Computer Representation of Building Codes Using NLP Techniques for Automated Compliance Checking

Melis Aldemir Toraman[1], Sibel Macit İlal[2]
[1,2]İzmir Democracy University
[1]meliss.aldemir@hotmail.com [2]sibel.macit@idu.edu.tr

Ensuring compliance with building codes is a critical yet complex and time-consuming process in the Architecture, Engineering, and Construction (AEC) industry. This study investigates the applicability of Natural Language Processing (NLP) techniques for developing computer representations of building codes to facilitate automated compliance checking. The research focuses on extracting rule expressions from the Izmir Municipality Housing and Zoning Code (IMHZCode) using NLP techniques. The methodology involves syntactic analysis, concept extraction, ontology-based knowledge structuring, and rule representation using the Zemberek NLP tool. Rule expressions are identified, analyzed morphologically, and transformed into structured formats containing concepts, attributes, operators, values, and units. The study demonstrates that NLP techniques can effectively transform textual building codes into structured, machine-readable formats, paving the way for integration with BIM-based workflows. Future work should focus on analyzing complex linguistic structures and explore advanced semantic parsing to further enhance the automation of building code compliance checking.

Keywords: *Building Code Representation, NLP, Automated Compliance Checking.*

INTRODUCTION

In the Architecture, Engineering, and Construction (AEC) industry, various technical standards and codes exist to ensure that buildings are designed and constructed to provide safe, healthy, and comfortable environments. Every building project is checked for compliance with relevant codes. However, building code compliance checking is a highly complex and time-consuming process which is usually manual. Possibility of mistakes due to human factors is high, leading to lost time and increased costs.

There are various studies in the literature on the automation of building code compliance checking process (Dimyadi and Amor, 2013; Aldemir Toraman and Macit İlal, 2023). Automating this process has the potential to ensure error-free compliance checks and improve efficiency in terms of time and cost. Research in this field focuses on the four fundamental stages of the process: developing computer representations of building codes, defining digital building information models, developing compliance checking algorithms, and reporting results.

The first stage in the automated building code compliance checking process is the development of a computer representation for building codes. Building codes are textual documents written in natural language with complex structures. To develop computer representations of building codes, rule expressions must be systematically extracted from these texts. In current approaches, this process is mostly carried out manually,

requiring expertise, hindering the fast and efficient progress of the process. Natural Language Processing (NLP) techniques, which enable computer systems to understand and process natural language, can be effectively used in such complex text analysis processes. However, various challenges exist, including the complexity of building codes, ambiguous expressions, domain-specific terminology, and the need for contextual analysis.

This paper aims to evaluate the applicability of integrating NLP techniques into the process of developing computer representations of building codes by reviewing existing approaches. The study focuses on using NLP techniques to automatically extract rule expressions from building code texts. As an example, the Izmir Municipality Building and Zoning Codes (IMHZCode) is analyzed. The research involves selecting the most suitable NLP tool for the Turkish language, conducting a syntactic analysis of the building code, extracting key concepts, creating a domain ontology, and performing automatic rule extraction to structure these rules systematically.

BUILDING CODE MODELS

Early research on automated code compliance checking in the AEC industry focused on enhancing the logical structure of building codes by transforming them from textual documents into machine-readable and processable formats. This transformation requires systematically structuring rule expressions to enable computational interpretation. Various models have been proposed to represent building codes in a computable format for automated compliance checking.

Decision Tables: The first study on representation of building codes was conducted by Fenves (1966). He used decision tables to represent building code rules. However, this method was inadequate in representing the relationships between rules.

Rule-Based Models: To address this limitation, rule-based models were introduced, where building code provisions were expressed using an "IF-THEN" logic structure (Garrett & Fenves, 1987). Despite its straightforward approach, this model failed to fully represent the organizational structure of regulations.

Logic-Based Models: An alternative approach involved modeling building code rules using predicate logic (Rasdorf & Lakmazaheri, 1990). While this method allowed user-defined predicates and logical operators, its practical application was hindered by complexity and implementation challenges.

Object-Based Models: Garrett and Hakim (1992) proposed representing building codes using an object-oriented framework, enabling better integration with Building Information Models (BIM). However, this approach required expertise in object-oriented programming, limiting its practical adoption.

Semantic Models: More recent approaches have focused on semantic representation of building codes. The RASE model proposed by Nisbet et al. (2009) introduced an annotation-based method for digital representation. Beach et al (2015) further refined semantic annotation techniques using predefined ontologies, while Solihin and Eastman (2015) developed a rule classification and semantic knowledge representation framework for BIM integration.

Hybrid Models: These models combine previous representation approaches to overcome individual limitations. Yabuki and Law (1996) developed an Object-Logic model that combines logic-based and object-based modeling. Kiliccote (1997) proposed a context-focused model using an object-based approach. Macit (2014) introduced a four-layered modeling framework integrating semantic modeling principles.

A major challenge in automating code compliance checking remains the manual extraction of information from continuously updated and text-based building codes. This

study investigates the applicability of NLP techniques for computer representation of building codes.

NATURAL LANGUAGE PROCESSING (NLP)

In recent years, AI-based methods have been used for automatic information extraction from building codes (Zhang and El-Gohary, 2012; Hettiarachchi et al, 2024). NLP techniques facilitate the automated interpretation and application of building codes by extracting meaningful information from texts.

NLP enables computers to process and understand natural language text and speech in a human-like manner (Zhang & El-Gohary, 2012). As a subfield of artificial intelligence, NLP also incorporates machine learning and deep learning methods (Yüksel & Karabıyık, 2022).

NLP operates at four levels:

- Lexical: Understanding the meanings of words.
- Syntactic: Analyzing the structure of sentences.
- Semantic: Extracting the meaning of sentences.
- Pragmatic: Interpreting words and meanings in context.

NLP also includes text generation, where the effort put into understanding natural language sentences is used to produce meaningful and grammatically correct sentences.

Challenges of Turkish in NLP

Turkish poses several challenges for natural language processing (NLP) due to its agglutinative and productive nature. The language allows the sequential attachment of numerous derivational and inflectional suffixes to root words, enabling the generation of almost unlimited number of new words. Additionally, Turkish has a highly flexible syntax, where elements such as subjects, objects, and adjuncts can appear in varying positions while meaning is preserved through morphological markers.

This flexibility introduces difficulties in applications such as machine translation, semantic parsing, and statistical modeling, often leading to data sparsity and ambiguity issues. The free word order makes it particularly challenging to infer intra-sentence relationships based solely on word order. These characteristics necessitate that Turkish be processed using structurally aware methods rather than classical NLP approaches.

Applications of NLP in the AEC industry

NLP techniques have been used in various areas of the AEC industry, including compliance checking by extracting information from building codes. Table 1 shows some applications of NLP in the AEC.

NLP tools for Turkish language processing

The accurate and meaningful grammatical analysis of textual data is a crucial step in representing building codes in a computable format. Natural Language Processing (NLP) tools play a fundamental role in this process, as they enable the linguistic structuring and semantic interpretation of textual content. The selection of an appropriate NLP tool depends on the linguistic characteristics of the language being process. Given the agglutinative structure and rich morphological features of Turkish, it is essential to employ NLP tools specifically designed or adapted for Turkish language processing. This section provides an overview of selected NLP tools that support Turkish language processing. Several NLP tools have been developed to process Turkish, each with distinct capabilities and limitations.

Title	Author(s)	Year
Automated classification of construction project documents	Caldas, Soibelman, & Han	2002
Syntactic approach to extracting key elements in change-order documents	Ko & Jeong	2020
Development of an automatic-extraction model for contract clauses using rule-based NLP	Lee, Yi, & Son	2019
Automated content analysis for construction safety	Tixier et al.	2016
Semantic NLP-based information extraction from construction regulations	Zhang & El-Gohary	2016

Table 1
Research on NLP in the AEC industry

Natural Language Toolkit (NLTK): NLTK is an open-source NLP library developed in Python and compatible with multiple operating systems. It provides fundamental text-processing functionalities such as stop-word removal, stemming, and sentiment analysis (Bird, 2019). However, its support for Turkish is relatively limited, making it less suitable for comprehensive Turkish language processing tasks.

StanfordNLP: Developed in Java by researchers at Stanford University, StanfordNLP is a multilingual NLP library designed for various text-processing tasks, including named entity recognition and sentiment analysis (McClosky & Bauer, 2014). While it includes preliminary support for Turkish, its capabilities remain restricted.

SpaCy: SpaCy is an open-source NLP library written in Python, designed to process entire documents efficiently without sentence segmentation (Honnibal & Montani, 2017). Although it offers multilingual support, its Turkish language processing functionalities are relatively limited compared to tools specifically developed for Turkish.

ITU Turkish Natural Language Processing Software Chain: Developed by researchers at Istanbul Technical University, this toolchain provides a range of NLP functionalities. However, it is not open-source, and access is primarily limited to demonstrations (Eryiğit, 2014).

Zemberek NLP: Zemberek is an open-source Turkish NLP tool developed in Java. It offers extensive features tailored to Turkish, including morphological analysis, spell checking, syllabification, and error correction (Zemberek, 2007).

Due to the agglutinative nature of Turkish, morphological disambiguation and compound sentence parsing remain ongoing challenges, which are only partially addressed by current tools. Within the scope of this study, Zemberek NLP was selected as the NLP tool. Its advantages include its open-source nature, its explicit focus on the Turkish language, and its ability to perform essential linguistic analysis tasks with high accuracy. These characteristics make Zemberek NLP a suitable choice for facilitating the automated representation of building codes in Turkish.

EXTRACTING INFORMATION FROM THE IZMIR MUNICIPALITY HOUSING AND ZONING CODE USING NLP

This study focuses on the use of NLP techniques for the automatic extraction of rule expressions from building code texts. As an application example, the Izmir Municipality Housing and Zoning Code (IMHZCode) was examined.

To represent building codes in a computable format, rule expressions must first be systematically analyzed. In this context, computer

representation of the IMHZCode using NLP is carried out in four stages:

- Analysis of the Building Code and Creation of a Dictionary
- Morphological Analysis with Zemberek NLP
- Defining the Context of Rule Expressions
- Structured Representation of Rule Sentences

To assess the effectiveness of the proposed NLP-based rule extraction, a preliminary evaluation was conducted. A sample of extracted rules was manually validated according to accuracy and precision by domain experts. Future studies should expand this evaluation with larger rule sets and comparative benchmarks against existing systems.

The methodology includes concept extraction, hierarchical relationship definition, syntactic analysis, and rule structuring using Zemberek NLP to enhance automation in building code compliance checking.

Analysis of the building code and creation of a dictionary

The analysis of IMHZCode has been carried out in two stages:

- Determining the types of rule sentences according to their structures
- Creating lists of concepts, attributes, operators, values, and units.

IMHZCode includes 233 rule sentences pertinent to buildings. These rules have been grammatically classified according to their sentences structures (Table 2)

Table 2
Types of rule sentences according to their structures

Sentence Type	Total Count
Simple Sentence	69
Compound Sentence	132
Complex Sentence	17
Compound-Complex Sentence	15

Out of a total of 233 sentences, 69 are simple sentences. Simple sentences contain a single clause and are structurally straightforward with a single predicate, making them easier to parse. Their semantic relationships are not complex. The proposed approach aims to model these simple sentence types. A large portion of the sentences are compound. Compound sentences include verbal nouns or participles that express additional actions or clauses beyond the main predicate. This adds structural complexity and makes parsing more difficult.

Zemberek NLP can detect suffixes like "-dığı," "-ınca," or "-madan" at the morphological level to identify such participial structures, but it does not interpret the logical relationship these suffixes convey (such as time or cause-effect). This makes modeling the semantic function of subordinate clauses more challenging.

The remaining 17 complex and 15 compound-complex sentences contain multiple clauses. Complex are linked by commas or semicolons, while compound-complex sentences are connected using conjunctions. These structures consist of logically independent clauses that are syntactically connected. Especially in compound-complex sentences, identifying the logical relationships created by conjunctions (contrast, cause-effect, etc.) can be difficult for the current system to handle.

The IMHZCode has been analyzed to create lists of concepts, attributes, operators, values, and units.

Concept List: All concepts found in the rule expressions of the IMHZCode were identified, and a "Concept List" specific to the building code was created. Concepts are expressions that represent a physical building element or architectural component (e.g., "Building," "Parcel," "Roof," "Door," "Window," "Staircase").

Defining concepts alone is not sufficient for meaningful information extraction from building codes; hierarchical relationships between concepts must also be established. A hierarchical

relationship represents a conceptual structure arranged from general to specific. A concept can be general, encompassing sub-concepts, or it can be a specific concept defining a particular situation or physical component. Establishing this hierarchical structure is important in the information extraction process because rules related to a parent concept can be generalized to its sub-concepts or restricted by specific rules. For example, the concept of "Building" is a general concept, and based on the "Usage" attribute, buildings can be categorized as "Official Building," "Public Building," and "Existing Building," forming sub-concepts of the "Building" concept. Thus, some rules may apply only to specific types of buildings, while others can be generalized to all buildings.

Attribute List: Quantifiable and semantic attributes of the concepts were identified, and an "Attribute List" was created. Each concept may have one or more attributes. For example, the "Building" concept may have quantifiable attributes such as "Height" and "Depth," as well as semantic attributes such as "Building Type." These attributes are used to determine which aspects of a concept are being examined during information extraction.

Operator List: Expressions used for comparisons in rule sentences were analyzed. These expressions were matched with numerical operators required for compliance checks. For example:

- "cannot exceed" <=
- "cannot be less than" >=
- "must be equal to" =
- "does not exist" = 0

Unit List: The unit list includes measurement units associated with the values mentioned in rule sentences. For example, units such as meters, centimeters, and square meters are included in this list. The different ways units appear in sentences were considered to ensure correct matching.

All these lists were defined as arrays using the Java programming language and made compatible with the root words obtained through lemmatization using the Zemberek NLP tool. This dictionary structure forms the fundamental infrastructure for the information extraction algorithm to analyze rule sentences and convert them into numerical formats.

Morphological analysis with Zemberek NLP

In this step, rule expressions from the building code were subjected to morphological analysis using the Zemberek NLP library. Morphological analysis aims to recognize different forms of a word and match them in a dictionary (Fautsch & Savoy, 2009). Thus, in the proposed domain dictionary, only the root form of a word is defined, and words found in rule sentences with inflections are converted to their root form using the "Root Finding and Lemmatization" step of the Zemberek NLP library and then matched with the dictionary.

Tokenization: In this step, sentences are broken down into words. This process aims to divide a sentence into meaningful linguistic units (tokens). Each word is defined as an independent analysis unit. After this step, subsequent processes such as part-of-speech tagging and root finding can be applied separately to individual words.

For example, the sentence "*Kapıların yüksekliği 2.10 metreden az olamaz*." was tokenized using the Zemberek NLP library, and the results are shown in Figure 1. (English: "Height of doors cannot be less than 2.10 m."

```
=== Tokenizasyon ==
Token: Kapıların
Token: yüksekliği
Token: 2.10
Token: metreden
Token: az
Token: olamaz
Token: .
```

Figure 1
Example of tokenization step with Zemberek NLP

Part-of-Speech (POS) Tagging: In this step, each word in the sentence is tagged according to its grammatical role. For example, a word is labeled as a noun, verb, adjective, or adverb based on its grammatical role. These tags help determine the function of the word in the sentence during the information extraction process.

For the example rule sentence, POS tagging was performed using the Zemberek NLP library, and the results are shown in Figure 2.

Figure 2
Example of PoS tagging with Zemberek NLP

```
=== Konuşma Parçası (POS) Bulma ===
Kapıların : Noun
yüksekliği : Noun
2.10 : Numeral
metreden : Noun
az : Adjective
olamaz : Verb
. : Punctuation
```

Root Finding and Lemmatization: In this step, words are stripped of derivational and inflectional suffixes and converted into their root forms. This process is known as root finding and lemmatization. The goal is to obtain the fundamental dictionary form of a word. For example, the word "height" ("yüksekliği") can be identified by the system as its root forms "high" ("yüksek") or "height" ("yükseklik"). These root words are then matched with previously created concept and attribute dictionaries, forming the foundation for information extraction.

For the example rule sentence, root finding and lemmatization were performed using the Zemberek NLP library, and the results are shown in Figure 3.

Figure 3
Example of root finding with Zemberek NLP

```
=== Kök Bulma ve Lemmalaştırma ===
Kelime: Kapıların
[kapı:Noun] kapı:Noun+lar:A3pl+ın:Gen
        Stems = [kapı]
        Lemmas = [kapı]
[kapı:Noun] kapı:Noun+lar:A3pl+ın:P2sg
        Stems = [kapı]
        Lemmas = [kapı]
```

Defining the Context of Rule Expressions

The goal of this step is to ensure that rule expressions are analyzed not only at the word level but also at the semantic and contextual levels, transforming them into a structured format. Each rule sentence contains components such as concept, attribute, operator, value, and unit, and for these components to be recognized by the system, the sentence must be analyzed contextually.

Example Rule Sentence: "*Height of doors cannot be less than 2.10 meters*."

- Concept List : **door**, building, staircase, …
- Attribute List : width, **height**, depth, …
- Operator List : **"cannot be less than" >=** "cannot be more than" <=
- Value List : **2.10**
- Unit List : **meter**, m^2, …

Table 3
Defining the Context of Rule Expressions

Word	**Root**	**POS**	**Matched List**
Doors (Kapıların)	door (kapı)	Noun	Concept
Height (yüksekliği)	height (yükseklik)	Noun	Attribute
2.10	-	Num	Value
meters (metreden)	meter (metre)	Noun	Unit
less (az)	less (az)	Adj	-
cannot be (olamaz)	be (ol)	Verb	Operator (>=)

In some rule expressions, meaning is conveyed not by a single word but by a phrase. This is particularly true for comparison operators (e.g., "cannot be less than," "cannot be more than," "must be equal to"), which consist of two or more words. Therefore, n-gram scanning is applied in the system when matching words.

For example, in the phrase: "cannot be less than" ("az olamaz")

The word "less" ("az") alone does not exist in the operator list. The system then considers the next word "cannot be" ("olamaz") to form the phrase "cannot be less than" ("az olamaz").

Once "cannot be less than" is found in the operator list, it is matched, and its corresponding numerical representation (>=) is used.

Structured Representation of Rule Sentences

As a result of the analysis, the sentence has been structured as follows:

Information extraction for the rule sentence: "The height of the doors cannot be less than 2.10 meters."

```
{
  "concept": "door",
  "attribute": "height",
  "operator": ">=",
  "value": 2.10,
  "unit": "meter"
}
```

This representation enables building codes expressed in natural language to be converted into a computable format that can be processed by computers. Structured representation of rule sentences allows computers to logically check building code rules and perform automatic compliance checking on building information models of projects.

CONCLUSION

This study evaluates the applicability of natural language processing (NLP) techniques in developing computer representations of building codes in Turkish, which is one of the key requirements for automated compliance checking. Within the scope of the research, NLP techniques were used to extract rule expressions from the IMHZCode, and a structured representation of these rules was created. The study demonstrates that NLP techniques can be utilized for the automatic extraction of rule expressions from written texts in the process of developing building code representations. In particular, syntactic analysis, concept extraction, and ontology-based information extraction have been observed to contribute significantly to the systematic structuring of rule expressions.

The main contribution lies in demonstrating a structured methodology for Turkish building code parsing using the Zemberek NLP tool. While promising, the approach requires further validation and testing across diverse and complex rule types. Limitations include reliance on rules with simple sentence structures and lack of comparative benchmarking. Future work should address these gaps to evaluate performance and improve generalizability.

Automating information extraction from textual documents in the process of creating digital representations of building codes is a crucial but highly challenging task for the automation of compliance checking. Challenges such as the complexity of building code texts, ambiguities, and the necessity for contextual analysis can limit the application of NLP techniques in this domain. Therefore, future studies should explore the use of machine learning and deep learning-based approaches for building code analysis. Additionally, beyond extracting information from simple sentence structures, NLP tools should be further investigated for analyzing relative clauses, compound sentences, and other complex sentence structures to improve the generation of digital rule representations.

REFERENCES

Aldemir Toraman, M., and Macit İlal, S. (2023). 'A Bibliometric analysis of published research in the field of building regulations automated compliance checking'. in *Proceedings of SPAD'23*, 15-17 December 2023, İzmir, Türkiye, 111-123.

Beach, T. H., Rezgui, Y., Li, H., and Kasim, T. (2015). 'A rule-based semantic approach for automated regulatory compliance in the

construction sector'. *Expert Systems with Applications*, 42(12), 5219-5231.

Bird, (2019). Natural Language Toolkit, http://www.nltk.org (Accessed 1 April 2025)

Caldas, C. H., Soibelman, L., and Han, J. (2002). 'Automated classification of construction project documents'. *Journal of Computing in Civil Engineering*, 16(4), 234-243.

Dimyadi, J., and Amor, R. (2013). 'Automated building code compliance checking– where is it, at.' in *Proceedings of CIB WBC*, 6(1), 172-185.

Eryiğit, G. (2014). 'ITU Turkish NLP web service'. *Proceedings of the Demonstrations at the 14th Conference of the European Chapter of the Association for Computational Linguistics*, pp. 1-4.

Fenves, S.J. (1966) *Tabular decision logic for structural design*, J. Struct. Div. ASCE 92, 473–490.

Garrett, Jr., J.H.; Hakim, M.M. (1992) 'An object-oriented model of engineering design standards', *Journal of Computing in Civil Engineering*, 6, 3, 323–347

Garrett, J.H. and Fenves, S.J. (1987) 'A knowledge-based standards processor for structural component design,' *Engineering with Computers*, 2(4), pp. 219–238. https://doi.org/10.1007/bf01276414.

Hettiarachchi, H., Dridi, A., Gaber, M. M., Parsafard, P., Bocaneala, N., Breitenfelder, K., Costa, G., Hedblom, M., Juganaru-Mathieu, M., Mecharnia, T., Park, S., Tan, H., Tawil, A. H., and Vakaj, E. (2025) 'CODE-ACCORD: A Corpus of building regulatory data for rule generation towards automatic compliance checking,' *Scientific Data*, 12(1). https://doi.org/10.1038/s41597-024-04320-x.

Honnibal, Matthew, (2017). *Spacy industrial-strength natural language processing in Python*, https://spacy.io/ (Accessed 1 April 2025)

Kiliccote, H. (1997) *A Standards processing framework in department of civil and environmental engineering*, Carnegie Mellon University, Pittsburgh, p.304

Ko, T. and Jeong, H.D. (2020b) 'Syntactic approach to extracting key elements of work modification cause in Change-Order documents,' *Construction Research Congress 2022*, pp. 134–142. https://doi.org/10.1061/9780784482889.015.

Lee, J., Yi, J.-S. and Son, J. (2019b) 'Development of Automatic-Extraction Model of Poisonous Clauses in international construction contracts using Rule-Based NLP,' *Journal of Computing in Civil Engineering*, 33(3). https://doi.org/10.1061/(asce)cp.1943-5487.0000807.

Macit, S. (2014) *Computer representation of building codes for automated compliance checking*, İzmir Institute of Technology, p.172

McClosky, and Bauer, (2014). Stanford CoreNLP: A Java suite of core NLP tools. https://github.com/stanfordnlp/CoreNLP (Accessed 1 April 2025)

Nisbet, N., Wix, J. and Conover, D. (2009) 'The future of virtual construction and regulation checking,' in *Blackwell Publishing Ltd eBooks*, pp. 241–250. https://doi.org/10.1002/9781444302349.ch17.

Rasdorf, W.J. and Lakmazaheri, S. (1990) 'Logic-Based approach for modeling organization of design standards,' *Journal of Computing in Civil Engineering*, 4(2), pp. 102–123. https://doi.org/10.1061/(asce)0887-3801(1990)4:2(102.

Solihin, W. and Eastman, C. (2015) 'Classification of rules for automated BIM rule checking development,' *Automation in Construction*, 53, pp. 69–82. https://doi.org/10.1016/j.autcon.2015.03.003.

Tixier, A. J. P., Hallowell, M. R., Rajagopalan, B., and Bowman, D. (2016). 'Automated content analysis for construction safety: A natural language processing system to extract precursors and outcomes from unstructured injury reports,' *Automation in Construction*, 62, pp. 45–56. https://doi.org/10.1016/j.autcon.2015.11.001.

Yabuki, N. and Law, K.H. (1993) 'An Object-Logic model for the representation and processing of design standards,' *Engineering With Computers*, 9(3), pp. 133–159. https://doi.org/10.1007/bf01206345.

Yüksel, A.S. and Karabiyik, M.A. (2022) 'Doğal dil işleme yöntemleriyle metinden SQL sorgusu tahmini üzerine bir çalışma,' *Ömer Halisdemir Üniversitesi Mühendislik Bilimleri Dergisi* [Preprint]. https://doi.org/10.28948/ngumuh.1120264.

Zemberek, (2007). NLP tools for Turkish. https://github.com/ahmetaa/zemberek-nlp (Accessed 1 April 2025)

Zhang, J. and El-Gohary, N. (2012) 'Extraction of Construction Regulatory Requirements from Textual Documents Using Natural Language Processing Techniques,' *Computing in Civil Engineering* [Preprint]. https://doi.org/10.1061/9780784412343.0057.

Zhang, J. and El-Gohary, N.M. (2013) 'Semantic NLP-Based Information Extraction from Construction Regulatory Documents for Automated Compliance Checking,' *Journal of Computing in Civil Engineering*, 30(2). https://doi.org/10.1061/(asce)cp.1943-5487.0000346.