Emmanuel C. Ayeleso*
eayel037@uottawa.ca

Contract Elements Extraction, Give it to AI

ABSTRACT

Information is hidden in the volume of contract documents, thus it is useful to extract these information, called information extraction (IE). In the recent times, there have been expansion in the field of information extraction. Many methods have been defined by the researchers to automate extraction of information from web text and documents [5]. Similarly, application of IE techniques to extract contract elements have been given attention by some researchers. It is believed that the success of extracting contract elements automatically will save organization from the the recurrent cost incurred on contracting and its management. For instance, huge amount of money expended on legal services to draft, modify and manage contracts across wide facet of disciplines would become thing of the past. It will also give birth to the possibility to generate structured data from a contract document. We attempted to reproduce an existing study in this problem domain. We used keras and Tensorflow to implement the training of a deep leaning layer BILISTM-CRF on Google Collab. We compared the existing state-of-the-art results. Obfuscation of the available dataset restricted our accomplishment. However, quite interesting research problems have been exposed and marked for future efforts.

KEYWORDS

Information Extraction (IE), Contract, BILSTM (Bidirectional Long Short Term Memory), CRF (Conditional Random Field)

ACM Reference Format:
Emmanuel C. Ayeleso. 2020. Contract Elements Extraction, Give it to AI. In *Proceedings of ACM Conference (Conference'17).* ACM, New York, NY, USA, 6 pages. https://doi.org/10.1145/nnnnnnn.nnnnnnn

1 INTRODUCTION

Contract is a document that represents trust between stakeholders involved in any dealing together. Most times, this document called "contract" and used across all fields contains clauses that specify the terms and conditions of the agreement between parties involved. For instance; bilateral relation treaties between countries, labor union agreement between employees and employer, memorandum of the understanding between bodies such as; institutions, companies and organizations are some of the typical examples of a contract.

Poor contracting can cause an organization to lose substantial value on a deal. For instance, non-compliance to the dictate and terms of a contract. On the contrary, if an organization is able to put a measure that can automatically extract terms of a contract as a structured data successfully, such data can be used to her advantage. For instance, such data can be processed by algorithms to give timely advisory tips on risk analysis of contracts. Also, such data can be used for quick regeneration of new contract with less human inputs. Automatic extraction of contract elements will save huge amount of human efforts spent across sectors to drafts and manage contracts on a daily basis by the legal practitioners. Lawyers may have to shift their services to counselling as regards the clauses of a contracts rather than drafting and reviewing of it that takes huge amount of time. We have looked at existing efforts in this problem domain and tried to reproduce the state of the art.

2 PROBLEM SPACE

Contracts are often prepared in a semi-structured format, meaning that they lack uniformity. Typically, a sheer numbers of contracts may need processing before a decision can be taken by the concerned parties. There are organizations that need to process hundreds of contracts in their daily running. Making it difficult to manage, update and comply with the letters of these contracts in practice. Therefore, exposing many of them to litigation and infringement of agreement. A typical strategy in processing contracts is to engage the human workforce to extract the specific and crucial pieces of information and action plans from these contracts. Some elements of contract document that are of essential information and processing are contracting parties, amending terms, date of expiration, legislation implication and references (Chalkidis, 2017a and Chalkidis, 2017b). The manual approach of reading contracts and retrieving these pieces of information can be time-consuming, lack consistency and do not provide any contract database that can be used for data management purposes.

In the recent years, the use of machine learning approaches in the field of information extraction has witnessed huge efforts. Such that some of the activities that used to be explicitly carried out by human beings only can now be taken by the machine models. Extracting contract elements from typical contract share a problem space of information extraction from unstructured text. This nature of problem is well researched with extensive methodologies developed in the field of information extraction.

2.0.1 Information extraction (IE). This is a system that searches text for specific types of entities and relations (structured data). This system takes raw text as input and generates a list of (entity relation entity) tuples as output. Figure 1 presents a sample of IE architecture. For instance, it can take in a drafted contract and establish relationship such as (Org: 'Company A') 'agrees with ('ORG: "Company B"). The conventional IE approaches are rule learning, classification and sequential labeling based method [5]: Figure 1 presents a simple architecture of an information extraction system.

2.0.2 Rule-Based method. This approach defines a set of rule syntax and other grammatical properties of a text and use these rules

*Both authors contributed equally to this research.

Conference'17, July 2017, Washington, DC, USA

ACM ISBN 978-x-xxxx-xxxx-x/YY/MM…$15.00
https://doi.org/10.1145/nnnnnnn.nnnnnnn

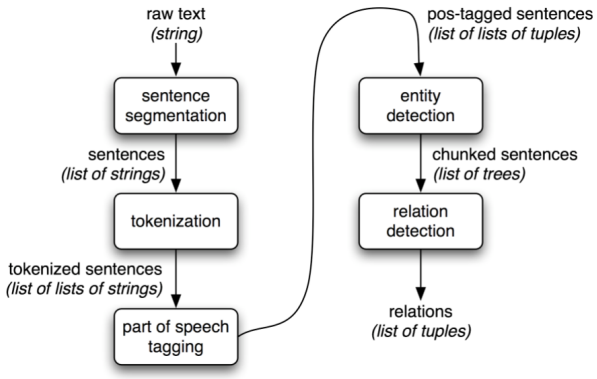

Figure 1: Simple Pipeline Architecture for an Information Extraction System. Sourced from [6]

to extract information. Purely rule-based methods often work well only within a very narrow specific domain. Especially, when processing semi-structured documents [5]. The drawback of this approach is that rules need to be updated regularly and be reflective of a document that this approach can be used on. Updating the rules can be laborious, time consuming and ineffective. Rule-based was used by [2] to extract contract element.

2.0.3 classification based method. This approach uses supervised machine learning to see IE as a classification problem in terms of statistical theory [5]. It has more generalization than rule-based method. And it can be quite difficult for a common user to understand. It was used by [2] also to extract contract elements.

2.0.4 sequential labeling based method. In sequential labelling, a document is seen as a sequence of tokens, and a sequence of labels are assigned to each token in order to indicate the property of that token[5]. The concept can make use of the dependencies between information to improve extraction performance. It is also based on statistical theory and thus had generalization capabilities. It can also be combined with rule based method to get hybridized result. An implementation of sequential labeling based method is typical of training a machine learning model. Such model is trained for the relations between the entities of a text. Often times, the success of this approach is subject to the availability of enough labeled data to train the model. Recently, RNNs based on Long Short-Term Memory (LSTM) cells or Gated Recurrent Unit (GRU) cells have been reported to give impressive state-of-the-art. The trending architecture is to build a cascade of models called layer as applied by [1] study that motivated this study.

3 PROBLEM FORMULATION

Let say we have a contract as input of sequence (sentences of words of in sentences)

$$x = (x_1, ..., x_m)$$

and an output states (the named entity tags- Contract Element Categorized)

$$s = (s_1, ..., s_m)$$

In conditional random fields (CRF) we modeled the conditional probability of the output state sequence given a input sequence by

$$p(s_1, ..., s_m \mid x_1, ..., x_m)$$

For doing this we define a feature map that maps an entire input sequence $\mathbf{x}$ paired with an entire state sequence $\mathbf{s}$ to some d-dimensional feature vector. Then we can model the probability as a log-linear model with the parameter vector $\omega\varphi(s_1, ..., s_m, x_1, ..., x_m) \in \mathbb{R}^d$

$$p(s \mid x; \omega) = \frac{exp(\omega).\varphi(x, s))}{\sum_s exp(\omega).\varphi(x, s)))}$$

4 CASE STUDY A

We studied existing efforts that motivated our study. The first study experimented about how to extract contract elements automatically. Firstly, they generation of a new benchmark contract dataset. Their dataset approximately contained 3,500 English contracts annotated with focus on 11 types of contract elements. These elements are Contract Title, Contracting Parties; Start, Effective, Termination Dates, Contract Periods, Values, Governing Law, Jurisdiction, Legislation Refs and Clause Headings. Each of the algorithm used was trained to detect contract element in the defined extraction zones independently.

4.0.1 Labeled Benchmark Dataset. 993 contracts (893 training, 100 test) annotated with gold (correct) clause headings. 2461 contracts (2,111 training, 350 test) with gold annotations for the other 10 types of contract elements. The gold contract element annotations of the labeled dataset were provided by 10 law students. Further details of [2] labeled benchmark dataset are:

```
-2000 Dimension pretrained word embeddings
-350 test contracts
-200 dimensional embeddings
-25 dimensional POS tags
```

- Token consisting of alphabetic upper-case characters, possibly including periods and hyphens (e.g., "AGREEMENT", "U.S", "CO-OPERATION")
- Token consisting of alphabetic lower-case characters, possibly including periods and hyphens (e.g., "registered", etc.; "third party")
- Token with at least two characters, consisting of an alphabetic upper case first character, followed by alphabetic lower-case characters, possibly including periods and hyphes (e.g., "Limited", "Inc.", "E-commerce")
- Token consisting of digits, possibly including periods and commas (e.g., "2009", "12,000", "1.1")
- Line break
- Any other token containing only non-alphanumeric characters (e.g., "@", "");
- Any other token (e.g., "3rd", "strangeTek", "EC")

4.0.2 Extraction zones. They classified a typical contract into zones that are commonly featured in a contract. For instance, a contract title would be found on the cover page. Figure 2 presents this classifications, examples of heading words and the zones part of a contract where the targeted elements are expected to be found.

Extraction Zones (at testing)	Example Clause Heading Words	Contract Elements Typically Included
Cover page and preamble	–	Contract Title, Contracting Parties, Start Date, Effective Date
Term clause	'Term', 'Period', 'Term of Agreement'	Termination Date, Contract Period
Termination clause	'Termination', 'Termination of Agreement'	Termination Date
Governing Law clause	'Governing Law', 'Applicable Law'	Governing Law, Jurisdiction
Jurisdiction clause	'Jurisdiction', 'Jury Trial', 'Venue'	Jurisdiction
Miscellaneous clause	'Miscellaneous', 'Entire Agreement'	Governing Law, Jurisdiction
Contract Value clause	'Lump Sum', 'Salary'	Contract Value
In the text after the recitals, zones starting up to 20 tokens before and ending up to 20 tokens after each line break, not crossing other line breaks		Clause Headings
In the entire contract, zones starting up to 20 tokens before and ending up to 20 tokens after each occurrence of words like 'Act', 'Treaty' etc.		Legislation References

Figure 2: Extraction zones where contract elements types are searched during testing. Sourced from [2]

4.0.3 *Models Used.*

- Manual-rule based
- Logistic regressor
- SVMs

4.0.4 *Algorithm.*

- Instead of working on the whole document, they extracted element zones
- Make use of Linear classifiers with hand-crafted features, word embeddings, and part-of-speech tag embeddings to classify tokens

```
TOKEN_2888[0] TOKEN_2889[0]
TOKEN_1490[TIT] TOKEN_6[TIT]
TOKEN_15[0] TOKEN_6[0] TOKEN_2384[0] TOKEN_263[0] TOKEN_28816[STD] TOKEN_28[STD]
TOKEN_25[STD] TOKEN_4376[STD] TOKEN_19[STD] TOKEN_1530[STD] TOKEN_31[0] TOKEN_78167[CNP]
TOKEN_5565[CNP] TOKEN_1539[CNP] TOKEN_19[0] TOKEN_66[0] TOKEN_12279[0] TOKEN_2207[0]
TOKEN_2167[0] TOKEN_22[0] TOKEN_2191[0] TOKEN_408[0] TOKEN_26[0] TOKEN_413[0]
TOKEN_1660[0] TOKEN_8133[0] TOKEN_194[0] TOKEN_57[0] TOKEN_10943[0] TOKEN_74489[0]
```

Figure 3: Sample file in the annotated corpus text file

5 CASE STUDY B

They adopted the Labeled Benchmark Dataset created by [2], being their previous study and explored deep learning models by stacking LSTM on top of the BILSTM, or adding a CRF layer on top of the BILSTM.

5.0.1 *Models Used.*

- BILSTM-LR Extractors
- BILSTM-LSTM-LR Extractors
- BILSTM-CRF Extractors

5.0.2 *Algorithm.*

- Instead of working on the whole document, they extract element zones
- BILSTM models (above variations), Glorot initialization, binary cross-entropy loss, Adam optimizer

6 CASE STUDY C

Patil (2018) attempted reproduction of [1] but reported difficulty to use the benchmarked dataset due to its obfuscation (encoding). He [4] struggled with the provided embeddings, part-of-speech tag and word length for each token which was provided by the benchmarked dataset.

6.0.1 *Dataset.*

-2000 Dimension pretrained word embeddings
-350 test contracts
-200 dimensional embeddings
-25 dimensional POS tags

6.0.2 *Models.*

- Embeddings+BILSTM+Dense+CRF

7 PROPOSED METHODOLOGY

Extraction of contract element is a sequence tagging problem. According to Wikipedia, a sequence tagging sequence labeling is a type of pattern recognition task that involves the algorithmic assignment of a categorical label to each member of a sequence of observed values. In the context of contract document, tagging of part of speech (POS), chunking and named entity recognition (NER) as a classic NLP task [3]. We proposed application of convolutional network based model bidirectional long short term memory (BILSTM) with conditional random field (CRF) layer to sequence tagging in this work [3].

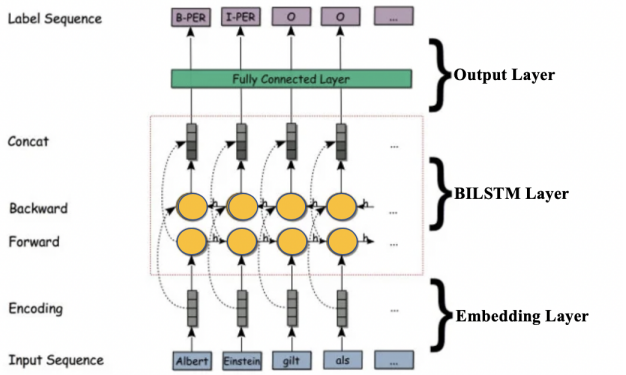

Figure 4: Proposed Methodology

7.1 Bidirectional LSTM Networks

LSTM is the same as recurrent neural networks (RNNs) except that the hidden layer updates are replaced by purpose-built memory cells. This makes them to be better at finding and exploiting long range dependencies in the data [3].BILSTM is a combination of two LSTMs in the opposite direction in order to have forward and backward propagation. This is typical of how human process text in the real life. Meaning is attached to a sentence from what have been read and the following words. With BILSTM we could have access to both past and future input features for a given time in the trained dataset.

7.2 CRF Networks

Conditional Random Field (CRF) focus on the sentence level features such as named entity recognition and part of speech tagging and used discriminative models best suited to prediction tasks where

Table 1: Comparison of our model's results with the case studies

Element Type	BILSTM-CRF[1]			BILSTM-CRF[4]			BILSTM-CRF (our model)		
	P	R	F1	P	R	F1	P	R	F1
Title	0.96	0.95	0.95	0.87	0.93	0.90			
Parties	0.98	0.92	0.95	0.84	0.82	0.83	0.000325	0.000194	0.00024
Start	0.92	0.98	0.95	0.74	0.70	0.72	4.298	0.139	8.594
Effective	0.95	0.89	0.92	0.76	0.07	0.12	8.292	0.063	1.658
Termination	0.65	0.93	0.77	0.65	0.14	0.23	3.828	0.066	7.652
Period	0.55	0.85	0.65	0.93	0.28	0.43	6.340	0.056	0.00012
Value	0.72	0.60	0.66	0.69	0.07	0.12	4.860	0.0321	9.707
Gov. Law	0.99	0.97	0.98	0.82	0.94	0.88	0.0001	0.0230	0.0002
Jurisdiction	0.90	0.88	0.88	0.82	0.64	0.72	4.15	0.066	8.304
Legisl. Refs.	0.82	0.94	0.87	0.88	0.79	0.83	0.0002	0.112	0.0003
Heading	0.99	0.97	0.98						

Algorithm 1 Bidirectional LSTM CRF model training procedure

```
for each epoch do
    for each batch do
        1) bidirectional LSTM-CRF model forward pass:
            forward pass for forward state LSTM
            forward pass for backward state LSTM
        2) CRF layer forward and backward pass
        3) bidirectional LSTM-CRF model backward pass:
            backward pass for forward state LSTM
            backward pass for backward state LSTM
        4) update parameters
    end for
end for
```

Figure 5: BILST CRF model training algorithm. Adopted from [3]

contextual information or state of the neighbors affect the current prediction [3].

7.3 BI-LSTM-CRF Networks

This is simply combination of BILSTM and CRF as showcased in figure 4. The importance of this is to boost accuracy. We adopted the Labeled Benchmark Dataset of [2] and faced the same challenge of encoded dataset that restarained [4]. According to the authors [2], the encoding was done in order to protect the privacy of the companies or organizations whose contracts are in the dataset. Such encoding made working with the dataset restrictive and difficult to work with. For instance, no human readable meaning could be traced to both labeled and unlabelled dataset as shown in Figure 3 . In the interest of time and resources that cannot permit us to create our own dataset, we only achieved reproduction of [4] efforts. Figure 4 presents our methodology and and figure 5 showcased our adopted algorithm. The embedding layer in Figure4 encodes the input sequence into a sequence of dense vectors of defined dimension. This has been shown to be vital to improving sequence performance tagging by [5]. We labeled and encoded contract element as the targeted categorical labels for our trained model to detect as shown in figure 6

```
|CONTRACT ELEMENT|   |CATEGORY ENCODING| |CLASS|

------------------------ + ------------------
None                    β> 0                 0
Contract Title          β> TIT               1
Contract Party          β> CNP               2
Start Date              β> STD               3
Effective Date          β> EFD               4
Termination Date        β> TED               5
Contract Period         β> PER               6
Contract Value          β> VAL               7
Governing Law           β> GOV               8
Jurisdiction            β> JUR               9
Legislation Refs.       β> LEG               10
```

Figure 6: Categorical encoding of the contract element. CLASS is the value predicted in the model

```
Model: "model_9"
_________________________________________________________________
Layer (type)                 Output Shape              Param #
=================================================================
input_19 (InputLayer)        (None, 69593)             0
_________________________________________________________________
embedding_10 (Embedding)     (None, 69593, 225)        38467350
_________________________________________________________________
bidirectional_10 (Bidirectio (None, 69593, 100)        110400
_________________________________________________________________
time_distributed_10 (TimeDis (None, 69593, 50)         5050
_________________________________________________________________
crf_10 (CRF)                 (None, 69593, 12)         780
=================================================================
Total params: 38,583,580
Trainable params: 38,583,580
Non-trainable params: 0
_________________________________________________________________
```

Figure 7: Summary of our trained

8 RESULTS

We limitedly trained a BI-LSTM-CRF model for the bench marked contract dataset published by [2]. Figure 7 present the summary of our trained model. Souce code written for the model training

can be found on GitHub with the url https://github.com/Emmcho/ContractElementExtraction. Our model's result as reported in table 1 is erroneous and inconsistent. This is as a result of bugs in our source code that keep aborting the training process of our algorithm. As a result, the generated model could not be fitted against the dataset. This researcher explored the use of keras function predict-generator() with custom data generator functions which was used to load our dataset to memory during training or predicting just to see what the results will look like. This researcher intends to go back into the code and fix the bug. Similarly, results reported by [4] in terms of precision, recall and F1 were quite lower than that of [1] because of the obfuscation of the dataset available. But, the results of [1] indicated a very impressive state-of-the-art performance of the trained model as shown in table 1.

9 CONCLUSION

In this project, we have compared the performance of BILSTM networks based models for sequence tagging. We attempted to reproduce the work of [1] and compare their results with that of [4] and our model. This project could not fully achieve the set objectives because of the following factors: dataset obfuscation constrains, limited researcher's technical know how as a result of working in this domain for the first time, time and resource limitations. However, this researcher has been exposed to sequence tagging field of research with interesting experience and line of action to explored further as regards this work.

10 FUTURE WORK

We intend to take this work a step further by implementing the model trained in this research. Such that, the working of the model will be usable to a lay man. That is, a transition from a model trained to application of the model to extract these contracts element in a real life scenario. Similarly, the approach of dividing the contract into zones for effectiveness and less computational cost by [1, 2] is impressive. However, practical implementation of this procedure may be costly and not operational. We intend to experiment with a contract-document based. That is, without zoning contract elements to any zone and compare our results.

REFERENCES

[1] Ilias Chalkidis and Ion Androutsopoulos. 2017. A Deep Learning Approach to Contract Element Extraction. In *JURIX*.

[2] Ilias Chalkidis, Ion Androutsopoulos, and Achilleas Michos. 2017. Extracting Contract Elements. In *Proceedings of the 16th Edition of the International Conference on Articial Intelligence and Law* (London, United Kingdom) *(ICAIL '17)*. Association for Computing Machinery, New York, NY, USA, 19–28. https://doi.org/10.1145/3086512.3086515

[3] Zhiheng Huang, Wei Xu, and Kai Yu. 2015. Bidirectional LSTM-CRF Models for Sequence Tagging. *ArXiv* abs/1508.01991 (2015).

[4] Patil Sharat. 2018. Contract Element Extraction Using Deep Learning. In *JURIX*.

[5] Jie Tang, Mingcai Hong, Duo Zhang, Bangyong Liang, and Juanzi Li. 2007. Information extraction: Methodologies and applications. *Emerging Technologies of Text Mining: Techniques and Applications* (2007). https://doi.org/10.4018/978-1-59904-373-9.ch001

[6] Wiebke Wagner. 2010. Steven Bird, Ewan Klein and Edward Loper: Natural Language Processing with Python, Analyzing Text with the Natural Language Toolkit. *Language Resources and Evaluation* 44, 4 (2010), 421–424. https://doi.org/10.1007/s10579-010-9124-x