Docket Number

APPLICATION FOR UNITED STATES LETTERS PATENT

Title: REPRESENTING CONTRACTS BY CLAUSE GENOME FOR MACHINE LEARNING APPLICATIONS

Inventors: Yogesh Kulkarni

Patent Laywer Name
Patent Lawer Reg. Number

Assignee Name
Assignee Address
Assignee City, State, Zip
Assignee Phone

REPRESENTING CONTRACTS BY CLAUSE GENOME FOR MACHINE LEARNING APPLICATIONS

FIELD OF THE INVENTION

[**0001**] The present invention relates generally to document management, and more particularly, but not exclusively, to analyzing document content.

[**0002**] The present invention proposes intermediate level abstraction for representing contract documents, based on clause categories. Such representation helps in various machine learning algorothms such as assessing contracts similarlity, grouping contracts based on contents, predicting type of a contract, etc.

BACKGROUND OF THE INVENTION

[**0003**] Contracts are legal documents describing agreements between two or more parties. They have been in practice for ages, evolving in aspects of medium, language and enforcement approaches. Analysis of contracts has become critical for Law firms, companies, governments to assess contents for finding similar documents, extracting important information, etc. Although rule-based approaches have been predominant in this aspect, use of Artificial Intelligence (AI) in form of Machine Learning and Natural Language processing are getting their way into mainstream legal informatics.

[**0004**] Machine Learning approaches need inputs to be in numeric form. For Natural Language Processing (NLP), it becomes imperative to convert input (and output, in some cases) to numeric form, such as vectors. Multiple approaches to map words or sentences or document to vectors are available. Some popular ones such as ‘Bag of Words (BoW)’, ‘Term-Frequency Inverse Document Frequency (Tf-Idf)’ are typically word frequency based, whereas Word Embeddings such as ‘Document to Vectors (Doc2Vec)’, Glove, ‘Bidirectional Encoder Representations from Transformers (BERT)’ are co-occurrence or contextual in nature. Domain specific embeddings tend to capture some form of semantics due to co-occurrences, i.e. vectors of related words are close in the embeddings space. But such word level vectors are too granular to capture higher level domain constructs or abstractions. For example, in case of legal contracts, Doc2Vec would be some sort of composition (say, averaging) of Word2Vec of individual words. This approach loses middle level abstractions such as clauses, i.e. Contract is composed of clauses; clauses are composed of words. The middle level abstract captures legal-domain semantics.

[**0005**] Various schemes are available to convert text to vectors for Machine Learning related applications in the domain of Natural Language Processing. Embeddings come in different forms such as Word (Word2Vec [2]), Sentence (Sent2Vec), Paragraph and Documents (doc2vec). In legal domain, [1] obtained (pre-trained) word embedding by applying word2vec to an additional unlabeled dataset of about 750,000 contracts.

OBJECTS OF THE INVENTION

[**0006**] The invention proposes a novel approach of represent contracts by vector of clauses. Once documents are converted to list of clause category labels, it’s a string of labels, like biological genome made up of letters such as A, T, C, G. Thus, the clause category-based representation of contract document is referred to as ‘Clause

Genome'. They act as signature or short from string of the original document, which can further be vectorized using traditional approaches mentioned above, and then used to Machine Learning algorithms such as Classification, Clustering, Similarity, etc.

SUMMARY OF THE INVENTION

[**0007**] The invention has following advantages and disadvantages.

- Advantages:
  - Leverages Contract domain specific higher-level abstractions, making it more domain-semantic.
  - Clause genome has lower space requirements and computationally efficient for further downstream applications.
  - Noise based on individual word frequency is smoothened
  - Order of the clauses is maintained i.e. document structure is retained
  - Explain-ability is better for debugging as well as customer trust
- Disadvantages:
  - Heavily depends on Clause Classifier efficacy
  - Disregards variations within a clause category
  - Loss of word-level information including linguistic structures.

BRIEF DESCRIPTION OF THE DRAWINGS

[**0008**] For a more complete understanding of the invention, reference is made to the following description and accompanying drawings, in which:

[**0009**] FIG. 1 is an example drawing created in Visio; and

[**0010**] FIG. 2 is an example drawing created in TpX.

INVENTION APPROACH

[**0011**] The approach takes document corpus as input, converts each document into "Clause Genome" based intermediate representation, which then taken further into standard Text Classification pipelines such as Vectorization, Classification model training and results generation. Major steps are enumerated below:

- Data Preparation: Documents, if non-textual format, are converted to text. Text is segmented into paragraphs.
- Clause Classification: Each paragraph gets classified into a clause category; thus a document becomes list of clause categories.
- Symbolization: For each document, using list of clause categories, a "Clause Genome" sequence or string is generated. This shortened form now represents the original document in a shortened form.
- Vectorization: Standard vectorizers like BoW or Tf-Idf are used on the shortened "Clause Genome" string.

- Application: Genome based vectors can be used in Machine Learning tasks such as Classification and Clustering.

Core steps have been explained in detail, in the following sub sections.

[**0012**] Symbolization: Typical Master Services agreement will have about 70-100 categories of clauses. Machine Learning based approaches such as Support Vector Machines, Naive Bayes, Neural Networks, can be used to build a classifier.

After ‘Data Preparation” and ‘Clause Classification’ phases, each document is in the form of list of clause categories in it, while maintaining the order of the clause sequence. Each category is mapped to a symbol or label. It can be letters like ‘a’, ’b’, ’c’, etc. But to accommodate large number of categorizes, labeling scheme as ‘c1’,’c2’, . . . is used. Sample symbolization map is shown below:

```
clause_cateory_to_symbol_map ={
"acceptance":"c1",
"publicity":"c2",
"amendment":"c3",
:
"intellectual␣property␣rights":"c9",
"penalties/remedies":"c10",
"business␣continuity":"c11",
:
"payment␣terms":"c24",
"force␣majeure":"c25",
"governing␣law":"c26",
:
"quality␣control":"c60",
"preamble":"c61",
:
```

}

For Sample documents like MSA_1 and MSA_2, the clause genomes look like:

MSA_1 = ‘‘c61 c53 c3 c40 c23 c7 c24 c24 c50 c7 c7 c7’’
MSA_2 = ‘‘c61 c61 c48 c48 c8 c28 c30 c24 c30 c9 c53’’

Note that both seem to have started with similar clause category, “c61”, and that’s obviously “preamble”. Pictorially they would look as in Fig. 0.1:

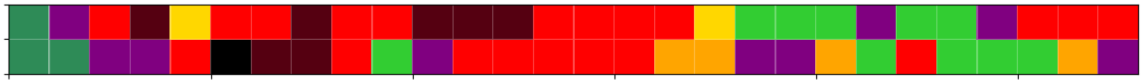
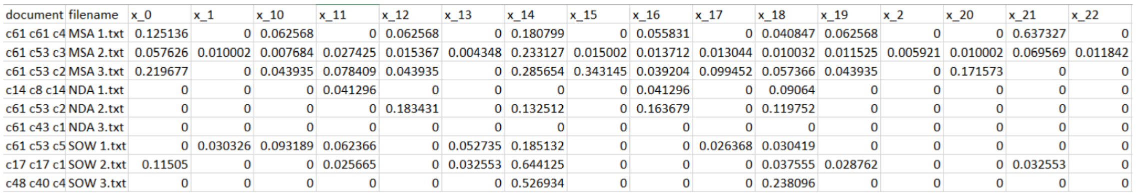


Figure 0.1Graphical representation of document clause genomes

[**0013**] Vectorization: Instead of the original text, just the genome strings now represent the documents. This new content then used to vectorize by Tf-Idf as shown in Fig. **??**.

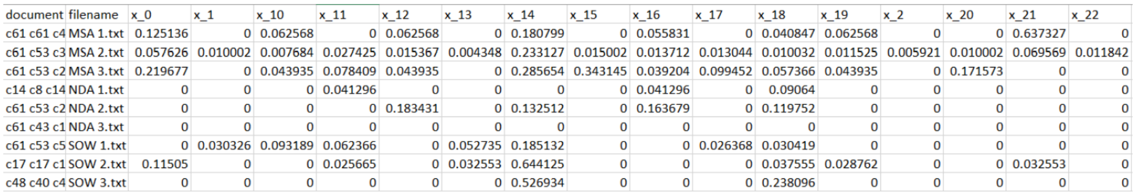


document	filename	x_0	x_1	x_10	x_11	x_12	x_13	x_14	x_15	x_16	x_17	x_18	x_19	x_2	x_20	x_21	x_22
c61 c61 c4	MSA 1.txt	0.125136	0	0.062568	0	0.062568	0	0.180799	0	0.055831	0	0.040847	0.062568	0	0	0.637327	0
c61 c53 c3	MSA 2.txt	0.057626	0.010002	0.007684	0.027425	0.015367	0.004348	0.233127	0.015002	0.013712	0.013044	0.010032	0.011525	0.005921	0.010002	0.069569	0.011842
c61 c53 c2	MSA 3.txt	0.219677	0	0.043935	0.078409	0.043935	0	0.285654	0.343145	0.039204	0.099452	0.057366	0.043935	0	0.171573	0	0
c14 c8 c14	NDA 1.txt	0	0	0	0.041296	0	0	0	0	0.041296	0	0.09064	0	0	0	0	0
c61 c53 c2	NDA 2.txt	0	0	0	0	0.183431	0	0.132512	0	0.163679	0	0.119752	0	0	0	0	0
c61 c43 c1	NDA 3.txt	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
c61 c53 c5	SOW 1.txt	0	0.030326	0.093189	0.062366	0	0.052735	0.185132	0	0	0.026368	0.030419	0	0	0	0	0
c17 c17 c1	SOW 2.txt	0.11505	0	0	0.025665	0	0.032553	0.644125	0	0	0	0.037555	0.028762	0	0	0.032553	0
c48 c40 c4	SOW 3.txt	0	0	0	0	0	0	0.526934	0	0	0	0.238096	0	0	0	0	0

Figure 0.2Vector representation of document clause genomes

[**0014**] Vectorization: Instead of the original text, just the genome strings now represent the documents. This new content then used to vectorize by Tf-Idf as shown in Fig. **??**.

Efficacy of the ‘Clause Genome’ based approach for Similarity calculations can be demonstrated by checking two Master Service Agreements (MSA) and then checking



document	filename	x_0	x_1	x_10	x_11	x_12	x_13	x_14	x_15	x_16	x_17	x_18	x_19	x_2	x_20	x_21	x_22
c61 c61 c4	MSA 1.txt	0.125136	0	0.062568	0	0.062568	0	0.180799	0	0.055831	0	0.040847	0.062568	0	0	0.637327	0
c61 c53 c3	MSA 2.txt	0.057626	0.010002	0.007684	0.027425	0.015367	0.004348	0.233127	0.015002	0.013712	0.013044	0.010032	0.011525	0.005921	0.010002	0.069569	0.011842
c61 c53 c2	MSA 3.txt	0.219677	0	0.043935	0.078409	0.043935	0	0.285654	0.343145	0.039204	0.099452	0.057366	0.043935	0	0.171573	0	0
c14 c8 c14	NDA 1.txt	0	0	0	0.041296	0	0	0	0	0.041296	0	0.09064	0	0	0	0	0
c61 c53 c2	NDA 2.txt	0	0	0	0	0.183431	0	0.132512	0	0.163679	0	0.119752	0	0	0	0	0
c61 c43 c1	NDA 3.txt	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
c61 c53 c5	SOW 1.txt	0	0.030326	0.093189	0.062366	0	0.052735	0.185132	0	0	0.026368	0.030419	0	0	0	0	0
c17 c17 c1	SOW 2.txt	0.11505	0	0	0.025665	0	0.032553	0.644125	0	0	0	0.037555	0.028762	0	0	0.032553	0
c48 c40 c4	SOW 3.txt	0	0	0	0	0	0	0.526934	0	0	0	0.238096	0	0	0	0	0

Figure 0.3Vector representation of document clause genomes

between one MSA and one NDA (Non-Disclosure Agreement). Results are:

- Cosine Similarity: MSA 1 and MSA 2: 0.734940271290997
- Cosine Similarity: MSA 1 and NDA 1: 0.31642358837691426

It can be clearly seen that the similar type of documents has high and dissimilar ones have low similarity score. Meaning, clause genome representation is quantitatively in agreement with the document language

What is claimed:

1. Contract documents can be effectlvely representated by their clause category labels, known as "clause genome".

2. "Clause Genome" effectiveness is proven in case of contracts similarity assessment.

ABSTRACT

Contracts drive large scale commercial transactions. Being a legal document, it is binding to all the parties, stakeholders involved. Composition contracts using clauses in an unambiguous manner is key to a good contract design. Contract analysis becomes critical in case of already drafted contract for detecting important clauses, obligations, key attributes, etc. Artificial Intelligence, in form of Machine Learning is being used effectively in some of these critical requirements. Finding similar contracts, grouping them is one such specific example.

Text needs to be converted to numbers to be used for Machine Learning applications. Various such vectorization techniques are available, which are primarily statistical in nature and deal with words in the documents. Word vectors are then composed into respective document vectors. This paper proposes vectorization of a higher-level abstraction, called clauses, there by bringing domain specific semantic composition into vectorization. Efficacy of the proposed approach has been demonstrated using Similarity and Clustering algorithms.

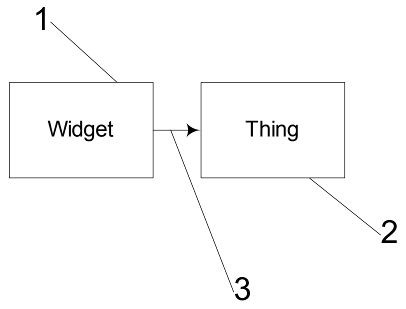


FIG. 1

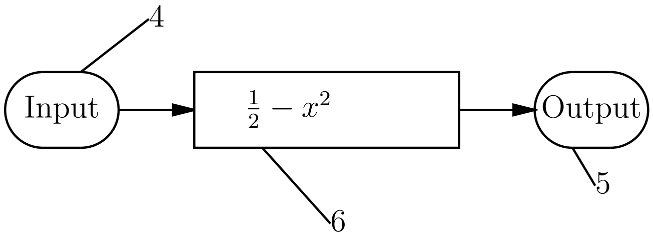


FIG. 2 PRIOR ART

Bibliography

[1] Ilias Chalkidis, Ion Androutsopoulos, and Achilleas Michos. Extracting contract elements. In *Proceedings of the 16th Edition of the International Conference on Articial Intelligence and Law*, ICAIL-17, pages 19–28, New York, NY, USA, 2017. Association for Computing Machinery. 2

[2] Tomas Mikolov, Ilya Sutskever, Kai Chen, Greg Corrado, and Jeffrey Dean. Distributed representations of words and phrases and their compositionality, 2013. 2