Towards Extracting Entity Relationship Diagrams from Unstructured Text using Natural Language Processing

Vaihunthan Vyramuthu
Department of Electronics and Computer Science
Aalen University
Aalen, Germany
email: vaihunthan@gmail.com

Gregor Grambow
Department of Electronics and Computer Science
Aalen University
Aalen, Germany
email: Gregor.Grambow@hs-aalen.de

***Abstract*—In computer science, the creation of applications usually involves the process of abstracting real world entities and relationships and creating models to be able to process these. One crucial part of this is data storage and management and therefore the creation of data models. As a first step, usually the Entity-Relationship (ER) model is used. However, the transformation from real world descriptions in natural language to standardized ER diagrams can be tedious and error-prone. Recently, Natural Language Processing (NLP) has gained much attention but this specific area is still mostly handled manually by humans. This paper describes a hybrid system for capturing ER model components from German texts using NLP. That way, time-consuming interpretation of textual database scenarios can be automated. We implemented and tested both rule-based and model-based approaches, whereas the main extraction is performed by the rule-based variant so that the entities, attributes, relationships and cardinalities can be strategically identified. The results of the model-based approach are used as a comparison to the rule-based results and can be applied for correctness checking and improvement of the results. Furthermore, we conducted a preliminary evaluation, which shows promising results. A hybrid approach can be better than a classical approach, as it combines the precision of the rule-based system with the flexibility of the model-based approach. This may lead to a more robust and reliable extraction, as errors in one of the approaches can be compensated by the other.**

***Index Terms*—Entity-Relationship Model, Natural Language Processing, Named Entity Recognition, POS-Tagging, SpaCy, LSTM**

I. Introduction

In today's data-driven world, large amounts of text are generated that can contain valuable information. Examples of this include police reports from online press portals, Twitter comments or Amazon reviews. Various analyses are possible using such text that can provide insights about future trends, user ratings and sentiment or other questions [1]. Extracting structured data from unstructured text, however, is a complex task that mostly requires manual processing. Automating this process can save time and resources and improve the efficiency of data analysis [2] [3].

At the same time, the use of database systems for storing and managing data is prevalent. A common approach to database modeling is the use of *Entity Relationship Models* (ERM), followed by conversion to a relational model and finally to *Structured Query Language* (SQL) statements to represent the real world data in a database [4].

Manual data extraction involves reading through the text and interpreting it correctly so that database modeling is implemented as efficiently as possible. The aim of this paper is to automate data extraction and thereby simplify the process of database modeling and integration. The results of this process are considered successful if the discrepancy between the human and software interpretation of a text with regard to correct ERM generation is as small as possible. This paper proposes an hybrid approach for the identification of entities and attributes, as well as the recording of entity relationships (including cardinalities). A hybrid approach may lead to increased error resistance. Rule-based systems alone are often prone to inaccurate results if the texts do not exactly match the expected patterns. The model-based approach compensates for this with its ability to learn from contexts and recognize variations. The analysis of semantic contexts and complex text structures that go beyond the recording of entities, attributes and relationships is not part of this work. The implementation of the approach focuses on the processing of texts in German language.

The paper is organized as follows: Section II presents related work, followed by Section III, containing details on the concept and architecture of the proposed approach. Implementations are presented in Section IV. Subsequently, we evaluate the implementation in Section V. Finally, the conclusion, including limitations, can be found in Section VI.

II. Related Work

In literature, there exist several approaches dealing with the creation of ER models from natural language. In the work of Ghosh et al. [5], a method is proposed that uses grammatical knowledge patterns and lexical and syntactic analyses of request texts to create ERMs. This system assumes that the input text consists only of simple subject-predicate-object sentences for correct information extraction. This sentence structure

makes it possible to detect entities (subject), relationships (verb) and attributes or other related entities (object). Until the step before the domain-specific database is used to identify the ERM components, only NLP techniques, such as sentence segmentation, word separation (tokenization) or *POS-Tagging* are used [5].

The segmentation was carried out in this work in such a way that cases in English, such as "Mr. Mustermann", in which (.) appears after Mr, are not recognized as the end of the sentence. In *POS-Tagging*, the individual words are assigned to the corresponding word types, e.g., noun, verb, pronoun, adjective, preposition etc. In this publication, the recognition of entities, attributes and relationships is performed using a database and the *Support Vector Machine* (SVM) classifier [5].

Six tables (word & synonym for each ERM component) are implemented, which contain domain-specific words and synonyms. To capture related entities, the synonyms of a word from the table are given the same ID. This prevents redundant SQL tables from being created. Pronouns such as "he", "she" or "it" are already recognized in the *POS-Tagging* phase. In this phase, the pronoun is identified based on the closest previous entity that is present either in the same sentence or in the previous sentences. The technical term for this is *coreference resolution* [5] [6].

In the publication by Kashmira et al. [7], the ERM components are recorded using neural networks. Three main modules are presented, namely the preprocessing module, the machine learning module and the ER modeling module. In the first step, the preprocessing module is implemented using *NLTK* to preprocess the text. This includes steps, such as converting the text into lowercase letters, tokenization into sentences, etc. The machine learning module is then implemented using supervised learning. This module is trained with an English dataset where words are categorized into different categories including entity, sub-entity, attribute and irrelevant category. Four classifiers are taken into account when training the model: *Random Forest*, *Naive Bayes*, *Decision Table* and *Sequential minimal optimization* (SMO) [7].

To address the problem of attribute selection for entities in an ER diagram, the proposed model uses a combination of *ontology* and *web mining*. By using ontology, an attempt is made to filter relevant attributes from the extracted entities. In addition, web mining is used to obtain further information from the web that can be helpful in determining the attributes [7].

The publication by Habib [8] also follows similar preprocessing steps at the beginning, like Ghosh [5] and Kashmira et al [7]. After the parsing process, the grammatical sentence structure is obtained so that the components of the ERM can be determined based on rules. The words are converted into a parse tree structure to understand how the individual parts of the sentence are related to each other. Using appropriate rules for sentence structures, entities, attributes, cardinalities and relationships can be determined.

There are several other publications that go in a similar direction and examine the topic of automatic ERM generation in the context of NLP in more detail. One example by Omar et al. [9] describes heuristic-based analysis options for ER model generation. In contrast, Omar and Abdulla [10] pursue the approach of training a machine learning model that can extract the entities from the text. Depending on the complexity of the input text and the scope of training, the model achieves precision values of up to 85%. The results of Btoush and Hammad [11] can also be placed in a similar context. Here, a method is presented which, like [8], defines and applies certain rules for extracting information from texts.

It can be stated that two basic methodological approaches are used for ER model generation. The ERM components are determined either rule-based or using neural networks or artificial intelligence. Both approaches have advantages and disadvantages. The biggest advantage of rule-based extraction is the more time-efficient implementation as, unlike neural networks, no data preprocessing and training is required. Furthermore, the unambiguous definition of the rules ensures one hundred percent extraction probability. In neural networks, a residual inaccuracy always remains. In contrast, model-based extraction is not limited to a few rules, but can learn complex and nested sentence structures to determine the ERM components. These sentence structures can contain linguistic variations and ambiguities, which can be recognized more easily by neural networks than by fixed heuristics. The heuristic approach is more suitable for small application areas and cannot maintain its effectiveness in large application areas.

Purely rule-based methods are prone to lack of generalization, while purely model-based approaches often depend on large training datasets and have difficulties in capturing rare or complex linguistic structures. This hybrid approach has the potential to overcome these limitations by combining the strengths of both methods: The rule-based method enables accurate extraction, while the model-based method helps to validate and improve the results, resulting in a more robust and adaptable solution to different text scenarios.

III. Overview of the architecture

Figure 1 shows a general overview of the individual processes of the proposed approach. The latter is divided into a rule-based and a model-based part. The model-based algorithm can be used either to train new or existing models or to extract the ER components from a text. However, these results are not used for the resulting ERM, but are only used to compare the rule-based results. This makes it possible to check whether the final result from the rule-based process may still need to be modified manually.

The input for both parts of the approach is a text that is saved in a *.txt* file. The *output.json* file contains all ER components and relationships found in a structure that can be read by an external ER modeling tool. The artifacts (in the image: ordinary rectangles or arrow labels) represent the created files or results. These files are required for the subsequent processes.

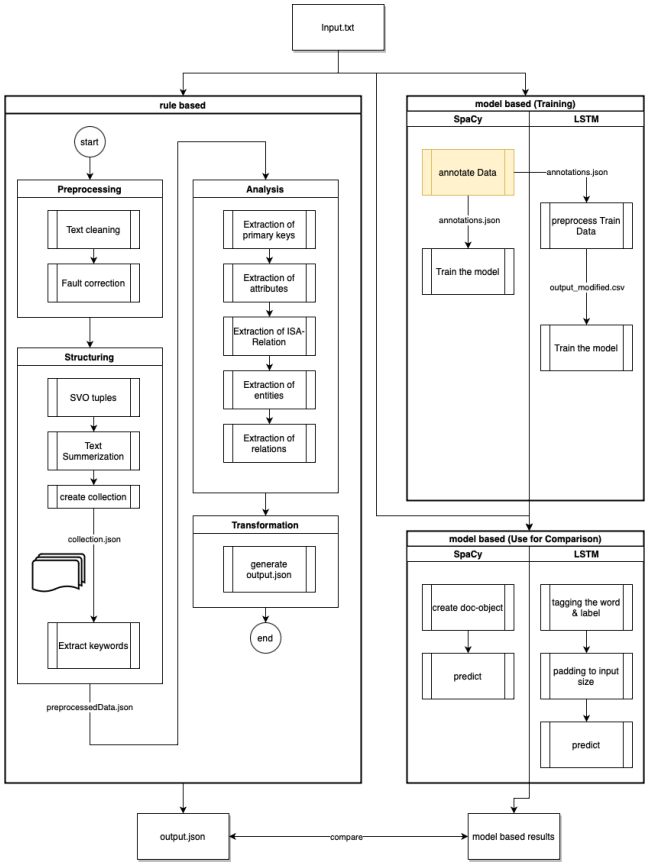

Fig. 1. Overall flow chart of the implementation.

The rule-based approach is made up of the sub-steps preprocessing, structuring, analysis and transformation. During preprocessing, the text data is cleansed and corrected for spelling errors. The structuring phase contains processes that break down the text into more meaningful parts and store them temporarily. In the main block of the analysis, the ER components are extracted one after the other.

In the model-based part, a different process path is carried out depending on the selected model type (*SpaCy*-Transformer or *LSTM*). Due to the individual input requirements of the models, the text data must be converted into the permitted form for both the training case and the use case. The process block highlighted in yellow indicates an annotation process that is carried out using an external tool. For the *LSTM* model, the output of the annotation tool must be pre-processed again into a *.csv* file so that it can be used for training. After the training processes, the model can be applied to new text data. For the *SpaCy* model, it is sufficient to convert the text into a doc object so that the analysis can be started. The text for the *LSTM* model, on the other hand, requires additional processing (analogous to the training process), which decodes the words and labels into numbers and scales the input word vector to a specified size (so-called *padding*). This hybrid combination provides an important advantage for continuous optimization. The results of the model-based approach can be used to dynamically adapt and further develop the rules of the rule-based system. This means that the hybrid system can become increasingly precise and effective over time through feedback and new data sets.

The selection of *SpaCy* as a model and NLP-Tool is based on its robust capabilities and user-friendly implementation. SpaCy is recognized for its comprehensive documentation and strong support from an active developer community. *LSTM* were chosen for their effectiveness in handling sequential data and their ability to capture contextual dependencies over extended text passages. This makes them particularly suitable for tasks that require understanding the relationships within long text sequences of information.

IV. Implementation

Starting with the rule-based part, it can be noted that it is important not to define too many special rules. Otherwise, the implementation will be too application-specific and errors for other text styles will sometimes occur. The first process in data preprocessing is text cleansing. This process is divided into three steps. Each result of the individual preprocessing and structuring steps is saved in the *preprocessedData.json*. At the beginning, the unstructured text is filtered from the *.txt* file.

In this first step, existing white-spaces or empty lines are also taken into account. In the second step of text cleansing, the individual sentences from the text are found and saved using the *SpaCy* model (*de_core_news_sm*). In the last step of the text cleansing process, unnecessary sentences that do not provide the necessary information for the ER diagram design are removed. In this process, sentences are removed using *RegEx* matches of certain words, such as “database” or “modeling”, are sorted out.

The next process in preprocessing is the error correction of words. The *Levenshtein-similarity* is used here. The Python library *pyspellchecker* checks whether there is an error for each word in a sentence. If this is the case, the word is replaced with the closest one.

In the structuring task block, the main focus is on capturing the subject-verb-object (SVO) sentence structure and some important term frequency analyses, with the help of which the text can be broken down into smaller information-rich parts. While only the *normalized word frequency* (TF) is used for the text summary, the *inverted document frequency* (TF-IDF) helps to capture the most significant keywords from the text. In contrast to SVO generation, these two structuring steps are only used optionally. These results are not actively used in the NLP workflow, because it is possible that relevant information may be lost. Another use case for these results is looking for the most essential keywords in the text in order to compare them with the entities and attributes found. It should also be noted that to calculate the TF-IDF for keyword extraction, a document corpus (*collection.json*) must be created in which all existing ER diagram sample texts are stored. The *TfidfVectorizer()* function provided by the *Scikit-*

learn library calculates the TF-IDF. No numbers are included in the keywords.

The text summary is implemented chronologically according to the following key points:

- count the number per word in the text (stop words excluded)
- calculate the normalized weight per word used by dividing the respective word count by the maximum word frequency occurring.
- calculate the sentence weight by adding the weight per word.
- search for sentences with the highest weighting.

The individual sentences are scored by adding the normalized TF for each word in the sentence. Depending on the original length of the text, a certain number of sentences is selected in descending order of the evaluation number. For this purpose, a corresponding factor is determined at the beginning, with which the number of sentences is calculated. If there are fewer than three sentences in the text, the text is not summarized.

To generate SVO tuples, the sentence must be analyzed using *dependency parsing* and *POS-Tagging* provided in *SpaCy*. Certain commands can be used to analyse the grammatical structure of sentences so that the visualization shown in Figure 2 is displayed. Similar to the tree structure, the successors or predecessors of a word are addressed with “children” or “parent”.

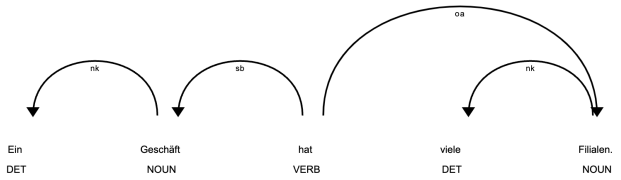

Fig. 2. Visualization of the grammatical sentence structure.

On the basis of this structure, a generally valid logic can be developed for the extraction of subject, verb and object, which is shown in the structure diagram in Figure 3. The sentences from the text are entered individually into the function. There may be several verbs in each sentence, but each verb must belong to exactly one subject and object. *Lemmatization* can be used for the output of recorded relationships in texts. This involves changing the verb from its inflected form back to its basic form. For example, the German word “angeboten” is “anbieten” after lemmatization.

A. Primary Key

The primary keys can be found using a *RegEx* comparison. The words “-id” or “-number” indicate a key candidate.

The words are first converted to lower case so that there is a certain amount of leeway in the comparison. However, the limitation is the hyphen, which must be contained in a primary keyword. The following rules are implemented as Python code:

- Determine all nouns and filter primary key via *RegEx*.

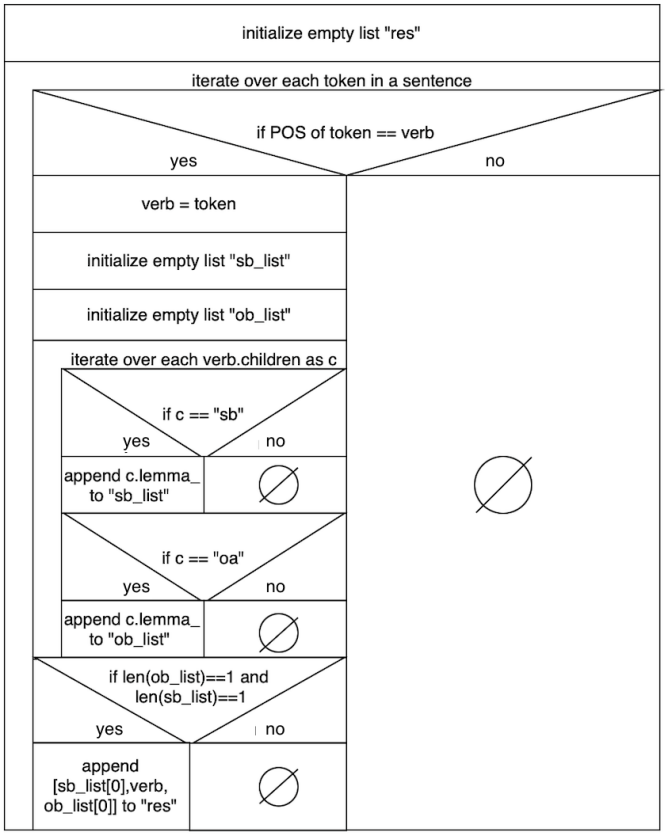

Fig. 3. Structogram for SVO extraction from a record.

- If the record only contains the word “unique”, then the closest noun should be the primary key with “noun ID”.
- If only the words “ID” or “id” appear, then the nearest noun should be the primary key with “noun ID”.

B. Attribute

Attributes are identified as soon as a sentence contains a list of more than two nouns. The first noun that occurs in the sentence is the entity to which the remaining nouns or attributes belong. This simple rule serves as a first step in the identification process, but it can be refined and enhanced through model-based results in the future.

C. ISA-Inheritance

To detect the specialization or generalization of entities, the following rules are followed:

- If a sentence contains the following verbs: [“include”, “consist”, “comprise”, “share”, “include”], then the sentence describes a generalization.
- If the sentence contains “type” as a word (noun), the first noun or entity is the generalization of the following nouns (entities).

It should be noted that the recognition of ISA relationships using rule-based approaches is limited. For example, the order of membership may be different for the entities or the ISA description may extend over several sentences.

D. Entity

In the *preprocessed.json* under the item in which the SVO tuples are stored, the subject and object in each record can be either an entity or an attribute. These tuple words are therefore compared again with the attributes and primary keys found so far. If the same tuple word is also contained in the list for attributes or primary keys, it is discarded. The final result is a list that only contains the final entities.

E. Relationship

The SVO tuples can again be used for the relationships. If both the subject and the object are contained in the list of final entities, the verb is a relationship between two entities. There are sentences that are not formulated in an ordinary SVO style, but which define further relations between entities. In order to take such sentences into account as well, a check is made to see whether two nouns occur in a sentence in addition to a verb, which are not contained in the attribute, ISA-Inheritance and primary key list. If this is the case, a relationship tuple can be extracted from this sentence again if it has not already been extracted from the SVO tuple.

F. Cardinality

Two min/max cardinalities must be determined for each relationship between two entities. The sentence in Figure 2 shows that the cardinalities can be taken from the determiners of the nouns. Once the SVO tuples have been reassigned to the complete sentences with the help of indexing, the corresponding determiners of each entity can be determined. These are then translated into the corresponding min/max value using a comparison. For the correct min/max notation, the cardinalities of the entities in an SVO tuple must be swapped.

When interpreting the adverbs “at most” and “at least”, the following word must also be taken into account, as this defines either the upper (max) or lower limit (min).

Due to the unlimited possibilities for translating this adverb, no rule-based application is suitable for this. This makes the rules too specialized for one use case.

The model-based approach primarily serves as a comparison tool for the ER components found in the rule-based algorithm. Therefore, the hybrid approach is also advantageous for performance reasons. The rule-based methods often deliver faster results as they do not rely on extensive calculations, while the model-based part intervenes where more in-depth analyses are required. This efficient applicability ensures that the goal is achieved faster without losing accuracy.

V. Limitations

The following aspects were not taken into account in the rule-based algorithm:

- Attributes for relationships.
- special cardinalities (“two”, “three”, etc.).
- weak entities, relationships, attributes, etc.
- described ISA-Inheritance across several sentences.
- multi-valued or complex attributes.

In the future, results from the model based approach could be used to include more ER components in the results. An extension to the rule-based approach for this would be rather difficult, as a generally valid formulation of the rules is difficult and often tied to a specific use case.

VI. Evaluation

The evaluation is based on several German texts that describe a specific DB scenario. This means that they contain specific formulations that describe the ER components. The rule-based algorithm only extracts ER components that correspond to the defined grammatical regularities. In contrast to the model-based approach, emphasis is placed on a qualitatively correct ER extraction instead of a quantitative result set.

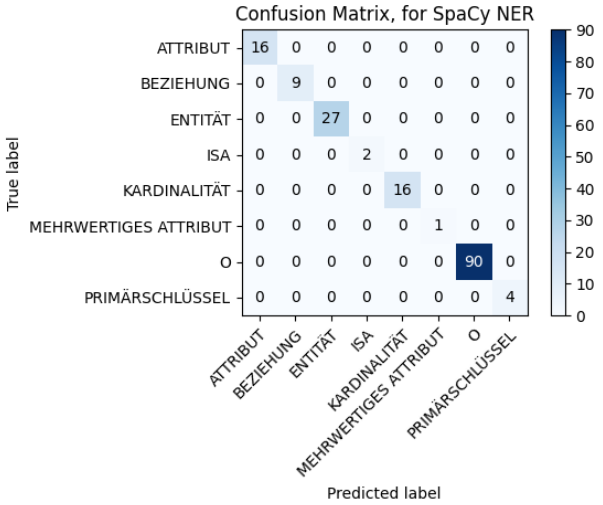

Fig. 4. Ideal reclassification result in the case of training and testing with the same data.

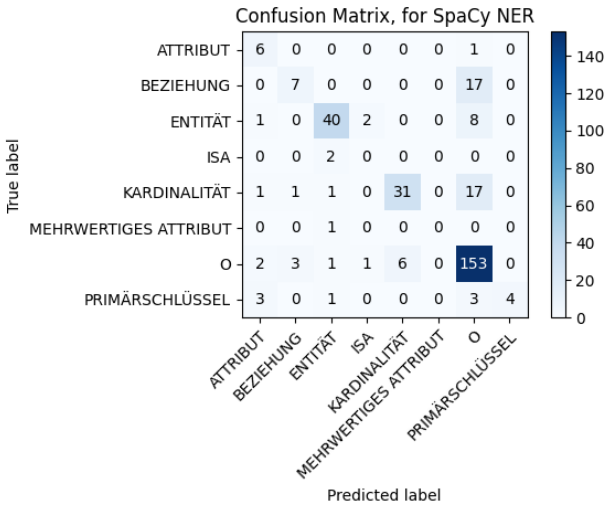

Fig. 5. Results of Crossvalidation with larger and different training and test data sets.

The Confusion Matrix in Figure 4 shows the reclassification case in which all previously labeled ER components are correctly classified during testing. However, the Confusion Matrix in Figure 5 provides more information about the generalization capability of the *SpaCy* model, because the cross-validation also examines text data that the model does not yet know and was therefore not part of the training process.

Analogous to the *SpaCy* model, the *LSTM* model is also evaluated using the learning curves shown in Figure 6. After 10 training epochs, a solid validation accuracy of 92.1% and a validation error of 0.53 are achieved.

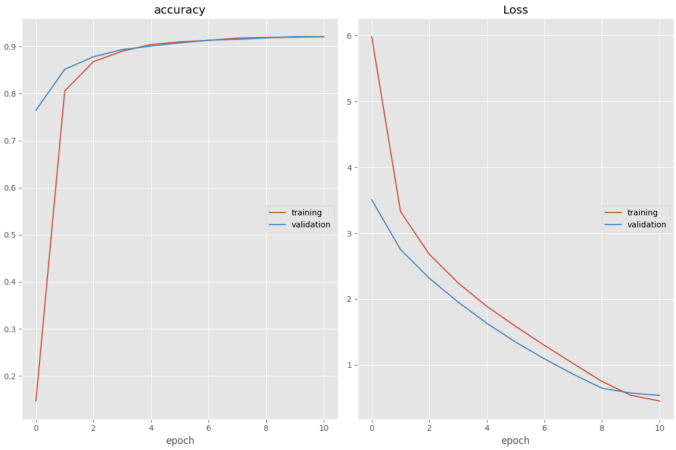

Fig. 6. Learning curves from the *LSTM* model training.

This can only serve as a first step for an evaluation. As part of our future work, we plan to compare our results to the results created by domain experts for a set of different models.

The results of the two approaches, both rule-based and model-based, complement each other very well to form a complete tool due to their complementarity. It can be stated that the challenge of the rule-based method lies in the assignment of the nouns found to the individual components. Furthermore, the connection of attributes and relationships to certain entities was not entirely trivial. On the other hand, the model-based comparison offers possibilities to correctly capture different cardinalities or ISA inheritances due to the increased flexibility in sentence structure and grammar structure.

VII. Conclusion

Extracting information from unstructured text is a complex task. Manual processing of large amounts of text is time-consuming, error-prone and not scalable. Recently, numerous approaches for automating this task have been proposed. However, there exist many cases where very specific information has to be extracted from unstructured text. These imply challenges based on the specifics and rules applicable for the desired result. One of these cases is the extraction of ER models. There exist several approaches, but they still lack different features for a complete and usable automation.

We proposed a hybrid approach to extract meaningful ER data from unstructured text. Two subsystems were developed, both of which can extract ER components from unstructured texts. Special NLP methods were used for the rule-based extraction of entities, attributes and relationships. Due to the higher reliability of the rule-based results, these were used for the final ER model. *SpaCy* and *LSTM* models can be used to validate the rule-based results. In the future, the results of the rule-based approach could be supplemented by an automated comparison of the results from the model-based approach.

In literature, there are still research gaps in the area of automatic extraction of ER models from texts using NLP techniques. This includes the integration of contextual information and the consideration of ambiguity. Another fundamental improvement is the training of the models with even more text data, so that even rarely occurring ER components, such as multi-value attributes, can be better learned. The pretrained models can be trained for other domains according to the principle of transfer learning so that the models can be used for other purposes, e.g., to create knowledge graphs.

Our future work will include the mentioned gaps: We will expand our approach to include more specific ER concepts. Further, we will concentrate on enabling a broad applicability for different domains. We will also investigate options for automated integration of the results from the two approaches. As a first step, we will extend our evaluation including use cases with real world texts and compare the results of our approach with ER models created by human experts.

References

[1] M. Kanakaraj and R. M. R. Guddeti, "Performance analysis of ensemble methods on twitter sentiment analysis using nlp techniques," in *Proceedings of the 2015 IEEE 9th international conference on semantic computing (IEEE ICSC 2015)*. IEEE, 2015, pp. 169–170.

[2] A. Rajput, "Natural language processing, sentiment analysis, and clinical analytics," in *Innov in health informatics*. Elsevier, 2020, pp. 79–97.

[3] R. Suganya, R. Krupasree, S. Gokulraj, and B. Abinesh, "Product review analysis by web scraping using nlp," in *Smart Data Intelligence: Proceedings of ICSMDI 2022*. Springer, 2022, pp. 427–436.

[4] V. T. N. Chau and S. Chittayasothorn, "A bitemporal sql database design method from the enhanced entity-relationship model," in *2021 7th International Conference on Engineering, Applied Sciences and Technology (ICEAST)*. IEEE, 2021, pp. 85–90.

[5] S. Ghosh, P. Mukherjee, B. Chakraborty, and R. Bashar, "Automated generation of er diagram from a given text in natural language," in *2018 Int'l Conf on ML and Data Eng (iCMLDE)*. IEEE, 2018, pp. 91–96.

[6] V. Bryl, C. Giuliano, L. Serafini, and K. Tymoshenko, "Supporting natural language processing with background knowledge: Coreference resolution case," in *The Semantic Web–ISWC 2010: 9th International Semantic Web Conference, ISWC 2010, Shanghai, China, November 7-11, 2010, Revised Selected Papers, Part I 9*. Springer, 2010, pp. 80–95.

[7] P. Kashmira and S. Sumathipala, "Generating entity relationship diagram from requirement specification based on nlp," in *2018 3rd Int'l Conf on Information Technology Research (ICITR)*. IEEE, 2018, pp. 1–4.

[8] M. K. Habib, "On the automated entity-relationship and schema design by natural language processing," *Int'l J Eng Sci*, vol. 8, no. 11, pp. 42–48, 2019.

[9] N. Omar, J. Hanna, and P. McKevitt, "Heuristic-based entity-relationship modelling through natural language processing," in *Proc. of the 15th Artificial Intelligence and Cognitive Science Conference (AICS-04)*. Artificial Intelligence Association of Ireland, 2004, pp. 302–313.

[10] M. Omar and A. Abdulla, "The entities extraction for entity relationship models from natural language text via machine learning algorithms," in *Proc 4th Int'l Conf of Basic Sci and Their Appl, Elbeida, Libya*, 2020.

[11] E. S. Btoush and M. M. Hammad, "Generating er diagrams from requirement specifications based on natural language processing," *Int'l J of Database Theory and Application*, vol. 8, no. 2, pp. 61–70, 2015.