AUTOMATED CONDITION DETECTION IN REQUIREMENTS ENGINEERING

Gärtner, Alexander Elenga (1,2);
Göhlich, Dietmar (1);
Fay, Tu-Anh (1)

1: TU-Berlin;
2: IAV GmbH

ABSTRACT

In product development, it is of great importance that a complete, unambiguous, and, as far as possible, contradiction-free target system is defined. Requirements documents of complex systems can contain several thousand individual requirements, derived in an interdisciplinary manner and written in natural language by many different stakeholders. Hence, errors, in the form of contradictions, cannot be completely avoided in these documents and today they must be corrected manually with high effort.

This paper presents an important building block for automated contradiction detection and quality analysis of requirements documents. We discuss the necessary identification of conditions in requirements and the extraction of the verbal expressions associated with condition and effect, respectively. We applied and analyzed natural language processing methods based on grammatical versus machine learning models. The models have been applied to 1,861 real-world requirements. Both approaches generate promising results, with an accuracy partly over 98%. However, in structured specification texts, a grammatical model is preferable due to lower effort in preprocessing and better usability.

Keywords: Requirements, Machine learning, Semantic data processing, Systems Engineering (SE), Evaluation

Contact:
Gärtner, Alexander Elenga
IAV GmbH
Germany
alexander.elenga.gaertner@iav.de

Cite this article: Gärtner, A. E., Göhlich, D., Fay, T.-A. (2023) ‘Automated Condition Detection in Requirements Engineering’, in *Proceedings of the International Conference on Engineering Design (ICED23),* Bordeaux, France, 24-28 July 2023. DOI:10.1017/pds.2023.71

1 INTRODUCTION

Requirements play a central role in the development process of virtually all products (Loucopoulos 2005; Gericke and Blessing 2012) because they are usually a central basis of communication when solutions are developed by several involved persons, areas or even companies in parallel or successively (VDI 2221 Blatt 1:2019-11). It is a widespread practice that requirements documents are formulated in a specification sheet, which should contain the totality of the requirements within a development project (DIN 69901-5:2009-01). In this context, it is of great importance that a complete, unambiguous and, as far as possible, contradiction-free target system is defined (Bender and Gericke 2021). However, specification sheets can contain several thousand individual requirements written in natural language and are typically derived in an interdisciplinary manner. Additionally, the variety of mechatronic products and the complexity of modern systems make distributed and concurrent development at different aggregation levels of the product development process indispensable. Typically, the requirements for each level are managed in different documents, for the overall product in a product requirements document and for the subsystems in component requirements documents (Göhlich et al. 2021). Therefore, it is not surprising that errors, e.g., in the form of contradictions, are often found in these documents.

To the best of our knowledge, currently, neither a comprehensive automated contradiction detection nor an automated quality analysis of industrial specification documents is available. Our work aims at providing the necessary building blocks for an automated quality assurance of specification documents. In a previous study (Gärtner et al. 2022), we presented different types of contradictions that can occur in requirements specifications and developed a method to classify contradictions. In this context, the question of whether the requirement includes a condition or not plays a central role. In fact, they give the decisive hint as to whether statements contradict each other. For example, the two statements *x must be 4* and *x must be 5* are contradictory unless the conditions state otherwise, e.g. *if y=3* and *if y=4*. Therefore, the detection of conditions and extraction of their constituents is crucial for reaching the goal of an automated contradiction detection. In this paper, we want to show how conditions can be detected automatically and which verbal expressions form the condition. In detail, this paper aims to:

- Identify all requirements in a specification that contain conditions.
- Identify the verbal expressions that form the condition.
- Evaluate which is the best-suited approach in this context: self-written rules or machine learning.
- Create a basis to identify contradictions automatically. According to our previous study, the conditions and effects must be compared regarding their verbal expressions to find contradictions (Gärtner et al. 2022).

For this purpose, we compare two natural language processing (NLP) techniques: the first one is a model that consists mainly of grammatical rules that are directly embedded into the source code. The second one uses a bag-of-words model for machine learning (ML). We, therefore, elaborate on the terminology, define rules on how to detect conditionals and the related verbal expressions, and finally present the results of our algorithms on 1,861 "real world" requirements.

Our main dataset is in German, therefore the terminology, as well as the method, are tailored to German grammar. As Mark Twain in *The Awful German Language* (1880) notes, the German language is much more complicated than English. In this respect, we found that rules which work well for requirements written in German can be transferred easily into rules for requirements in written English, but not vice versa. We demonstrate this by translating and analyzing a public English dataset gathered by Fischbach et al. (2021a) into German.

2 TERMINOLOGY

Conditions are sub-statements of sentences that are *"essential to the appearance or occurrence of something else"* (Merriam-Webster 2022). In other words, the condition must be true for the effect to happen. In German, the condition can be formulated either in a subordinate clause (SC) or in an adverbial (ADVB) within the main clause (Duden 2006; Eisenberg 2016).

SC-clauses are, in German, usually marked by conjunctions such as *wenn, falls, sofern* (eng: *if, when, in case*) (1), or by a special word position, the so-called verb-first position (2). The second case is non-

https://doi.org/10.1017/pds.2023.71 Published online by Cambridge University Press

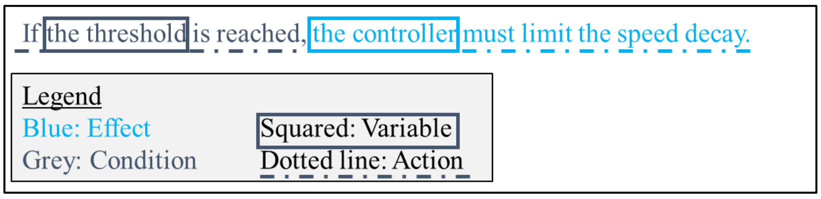
existent in English and is always translated by using one of the mentioned conjunctions. In the examples below, the conditions are noted in italic:

1. *Wenn drei Sekunden vergangen sind*, dann muss das Licht ausgehen. Eng.: *When three seconds have passed*, the light must go out.
2. *Sollten drei Sekunden vergangen sein*, dann muss das Licht ausgehen. Eng.: *When three seconds have passed*, the light must go out.

Adverbial conditions occur within the main clause without the need for a subordinate clause. It is important to note, that adverbs are defined differently in English. In German, they express the closer circumstances of an action, a process, or a condition (Duden 2006). They are usually marked by *bei, nach, während, im Falle*, and more (Eng.: *at, after, while, in case of*, and more). For example: *In case of a message timeout, a message should be sent to the error manager*. Although there is a comma after *timeout*, the first part of the sentence is not a subclause.

It is important to note, that conditional statements are not statements of causality. The *strike of a match* is a cause of the match lighting. The *presence of oxygen* is a condition for the match lighting (Broadbent 2008). Confusion arises often, as both conditional and causal statements can be introduced by *If …, then*: *If I strike a match, then the match lights* (causal statement) and *If oxygen is present, then the match can light* (conditional statement). In requirements engineering (RE), causal statements would be classified as information rather than as requirements, as they describe why something happens and not how.

Another research topic of this paper is to identify the verbal expressions that belong to the condition as well as the verbal expressions that belong to the subject and the verb of the clause. These terms are also referred to as constituents (words or groups of words that function as a unit). Unlike Fischbach's terminology (2021a) which we adopted in our previous paper, (Gärtner et al. 2022), we now propose the following terms: variable and action. For example, in the sentence *If the threshold is reached, the controller must limit the speed decay*, we must first differentiate between the condition (*If the threshold is reached*) and the effect (*the controller must limit the speed decay*), as shown in Figure 1. For the condition, *the threshold* is the variable, and *if is reached* is the action. For the effect, *the controller* is the variable, and *must limit the speed decay* is the action. In other words: the variable is the protagonist and the action that what shall happen.



If the threshold is reached, the controller must limit the speed decay.
Legend
Blue: Effect
Grey: Condition
Squared: Variable
Dotted line: Action

Figure 1: variable and action

3 RELATED WORK

In the literature we found several approaches to automatically identify conditions and effects. While rule-based methods were predominant in the past, machine-learning-based methods have emerged with increasing computing power (Asghar 2016).

The rule-based approach from Khoo et al. (1998) extracts conditionals using linguistic clues and pattern-matching, reaching an accuracy of 68 %. It was designed based on prose text from the wall street journal, and not specifically designed for RE, which has more linguistic restrictions than a newspaper article. This approach, as well as an approach from Liu et al. (2021), searches for specific trigger words. Although this is a good approach, especially for English, in the analysis of our dataset for this paper we found that 17 % of the conditions were not based on trigger words but on special grammatical forms, as explained in section 2. Águeda and Olivas (2008) present an approach to automatically extract conditionals for search engine optimization. However, the constraints are too narrow, as *"the cause must precede the effect"* (Águeda and Olivas 2008). In RE, but also in general, the effect can precede the cause, e.g., *the engine must shut down, if the engine speed is too high.*

In multiple papers, Frattini and Fischbach present different machine-learning-based approaches for the automatic detection of causalities (Frattini et al. 2022; Fischbach et al. 2021a; Fischbach et al. 2020) and the automatic detection of their constituents (Fischbach et al. 2021b). Although their theory is

https://doi.org/10.1017/pds.2023.71 Published online by Cambridge University Press

explicitly designed for RE, they do not focus on conditionals, but on causalities. However, they seemed to have looked at both conditionals and causalities, which could be due to an incorrect definition of the terms, as explained in section *2 Terminology*. In addition, they have classified sentences as causal, which are not causal nor conditional, e.g.: *The fire is burning down the house*. Although the fire is indeed the cause of the house burning down and there is an implicit causal relationship, the sentence is grammatically not causal. Furthermore, they define causality as a causal relation that requires the effect to occur if - and only if - the cause has occurred. However, this definition is not useful in the RE context, since effects can be triggered by multiple conditions: *If the speed exceeds the threshold, the motor is to be switched off in an emergency* and *if the torque exceeds the threshold, the motor is to be switched off in an emergency*. They investigated 14,983 sentences to track the extent and form in which causality occurs. Their approach achieved an accuracy of 82%.

4 STUDY

In this paper, we compare the performance of two approaches: a rule-based grammatical model versus two ML-based models. The former applies the rules mentioned in section 2 *Terminology*. The latter is implemented using a bag-of-words model for the document and two popular classifying theorems for NLP tasks: Naïve Bayes (NB) and k-Nearest Neighbor (kNN) (Bramer 2013). The bag-of-words algorithm is used to represent the document, "*depicting [it] as a bag and each vocabulary in the texture as the items in the bag*" (Ersoy 2021). The NB classifier uses probability to find the most likely of the possible classifications. kNN estimates the classification using the classification of its neighbors, with k representing the number of the closest instances to consider (Bramer 2013). For this study, it was deemed appropriate to limit the approach to utilizing these two simple and lightweight methods due to their demonstrated efficacy in section *5 - Results*. Hence, the advanced features of more complex methods, such as GloVe and BERT, including improved contextual comprehension and semantic representation, were not considered for the current task and were not incorporated into the analysis.

4.1 Method

All 1,861 requirements of our datasets, see section *4.2 Data*, were manually divided into conditionals and non-conditionals according to section *2*. This classification was verified independently by 3 persons, all having extensive experience in requirements engineering. The classification is necessary to train the ML models and to determine the accuracies of both the ML and the grammatical models.
Our method as well as the software which was developed in form of a Python code are structured in four parts as shown in Figure 2:

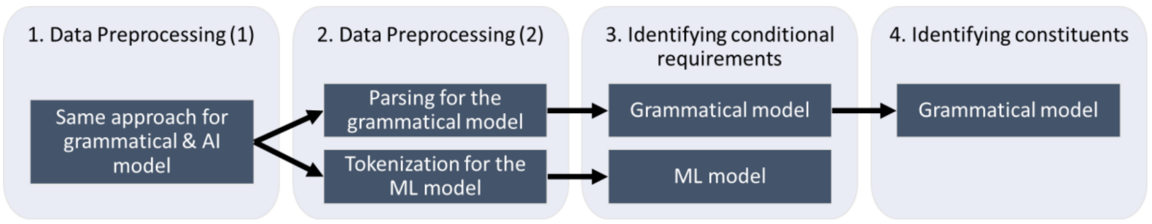

1. Data Preprocessing (1)
Same approach for grammatical & AI model
2. Data Preprocessing (2)
Parsing for the grammatical model
Tokenization for the ML model
3. Identifying conditional requirements
Grammatical model
ML model
4. Identifying constituents
Grammatical model

Figure 2: code structure

1. Data Preprocessing (1): *"Data directly taken from the source will likely have inconsistencies, errors or most importantly, it is not ready to be considered for a data mining process."* (García et al. 2014). Therefore, data reduction techniques must be applied to remove irrelevant and noisy elements from the dataset, for example by replacing "<" with "is smaller than".
2. Data Preprocessing (2): *"Parsing is the task of analyzing grammatical structures of an input sentence and deriving its parse tree*." (Bojić and Bojović 2017). This task is sometimes considered to be part of the preprocessing (Gudivada 2018). For the grammatical model, we used parse trees, also called dependency trees, and Part-of-Speech (POS) tagging. A parse tree is a graphical representation of the dependencies between words. POS tagging categorizes words in correspondence with a particular part of speech, e.g., nouns, verbs, adverbs, conjunctions, etc. The parsing was outsourced to the dependency parser ParZu, by Sennrich et al. (2013). For the ML models tokenization was done via countvectorizer, a method to convert

https://doi.org/10.1017/pds.2023.71 Published online by Cambridge University Press

text to numerical data, making a separate tokenizer redundant. While it does not employ lemmatization techniques, this is not critical for the specific task at hand, as semantic considerations do not play a crucial role in this classification problem.

3. Identifying conditional requirements: The grammatical model uses grammatical rules, trigger words, and POS tags to detect conditions, as explained in section *2*. The ML models are trained using the labeled German dataset. An 80/20 split was chosen, i.e., 1,249 requirements for training and 313 requirements for testing. For kNN, hyperparameter tuning was done via gridsearch. This is a technique to determine the optimal values for a given model. The best results were achieved with leaf_size = 1, metric = minkowski, n_neighbors = 1, p = 2 and weights = uniform. The results are discussed in sections *5.1* and *5.2*.
4. Identifying constituents: The final task is to identify the constituents, as explained in section *2*. From the POS tags and the parse tree, the verbal expressions associated with the condition and the effect can be determined, as well as the variables and actions. We did not use ML for this task, as the dataset is too large to label all variables and actions manually. The results for the grammatical model are discussed in section *5.3*.

4.2 Data

We used two different datasets. The first dataset originates from a recent project for electric buses. In this context a complete requirements package was derived, that describes a modular system, which shall replace conventional bus powertrains with new, electric powertrains. The role of requirements in the development process of electric buses can be found in *Design of urban electric bus systems* (Göhlich et al. 2018). We based our study on 1,561 functional and non-functional requirements. The specifications are written in German and the examples discussed in this paper were translated into English.

The second dataset is public, originating from Fischbach (Fischbach et al. 2020). It is an accumulation of approximately 15,000 requirements written in English from many different sources available online. From this dataset we used 300 randomly selected requirements, available online at Swarm-Engineer (2023). For our analysis, we translated them into German via DeepL. We used this dataset to show the feasibility to get correct results with non-automotive requirements and an originally English dataset.

5 RESULTS

The models were validated with two different datasets. In sections *5.1* and *5.2* the results of the questions on how to automatically detect conditional requirements are shown and discussed. In section *5.3* the result for identifying the verbal expressions assigned to the condition/effect and to the action and variable are shown.

5.1 Dataset 1

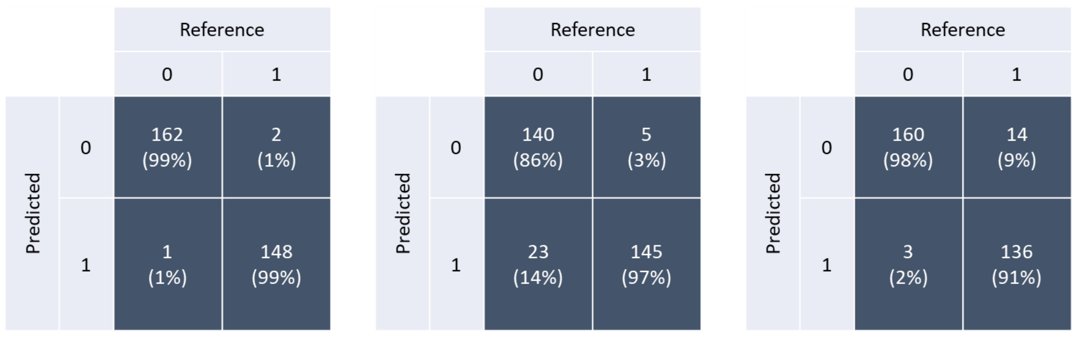
Out of the 1,561 analyzed requirements, 785 (50,3%) were labeled as conditional by us, according to section 2 *Terminology*. The labeling was verified by three persons to increase confidence in the correctness of the labels. For the ML algorithms, an 80/20 split was chosen, i.e., 1,249 requirements for training and 313 requirements as a test set.

The overall accuracies are:

- grammatical model: 99%
- NB model: 91%
- kNN model: 94%

To measure the effectiveness of the models, we used confusion matrices, as shown in Figure 3. They are used for performance measurements for machine learning classification problems, where the output can be two or more classes (Narkhedem 2018). In these tables, the different combinations of predicted and reference values can be examined. The combinations are called true negative (reference: 0; predicted 0), false negative (reference: 1; predicted: 0), false positive (reference: 0; predicted: 1), and true positive (reference: 1; predicted: 1).

https://doi.org/10.1017/pds.2023.71 Published online by Cambridge University Press



(1) Grammatical Model

		Reference 0	Reference 1
Predicted	0	162 (99%)	2 (1%)
	1	1 (1%)	148 (99%)

(2) Naive Bayes

		Reference 0	Reference 1
Predicted	0	140 (86%)	5 (3%)
	1	23 (14%)	145 (97%)

(3) Nearest Neighbor

		Reference 0	Reference 1
Predicted	0	160 (98%)	14 (9%)
	1	3 (2%)	136 (91%)

Figure 3: confusion matrices for dataset 1 based on 313 requirements

The grammatical model (Figure 3(1)) only has a few false predictions compared to the other models. It misclassified only 3 requirements, consisting of 1 false positive and 2 false negatives. The Naïve Bayes model (Figure 3(2)) had the biggest difficulties with false positives, as 23 non-conditional requirements were misclassified as conditionals. Whereas the k-Nearest Neighbor model (Figure 3(3)) had the biggest difficulties with false negatives, as 14 conditional requirements were misclassified as non-conditionals.

Although we have results for the grammatical model classifying all 1,561 requirements, for the ML models we only have results on the test set consisting of 313 requirements. Comparing the results between 1,561 requirements and 313 would not be fair, which is why we determined the accuracy of the grammatical model also using the 313 requirements. The accuracies are calculated as the number of all correct predictions divided by the total number of the dataset:

$$accuracy = \frac{TP+TN}{P+N} \qquad (1)$$

5.2 Dataset 2

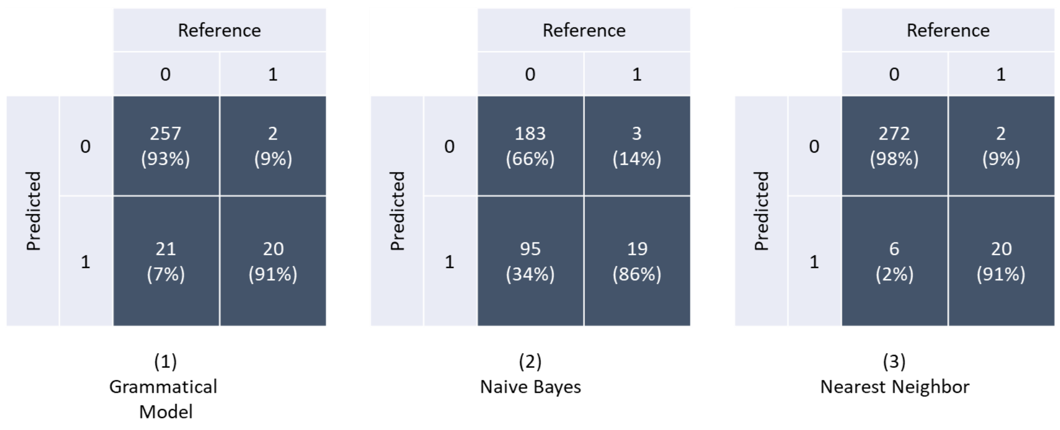
To show that the algorithm is not overfitted to the main German dataset, it was tested against a public English dataset. 300 out of approximately 15,000 sentences were randomly selected and then labeled so that they could now be used as a second test set, as explained before. The same models trained on the previous dataset were used and no new ML training was applied since the behavior of the models was to be tested with unknown data from different fields than automotive. The overall accuracies are:

- Grammatical model: 92%,
- NB model: 67%
- kNN model: 91%.

In contrast to the ML models, the reasons for the poorer accuracy of the grammatical model (99% with dataset 1 compared to 92% with dataset 2) can be clearly identified: some requirements were formulated in an inconsistent and complex way. Particularly adverbial trigger words were used much more liberally compared to the first dataset, causing the model to incorrectly identify ADVB-conditions.

The confusion matrices for dataset 2 are shown in Figure 4. As expected, the accuracies of all models are lower than for the dataset 1. On the one hand, this is because we built our theory using the dataset 1. On the other hand, the requirements in the second mostly don't follow standard RE documentation guidelines, for example described in Pohl and Rupp (2021). This complicates the analysis, as patterns do not exist or cannot be identified. Nevertheless, the grammatical model and the kNN model were able to achieve solid results. Although the accuracy for the grammatical model dropped from 99% to 92% and the accuracy from the kNN model dropped from 94,1% to 91%, the results are still good. Thus, we were able to show how the models can handle English datasets from other disciplines, although many requirements were formulated differently than in our first, automotive dataset. The NB model, however, misclassified many requirements, especially when analyzing non-conditional requirements. Here, only 66% were correctly classified, while 34% were misclassified. This suggests that the algorithm was overfitted to the original dataset.

https://doi.org/10.1017/pds.2023.71 Published online by Cambridge University Press



(1) Grammatical Model

		Reference 0	Reference 1
Predicted	0	257 (93%)	2 (9%)
Predicted	1	21 (7%)	20 (91%)

(2) Naive Bayes

		Reference 0	Reference 1
Predicted	0	183 (66%)	3 (14%)
Predicted	1	95 (34%)	19 (86%)

(3) Nearest Neighbor

		Reference 0	Reference 1
Predicted	0	272 (98%)	2 (9%)
Predicted	1	6 (2%)	20 (91%)

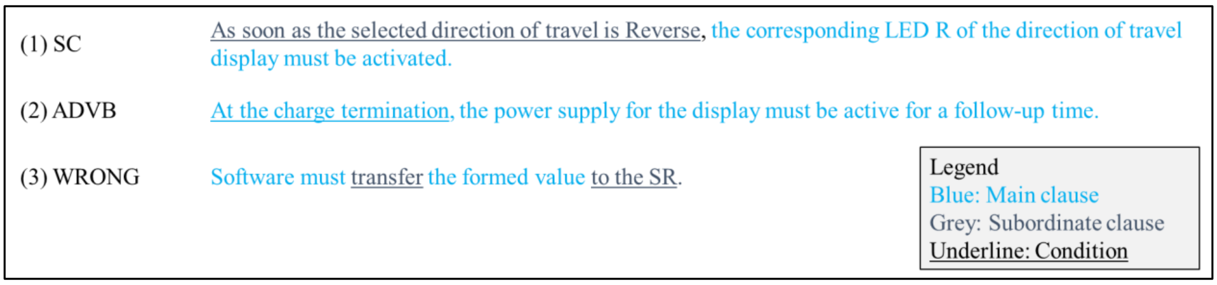
Figure 4: confusion matrices for dataset 2 based on 300 requirements

This leads to the following insight: some ML approaches, e.g. Naïve Bayes, are not well suited for our kind of problem, as the data that one wants to use in practice must correspond to the shape of the data of the training set. Especially important in this respect is the distribution of the labels True and False. In the German training dataset, there was a distribution of about 1 True to 1 False. Naturally, the German test set had the same distribution, which is why the accuracy is high, see Figure 3(2). The English test set though had a distribution of approximately 1 True to 10 False. The algorithm, however, expects a distribution of 1/1 and includes this in its classification. This can be seen in the result of Figure 4(2). More conditionals were recognized than there actually are: namely 114 (95 + 19) True predictions instead of 22 (3 + 19) True references.
The kNN approach looks at the classification of its nearest neighbor, which is why this ML algorithm is not as biased toward the original label distribution. The grammatical model also ignores the original label distribution, as it considers a fixed rule set.

5.3 Detection of constituents

Another research point of this paper is to identify the verbal expressions that are linked to the condition and the effect, as well as the verbal expressions that are linked to the variable and action, as explained in section *2*. In this case, we did not use ML as a comparison. First, it is very time-consuming to label a training set accordingly. Second, and much more important, if the basis (condition-recognition) is correct, see Figure 2 steps 1-3, we can accurately match all verbal expressions to the condition, using a fixed rule set. This problem is not complex enough to expect a better solution with ML. In the previous problem of condition recognition, for example, we could not be sure of the grammatical model being better than ML, because there are many variables involved and ML could have found a better way - which still it did not.
The results for three exemplary sentences of the first dataset can be seen in Figure 5: correct verbal expression assignments of an SC-condition and an ADVB-condition as well as an unsuccessful assignment. Figure 5(1) shows, the correctly determined SC and the resulting assignment of the verbal expressions that correspond to the condition. Figure 5(2) shows, that while no SC was detected, the ADVB was correctly determined and the verbal expressions that correspond to the condition were correctly assigned. Figure 5(3) shows a requirement that could not be processed by our algorithm because the parser had difficulty interpreting the input: For unknown reasons, it detected a main clause with the verb *must* and a subordinate clause with the alleged verb *transfer*. Therefore, the output resulted in a completely wrong assignment. The correct interpretation would be that *must transfer* is a compound main clause verb and thus no subordinate clause prevails as a condition.

https://doi.org/10.1017/pds.2023.71 Published online by Cambridge University Press



(1) SC As soon as the selected direction of travel is Reverse, the corresponding LED R of the direction of travel display must be activated.

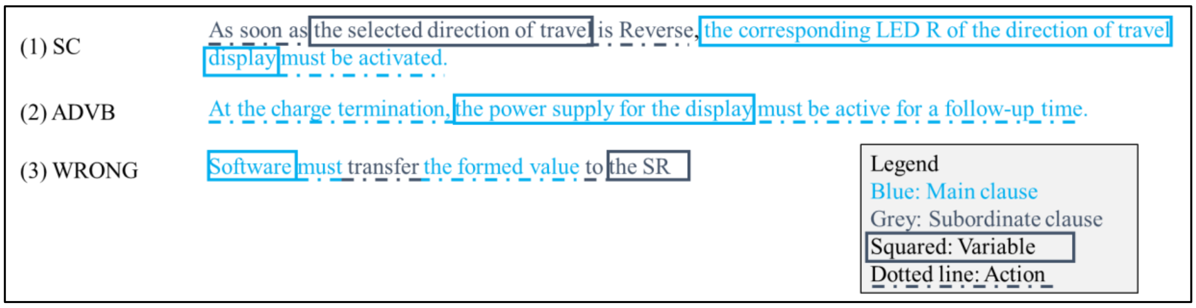
(2) ADVB At the charge termination, the power supply for the display must be active for a follow-up time.

(3) WRONG Software must transfer the formed value to the SR.

Legend
Blue: Main clause
Grey: Subordinate clause
Underline: Condition

Figure 5: condition/effect verbal expression detection

Figure 6 shows the results for variable and action detection. The first two examples show correct verbal expressions assignments, while the third one shows a failed assignment. The correctness of the assignments is directly linked to the correctness of the assignments of the verbal expressions for condition and effect, see Figure 5 - and also fails for the same reasons.



(1) SC As soon as the selected direction of travel is Reverse, the corresponding LED R of the direction of travel display must be activated.

(2) ADVB At the charge termination, the power supply for the display must be active for a follow-up time.

(3) WRONG Software must transfer the formed value to the SR

Legend
Blue: Main clause
Grey: Subordinate clause
Squared: Variable
Dotted line: Action

Figure 6: action/variable verbal expression detection

5.4 Discussion of validity

Descriptive validity (Maxwell 1992) deals with the risk of not having remained objective when conducting a study. Although we defined conditions based on fixed criteria, there may have been inconsistencies when labeling the dataset. We have tried to reduce this risk by having three persons review the labeling of the dataset. In addition, the grammatical model has an advantage here, because a different understanding of a condition can immediately be implemented, whereas for ML everything would have to be re-labeled and re-trained. Generalizability (Maxwell 1992) is given when the results can be applied to other situations that are outside the present research, which is also a threat to the study's validity. We addressed this, by testing and validating our method with a non-automotive dataset. That is why we believe that other industries write requirements in a similar way to the automotive industry. This is also supported by Göhlich et al. (2021) who found that processes to manage requirements and specifications do not differ significantly with regard to the industrial context. However, further testing should be conducted in the future to verify its applicability in different industries.

6 SUMMARY, CONCLUSION AND OUTLOOK

In this paper, we have provided a building block for how to make requirements engineering (RE) and requirements management intelligent using automated methods. In specific, we made a proposal on how to automatically detect conditionals and how they occur in the RE. In section *2*, we elaborated on the terminology and on how to detect conditional sentences. Furthermore, we laid the foundation to identify the verbal expressions respectively associated with the variables and actions. In section *4*, we presented the results on 1,861 requirements while comparing a grammatical model with two machine learning (ML) models.

We found that in structured texts, such as usually found in specifications, grammatical models are well suited for identifying conditionals and their constituents. Grammatical models show better or at least similar results than ML approaches. Some ML algorithms e.g. Naïve Bayes, are not well suited for our kind of problem. For example, for this algorithm, the data that one wants to use in practice must correspond to the shape of the data of the training set. Furthermore, every dataset used for ML methods must first be labeled, which can be very time-consuming and prone to human errors.

https://doi.org/10.1017/pds.2023.71 Published online by Cambridge University Press

Therefore, we conclude that the grammatical model is preferable as the rules can be tracked and easily adjusted if needed, for example, by changing trigger words. It is important to note, that this is not the case for every natural language processing problem. These findings do not apply to unstructured text, such as in newspapers or books. In RE, however, there seem to be certain explicit or implicit rules - depending on the industry - according to which sentences are formulated, which massively reduces the complexity of the problem and makes machine learning redundant.
Complete and error-free requirements specifications are crucial for a good product design. Some specifications contain several thousand individual requirements. Therefore, it is obvious, that automatization would result in an enormous leap in the manageability of such large datasets. Moving in this direction, this paper creates the possibility to identify contradictions between two requirements automatically. The approach is, that if two requirements had the same condition and in the effect the same variables, but different actions, this would indicate a contradiction (Gärtner et al. 2022). This research contributes a building block for this approach by identifying these corresponding verbal expressions. Such a method could, for example, help developers to identify critical requirements already during the design process or even in later stages like the review phase. This will be elaborated further in a future paper.

7 AUTHOR CONTRIBUTIONS

Conceptualization, A.E.G., D.G. and T.-A.F.; methodology, A.E.G.; validation, A.E.G.; formal analysis, A.E.G.; investigation, A.E.G.; resources, A.E.G.; data curation, A.E.G., D.G. and T.-A-F.; writing—original draft preparation, A.E.G.; writing—review and editing, D.G. and T.-A.F.; supervision, D.G. and T.-A.F. All authors have read and agreed to the published version of the manuscript.

ACKNOWLEDGMENTS

We thank IAV GmbH for providing us with the requirements specification on electric buses.

REFERENCES

Águeda, Cristina Puente/Olivas, José A. (2008). Analysis, detection and classification of certain conditional sentences in text documents. ReseaerchGate.
Asghar, Nabiha (2016). Automatic Extraction of Causal Relations from Natural Language Texts: A Comprehensive Survey. https://doi.org/10.48550/arXiv.1605.07895.
Bender, Beate/Gericke, Kilian (2021). Entwickeln der Anforderungsbasis: Requirements Engineering. In: Beate Bender/Kilian Gericke (Eds.). Pahl/Beitz Konstruktionslehre. Berlin, Heidelberg, Springer Berlin Heidelberg, 169–209.
Bojić, D./Bojović, M. (2017). A Streaming Dataflow Implementation of Parallel Cocke–Younger–Kasami Parser. In: Creativity in Computing and DataFlow SuperComputing. Elsevier, 159–199.
Bramer, Max (2013). Introduction to Classification: Naïve Bayes and Nearest Neighbour. In: Max Bramer (Ed.). Principles of Data Mining. London, Springer London, 21–37.
Broadbent, Alex (2008). The Difference Between Cause and Condition. Proceedings of the Aristotelian Society (Hardback) 108 (1pt3), 355–364. https://doi.org/10.1111/j.1467-9264.2008.00250.x.
DIN 69901-5:2009-01. Project management - Project management systems - Part 5: Concepts, 2009. Berlin.
Duden (Ed.) (2006). Die Grammatik. Das Standardwerk zur deutschen Sprache. 7th ed. Mannheim, Dudenverl.
Eisenberg, Peter (2016). Duden - die Grammatik. Unentbehrlich für richtiges Deutsch. Hg. von Angelika Wöllstein. 9th ed. Berlin, Dudenverlag.
Ersoy, Pinar (2021). Naive Bayes Classifiers for Text Classification. Types of Naive Bayes Classifiers for Different Text Processing Cases. Available online at https://towardsdatascience.com/naive-bayes-classifiers-for-text-classification-be0d133d35ba (accessed 10/26/2022).
Fischbach, Jannik/Frattini, Julian/Spaans, Arjen/Kummeth, Maximilian/Vogelsang, Andreas/Mendez, Daniel/Unterkalmsteiner, Michael (2021a). Automatic Detection of Causality in Requirement Artifacts: the CiRA Approach. Available online at http://arxiv.org/pdf/2101.10766v1.
Fischbach, Jannik/Hauptmann, Benedikt/Konwitschny, Lukas/Spies, Dominik/Vogelsang, Andreas (2020). Towards Causality Extraction from Requirements. https://doi.org/10.48550/arXiv.2006.15871.
Fischbach, Jannik/Springer, Tobias/Frattini, Julian/Femmer, Henning/Vogelsang, Andreas/Mendez, Daniel (2021b). Fine-Grained Causality Extraction From Natural Language Requirements Using Recursive Neural Tensor Networks. Available online at http://arxiv.org/pdf/2107.09980v2.

Frattini, Julian/Fischbach, Jannik/Mendez, Daniel/Unterkalmsteiner, Michael/Vogelsang, Andreas/Wnuk, Krzysztof (2022). Causality in requirements artifacts: prevalence, detection, and impact. Requirements Engineering. https://doi.org/10.1007/s00766-022-00371-x.

García, Salvador/Luengo, Julián/Herrera, Francisco (2014). Data Preprocessing in Data Mining. s.l., Springer-Verlag.

Gärtner, Alexander Elenga (2023). condition detection. Available online at https://swarm-engineer.me/2023/condition-detection/ (accessed 2/7/2023).

Gärtner, Alexander Elenga/Fay, Tu-Anh/Göhlich, Dietmar (2022). Fundamental Research on Detecting Contradictions in Requirements: Taxonomy and Semi-Automated Approach. Applied Sciences 12 (15), 7628. https://doi.org/10.3390/app12157628.

Gericke, Kilian/Blessing, L. (2012). An analysis of design process models across disciplines. In: D. Marjanovic/M. Storga/N. Pavkovic et al. (Eds.). DESIGN 2012. Proceedings of the 12th International Design Conference, May 21 - 24, 2012, Dubrovnik, Croatia. Zagreb, Fac. of Mechanical Engineering and Naval Architecture, pp. 171-180.

Göhlich, Dietmar/Bender, Beate/Fay, Tu-Anh/Gericke, Kilian (2021). PRODUCT REQUIREMENTS SPECIFICATION PROCESS IN PRODUCT DEVELOPMENT. Proceedings of the Design Society 1, 2459–2470. https://doi.org/10.1017/pds.2021.507.

Göhlich, Dietmar/Fay, Tu-Anh/Jefferies, Dominic/Lauth, Enrico/Kunith, Alexander/Zhang, Xudong (2018). Design of urban electric bus systems. Design Science 4. https://doi.org/10.1017/dsj.2018.10.

Gudivada, Venkat N. (2018). Natural Language Core Tasks and Applications. In: Computational Analysis and Understanding of Natural Languages: Principles, Methods and Applications. Elsevier, 403–428.

KHOO, C. S. G./KORNFILT, J./ODDY, R. N./MYAENG, S. H. (1998). Automatic Extraction of Cause-Effect Information from Newspaper Text Without Knowledge-based Inferencing. Literary and Linguistic Computing 13 (4), 177–186. https://doi.org/10.1093/llc/13.4.177.

Liu, Chun/Zhao, Zhengyi/Zhang, Lei/Li, Zheng (2021). Automated Conditional Statements Checking for Complete Natural Language Requirements Specification. Applied Sciences 11 (17), 7892. https://doi.org/10.3390/app11177892.

Loucopoulos, Pericles (2005). Requirements engineering. In: John Clarkson/Claudia Eckert (Eds.). Design process improvement. London, Springer London, 116–139.

Maxwell, Joseph (1992). Understanding and Validity in Qualitative Research. Harvard Educational Review 62 (3), 279–301. https://doi.org/10.17763/haer.62.3.8323320856251826.

Merriam-Webster (2022). condition. Available online at https://www.merriam-webster.com/dictionary/condition (accessed 10/4/2022).

Narkhedem, Sarang (2018). Understanding Confusion Matrix. Available online at https://towardsdatascience.com/understanding-confusion-matrix-a9ad42dcfd62 (accessed 10/11/2022).

Pohl, Klaus/Rupp, Chris (2021). Basiswissen Requirements Engineering. Aus- und Weiterbildung nach IREB-Standard zum Certified Professional for Requirements Engineering Foundation Level. 5th ed. Heidelberg, dpunkt.verlag.

Sennrich, Rico/Volk, Martin/Schneider, Gerold (2013). Exploiting Synergies Between Open Resources for German Dependency Parsing, POS-tagging, and Morphological Analysis 2013.

Twain, Mark (1880). The Awful German Language. In: Mark Twain: A Tramp Abroad. Toronto, Belford & Co., 879–897.

VDI 2221 Blatt 1:2019-11. Design of technical products and systems - Model of product design, 2019. Berlin.

https://doi.org/10.1017/pds.2023.71 Published online by Cambridge University Press