engineering proceedings

MDPI

Proceeding Paper

Application of a Syntax-Based Text Extraction Algorithm on Airworthiness Security Regulations †

Adrian Hechelmann

Aerospace Engineering, Campus Friedrichshafen, Cooperative State University Baden-Württemberg Ravensburg, Fallenbrunnen 2, 88045 Friedrichshafen, Germany; hechelmann.a@dhbw-ravensburg.de

† Presented at the 15th EASN International Conference, Madrid, Spain, 14–17 October 2025.

Abstract

In the context of digital system development, characterized by the utilization of digital methods and tools such as Model-based Systems Engineering (MBSE) and artificial intelligence (AI), the automated extraction of requirements from text-based documents becomes realizable. As a result, time-consuming tasks regarding the management of requirements can be optimized. This paper demonstrates the results of the application of a syntax-based text extraction algorithm on different airworthiness security regulations. Extracting non-functional requirements from document-based regulations, as the requirements in airworthiness security regulations, using AI models is complex. This is because there is a lack of appropriate training data sets to train AI models. In addition, the creation of a large, high-quality data set requires time-consuming preparatory work. Consequently, an algorithm was developed that extracts non-functional requirements from document-based regulations independently of the existence of appropriate training data sets. The algorithm compares individually defined templates that are composed of syntactical functions with text-based regulations, stores matches and generates syntax trees. The algorithm further includes an automated data-labeling functionality that enables the simplified creation of training data sets for the training of AI models. It was found that with only a few well-defined individual templates, a large number of requirements can be identified.

Keywords: text extraction; natural language processing; requirements engineering; airworthiness security regulations

1. Introduction

As part of the development of airborne systems using system engineering, the management of requirements plays a central role. New systems must be developed sensibly under consideration of certification specifications [1]. When regulatory authorities revise regulations or define new regulations, design organizations have to make adaptions in their development processes and adapt requirements regarding system design and certification. System engineering approaches aim to handle the complexity of aerospace systems. MBSE is the future of system engineering [2]. MBSE fosters different disciplines to collaborate targeted and design system models [3]. The benefit of using MBSE is to generate a system model that is easy to understand. The system model serves as a "single source of truth", considering all relevant development information in the model [4] and capture design decisions as model elements [5]. In this context, requirement engineering is particularly essential for model-based developments. It is common practice to define requirements in natural language [6]. This is applicable to the definition of requirements as well as to

Academic Editors: Spiros Pantelakis, Andreas Strohmayer and Gustavo Alonso

Published: 29 May 2026

requirements in regulations, which are in fact text documents. Consequently, the definition of requirements as model elements necessitates a recording process. It was found that although the automated recording of design, functional, and performance requirements in the aerospace domain has been established [7], this recording process is typically carried out manually regarding non-functional requirements. Further, the recording process necessitates the involvement of highly skilled engineering professionals, which is inherently costly in terms of both time and resources. The recording of non-functional requirements from document-based aerospace regulations, especially security-related airworthiness regulations, is complex: natural language processing (NLP) requires an appropriate training data set [8]. In addition, the creation of a large, high-quality data set requires time-consuming preparatory work. The research indicates that, regarding non-functional requirements in the aerospace domain, no appropriate data sets exist. Nevertheless, development companies may profit from solutions for automated requirement recording, in the form of time saving. Due to this, there is a demand for a solution that enables the recording of non-functional requirements in document-based aerospace regulations, independently of existing training data sets and AI models. In this paper, an algorithm that uses the principle of syntactical functions to identify and extract non-functional requirements is presented. To be able to train AI models in the future, an automated data-labeling function has been implemented in the syntax-based algorithm as a key feature. This enables the generation of training data sets for subsequent use.

The development of this algorithm was motivated by the research project i+sCabin2.0 [9]. As a project partner, the Cooperative State University Baden Württemberg Ravensburg is responsible for the research and development of an MBSE framework contributing to the airworthiness certification process of aircraft cabin systems. One objective is to capture and transfer the non-functional requirements from the airworthiness security regulations into the model-based framework. The algorithm presented here was designed to support this process.

2. Technical Foundation

This section presents the technical foundation of this study. Relevant work regarding NLP, applications of NLP for MBSE purposes, and fundamentals regarding syntax extraction are summarized below.

2.1. Natural Language Processing

NLP is a discipline consisting of computer science, artificial intelligence, and linguistics [10]. The application of NLP techniques assists systems engineers in creating effective specifications, which is achieved by eliminating ambiguity, identifying incompatible requirements, and evaluating the influence of requirements on the final design [2].

NLP techniques are commonly used by analysts and engineers for requirement-related tasks as support in extracting and classifying requirement-related knowledge [11]. Transformer models have become the de facto standard for handling a range of language processing tasks in industry and science [12]. Although frameworks like Bidirectional Encoder Representations from Transformers (BERT) are used to identify non-functional requirements [13], several different techniques for the identification of non-functional requirements have been established. Examples include the usage of supervised machine learning [14] or support vector machines [15]. In addition, researchers are working on data sets for requirement processing [16], and mapping studies regarding requirement engineering and machine learning have been carried out [17,18]. Most of these examined approaches do not address the aerospace domain, especially not airworthiness security regulations. Research in the aerospace domain regarding requirement engineering has been carried

out by Ray et al. [7]. While the authors used BERT to extract design requirements, functional requirements, and performance requirements [7], the extraction of non-functional requirements and airworthiness security requirements remains limited.

2.2. Natural Language Processing and Model-Based Systems Engineering

Natural language processing techniques can be employed to transform unstructured text-based requirements into a format that is suitable for model-based system engineering. Riesner et al. [3] presented an approach based on Named Entity Recognition (NER) to generate SysML requirement tables from requirements written in natural language and found that NLP is a valuable tool for MBSE to record information and reduce manual workload. Riesner et al. [3] used Python and spaCy toolbox for development, and they evaluated their findings through a case study of satellite requirements. The authors did not investigate the effectiveness of non-functional requirements. Beyond Riesner et al. [3], other researchers have addressed the disciplines of NLP and MBSE. The research conducted by Kulcsár et al. [19] aimed to investigate the natural language understanding of MBSE artifacts. The authors state that despite structured description of engineering data, their inherent semantics often remain hard to explore. Kulcsár et al. [19] proposed to use text generators to generate descriptive text, on which semantic search and analysis techniques can be applied.

2.3. Formal Tools

Requirements in a constrained natural language and formalization can be articulated as a temporal logic to facilitate analysis and verification using appropriate tools. A well-established tool is the Formal Requirements Elicitation Tool (FRET), which was published by NASA [20]. FRET is a framework designed to streamline the elicitation, formalization, and comprehension of requirements. Requirements written by following the rules of FRET are semantically unambiguous [20]. While these rules can be employed to define requirements, it cannot be assumed to be present in all cases: requirements in text-based regulations may not align with the rules of FRET.

2.4. Syntax and Syntax-Based Text Extraction

Syntax fundamentals and the state of research regarding syntax-based text extraction is summarized in the following. The process of analyzing a sentence syntactically consists of attributing a function and a form to the smaller units that constitute the sentence. For a successful syntactic analysis, it is relevant to know the difference between functions and forms. Functions are either syntactic functions like subject, verb object, etc., or phrase-internal functions like, for example, head, premodifier, postmodifier, etc. Forms are phrases (e.g., noun phrase or verb phrase) and clauses (e.g., that-clause) or word classes (e.g., noun, adjective, preposition. . .). Seven syntactic functions exist, which can be found on the first syntactic level when segmenting sentences. The forms must be alongside the functions during segmentation [21].

These syntactic rules are relevant for NLP and text extraction algorithms [22] and are already implemented in toolboxes such as the Matlab 2023b Text Analytics Toolbox and Industrial-strength NLP toolbox in Python (spaCy 3.0). Liu et al. [23] developed a method for acquiring cross-domain requirements for app development based on feature extraction and similarity matching. This proposed method is divided in two steps. Firstly, features from descriptions are extracted by adding semantic actions to syntax rules. Secondly the similarity of features in different domains is computed. Liu et al. [23] found that their method is scalable and accurate and enables developers to accumulate experience. As already mentioned, transformer models are used for NLP tasks. In addition, a lot of research has been conducted in the field of transformer models. Different groups are investigating

transformer models in terms of syntactical rules, aiming to increase their performance using syntax trees. Sachan et al. [24] explored the utility of incorporating syntax information from dependency trees into pre-trained transformers and applied it to the information extraction tasks of semantic role labeling (SRL), NER, and relation extraction (RE). Sachan et al. [24] found that syntax representations are most helpful for SRL and RE tasks when incorporated on top of pre-trained representations. Sachan et al. [24] further found that models do not provide any performance improvements on NER. The research of Bai et al. [25] aligns these findings by incorporating the inductive bias of syntax trees into pre-trained transformer models. Bai et al. [25] compared three different transformer models and found that the incorporation of syntactic knowledge improves the effectiveness of transformer-based models in natural language understanding.

3. Syntax-Based Text Extraction Algorithm

To identify and record non-functional requirements, this paper presents a syntax-based text extraction algorithm. The Matlab 2023b development tool and a text analytics toolbox were used. To extract requirements based on the syntax, several steps must be performed. These steps are visualized in Figure 1.

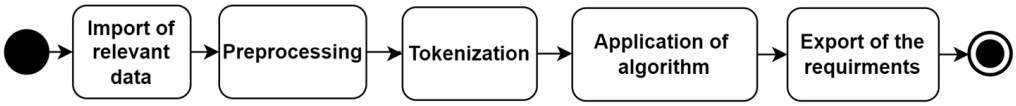

Figure 1. Flowchart demonstrating the five steps required to extract non-functional requirements with the proposed algorithm.

As the first step, the extraction requires the requirement document to be imported as a pdf file and a table containing syntactical information of non-functional requirements. In this step, the pdf is converted into a data string, and the table is converted into a cell array. In the remainder of this paper, the aforementioned table will be referred to as the "syntax template". The second step is preprocessing. In this step, data are extracted from the metadata of the pdf file, and then employed for the subsequent labeling of the extracted requirements, thereby enhancing traceability. After preprocessing, tokenization is carried out. Tokenization is essential to preparing the text for further processing with NLP. In general, tokenization involves breaking sentences into individual words [10]. This step is followed by the application of the algorithm to identify the requirements. Subsequently, all identified requirements are exported and further integrated into the project's system model. Figure 2 shows a flowchart that visualizes the major steps of the algorithm and an exemplary syntax template. The syntax template is shown in Figure 3.

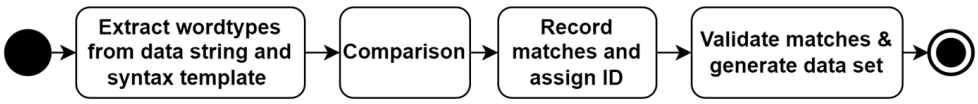

Figure 2. Flowchart presenting the functionality of the extraction algorithm.

The algorithm is explained as follows. The initial step is to ascertain each individual template of the syntax template. Figure 3 shows the schematic concept of a syntax template. In each column of the shown syntax template, an individual template is defined. To define an individual template, a syntactic function is assigned to each cell of the defined individual template. There is no limitation of the length of an individual template. The endpoints of each individual template are then indicated with the term "end". The syntax template may contain several individual templates.

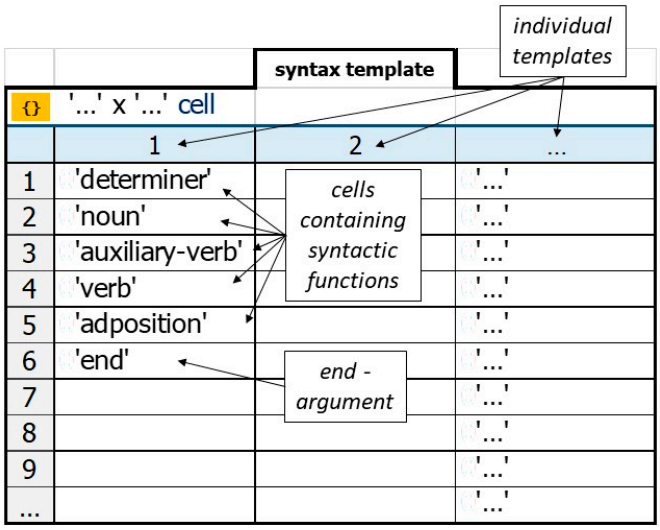

Figure 3. Schematic concept of the syntax template.

After the initial step, the parts of speech are read from the syntax template cell array and compared with the parts of speech of the data string. This process is repeated until the algorithm identifies a match between the individual template and data string. In the case that a match is identified, the algorithm records the match, the number of the individual template, and the location in the data string. In addition, the algorithm assigns an ID. The purpose of collecting this information is to enable the labeling of the data string. After every individual template of the syntax template cell array has been checked, the records of the matches are processed and validated. Next, the algorithm labels the data string automatically using the validated records. The labeled data string can be used as input to train AI models. By means of this automatic data set generation, the developed algorithm reduces time-consuming manual data labeling. Currently, records are validated manually. To support the validation step, syntax trees of the matches are generated by the algorithm. Figure 4 shows the generated syntax tree of a validated non-functional requirement.

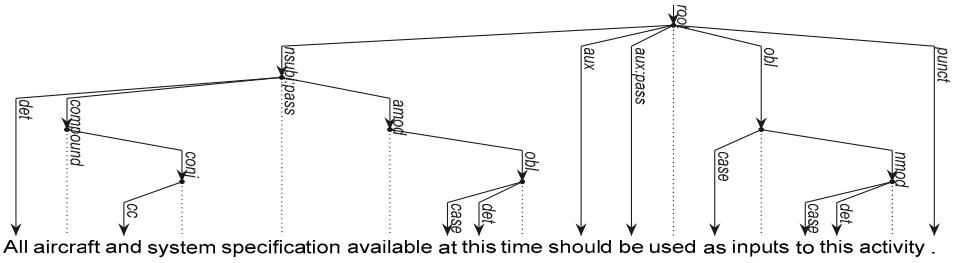

Figure 4. Syntax tree of an extracted and validated non-functional requirement.

It was found that the generation of syntax trees contributes to the process of manual validation. Each validated requirement is exported in a format that can be further integrated into a requirement or an MBSE tool. It is conceivable to automate the manual validation step using transformer models y relying on approaches described in [13] or [24].

4. Results

To apply the algorithm, a reference data set was derived from the airworthiness security specification ED-203A. The reference data set comprises 46 pages and contains a total of 23 non-functional requirements. The syntax template is designed to identify all requirements, and it consists of eight individual templates. Despite the templates being constructed by hand, the construction time required can be considered low. The algorithm

was applied to the reference data set after the syntax template was created. Subsequently, to evaluate the effectiveness of the individual templates in the syntax template set, the respective matches of the individual templates were identified. Figure 5 presents the results as a pareto diagram, revealing that 78.3% of the requirements of the reference data set could be identified with only three of the eight individual templates. The template that achieved the highest number of matches was able to identify eight requirements, which corresponds to a percentage of 34.8%.

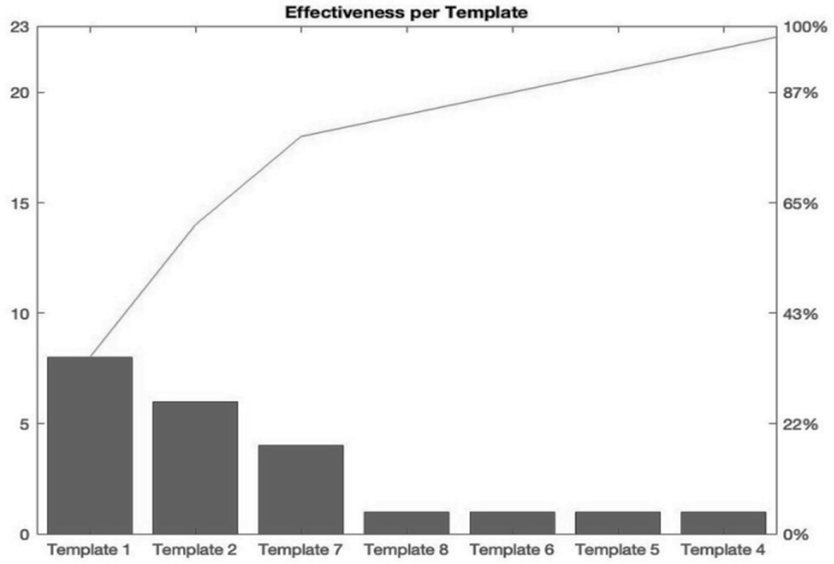

Figure 5. Pareto diagram demonstrating the matching rate of the individual templates in accordance with the reference data set.

At this point, it can be deduced that the extraction of non-functional requirements in airworthiness security regulation based on the syntax is possible in principle. This is mainly due to the wording used in the selected regulations. As already mentioned, the wording set out in text-based regulations does not necessarily align with the rules of a formal language as proposed in [20]. However, such formal rules would have a beneficial effect on the performance of the algorithm and contribute to the achievement of a high matching rate with well-defined templates. The achievement of a high matching rate based on well-defined templates is an essential advantage of the proposed algorithm. In addition to that, the algorithm offers the automated data-labeling functionality. This functionality reduces the time-consuming step of data labelling, which is needed when using AI, as proposed in [14]. As an additional test, the same syntax template (without any adaptions) was applied to the airworthiness security regulations. The results indicate that using the same syntax template for different documents results in different performances. Not all individual templates of the syntax template triggered a match. The syntax of requirements for different regulations varies, and the syntax template should be modified based on the application target. At this point, it can be argued that formal rules like those proposed in [20] would have a beneficial effect on the performance of the algorithm. The results are summarized in Table 1, showing a varying matching rate in relation to the application target. The performances of the best individual templates are between 57% and 83%. It can be concluded that high-quality single templates are essential for the targeted application of the proposed algorithm.

Table 1. Results of the application of the approach on different airworthiness security regulations using the same syntax template.

Regulation	Matches	Verified	Relation Verified/Matches (%)	Successful Templates (Nr)	Best Single Template	
					(Nr)	Performance (%)
ED 201-A	21	7	33%	1, 2, 4	2	57%
ED 202-A	17	6	35%	1, 2, 7, 8	1	83%
ED 204-A	10	5	50%	1, 2, 3	2	60%

In the underlying research project i+sCabin2.0, the algorithm was used specifically for the targeted identification of non-functional requirements in airworthiness security regulations. As a result, the integration of the non-functional requirements into the MBSE framework was achieved. Because of the flexibility in the definition of the syntax template, the proposed algorithm offers a versatile area of application that is not limited to the aerospace domain. However, it is difficult to predict the effectiveness of the proposed algorithm in different areas, since requirements in other regulations may have a fundamentally different syntax and structure and may not follow formal language rules.

5. Conclusions

This paper presents the functionality of a syntax-based text extraction algorithm and the application results of this algorithm on airworthiness security regulations. The introduced algorithm was tested in the context of extracting non-functional requirements for integration into the model-based framework of the i+sCabin2.0 research project. The fundamental characteristic of the proposed algorithm is the utilization of the syntax template, for which individual templates can be easily defined. The algorithm extracts requirements using only a few well-defined templates. The proposed algorithm exhibits broad applicability across multiple domains. The automated labeling of the input data is an additional benefit of the presented algorithm. It supports the definition of training data sets for AI models. Currently, the validation of the identified requirements of the algorithm is performed in a manual manner, supported by the automated generation of syntax trees. Future work will focus on optimizing the algorithm to automate the validation steps using transformer models. In addition, the training and testing of AI models is planned, where the extracted data are used as the training data set.

Funding: This work, as part of the i+sCabin2.0 research project, is supported by the German Federal Ministry for Economic Affairs and Climate Action (BMWK).

Institutional Review Board Statement: Not applicable.

Informed Consent Statement: Not applicable.

Data Availability Statement: The data presented in this article are not readily available because the data are part of an ongoing study. Requests to access the data sets should be directed to the author.

Acknowledgments: The author acknowledges the insights and collaborative efforts of the i+sCabin2.0 project partners. The author acknowledges Keanu Gehring's programming support in implementing the text extraction algorithm.

Conflicts of Interest: The author declares no conflicts of interest.

Abbreviations

AI	Artificial intelligence
BERT	Bidirectional Encoder Representations from Transformers

FRET	Formal Requirements Elicitation Tool
MBSE	Model-based Systems Engineering
NER	Named Entity Recognition
NLP	Natural Language Processing
RE	Relation Extraction
spaCy	Industrial-strength Natural Language Processing (Python Toolbox)
SRL	Semantic Role Labeling

References

1. Hinsch, M. *Industrial Aviation Management: A Primer in European Design, Production and Maintenance Organisations*; Springer: Berlin/Heidelberg, Germany, 2019. [CrossRef]
2. 'Systems Engineering Vision 2035'. International Council on Systems Engineering. 2021. Available online: https://www.incose.org/publications/se-vision-2035 (accessed on 10 September 2025).
3. MRiesener; Dölle, C.; Becker, A.; Gorbatcheva, S.; Rebentisch, E.; Schuh, G. Application of natural language processing for systematic requirement management in model-based systems engineering. *INCOSE Int. Symp.* **2021**, *31*, 806–815. [CrossRef]
4. Hechelmann, A.; Mannchen, T. Applicaton of SysML SYSML in the Development of Aircraft Cabin Health Management. In Proceedings of the 34th Congress of the International Council of the Aeronautical Sciences, Florence, Italy, 9–13 September 2024.
5. Delligatti, L. *SysML Distilled: A Brief Guide to the Systems Modeling Language*; Addison-Wesley: Upper Saddle River, NJ, USA, 2014.
6. Sandhu, G.; Atish; Pal, S.; Pal, P. Knowledge Extraction in Requirement Engineering with Machine Learning Perspective. *Int. J. Comput. Appl.* **2015**, *COGNITION2015*, 3.
7. Ray, A.T.; Cole, B.F.; Fischer, O.J.P.; White, R.T.; Mavris, D.N. aeroBERT-Classifier: Classification of Aerospace Requirements Using BERT. *Aerospace* **2023**, *10*, 279. [CrossRef]
8. Dang, V.M.H.; Verma, R.M. Data Quality in NLP: Metrics and a Comprehensive Taxonomy. In *Advances in Intelligent Data Analysis XXII*; Miliou, I., Piatkowski, N., Papapetrou, P., Eds.; Lecture Notes in Computer Science; Springer Nature: Cham, Switzerland, 2024; Volume 14641, pp. 217–229. [CrossRef]
9. Diehl Stiftung & Co. KG. i+s Cabin. Available online: https://microsite.diehl.com//i-s-cabin/en/ (accessed on 4 December 2025).
10. Vajjala, S.; Majumder, B.; Gupta, A.; Surana, H. *Practical Natural Language Processing: A Comprehensive Guide to Building Real-World NLP Systems*, 1st ed.; O'Reilly: Beijing, China; Boston, MA, USA; Farnham, UK; Sebastopol, CA, USA; Tokyo, Japan, 2020.
11. Abad, Z.S.H.; Gervasi, V.; Zowghi, D.; Far, B.H. Supporting Analysts by Dynamic Extraction and Classification of Requirements-Related Knowledge. In *Proceedings of the 2019 IEEE/ACM 41st International Conference on Software Engineering (ICSE), Montreal, QC, Canada, 25–31 May 2019*; IEEE: Piscataway, NJ, USA, 2019; pp. 442–453. [CrossRef]
12. Tunstall, L.; von Werra, L.; Wolf, T.; Géron, A. *Natural Language Processing with Transformers: Building Language Applications with Hugging Face*, Revised ed.; O'Reilly: Beijing, China; Boston, MA, USA; Farnham, UK; Sebastopol, CA, USA; Tokyo, Japan, 2022.
13. Le, F.; Wertheimer, D.; Calo, S.; Nahum, E. NorBERT: NetwOrk Representations Through BERT for Network Analysis & Management. In *Proceedings of the 2022 30th International Symposium on Modeling, Analysis, and Simulation of Computer and Telecommunication Systems (MASCOTS), Nice, France, 18–20 October 2022*; IEEE: Piscataway, NJ, USA, 2022; pp. 25–32. [CrossRef]
14. Kurtanovic, Z.; Maalej, W. Automatically Classifying Functional and Non-functional Requirements Using Supervised Machine Learning. In *Proceedings of the 2017 IEEE 25th International Requirements Engineering Conference (RE), Lisbon, Portugal, 4–8 September 2017*; IEEE: Piscataway, NJ, USA, 2017; pp. 490–495. [CrossRef]
15. Slankas, J.; Williams, L. Automated extraction of non-functional requirements in available documentation. In *Proceedings of the 2013 1st International Workshop on Natural Language Analysis in Software Engineering (NaturaLiSE), San Francisco, CA, USA, 25 May 2013*; IEEE: Piscataway, NJ, USA, 2013; pp. 9–16. [CrossRef]
16. Ferrari, A.; Spagnolo, G.O.; Gnesi, S. Towards a Dataset for Natural Language Requirements Processing. In Joint Proceedings of the REFSQ-2017 Workshops, Doctoral Symposium, Research Method Track, and Poster Track (REFSQ-JP 2017), Essen, Germany, 27 February 2017.
17. Zamani, K.; Zowghi, D.; Arora, C. Machine Learning in Requirements Engineering: A Mapping Study. In *Proceedings of the 2021 IEEE 29th International Requirements Engineering Conference Workshops (REW), Notre Dame, IN, USA, 20–24 September 2021*; IEEE: Piscataway, NJ, USA, 2021; pp. 116–125. [CrossRef]
18. Sonbol, R.; Rebdawi, G.; Ghneim, N. The Use of NLP-Based Text Representation Techniques to Support Requirement Engineering Tasks: A Systematic Mapping Review. *IEEE Access* **2022**, *10*, 62811–62830. [CrossRef]
19. Kulcsár, G.; Constant, O.; Pruvost, G.; Ráth, I.; Füzesi, M.; Harmath, D. Natural Language Understanding of Systems Engineering Artifacts. *INCOSE Int. Symp.* **2022**, *32*, 1373–1387. [CrossRef]

20. Giannakopoulou, D.; Pressburger, T.; Mavridou, A.; Rhein, J.; Schumann, J.; Shi, N. Formal Requirements Elicitation with FRET. In Joint Proceedings of the REFSQ-2020 Workshops, Doctoral Symposium, Live Studies Track, and Poster Track (REFSQ-JP 2020), Pisa, Italy, 24 March 2020.
21. Arendholz, J. (Ed.) *English Syntax: Basic Facts and In-Depth Analyses*; UTB GmbH: Stuttgart, Germany, 2022. [CrossRef]
22. Seretan, V. Syntax-Based Collocation Extraction. In *Text, Speech and Language Technology, No. 44*; Springer: Dordrecht, The Netherlands, 2011. [CrossRef]
23. Liu, H.; Zhang, M.; Liu, L.; Liu, Z. A method to acquire cross-domain requirements based on Syntax Direct Technique. *Softw. Pract. Exp.* **2022**, *52*, 236–253. [CrossRef]
24. Sachan, D.; Zhang, Y.; Qi, P.; Hamilton, W.L. Do Syntax Trees Help Pre-trained Transformers Extract Information? In *Proceedings of the 16th Conference of the European Chapter of the Association for Computational Linguistics: Main Volume*; Online; Association for Computational Linguistics: Stroudsburg, PA, USA, 2021; pp. 2647–2661. [CrossRef]
25. Bai, J.; Wang, Y.; Chen, Y.; Yang, Y.; Bai, J.; Yu, J.; Tong, Y. Syntax-BERT: Improving Pre-trained Transformers with Syntax Trees. *arXiv* **2021**. [CrossRef]