling. In edit distance, three types of edit operations are defined: removal, insertion, and substitution (Jurafsky 2000). Each

operation adds one to the distance between the two sentences. For example, the edit distance between proposition units “A0 A1 V” and “A0 A1 LOC V” is 1 because the first unit can be converted into the second unit after one insertion operation of the tag “LOC”.

Agglomerative hierarchical clustering. Different agglomerative hierarchical clustering methods were tested and evaluated based their average silhouette coefficients (Rousseeuw 1987): single, complete, average, centroid, McQuitty, median, and Ward’s. The centroid method achieved the highest performance, with an average silhouette coefficient of 0.5, and was therefore selected. The centroid method defines the pairwise similarity as the cosine similarity between the centroids of two groups (Steinbach et al. 2000). The average silhouette coefficient is the average of the silhouette coefficients of all the data. The silhouette coefficient of a datum is computed as per the following equation, where $a(i)$ is the average difference between datum i and the other data in the same cluster, and $b(i)$ is the lowest average difference between i and the other clusters. The silhouette coefficient ranges from -1 to 1. A value near 1 indicates that the datum is far from the neighboring clusters; and a negative value indicates that the datum might be assigned to a wrong cluster. A higher average silhouette coefficient indicates better clustering result, which is essential for the following steps of template induction. If the size of a cluster is too small compared to the average size of all the clusters, the sentence fragments corresponding to the cluster are treated as outliers and are neglected in the following template generation steps. Figure 2 shows an example of the clustering results.

$$\text{silhouette(i)} = \frac{b(i) - a(i)}{\max\{a(i), b(i)\}}$$

Step 3: Template Generation.

Defining the fixed parts. A template was generated for each type of sentence or sentence fragment. The fixed parts in the templates were identified based on the frequent words. First, the frequent words are identified. Then, for each sentence fragment template, the frequent words form the fixed parts of the template. Term frequency was used to find the frequent words in each cluster. The term frequency of every word in all sentence fragments was calculated for each cluster. Articles such as “a”, “an”, and “the”, symbols, and punctuations were neglected when calculating the term frequency. A threshold n was set to define the frequent words. The top n percent of words in terms of frequency (or words with the highest frequency if the size of the vocabulary is too small) were treated as frequent words.

Interpreting and annotating the slots. A slot in a template is the non-fixed (or blank) part between two consecutive fixed parts. The complete template is, thus, a mixed sequence of both frequent words and slot labels. The slots were annotated using two sets of labels: syntactic and semantic labels. The syntactic labels are the phrasal tags at the first level below the root of the constituency parsing tree. The slots were labeled with the phrasal tags of the fragments corresponding to the slots. For example, the slot in the sentence fragment “an airspace of not less than 1 inch” corresponding to “an airspace” was labeled as NP. The semantic labels are the semantic roles based on the semantic information elements by Zhang and El-Gohary (2015) (e.g., subject, compliance checking attribute, and quantity value) and the proposition units. For example, a slot was labeled as “subject” if the corresponding fragment represents a thing (e.g., building element) that is subject to a requirement; and a slot was labeled as “location” if the slot is in the “LOC” in a proposition unit, and follows a preposition such as “in”, “above”, or “below”. Figure 2 shows an example of the template generation results.

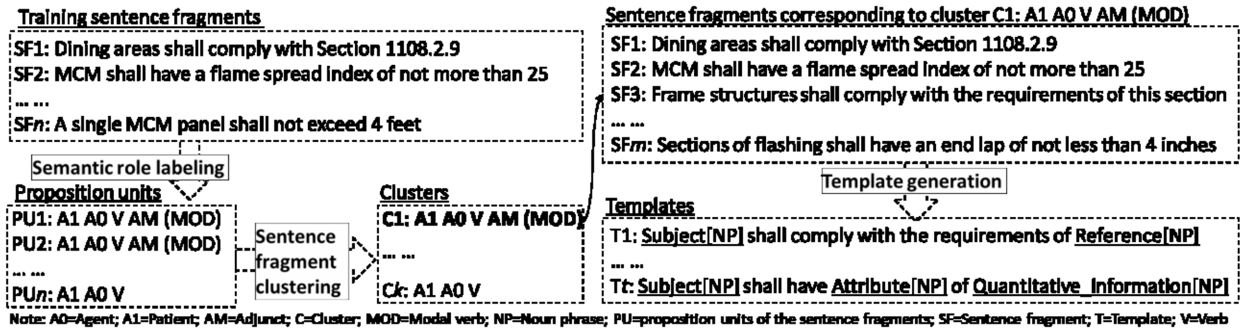

Training sentence fragments
SF1: Dining areas shall comply with Section 1108.2.9
SF2: MCM shall have a flame spread index of not more than 25
... ...
SFn: A single MCM panel shall not exceed 4 feet
Semantic role labeling
Proposition units
PU1: A1 A0 V AM (MOD)
PU2: A1 A0 V AM (MOD)
... ...
PUn: A1 A0 V
Sentence fragment clustering
Clusters
C1: A1 A0 V AM (MOD)
... ...
Ck: A1 A0 V
Sentence fragments corresponding to cluster C1: A1 A0 V AM (MOD)
SF1: Dining areas shall comply with Section 1108.2.9
SF2: MCM shall have a flame spread index of not more than 25
SF3: Frame structures shall comply with the requirements of this section
... ...
SFm: Sections of flashing shall have an end lap of not less than 4 inches
Template generation
Templates
T1: Subject[NP] shall comply with the requirements of Reference[NP]
... ...
Tt: Subject[NP] shall have Attribute[NP] of Quantitative_Information[NP]
Note: A0=Agent; A1=Patient; AM=Adjunct; C=Cluster; MOD=Modal verb; NP=Noun phrase; PU=proposition units of the sentence fragments; SF=Sentence fragment; T=Template; V=Verb

Figure 2. An example of the clustering-based template generation.

Step 4: Template Evaluation. Five metrics were used to evaluate the templates: the total number and the average length of templates, coverage, entropy, and accuracy. The total number of templates and the average length of templates are two basic indicators of the complexity of a template system. The larger the total number of templates and the average length of templates, the more complex the template system is. In addition, coverage and entropy were computed to evaluate the templates (Proskurnia et al. 2017). The coverage is how much of the training sentences can be represented using the template system. The entropy indicates how rigid/inflexible the templates are; the higher the probability of the word occurrence in the fixed parts, the higher the entropy. One important step of template generation is to find the optimal frequent-word threshold to maximize the coverage and minimize the entropy simultaneously.

Coverage was computed as per the following equation, where $l(t_i)$ is the number of words in the fixed parts of the ith template, T is the number of templates, $l(s_i)$ is total number of words in the jth sentence, and S is the number of sentences.

$$\text{coverage} = \frac{\sum_{i=1}^{T} l(t_i)}{\sum_{j=1}^{S} l(s_j)}$$

Entropy was computed as per the following equation, where $P(w_i)$ is the probability of w_i occurring in the fixed parts of the templates and is computed by dividing the word frequency of w_i by the total length of all templates, and N is the total number of frequent words.

$$\text{entropy} = -\sum_{i=1}^{N} P(w_i)\log_2 P(w_i)$$

Accuracy was computed as the portion of the testing sentence fragments that are matched to at least one of the templates to the total of testing sentence fragments. To develop the testing dataset, a total of 160 sentence fragments were randomly selected from the rest of the chapters of IBC 2009 (i.e., from Chapters 1 to 10 and Chapters 17 to 35).

PRELIMINARY EXPERIMENT RESULTS AND DISCUSSION

Sentence Fragment Clustering. A total of 3,000 sentence fragments were generated from Chapter 11 to 16 of the IBC 2009 (ICC 2009). The corresponding proposition units were clustered using the agglomerative hierarchical clustering method. The resulting average silhouette coefficient is 0.5, which indicates good clustering performance. The clusters with size lower than 30 were neglected, resulting in a total of eight clusters that were used for template induction. Table 1 shows the proposition unit structures corresponding to the eight clusters and the size of each cluster. The proposition unit roles that exist in the majority of data in the cluster are bolded.

Table 1. Proposition Unit Structures of the Clusters.

Cluster	Proposition unit structure	Cluster size
1	**A1** AM (PNC/MNR/LOC/**MOD**) **V**	243
2	**A1 A0 V AM (MOD)**	220
3	**A1 A0** A2 R **V**	109
4	**A1 A2** AM (LOC/**MOD**) **V**	389
5	**A1 A2 V**	305
6	**A1 A0 V**	414
7	**A1** AM (MNR/ LOC) **V**	985
8	**A1 A2** AM (PNC/MNR/LOC/ADV/**MOD**) V	111

Note: A0=Agent; A1=Patient; A2=General argument with no specific semantic meaning; AM=Adjunct LOC=Location; MNR=Manner; MOD=Modal verb; PNC=Purpose; R=Reference argument; V=Verb

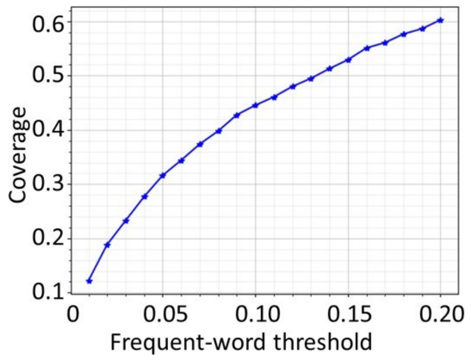
Template Development. Four example templates and the corresponding example sentences or sentence fragments that were used to generate the templates are shown in Table 2. The left side of each template is the proposition unit element and the right side includes the slots, slot labels, and the fixed parts. The sentence parts corresponding to the template slots are underlined.

Table 2. Example Templates.

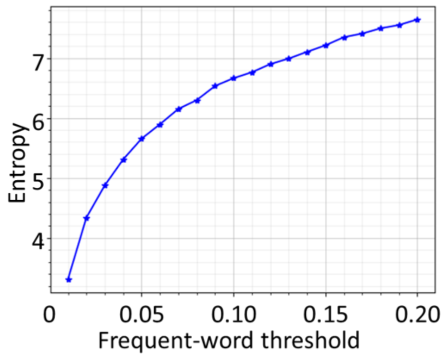
Example templates			Example sentence or fragment
A1	Slot	<u>Subject [NP]</u>	"<u>a drainage system</u> installed in accordance with <u>Sections 1805.4.2 and 1805.4.3</u>"
V	Fixed part	installed	
MNR	Fixed part	in accordance with	
	Slot	Reference [NP]	
A1	Slot	<u>Subject [NP]</u>	"<u>freestanding press boxes</u> that are <u>elevated</u> above grade <u>12</u> feet minimum"
R	Fixed part	that are	
V	Slot	<u>Quantitative relation [V]</u>	
	Fixed part	above grade	
A2	Slot	<u>Quantity value [CD]</u>	
	Fixed part	feet minimum	
LOC	Fixed part	in	"In <u>multilevel parking structures</u>, <u>van-accessible parking spaces</u> are permitted on <u>one level</u>."
	Slot	<u>Location [NP]</u>	
A1	Slot	<u>Subject [NP]</u>	
V	Fixed part	permitted	
A2	Fixed part	on	
	Slot	Location [NP]	
A0	Slot	<u>Subject [NP]</u>	"<u>A building, room or space used for assembly purposes with fixed seating</u> shall comply with <u>Sections 1108.2.1</u> through <u>1108.2.5</u>."
MOD	Fixed part	shall	
V	Fixed part	comply	
A1	Fixed part	with	
	Slot	<u>Reference [NP]</u>	
MNR	Fixed part	through	
	Slot	Reference [CD]	

Note: A0=Agent; A1=Patient; A2=General argument with no specific semantic meaning; CD=Cardinal number; LOC=Location; MNR=Manner; MOD=Modal verb; NP=Noun phrase; R=Reference argument; V=Verb

Template Evaluation. A sequence of frequent-word thresholds ranging from 0.01 to 0.2 with a step of 0.01 were tested. Based on the empirical analysis of the coverage and entropy results, a threshold value of 0.05 was found optimal. The coverage and entropy results are illustrated in Figure 3(a) and (b), respectively. As shown, the coverage and entropy of the templates increase as the threshold increases. The coverage increases because more words in the training sentences are used in the fixed parts of the templates; and the entropy increases because the rigidity of the templates increases. Similarly, the average length of the templates and the total number of templates both increase as the threshold increases. Using the testing dataset, the accuracy of the final templates is 0.76.



(a)



(b)

Figure 3. (a) Plot of coverage; (b) Plot of entropy.

CONCLUSIONS AND FUTURE WORK

This paper proposed an unsupervised learning-based template extraction approach for analyzing building code requirement sentences. The training building code sentences were first parsed by a constituency parser and labeled by a shallow semantic role labeler. The proposition units generated in the process of shallow semantic role labeling were clustered using an agglomerative hierarchical clustering method based on pairwise edit distance. The sentence fragments corresponding to the resulted clusters were then used to induce the templates: first, frequent words were found and assembled into the fixed parts of the templates; second, the slots were interpreted using syntactic and semantic labels. A number of building code requirement sentence templates were generated and evaluated using five metrics: total number of templates, average size of templates, coverage, entropy, and accuracy. An accuracy of 0.76 was achieved for the final set of templates, at a 0.05 frequent-word threshold value.

This paper contributes to the body of knowledge in two primary ways. First, these preliminary results show that the clusters learned from the syntactic and semantic features are potentially effective for template generation. Second, the results also indicate that sentence templates could help improve building code analytics by enabling the analysis of text and the extraction of information using these templates.

In their future work, the authors plan to refine the semantic annotation of the template slots using more compliance checking-related semantic roles; integrate the templates with ontologies to enhance the quality of semantic annotation for improving coverage and decreasing entropy; and leverage the templates to improve the tagging and extraction of information from building codes and thus the performance of automated/semi-automated compliance checking.

REFERENCES

Aggarwal, C.C., and Zhai, C. (2012). “A survey of text classification algorithms.” *Mining Text Data*, Springer, US, 163-222.

Arantes, L.O., and Falbo, R.A. (2010). “An infrastructure for managing semantic document.” *Proc., 14th Int. Enterprise Distributed Object Computing Conf. Workshops EDOCW*, 235-244.

Avitru. (2018). “E-Specs.” http://e-specs.com/products/e-specs-master-specification-support. (Oct 15, 2018)

Carreras, X., and Màrquez, L. (2005). “Introduction to the CoNLL-2005 shared task: semantic role labeling.” *Proc., 9th Computational Natural Language Learning Conf.*, 152-164.

Chambers, N., and Jurafsky, D. (2011). “Template-based information extraction without the templates.” *Proc., 49th Annual Meeting of the Association for Computational Linguistics: Human Language Technologies -Volume 1*, 976-986.

Clack, C.D., Bakshi, V.A., and Braine, L. (2016). “Smart contract templates: foundations, design landscape and research directions.” arXiv preprint arXiv:1608.00771.

Das, D., Kumar, M., and Rudnicky, A.I. (2008). “Automatic extraction of briefing templates.” *Proc., 3rd Int. Joint Conf. on Natural Language Processing: Volume-I.*

Digicon. (2018). “BIMdrive Specification Management Software.” http://www.digicon.ab.ca/services.aspx. (Oct 15, 2018)

ICC (International Code Council). (2009). *International Building Code 2009*. US.

Jurafsky, D. (2000). *Speech & language processing*. Pearson Education India, India.

Manning, C.D., Surdeanu, M., Bauer, J., Finkel, J.R., Bethard, S., and McClosky, D. (2014). “The Stanford CoreNLP natural language processing toolkit.” *In ACL (System Demonstrations)*, 55-60.

Proskurnia, J., Cartright, M. A., Garcia-Pueyo, L., Krka, I., Wendt, J. B., Kaufmann, T., and Miklos, B. (2017). “Template induction over unstructured email corpora.” *Proc., 26th Int. Conf. on World Wide Web*, 1521-1530.

Rousseeuw, P.J. (1987). “Silhouettes: a graphical aid to the interpretation and validation of cluster analysis.” *J. Comput. Appl. Math*, 20, 53-65.

Ryoo, B.Y., Skibniewski, M.J., and Kwak, Y.H. (2010). “Web-based construction project specification system.” *J. Comput. Civ. Eng.*, 24(2), 212-221.

Steinbach, M., Karypis, G., and Kumar, V. (2000). “A comparison of document clustering techniques.” KDD workshop on text mining, 400(1), 525-526.

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