Lex2KG: Automatic Conversion of Legal Documents to Knowledge Graph

Muhamad Abdurahman*, Fariz Darari (✉)*, Hans Lesmana†,
Muhtar Hartopo†, Immanuel Rhesa†, Berty Chrismartin Lumban Tobing†
*Faculty of Computer Science, Universitas Indonesia
Depok, Indonesia
fariz@ui.ac.id
†CATAPA
Jakarta, Indonesia

2021 International Conference on Advanced Computer Science and Information Systems (ICACSIS) | 978-1-6654-4264-0/21/$31.00 ©2021 IEEE | DOI: 10.1109/ICACSIS53237.2021.9631310

***Abstract*—Legal documents are generally available in the form of PDF which is not machine-readable. A knowledge graph (KG) is a graph describing real-world entities and their relationships, providing structured, machine-readable information. In this paper, we present Lex2KG, a framework for converting legal (PDF) documents into a KG. The legal KG contains various kinds of structured data, such as metadata, document structures, textual content, and relations between legal resources (e.g., amendments and citations). Through Lex2KG, we have successfully converted 784 Indonesian laws into a KG with a total size of over 1.1 million triples. We also present use cases of the legal KG for SPARQL querying, simple chatbots, and legal visualizations, showing how the legal KG generated can be useful in practice.**

***Keywords*—Law, PDF, Conversion, RDF, SPARQL, Legal Documents, Knowledge Graph**

I. Introduction

In the law history, the Hammurabi's Code is considered to be one of the oldest laws in the world, carved on a black stone monument and established during the reign of Hammurabi (1795–1750 BC) [1]. Fast forward to today, laws are typically available in the PDF format,[1] facilitating easy access to humans. The advancement of digital and artificial intelligence (AI) technologies has called for the proliferation of computational law, concerning the provision of laws in a form that is also machine-readable in order to support complex, automated legal processing [2]. To that end, knowledge graphs (KGs) [3], which provide (structured) descriptions about entities of interest and relationships among them, can be leveraged as a viable means. Indeed, in the enterprise context (e.g., Google, Microsoft, and IBM), the use of KGs has been shown successful in delivering knowledge-based features within company products [4]. In the legal context, a few initiatives have existed to foster advancements based on KG technologies, which we will elaborate on below.

Related Work. The European Legislation Identifier (ELI) initiative seeks to achieve a common representation of legislation metadata across European Union (EU) countries [5]. ELI has inspired the development of the legal extension of Schema.org,[2] enhancing the flexibility in adding legislation metadata to web pages. Through ELI and Schema.org, one can offer legislation metadata in the machine-friendly format of Resource Description Framework in Attributes (RDFa)[3] and JavaScript Object Notation for Linked Data (JSON-LD).[4] Potential use cases of such legal KGs include smart compliance services [6], contract analysis [7], and similar legal case matching [8].

Both ELI and the legal extension of Schema.org concern more on the (manual) provision of legal metadata. To the best of our knowledge, however, gaps still exist as to how not only the metadata, but also document structuring, text content, and legal relationships, can be automatically extracted from

[1]For example, Indonesian laws in PDF format are available online at https://www.dpr.go.id/jdih/uu/year/2020.

[2]https://schema.org/Legislation

[3]https://www.w3.org/TR/rdfa-core/

[4]https://www.w3.org/TR/json-ld11/

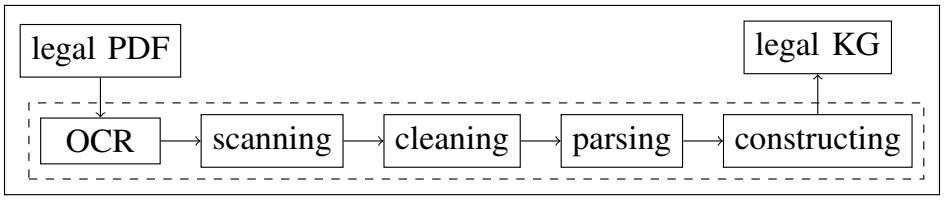


Fig. 1. Lex2KG Architecture

legal documents in the PDF format and provided as a legal KG.

Contributions. In this paper, we present Lex2KG,[5] a framework for automated conversion of legal documents to a legal KG. The Lex2KG framework aims to extract the metadata, structure, and textual content of legal documents, particularly laws, as well as relationships among legal resources, and present the extracted information as a (legal) KG. We highlight industry-inspired use cases to show the benefits of the generated legal KG in legal querying using SPARQL, simple legal chatbot, and legal data visualization.

At the moment, we focus on building a legal KG of Indonesian laws, though we envision that Lex2KG can also be adopted for laws of other countries. Also, handling other types of legal documents beside laws (in Indonesian, Undang-Undang) is in our to-do list.

Outline. The rest of the paper is organized as follows. Section II presents the Lex2KG framework, describing its architecture, parsing program, schema, and data. Section III highlights use case evaluation of the Lex2KG framework. We conclude our work in Section IV.

II. Lex2KG Framework

This section discusses the Lex2KG framework. First, we describe the architecture, followed by the parsing program. Then, we present the schema and data of the Lex2KG framework.

A. Architecture

The process of converting legal PDF documents to a legal KG follows the architecture as shown in Fig. 1.

[5]"Lex" is the Latin word for "law".

The optical character recognition (OCR) part is responsible for transforming image-only PDFs to text-scannable ones and standardizing the formatting of all PDFs. Next, we perform scanning over the PDF files to extract the raw text and its positioning. We call every scanned text line as span. Afterwards, spans that are identified as irrelevant content, such as repeated headers and footers, are filtered out. The resulting, clean spans, are then parsed into meaningful, structured (legal) data.

Finally, we construct a legal KG from the parsing result. The construction is done by generating a URI for each resource in the parsing result, assigning respective classes into those resources, and converting the data about those resources into triples. Each legal KG generated from legal PDF documents will be merged into one global KG.

B. Parsing Program

Here, we describe how our program parses legal text into structured data. A span consists of raw text and its position. Spans are grouped into (legal) components such as "Chapter 1", "Chapter 2", and so on. A component may contain subcomponents (which they themselves may contain subsubcomponents, and so on); for example, "Chapter 1" may contain "Article 1", which itself may contain "Section 1". We call the first span in the component the start span.

The program iterates over spans, and whenever a new start span is detected, all spans from the latest start span up to (but not including) the new start span will be put into a group. The detection of start spans is done by regex matching and positional checking of the text. Once all spans are grouped into components, the program then iterates over the spans in each component to find subcomponents. This grouping process will be repeated until there

```
@prefix dct:      <http://purl.org/dc/terms/> .
@prefix o:        <https://example.org/lex2kg/ontology/> .
@prefix owl:      <http://www.w3.org/2002/07/owl#> .
@prefix rdf:      <http://www.w3.org/1999/02/22-rdf-syntax-ns#> .
@prefix rdfs:     <http://www.w3.org/2000/01/rdf-schema#> .
@prefix schema:   <https://schema.org/> .

<https://example.org/lex2kg/ontology> a owl:Ontology ;
    dct:title "Lex2KG Ontology" .

o:Legislation a rdfs:Class ;
    rdfs:label "Legislation"@en , "Peraturan"@id ;
    rdfs:subClassOf schema:Legislation .

o:partOf a rdf:Property ;
    rdfs:label "partOf"@en , "bagianDari"@id ;
    rdfs:subPropertyOf  schema:isPartOf .

o:cites a rdf:Property ;
    rdfs:label "cites"@en , "merujuk"@id ;
    rdfs:subPropertyOf  schema:citation .

o:text a rdf:Property ;
    rdfs:label "text"@en , "teks"@id ;
    rdfs:subPropertyOf  schema:text .
```

Fig. 2. RDF Turtle Snippet of the Lex2KG Schema

are no more new subsubcomponents (or even deeper components) detected.

We can imagine the parsing result as a tree, where non-leaf nodes represent the structure of the document (sub*-components), and leaf-nodes represent the data of the document, such as metadata and textual content. Finally, the program extracts the citation and amendment data among legal components using regex matching complemented with positional checking.

The capability of the parsing program is limited to regex and positional checking. Hence, complex data such as non-regular language can not be parsed. Also, as positional checking expects equivalent patterns to be located on the same position, poorly scanned documents may fail to be parsed.

As a validation to the parsing result, we pick eleven documents as a representative dataset for which we monitor the parsing quality. This means that the parsing result of those documents are regularly evaluated whenever there are (significant) updates to the parsing program. At the end, we observe that the parsing result is of good quality for the documents. Some small details may have been overlooked, but there is no part of the documents which is unparsed. As a consideration, the parsing result of Lex2KG heavily depends on the consistency and overall quality of the scanned documents. In general, the parsing program can be augmented with manual effort by humans for quality checking over (a subset of) the result.

C. Schema

Lex2KG relies on the Lex2KG schema for its KG construction. The schema is modeled using RDF Schema (RDFS), facilitating the listing of the terms used in the legal KGs. In number, there are currently 15 classes (e.g., Legislation, Article), 28 properties (e.g., partOf, cites, legislationType), and 6 (constant) individuals (e.g., Law – as a value of legislationType). We also align the Lex2KG schema to Schema.org to support interoperability.

Fig. 2 shows a snippet of the Lex2KG schema in RDF Turtle.[6] The snippet first lists the prefixed namespaces of the schema. Then, it gives the metadata of the schema (i.e., the title). Next, the snippet defines the class and properties of Lex2KG. Note

[6]https://www.w3.org/TR/turtle/

that the snippet displays an English translation of the schema to accommodate the paper audience. The full schema is available at https://bit.ly/lex2kg-o.

D. Data

We have run Lex2KG over Indonesian laws from https://www.dpr.go.id/jdih/uu. The legal KG contains the data of 784 laws with a total size of 1,163,051 RDF triples. Table I shows the number of triples grouped by information types. The whole KG is available at https://bit.ly/data-lex2kg.

TABLE I
STATISTICS OF THE LEGAL KG

Information Type	#Triples
Metadata	6,869
Structure	714,888
Text	152,301
Citations in the same law	22,829
Citations between laws	1,401
Citations to non-existing resources	7,095
Amendments	1,252
Class assignments	256,416

III. USE CASE EVALUATION

In this section, we discuss the use case evaluation of the Lex2KG framework. The section starts with the story from industry, followed by the presentation of the three use cases. We refer the reader to Fig. 3 for our use case illustration.

A. Story from Industry

CATAPA,[7] an AI-based compensation and benefit platform, intends to assist HR divisions of companies to stay current with respect to legal aspects by easily discovering HR-related legal resources and the relationships between them. We realize that KG technologies could help CATAPA achieve its goals by delivering the following features: (i) legal querying using SPARQL; (ii) simple QA using a legal chatbot; and (iii) legal visualization.

B. Use Case 1: Legal Querying Using SPARQL

Legal querying can be done by loading the KG into a SPARQL server (e.g., Apache Jena Fuseki)[8] and then querying into it. SPARQL queries feature join, union, optional, and negation patterns. SPARQL also provides a full set of analytical query operations such as filters, aggregates, and property paths. As shown in Fig. 3, the SPARQL query requests all pairs of a law and its cited law. The evaluated query over the legal KG then returns the pair `law:2020/11` and `law:2003/13` as the citator law and cited law, respectively.

C. Use Case 2: Simple QA Chatbot

A question answering (QA) chatbot provides a conversational experience to users to retrieve the knowledge underlying the chatbot. QA over legal KGs can be done by converting the natural language (NL) question into a SPARQL query, evaluating the query, and then showing the query result as an answer. A simple approach could be by means of SPARQL templates with NL interfaces. Fig. 3 shows the translation of a question "Which laws talk about employment agreement?" into a SPARQL query. The chatbot then returns all laws containing the substring "employment agreement" in the text of laws' components (e.g., `law:2020/11`). More complex use cases can be supported by employing advanced NLP techniques, such as named-entity recognition, natural language generation, and word embeddings [9]. In this paper, the chatbot was implemented as a simple command-line inferface (CLI), which takes user input and prints the result of the QA process as explained above. The chatbot was built using Node.js[9] and the queries were constructed in the similar way as in Use Case 1.

D. Use Case 3: Legal Visualization

Visualization increases the visibility and explainability of legal data. As for legal KGs, we may visualize the nodes and relations among them, as well as their statistics. Fig. 3 depicts the number of components contained in laws as a bar chart using visualization libraries (e.g., VizKG from https://pypi.org/project/VizKG/ [10]).

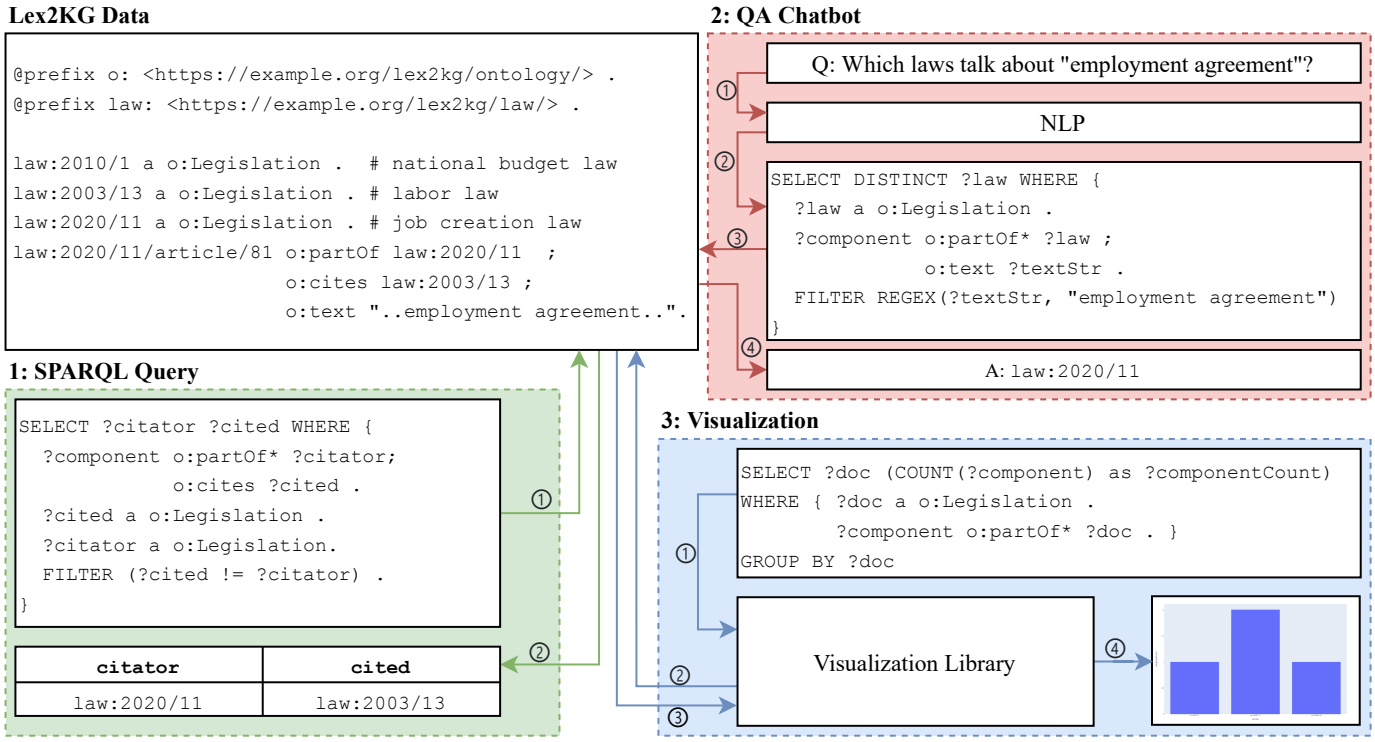
IV. CONCLUSIONS

In this paper, we present Lex2KG, a framework for automatic conversion of legal documents to knowledge graph (KG). The framework aims to deliver KG technologies to the legal domain. Our

[7]https://catapa.com/

[8]https://jena.apache.org/documentation/fuseki2/

[9]https://nodejs.org/



* For readability purposes, "\/" in URIs is written as simply "/". Also, data written in Indonesian is translated to English.

Fig. 3. Lex2KG Data and Use Cases

framework goes beyond the existing work, that is, ELI and the legal extension of Schema.org, in that our framework not only concerns the legal metadata, but also structure, textual content, and relationships among legal resources. We have applied our framework to automate the conversion of 784 Indonesian laws (or Undang-Undang, in Indonesian) into a legal KG with a size of 1.1 million triples. We provide use case evaluation of the legal KG generated for legal SPARQL querying, simple QA chatbot, and legal visualization. For future work, we aim to handle regulations other than laws, and also of other languages than Indonesian.

REFERENCES

[1] "The Avalon Project: The Code of Hammurabi," https://avalon.law.yale.edu/subject_menus/hammenu.asp, accessed: 2021-06-21.

[2] G. Riva, "The Potential and Limitations of Computational Law for Data Protection," *MIT Computational Law Report*, 2020.

[3] A. Hogan, E. Blomqvist, M. Cochez, C. d'Amato, G. de Melo, C. Gutiérrez, S. Kirrane, J. E. L. Gayo, R. Navigli, S. Neumaier, A. N. Ngomo, A. Polleres, S. M. Rashid, A. Rula, L. Schmelzeisen, J. F. Sequeda, S. Staab, and A. Zimmermann, "Knowledge Graphs," *ACM Comput. Surv.*, vol. 54, no. 4, pp. 71:1–71:37, 2021.

[4] N. F. Noy, Y. Gao, A. Jain, A. Narayanan, A. Patterson, and J. Taylor, "Industry-Scale Knowledge Graphs: Lessons and Challenges," *CACM*, vol. 62, no. 8, pp. 36–43, 2019.

[5] T. Francart, J. Dann, R. Pappalardo, C. Malagon, and M. Pellegrino, "The European Legislation Identifier," in *Knowledge of the Law in the Big Data Age*, ser. Frontiers in AI and Applications, vol. 317. IOS Press, 2018, pp. 137–148.

[6] E. Montiel-Ponsoda, V. Rodríguez-Doncel, and J. Gracia, "Building the Legal Knowledge Graph for Smart Compliance Services in Multilingual Europe," in *Workshop on Technologies for Regulatory Compliance*, 2017.

[7] J. M. Schneider, G. Rehm, E. Montiel-Ponsoda, V. Rodríguez-Doncel, A. Revenko, S. Karampatakis, M. Khvalchik, C. Sageder, J. Gracia, and F. Maganza, "Orchestrating NLP Services for the Legal Domain," in *LREC*, 2020.

[8] H. Zhong, C. Xiao, C. Tu, T. Zhang, Z. Liu, and M. Sun, "How Does NLP Benefit Legal System: A Summary of Legal Artificial Intelligence," in *ACL*, 2020.

[9] D. Jurafsky and J. H. Martin, *Speech and Language Processing: An Introduction to Natural Language Processing, Computational Linguistics, and Speech Recognition, 2nd Edition*, ser. Prentice Hall series in Artificial Intelligence. Prentice Hall, Pearson Education International, 2009.

[10] H. Raissya, F. Darari, and F. J. Ekaputra, "VizKG: A Framework for Visualizing SPARQL Query Results over Knowledge Graphs," in *VOILA*, 2021.