Information Extraction from Legal Documents using spaCy

Anustup Das, [1] Yamuna Nagasandra Rajaiah, [2] Sowmya Prakash [3] and Ahamad Shaik [4]
Otto-von-Guericke University Magdeburg, Germany
Data and Knowledge Engineering
Email: anustup.das@st.ovgu.de, yamuna.nagasandra@st.ovgu.de, sowmya.prakash@st.ovgu.de, peer.shaik@st.ovgu.de

***Abstract*— The development of knowledge extraction in the field of computer science has made advancements in mining useful information from heterogeneous sources. Latest evolution in legal science has shown that legal data analysis is a promising field of interest to data scientists. In this perspective it is important to extract domain specific information from law-based corpus. Due to high chances of new laws being formed we also need to build a solution that is capable of integrating expert knowledge into a legal decision support system. In order to achieve efficient information extraction and integration into exiting law-based corpus, there is a need to propose a solution for extracting context-based information from legal text books. In our work, we focus on knowledge extraction from a German legal corpus. Nowadays, German law system has made its corpus available in both PDF as well as XML documents. Our work is limited to extract content based files from German legal text books available in plain pdf data formats. In order to extract required contents, there are plethora of tools and technologies available in today's market. One of the existing tools from which the solution can be achieved is GATE. There are several drawbacks of Gate discussed in this paper, which demands for an improvement of existing NLP pipeline built on GATE. The solution recommended in this work not only improves the execution pipeline but also is expected to overcome previously seen user-interface issue of GATE. After performing enormous amount of research entangled in the domain of NLP, we were able to come up with a solution of building nlp pre-processing pipeline using spaCy. The objective of proposing spaCy as a solution is to perform efficient rule-based content annotation and extraction using Dbpedia Tagger. The evaluation of the final result of spaCy in comparison to Gate provides run-time values at each step of execution. This work also triggers an idea about applying neural network models to achieve better accuracy in terms of rule-based content extraction.**

I. INTRODUCTION

The field of information extraction has become a major part of information processing in the digital age. This subject is evolving rapidly in the domain of Artificial Intelligence (AI) and Natural Language Processing (NLP) to deliver new methods and technologies for the benefit of human kind [14]. The application of NLP and AI on legal science will help the legal experts to draw conclusions quickly, since the legal knowledge engineers and editorial staff in organizations can find it difficult to monitor the increasing number of law terms available from different legal domains like criminal or civil [11]. In order, to make sure no details are missed out, manual work is preferred. But, the law firms always lookout for automated tools (Eg. Westlaw Search Engine) to handle the existing information as well as maintain platform which will add new updations in legal texts, e.g., if a new law is added to the legal corpus then related laws have to be updated accordingly [13]. Thus, context sensitive search would be beneficial for such situations where appropriate information needs to be extracted and updated often. The application of natural language processing on German legal corpus using Natural Language Processing (NLP) modules like tokenization, parts-of-speech tagging, named-entity recognition and Named-entity resolution would enables rule-based annotation extraction of terms from document to determine linguistic and structural dependencies.There are various tools like GATE (with jape grammar) already been proposed to produce efficient results from legal texts. JAPE rules are simple yet effective rule matching approach [16] but gate has certain inefficiencies related to GUI such as output file obtained post annotation will be lost on the closure of GATE GUI, lack of support to neural network model [14].

So, consistent amount of work is performed to overcome the issues of gate tool by the extraction of semantic as well as syntactic relevant specific terms accurately from the legal corpus to check the feasibility of existing gate pipeline with that of spaCy framework and also, a methodology implemented to build NLP pipeline for legal text processing in spaCy efficiently to evaluate the text execution time taken for content extraction.While, the next advancement would be to focus on achieving time efficient results by utilizing the NLP techniques of spaCy in text processing pipeline in information extraction phase. The process of legal information extraction gets underway by including processing, generation and persistence of structured as well as unstructured information and is implemented by spaCy Architecture with open NLP library, which allows Natural Language processing to analyze, summarize, explore and visualize law corpus which includes new representation of text based annotations which are applied on legal documents [8]. Lawsuits in German courts are available in the data formats such as pdf or xml files. Usually, these data files consists of information objects. For instance, structure of law sections are of the form "(dt.Orientierungssatz/Leitsatz)", offence elements "(dt.Tatbestand)", and elements for reasoning "(dt.Grnd)". These structures are cumbersome in order to extract knowledge. The implementation work focused on the specific context-based information extraction using rule-based pattern matching. The steps we follow in terms of knowledge extraction is as follows. Firstly, we will gather

date and time-stamp information within the text. The data format usually differs based on geographical areas and also, the format depends on the use-case. If we consider date formats , they can be of the type: ”DD.MM.YYYY”, ”MM.DD.YYYY”, ”03. January 1993” or it can be of certain time stamp information such as ”2003 bis 2019” [5]. These refinements of data formats have been considered in our work to produce a result set which is capable of identifying different date formats. Next step, is to determine the text and sentences which includes terms like ”Inhaltsverzeichnis”, ”Inhalts” and other sentences which indicates the Chapter and Subchapter information like: ”Kap” or ”Kap I” or ”Kapitel”. These are achieved leveraging the tokenizer. This process is followed by applying rule-based annotation and custom rules for text annotation purpose [15]. We begin our work by implementation of pipeline for the Information pre-processing task which includes Tokenization, Sentence Splitting, Parts-of-Speech Tagging, Named Entity recognition and rule based annotations for the given corpus. On the second phase of implementation, we work on the tools and specifications which supports specified legal corpus. In the final stage, results obtained will be evaluated with the existing tools like GATE.

In evaluation section a brief comparison is discussed about the GATE and spaCy. This obtained result has produced better time efficient results in perspective of faster NLP pipeline execution. The solution provided has partially resolved issue of GATE with respect to visual interface. Adding on, the annotated files are not lost as in case of spaCy which is great benefit to the user. This work has opened new doors for information extraction process using neural network models in spaCy that has been further discussed in this paper.

II. MOTIVATION

The attention in the advancement of domain specific information extraction of legal text corpus with semantic and syntactic relatedness is necessary to maintain compliance as and when new law has been updated on regular basis. Domain specific collection of information gives the emphasis on specific content and understanding the linguistic structure of corpus.This is to say, that the present work is influenced by the basic ability to extract syntactically and semantically annotated corpus which is specific to legal texts [8]. There are several dedicated teams of experts constantly working in research areas of Natural Language Processing, Text Mining, Neural Networks and many more to name. To leverage the existing models or need for model to serve the needs of specific context based text retrieval for law-based corpus is main motivation behind this work.

After analyzing current approaches leveraged for annotation in linguistic stream, our idea was to propose a model with ability to write own rules. This happens to be a tedious job if one considers current tools like GATE which focuses on giving user interface to build the pipeline. GATE GUI platform has its own consequences in terms of processing time and complexities that involves extra effort by the user to understand the visual interface which is non-standard in terms of tool implementation. Also, the factor that GATE having complexity when annotated output files is lost once GATE GUI is terminated gives rise to questions to come up with an alternatives that is able to overcome these tool glitches. In order to address this our work is intended to build processing pipeline on spaCy’s platform [16]. There are plethora of tools available on the internet which are capable of extracting annotated csv files from the text books for example Apache UIMA, Brat Rapid, MAE, Rapid Miner and many more. There are very few of these tools which are capable of providing a platform which focuses on building custom rule based annotations which can leverage neural network models. Also, there is a need for a tool which not only able to successfully obtain the annotation but also makes the pre-processing pipeline easier and error free to some extent by requiring less effort by the user. spaCy is one such platform that is able to provide the user to write custom rules and which can also work on neural network models. Neural network models gains its importance when model is trained on specific corpus and validated on the other set of linguistic resource. This model is well explained in upcoming sections [12]. The extracted csv files provide information based on the required context. In this work, our focus is to mainly extract two of the annotated files for TOC (Table of Content) and RFC (Reason for citing). In the present GATE tool implementation a pre-processing pipeline is built and rule-based annotation is applied (Chapter, Part, Subchapter,Subsubchapter) in order to obtain the csv file [16]. The future aim of this work is to benefit legal knowledge engineers, editorial staff within enterprises who require drawing out legal interpretation and term usage details from law based corpus.

III. RELATED WORK

Processing the corpus based on the domain relevance has achieved major interest among the researchers. Large number of scientific works, publications and conferences have been proposed to provide an overview of the current trends and technology related to Information extraction of structured data which will be covered under this topic. We first start with a brief discussion on NLP techniques used to extract content based information from PDF documents. We also discuss on the approach used in current work and the demands of building pipeline for information extraction. We then narrow down our discussion on the survey’s conducted by many papers with respect to tools and technologies. Later which we propose the need for considering model building strategies and their importance in achieving better results based on the accuracy and precision as the measures in information extraction pipeline. We then conclude by specifying the steps involved in Machine Learning approach followed in the current systems [14].

The advancement in the area of information analytic systems and NLP approaches provide semantic text analysis approaches to extract domain specific information. Also, there are systems used in the field of textual analysis which focuses majorly on the Ontology-based approaches to extract

relevant information from PDF documents. Also, there is another area of Natural Language processing striving to build models based on the training samples and incorporate suitable algorithm to generate a model that best fits the semantic and structural relatedness for domain specific corpus [13].
In the current approach we tried to narrow down the research towards extracting information in document using entity recognition and resolution, which is more to do with Named entities like person, place organization to identify the segments of tokens. Entity recognition is applied in the processing pipeline in the Information Extraction methodology to find useful entities. These entities are useful pieces of tokens which form the basic units to recognize and can be classified based on the semantic type. These entities are usually referred to as Named entities which can be later leveraged by providing hypertext links for resolution purpose in recent systems. In case of law-based entity recognition we consider by providing certain types of entities such as jurisdictions, courts, law-based sections. The generic goal of Named entity resolution is to map names to the class labels of specific records considering class as a file. In case of law-based corpus class file contains the domain specific files for witnesses, sections, law-firms, enterprise or company and courts [14]. Named entity recognition provides citation and Table of Contents as the resolution framework. Table of content is used as a look-up table. It is to then verify these entities in legal text to evaluate accuracy. In order to build a data model to best fit the cases for Entity recognition and resolution,plethora of tools and techniques are available [5]. The source for information retrieval is taken from various digital formats (e.g., Microsoft Word, PDF, XML). Adding on, in several scientific proposals there are toolbox applications for e.g. GATE and Apache UIMA been used for task of extracting domain relevant information from legal text corpora. These tools are leveraged to perform initial data pre-processing tasks to make sense out of the text that is extracted. The relevancy is subjected to the semantic and structural closeness of the domain relevant corpus [13].
In the field of Information extraction, one can observe that the extraction process is not limited to semantic and structural closeness. We can potentially extend the topic towards method and techniques used from rule-based and statistical text retrieval systems.
In this work the focus is to extract the information from formatted data source such as pdf document. Working on such document would need a thorough understanding on the structure and the layout of source before extracting the entities tied to such data format. In order to work on such structure there are mainly two major pre-processing steps involved. One method is to extract the information in the form of List or to draw hierarchies within in the text structure. Several researches been done in this area but we focus on entity-based extraction the same will be covered in this paper. The methods of rule-based and statistical techniques focus on model driven algorithms. Since the work focus on the rule-based methods the part for statistical techniques are not discussed. To briefly give an idea about rule-based extraction strategy is to apply rules on the tokens of pre-processed data. these rule-based approaches are done on the tokens and the context of the token. To be more specific, rules are either hand-coded by domain experts or automatically learned by the systems.In case of rule-based methodology it is rule-learning algorithms used to define simple small set of rules for the training cases with attempt to maintain high precision. However, several use-cases provide us a naive idea about the rule-learning algorithm implemented in such a scenario would be cumbersome. This is due to the fact that rule-learning algorithm would follow a greedy strategy while applying these rules in the information extraction pipeline [12].
Knowledge extraction using Machine Learning models is another major field of research. Machine Learning models are built to learn the data over time so that the learning behaviour of the model could be improved. The models mainly extract temporal patterns in the data, which in some cases be hidden in high dimensional spaces [11]. Machine Learning models are developed to be able to learn from the training samples and designed to be able to predict the outcome from data about cases, judges and trends, these data are usually available from Court Database. Building a Machine learning model from legal text corpora is vital part of discussion. The algorithm leverages the approaches to classifying data based on the already studied instances and predict accurate results with prior records and facts obtained in terms of corpus of legal cases. Machine Learning approach consists of several steps in applying the context-based information extraction methodology. The steps are: Data pre-processing, integration, mapping, learning algorithm, Visualization, Entropy based Data Mining [6].

IV. Background

A. Overview

In this section we give a brief overview about the implementation phases in order to extract the content files with respect to GATE and spaCy. We then discuss on the comparison of GATE pipeline with respect to spaCy and summarize on how spaCy is able to achieve the pre-processing tasks and other annotation tasks better than or equivalent to GATE. In upcoming section we will discuss the processing pipeline of GATE which then gives clear idea on the aspects discussed in Overview section.
In todays digital era there is a need for information extraction in almost all the fields. Due to its popularity and usefulness it is now gaining most attention from researchers and industry experts. As we know that the source of information is in digital PDFs documents for law-based corpus, there is a need to consider heterogeneous ontology-based context extraction approaches. This approach to some extent can help in reducing the search space in order to extract contents based on context-based extraction among vast collection of law-based corpus. Annotation is one of the concept to extract conceptually relevant documents for references. Initially let us discuss on the implementation pipeline. Refer to the Figure 2 which clearly depicts the

workflow for processing the documents in order to extract the reference documents. Steps are described as follows: [16]
Step 1: Convert digital PDF literature to machine readable file.
Step 2: The file is then subjected to pre-processing step which is intended to eliminate Stopwords, tokenization, Sentence Splitter, Orthographic coreference resolution and Parts-of-Speech tagging, Roman numeral Identification, Named Entity Resolution in GATE usually using DBpedia.
Step 3: Apply Rule-based annotation to match components.
Step 4: Matched components are then extracted into a machine-readable file (content file) with contents of TOC and their annotations (TOC, RFC)
Step 5: This content file is then treated as lookup table for further processing in information extraction process.

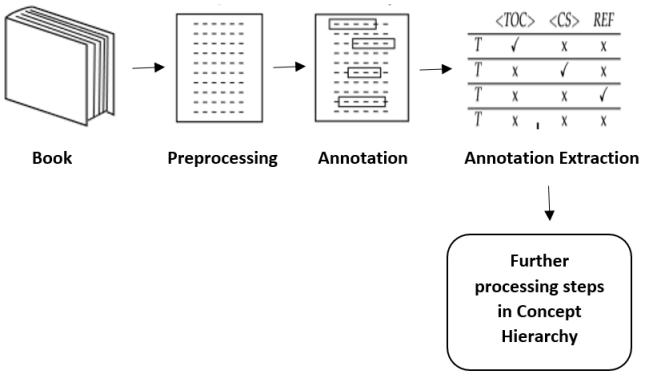

Fig. 1. Workflow to extract content files

Let us now discuss on the spaCy's ability to build pipeline. spaCy is one of the developing software intended to provide the user to create own processing pipeline and ability to reuse components. The spaCy is capable of building custom rules for annotation purpose from a given corpus this is discussed in detail in spaCy's implementation section. Initially the digital documents need to be collected from large number of legal text corpora, next focus is to transform the data into desirable machine-readable format for initial nlp based data pre-processing tasks. In the first phase, obtained machine readable text is tokenized using spaCys program which fragments the text file into smaller chunks of words and punctuation called tokens. Tokenization is based on OntoNote 5 corpus which supports multi-lingual annotation and works on multiple annotation layers. The function calls in spaCy is capable of returning a tokenizer referred as newly constructed object.
Parts of Speech Tagging in spaCy is based on the Neural Network models and visualizing using dispaCy. spacys architecture provides user the ability to perform training on neural network models.This concept is covered in more detail in spaCy's Neural network models section. The next is to apply rule-based annotation spaCy is capable of coming up with custom rules for rule-based annotation. In this work our interest is to come up with a model to extract law-based information which supports German language. spaCys models are developed to work on law-based corpus. spaCy provides simple yet effective functionality for built in entity types trained on OntoNote5 corpus called law-based taggers. This makes spaCy user-friendly due to which it is gaining most attention from most of the industry experts and researchers. spaCy also provides ability for the user to initially train the data and build custom rules which is another major idea to overcome issues of rule-based annotation issues of GATE which is explained in later sections of this paper. spaCy's multi-language support is another factor that makes it more useful independent of the language being used in this platform. spaCys architecture is constantly under the research and there are developing modules in terms of certain functionality which is addressed in implementation. spaCys ease of use and its custom defined rule functionality are the major focus which is discussed in this document.

B. Processing pipeline in GATE

This section provides a sample work-flow of GATE. Initially GATE begins its workflow by importing law-based document which is converted from PDF to machine readable format using pdftotext. Refer to the Figure 2 [4] which gives a PDF layout. [4] The process is then followed by applying

Inhaltsverzeichnis

Fig. 2. PDF Layout with double columnar format

required GATE plugins as shown in Figure 3. The resources are obtained by importing readily available plugin into GATE and then used in pre processing. Resource as shown in the figure 3 are reset, Tokeniser, Tokeniser postprocessor, Sentence splitter, German gazetteer and orthomatcher, Roman Numerals Tagger, Standford POS Tagger and many more.
The process then follows a series of steps involved to get the annotation list based on the required section. Refer to figure 4 which gives annotation list based on DBpediaLink.
After carefully implementing the pipeline and applying JAPE rules the resulting two set of files contains concept based information. One file provide the TOC content which is information regarding tokens, the other file provides the sentences that contains reference RFC (Reasons For Citing) and REL (relationship component). These files are a kind

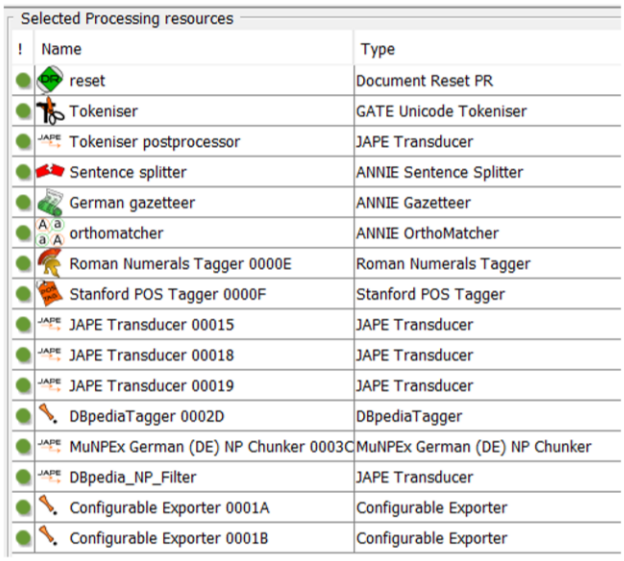

Fig. 3. GATE processing pipeline resources

of csv format. Now the GATE has provided with necessary extracted content files.

C. spaCy's capability

spaCy is an open-source natural language processing tool which supports Python and Cython programming languages. spaCy consists of features such as POS tagging, Dependency Parsing, Named Entity resolution, tokenization, sentence segmentation and rule based matching. The Features are discussed briefly below: [7]

- The Non-destructive tokenization: This will divide the sentence and will create Doc objects in the given sentence.
- Named entity recognition: Utilized to discover named entities in documents through annotations.
- Multi-language support: Around 34+ languages help in identifying unique named entities in various languages.
- Statistical models: spaCy has statistical models trained for about 8 languages to handle application requirements as per the need of the workflow.
- Pre-trained word vectors: This would perform a comparison of objects and will predict similarity of the content to flag as duplicates.Mostly general purpose similarity is performed on multiple languages.
- Easy deep learning integration:Performed by statistical models built on mutli-lingual corpus.
- Part-of-speech tagging:Uses statistical models to predict the tag or label based on the context.
- Labelled dependency parsing: Parser will check the sentence partitional frontier and detects the nouns or phrases present in the sentence.
- Sentence Segmentation: Based on the contents, complete analysis is performed by dependency parser.
- Named entity resolution: This would be useful in resolving named entities and is performed by entity types trained on OntoNote5 corpus called law-based taggers.

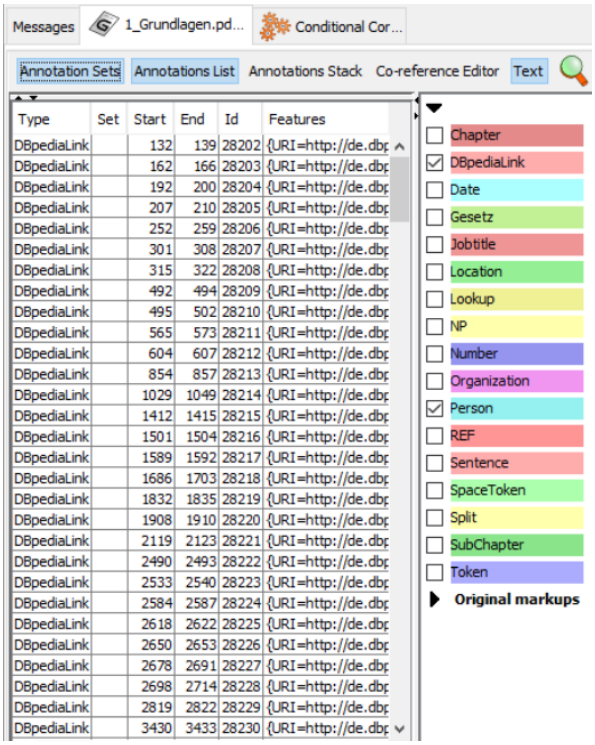

Fig. 4. Annotated table of contents DBpedia components

- Efficient binary serialization: The task of translating the content and structure as required file format such that all contents including tokens or vocabulary are stored.
- Evaluated accuracy: spaCy performs well with respect to NLP pre-processing tasks such as tokenization, roman numeral tagging, rule matching but lacks behind in DBPedia tagging, which shows certain inaccuracies to be addressed in future enhancement.
- spaCy is capable of building Machine Learning model to train for Named Entity recognition. The main use of such Machine Learning models are to be able to learn from the training samples and designed to be able to predict the outcome from data. spaCy provides Thinc as a machine learning library with abilities to optimized for CPU usage and deep learning with text input [6].

D. spaCy's Neural Network models [7]

The Artificial Neural Network(ANN) is a connected group of neurons, which work together for the purpose of information processing in case of NLP. An example of ANN is Multi layer perceptron(MLP) which can have an input with multiple hidden layers and one output layer [10].The input will be obtained from word embedding process. The hidden layers will capture the input such that only relevant features are utilized by the output to obtain required information. MLP also has an activation function in hidden layer neurons such that whenever threshold set is reached by the input, the vector gets activated such that desired result is obtained as output [10]. The figure 5 explains the neural network model with input that propagates on information to the multiple

hidden layers. These hidden layers with activation function send further data to output layer. Another type of ANN is Convolutional Neural network(CNN) works well in image processing purpose as they can find relevant patterns in close networks [17]. Usually,the CNN requires attention layer to get good visual insight of the image by classification or segmentation technique [1].

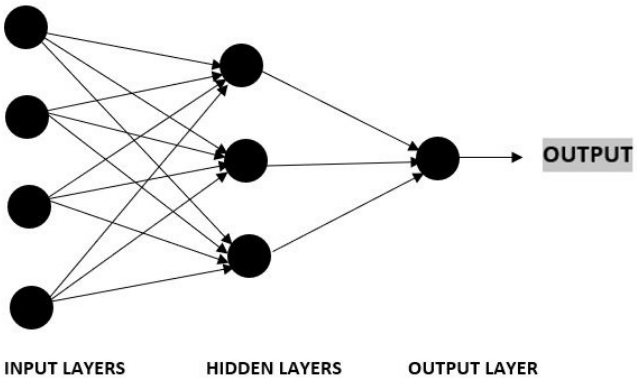

Fig. 5. Neural Network structure

The spaCy architecture uses Multilayer perceptron, which utilizes numerous hidden layers to classify data and Convolutional Neural Network is mainly helpful to extract high-level features from complex sentence structure. Deep Learning constitutes four components for encoding a sentence, which are: embed, encode, attend and predict.The prediction of statistical models show how neural network is structured and can be understood by embed,encode,attend and predict framework.

The Words represented in the Embed stage would be re-calculated such that the contextual meaning from dictionary is obtained. This representation of sentence is converted to summary vector, which is then condensed to predict information. The different modules of an encoded sentence can be understood by varying data shapes and the manipulated data shapes are said to be a small inventory of items with varying sentence structures. There are different types of shapes, where one of them is Integers with category labels, which could be words as well as part of speech tags or any discrete representation which predict output identifier.

In Embed step, spaCy learns word representation in documents by embedding terms in Doc2 Array procedure, where four attributes are extracted and analyzed. The attributes are : norm, which will normalize string form which may be lowercase or any adaptable form which will calculate string transformations, prefix attribute as well as suffix attribute usually having default size of length 3 and word shape attribute, which will replace the ”digits” with letter d or ”lowercase characters” with lowercase w and ”uppercase characters” with uppercase W. These attributes are useful for unknown words in the model, in case if model encounters not seen norm, then it comes up with a representation which is understandable. By this procedure, extraction of four columns views in document will be done, which is similar to feature extraction. Finally, we get the result in matrix of numerals with four columns and one row per word in document, each column is then embedded into a table and the embedding table uses hashing trick which is termed as bloom embeddings. In NLP, usually there is a fixed inventory present in the embed table and all words outside that inventory is saved as single-out-of vocabulary vector, but spaCy will mod the hash strings into a table, so lots of words end up in same vector representation from key.Thus, the solution to get distinct representation is to calculate different keys with random seed and mod them. Around four keys are used usually to get the solution, where vector for norm is sum of four different representations in the table. This feature was tested in spaCy by taking Spanish Part of speech document 200 rows, the words were learnt well through the universal dependency parsing, which analyses the word forms in multi-lingual document by grabbing the tokens and this process was facilitated by displaCy feature of spaCy [7]. Hence, we can conclude that we dont require multiple rows and embedding strategy to take words into account. Another example for english words with 1000-2000 also worked well even though the embedding table was small and dense because it had word representations running without having much cost of updating the whole table. So, once embedding table is completed for each feature, then concatenation of all the features together is performed, which finally outputs a vector form.

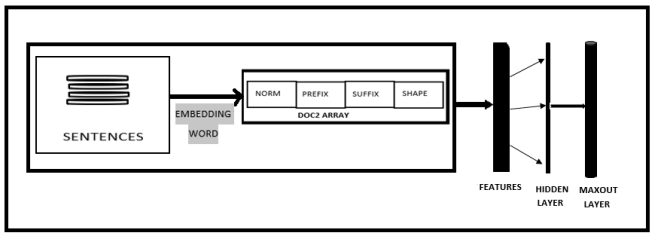

Fig. 6. Convolutional Neural Network structure

The Figure 6 is an illustration, which uses Pipelines to show function concatenation with four attributes i.e., norm, prefix, suffix and shape, which outputs the vector from word embedding procedure and this will be the input to the neural network, which consists of one hidden layer and one maximum output.The feed forward neural network gives 128 dimension vector per word that considers sub word features and thus, we learn vocabularies which are arbitrary sized. Once the completion of word embedding is performed for each word to output an vector. Next step would be to learn the representations by looking into neighboring vectors present in the document through Convolutional Neural Networks.Basically, CNN is selected as it is computationally cheaper due to support to parallel processing. Usually, CNN performs dimensionality reduction by taking input sentence for which tokenization process leads to an word embedding matrix from which features extracted to output precise vector

of final sentence. Apparently, spaCy utilizes tri-gram CNN layer as building block, which takes three words together at a time as input and feed into MLP that combines the information to get vector with dimensionality reduced and these features produced in output space, will be fed into input forward.

An Attention Mechanism reflects appropriate part of an image or information based on user needs. Usually, to learn weighted summary of query we use an attention mechanism output in the neural network But, spaCy will extract features manually, since the Features are less satisfying like buffer or entities that can be mapped back into document. The manual feature extraction layer has translation layer which then goes into hidden layer, but there could be some wait time. Thus, we consider the vector assigned to the first word of last named entity and the last word of previous entity and there is no boundary to consider the preceding entities. We use greedy search technique using which arbitrary feature functions, which can be written instead of dynamic programming The Predict layer is the basic layer of Multi-Layer Perceptron and consists of parsing loop where insert word embed the words of document and feed into tri-gram CNN, so that we get one row per vector considering context of word sensitive to phrases.Then, we compute attention layer which is the first hidden layer, where all computations are performed, where one word at time computation is considered. Once features are computed, we use basic MLP perceptron to get action probabilities on procedure to validate which action is valid for given sentence to come up with suitable action to be performed.The best valid action once performed will be proceeded forward in loop. CNN used here as transition based works well as it is mostly equivalent to sequence tagging.

V. Implementation Details

A. WorkFlow

In this solution we designed and implemented the work-flow that tries to replicate and achieve the tasks and results produced by the processing pipeline of **GATE**. In the below figure 7 the workflow of the proposed solution using Spacy's API is mentioned. The work-flow loads the PDF documents from the file and extracts text from the file. This extracted text is then passed into SPACY's NLP pipeline. When nlp is called on a text, the text is first tokenized and a DOC object is produced. This DOC object is passed into the processing pipeline where it is treated in several different steps. The default pipeline is composed of several modules namely *spaCy's POS tagger*, a *parser* and an *entity recognizer*. At each stage of the pipeline, spaCy's component returns the processed Doc, later which obtained Doc is passed on to the next component. The *tokenizer* breaks the text into chunks of tokens. Then for these each tokens in the Doc Object the *tagger* assigns a parts of speech tag. The *parser* assigns the dependency labels and also segments the sentences in the text. Later the *NER* detects and labels the named entity for each token and hence different features related to each token is extracted.

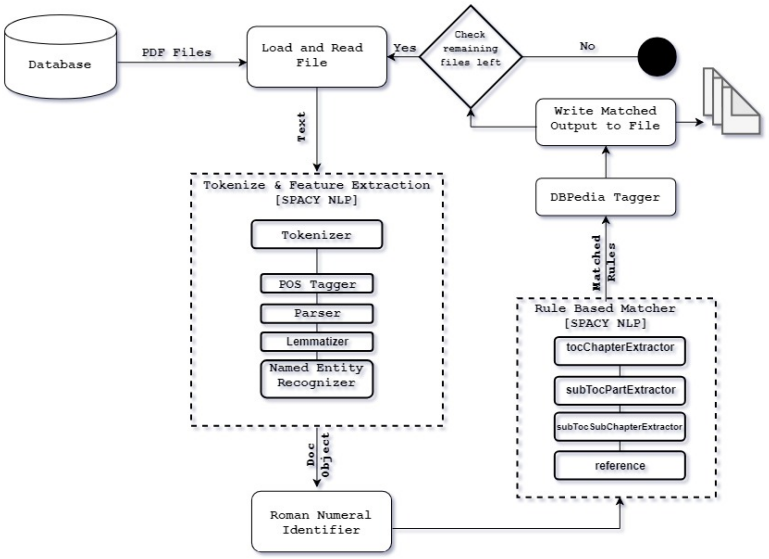

Fig. 7. Workflow of the Implementation

Once the Doc object is received each token is checked to identify the Roman Numerals. In the next modules custom rules are developed as written in Gate for matching patterns and phrase using the features of the tokens. Custom attributes are also added to Spacy's *nlp.vocab* in order to create the custom patterns. This eventually calls Spacy's *matcher* function which identifies the matched phrases in the text according to the stated rules. In the DBPedia tagger module the matched phrases are tagged with DBpedia links. Finally each of the matched phrases are store into a file along with their respective rule names, DBPedia links and position in the text. This process continues until no files are left to be processed. The following subsections describe each module in detail.

B. Tokenization

During the pre-processing step, prior to applying Stopword removal or Stemming process it is important segment the words into smaller chunks called tokens. Tokenization is one of the vital process to match and analyse the text based on the semantic and syntactic structure. spaCy's tokenizer gives ability to tokenize the strings and text streams. Several methods provided by spaCy are capable of tokenizing in terms of certain strategy to leverage the context later on. Based on the required format needed to use in Information extraction process. Certain methods used in spaCy that strives to return tokens based on prefixes, suffixes and infixes. For an instance ORTH field represents attributes to match exact strings. Using these methods spaCy uses its program to modify and work on the text and documents provided for initial preprocessing step of data. Also, function call supported in spaCy returns "tokenizer" as a newly constructed object [7].

C. Feature Extraction

Extracting relevant features from the corpus that is not only relevant in terms of entity recognition but also which

makes most sense with respect to specific domain is the key for a successful model implementation. In our solution we have worked on Feature Extraction by selecting certain set of data and transformed data to make the feature selection process easier [14]. Due to the reason where spaCys model support multi-lingual approach, it makes it easier to simply import the german model into spaCys workspace and work on feature construction using spaCys available set of technologies to transform the data to desirable context. spaCy provided ability to extract features based on technologies such as: Parts-Of-Speech using linguistic annotations for token attributes. In this technique, spaCy provides libraries that are efficient by reducing the memory consumption by encoding strings to hash value. Dependency labels and named entities also support spaCys feature extraction. This was achieved by writing customized tokens. which in turn was useful to work with rule-based matcher. Another approach is using Rule-based morphology which is not taken into serious consideration in our approach although would give the information processing pipeline to extract morphological features from the corpus [5]. In our implementation we have taken Dependency parser into consideration which as explained in spaCys Neural Network model.

D. spaCy's POS tagger

Part of Speech (POS) tagging is a pre-processing step which would eliminate irrelevant syntactic information extracted by parser, that results in speed up of the parsing process.
The stage of POS tagging in information extraction pipeline gives the ability to leverage statistical model. This model in turn enables to predict contextually relevant labels.The unnecessary Part of speech words dont help in extracting any required information but instead make summarize words like adjectives, delimiters will be eliminated through the code implementation [2].
There are two prime tasks concentrated during POS tagging: first is to induce categories that help in generalizing language based models for text recognition and next would be to check on how effective the linguistic information can be liable. Our focus would be mainly on certain rare words where range of languages are adequate and secondly, we look on how to incorporate morphological linguistic information.For good performance,correct model is to be created for infrequent words, since we already have information on frequent words. Thus, the Tagger would extract sequence of words for each parse and then, checks if correct class is assigned by the model created. Thus, we concentrate on how most of disambiguate tagging can be performed.These above steps help in improving accuracy of the result [7].

E. Roman Numbers Tagger

The Roman Numeral tagger will effectively help in identifying roman numerals present within the corpus. Since, table of contents in the given document has the list available as roman number, it is necessary for roman numeral tagging to check specific structure of document[7].
The implementation of roman numerals in this work is such that for every token in the document, the tagger will look if the letters in the token contain only X,V,I,C,L,M,D. If true, the tagger will parse the token as per the roman numeral function, which is based upon tokens matched. Else, moves ahead.Size action will be performed by the tagger.

F. Pattern Matcher

The Pattern Matching in spaCy is performed using PhraseMatcher or Matcher. The Phrase Matcher recognizes patterns and matches the characters as Doc objects. While, Matcher matches all the sequences in the document according to the description of token [7].
In spaCy, the PhraseMatcher would look into token sequence present in document. In the current work, Matcher has been used to do Pattern Matching of input streams. The Matcher refers to the supplied patterns those are written in order to match tokens in the given document. From the machine readable data format we use single string variable to embed the total document. This matches stream of terminologies in document and return for further processing. The rule based pattern matching is done based on several criteria in mind. While in GATE the obtained rules are created from JAPE grammar. In spaCy, we have used Python in combination with Pattern Matcher to obtain the required contents from German legal sections. The rules written in spaCy are:

- toc-chapter: extracts chapters from Table of Contents.
- part: extracts sub chapter from Table of Contents
- subChapter: extracts identified number, date features.
- SubSubChapter: extracts identified Roman numeral features.
- Reference: extracts the features for Reasons for Citing file RFC those are the law section details.

The comparison of GATE and spaCy's run-time for each of these feature extraction task for pattern matcher has been further explained in detail in Table II.

G. DBPedia Tagger

DBpedia Spotlight is used in spaCy for annotating purpose. The tool is capable of annotating the required contents from DBpedia resources automatically. The extracted DBpedia text contents are usually in unstructured data format. DBpedia Spotlight tool is leveraged as a solution to link these extracted unstructured information [3]. In this module each text phrase matching its respective rules are fed in to DBPedia Spotlight API and it returns results if there is a confidence score of 90 which is also set as a parameter.

H. Writing Output

The matched phrases along with the respective DBPedia tags are inserted in the output list in form of the following structure:
{*Matched Phrase, Start-End Position, Rule Name, Sub-Rule Name*,* {*DBPedia Details*}* } This output list is then written into the *Rules_FileName.txt* file in the project directory.

VI. RESULTS & EVALUATION

In order to maintain the grounds of evaluation as neutral as possible results on the input file *"2_Geschftsbeziehung_und_Bankvertrag.pdf"* from both the presented solution and Gate were compared. Although the architecture of both the solutions are different they were evaluated on the basis of Run-time and Number of Matched Phrases for each Rules. The evaluation results are presented in the following tables and figures.

	GATE	Spacy Solution
Tokenization & Feature Extraction	37.929	04.50
Roman Numeral	0.01	0.02
Rule Matcher	0.491	04.11
DBPedia Tagger	9.155	54.12
Others	12.402	1.19
Total Time	60.514	63.94

TABLE I
COMPARISON OF RUN-TIME FOR EACH MODULE IN SECONDS.

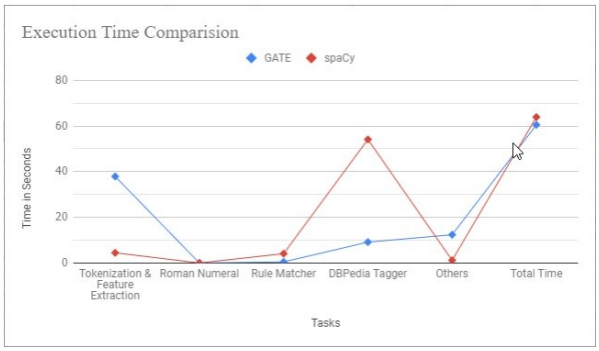

Fig. 8. Comparison with respect to No of Rules Matched

While comparing the run-time taken by each module as well as the total time between GATE and Spacy some valuable insights were gained. These insights can be realized from Table I and Figure 8. While Gate goes through a number of processes namely *Tokeniser, Sentence_splitter, German_gazetteer, German_grammar, orthomatcher, Stanford_POS_Tagger* for the feature extraction step it takes up most of the run-time. On the otherhand Spacy's NLP pipeline gets executed in a much shorter time. The comparison of the **DbPedia Tagger** may vary as it depends on the connection speed of the network. Overall with respect to time complexity GATE is 3.426 seconds faster than the proposed solution using Spacy.

Evaluating both the tools with respect to the number of phrases from the Table II and figure 9 matched spaCy performed better or at par with most of the rules. Even though there are some false positives in spaCy it is observed

	GATE	Spacy Solution	
tocChapterExtractor	7	7	**toc-chapter**
subTocPartExtractor	4	3	**part**
subTocSubChapter Extractor	1	5	**subChapter**
subTocSubSubChapter Extractor	6	13	**SubSubChapter Number**
reference	42	43	**Reference**
Total Matched Rules	60	71	

TABLE II
AN EXAMPLE OF A TABLE

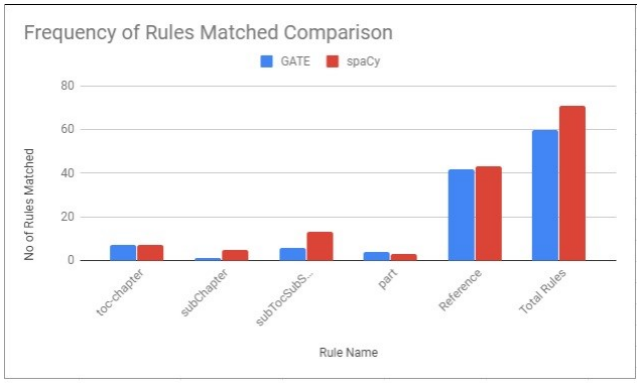

Fig. 9. Comparison with respect to Execution Time

that spaCy's Rule based matcher is identifying most of the phrases that are cover by the Jape rules matcher in GATE.

VII. DISCUSSION & FUTURE WORK

Although in the project it was possible to recreate the required functionality of GATE by using Spacy the architecture of both the tools are quite different. Spacy being a great tool for Natural Language Processing it is still in its early stages of developments and needs quite a bit of feature enhancements.

- While working with the NER models used by Spacy, it provides us with only 4 classes for the German Language namely PER, LOC, ORG and MISC entities. This made the construction of the Rules very difficult.
- Writing rules / patterns to match phrases of words, Gate uses complex logic that include
  - OR conditions in-between sequences of tokens.

```
({Token.kind==word, Token.orth!=lowercase}|({Token.string!="Abs"}
{Token.kind==punctuation}{Token.kind==number}))
```

  - Nested quantifiers for multiple tokens in a sequences.

```
(({Token.kind==word}|({Token.string==","}{Token.kind==number}))?)
```

– The rules also treats phrases matched by other rules as input tokens.

```
({SubChapter}|{Part})
```

– It also specifies patterns having tokens with multiple attribute of the same type and also mentioning the range of the length of the tokens.

```
{Token.kind==word, Token.length<3, Token.string!="Rn", Token.string!="Art"}
```

Spacy's Rule-Based-Matcher is still not ready to handle complex logic like the above mentioned. In future releases these features will be implemented to the tool as feature request is made to the development team of Spacy [9]. Moreover GATE can process one file at a time and everytime one has to manually insert the input files into the tool. In the proposed solution a new feature is included with which all the .pdf files present in a specific directory will be passed as input and for each input file a respective output rule file will be generated automatically. Adding to the future scope one can train Spacy's NER model with OntoNotes 5 corpus and also try to improve the Named Entity taggings.

VIII. CONCLUSION

In this work we have given brief overview about the processes involved in Information extraction process and how important it is to come up with domain specific information extraction solution in this digital era. The objective of this work is to overcome the existing issues of GATE by building nlp pipeline in spaCy. These results have been captured in the Evaluation section giving an idea about run-time complexities of GATE versus spaCy. We have discussed on the existing tools and technologies available to perform basic nlp tasks and we begin our work by introducing nlp pipeline built by GATE and its rule construction using JAPE rules. After examining enough techniques entangled with rule-based annotation from DBpedia tagger, we have settled down to use spaCy's Named Entity Recognizer. Thereafter, we have explained reasons for choosing spaCy for context file extraction. We also have ignited the spark about usage of Neural Network models that supports multi-lingual corpus to improve feature extraction in spaCy. High relevant feature extraction is achieved through word embedding process via embed, encode, predict and attend framework. These features are then fed into CNN that predicts appropriate actions. In addition to this, time computation for content-based extraction from corpus is minimized.

As we see in evaluation process the performance comparisons of GATE versus spaCy has been clearly depicted, GATE is able perform in very short span in terms of Rule Matcher, DBpedia Tagger and Roman Numeral Tagger as in case of spaCy, it is avoiding latency in terms of Tokenization and Feature Extraction. GATE is capable of matching pattern-based rules hierarchically using JAPE grammar, while in spaCy it is still the future enhancement to be achieved. The existing rule-based pattern matching approach in spaCys model is done using pattern matcher. This acts as an intermediate solution for the non-existence of hierarchical rule matching. The pattern matcher in spaCy considers one rule at a time. However, based on our findings with the development team of spaCy, we discovered that there are constant workarounds going on for spaCys Named Entity Tagger, supporting hierarchical rule matching and Neural network components that serves as a major advantage to capture efficient results in future.

REFERENCES

[1] Danqi Chen and Christopher Manning. "A fast and accurate dependency parser using neural networks". In: (2014), pp. 740–750.

[2] Mary Dalrymple. "How much can part-of-speech tagging help parsing?" In: *Natural Language Engineering* 12.4 (2006), pp. 373–389.

[3] *DBpedia Spotlight Shedding light on the web of documents*. URL: https://www.dbpedia-spotlight.org/api.

[4] Peter Derleder, Kai-Oliver Knops, and Heinz Georg Bamberger. *Handbook on German and European Banking Law*. Springer Science Business Media, 2008.

[5] Christopher Dozier et al. "Named entity recognition and resolution in legal text". In: (2010), pp. 27–43.

[6] Andreas Holzinger. "Introduction to machine learning and knowledge extraction (MAKE)". In: *Mach. Learn. Knowl. Extr* 1.1 (2017), pp. 1–20.

[7] Montani Ines Honnibal Matthew. "Everything you need to know' 2016". In: (2016). URL: https://spacy.io/usage/spacy-101..

[8] K Tamsin Maxwell and Burkhard Schafer. "Concept and Context in Legal Information Retrieval." In: (2008), pp. 63–72.

[9] Ines Montani. "Spacy Feature Update Request." In: Spacy. 2019. URL: https://github.com/explosion/spaCy/issues/3091#issuecomment-457833207.

[10] Kang Nahua. "'Multi-Layer Neural Networks with Sigmoid Functionfffdfffdfffd Deep Learning for Rookies', 2017." In: (2017). URL: https://towardsdatascience.com/multi-layer-neural-networks-with-sigmoid-function-deep-learning-for-rookies-2-bf464f09eb7f..

[11] Jesus Manuel Niebla Zatarain. "Artificial Intelligence and Legal Analytics: New Tools for Law Practice in the Digital Age". In: (2018).

[12] Sunita Sarawagi et al. "Information extraction". In: *Foundations and Trends® in Databases* 1.3 (2008), pp. 261–377.

[13] Giulia Venturi. "Design and development of TEMIS: a syntactically and semantically annotated corpus of italian legislative texts". In: (2012), pp. 1–12.

[14] BERNHARD Waltl et al. “Automated extraction of semantic information from german legal documents”. In: (2017).

[15] Bernhard Waltl et al. “LEXIA: A data science environment for Semantic analysis of german legal texts”. In: *Jusletter IT* 4.1 (2016), pp. 4–1.

[16] Sabine Wehnert et al. “Concept Hierarchy Extraction from Legal Literature”. In: (Oct. 2018).

[17] Krzysztof Wróbel et al. “Convolutional neural network compression for natural language processing”. In: *arXiv preprint arXiv:1805.10796* (2018).