machine learning & knowledge extraction

MDPI

Article

A Text-Based Predictive Maintenance Approach for Facility Management Requests Utilizing Association Rule Mining and Large Language Models

Maximilian Lowin

Chair of Information Systems and Information Management, Goethe University Frankfurt, 60629 Frankfurt, Germany; lowin@wiwi.uni-frankfurt.de

Abstract: Introduction: Due to the lack of labeled data, applying predictive maintenance algorithms for facility management is cumbersome. Most companies are unwilling to share data or do not have time for annotation. In addition, most available facility management data are text data. Thus, there is a need for an unsupervised predictive maintenance algorithm that is capable of handling textual data. Methodology: This paper proposes applying association rule mining on maintenance requests to identify upcoming needs in facility management. By coupling temporal association rule mining with the concept of semantic similarity derived from large language models, the proposed methodology can discover meaningful knowledge in the form of rules suitable for decision-making. Results: Relying on the large German language models works best for the presented case study. Introducing a temporal lift filter allows for reducing the created rules to the most important ones. Conclusions: Only a few maintenance requests are sufficient to mine association rules that show links between different infrastructural failures. Due to the unsupervised manner of the proposed algorithm, domain experts need to evaluate the relevance of the specific rules. Nevertheless, the algorithm enables companies to efficiently utilize their data stored in databases to create interpretable rules supporting decision-making.

Keywords: predictive maintenance; facility management; temporal association rule mining; sentence transformer; semantic similarity

Citation: Lowin, M. A Text-Based Predictive Maintenance Approach for Facility Management Requests Utilizing Association Rule Mining and Large Language Models. *Mach. Learn. Knowl. Extr.* **2024**, *6*, 233–258. https://doi.org/10.3390/make6010013

Academic Editors: Irena Spasić and Karin Verspoor

Received: 17 October 2023
Revised: 23 January 2024
Accepted: 24 January 2024
Published: 26 January 2024

1. Introduction

Maintenance represents a substantial share of work in various industries. Due to its significant financial impact [1], industry and research focus on improving the effectiveness of maintenance. Predictive Maintenance (PM) is one way to reduce costs and downtimes by planning maintenance work based on an asset's actual condition rather than relying on fixed time-based maintenance cycles [2]. Multiple industries like aviation [3,4], manufacturing [5,6], and chemistry [7,8] vastly apply PM into their operational routine. Also, other industries like construction and facility management can benefit from improved maintenance methodologies. Building information modeling (BIM) and computer-aided facility management (CAFM) generate a large amount of data that can potentially be used for PM [9]. Such systems utilize digital twins to map the properties of physical entities like buildings or machines into the virtual world [10]. However, research on PM for facility management is rare. Some works in this area, such as those presented in [11] or [12], concentrate on the maintenance of specific infrastructure in buildings like heating, ventilation, and air conditioning (HVAC). Other works like [13] utilize sensor information to predict an infrastructure's condition. However, most data are unstructured and are in text form [10,14]. This textual form is problematic since most PM algorithms require structured data, e.g., sensor measurements, to transform them into information ready for decision-making. Unlike numeric information, computers cannot interpret textual information directly. For instance,

different textual formulations may have the same semantic meaning, while one word in a text can drastically change its whole meaning.

Nevertheless, neglecting textual data misses the opportunity to apply PM to areas where only textual data are available, e.g., in facility management, where facility owners rarely install sensors or are not allowed to store sensor information to protect the inhabitants' or employees' privacy. Thus, it is essential to utilize the already existing textual data by using knowledge extraction from textual databases [14]. Maintenance request analysis is an upcoming field in this setting [9]. Maintenance requests are textual information about specific maintenance tasks that describe defect infrastructure and add more contextual information. Facility managers often store them in CAFM software to track status updates and staff assignments. Utilizing this information for PM can potentially optimize processes and reduce breakdowns. However, there is little research on this topic [15], and most work still focuses on PM based on structured data [10].

Another problem in facility management and construction is the phenomenon of data monopoly: most companies that manage extensive facilities cannot share their data with others, making the datasets needed for PM challenging to access [9]. However, big datasets containing data from several companies are a crucial prerequisite for adopting most supervised PM models that require explicit labels. These labels rarely exist in the facility management context since they are expensive to collect. Additionally, relying on supervised labels is inadequate in this context since the nature of maintenance works is an ongoing process for PM and facility management [16]. A possible solution to overcome this limitation is using semi-supervised learning, transfer learning, and pre-trained models [9]. However, based on current understanding, there is no literature about applying such algorithms for PM in facility management. Therefore, this paper aims to answer the following research question: How do we utilize knowledge discovery on maintenance requests for predictive maintenance?

This paper sheds more light on this research question by combining association rule mining (ARM) with pre-trained large language models (LLMs) and adopting it on maintenance requests. Maintenance requests are textual descriptions of damages or upcoming work that potentially contain relevant information for PM in a facility management context. ARM is a powerful group of algorithms suitable for PM in many industries [7]. Furthermore, LLMs based on transformer models allow a computer to understand the semantic meanings of natural-written texts [17]. Combining these research fields allows for adding predictive functionalities to almost every CAFM system by applying an unsupervised PM algorithm. The benefit of this approach is that it does not rely on labeled data. Furthermore, it allows companies to utilize the data they have already collected to better understand the interplay between infrastructures and predict impending failures on a wide range of infrastructures. Moreover, this paper studies the applicability of four different language models and compares their performances when coupling them with ARM.

This paper is structured as follows to answer the abovementioned research question: The next section introduces the theoretical background of this paper. Section three outlines the proposed methodology by adopting Apriori, a widely used ARM algorithm, to the setting of maintenance requests and introduces the data utilized in this paper. Section four overviews the results of the proposed algorithm's solution. Section five discusses its theoretical and practical relevance. The last section concludes this work and outlines future research.

2. Theoretical Background

2.1. Predictive Maintenance for Facilities

Traditional maintenance management techniques are run-to-failure and preventive maintenance [2]. Run-to-failure maintenance strategies perform no maintenance work until a specific infrastructure breaks. Preventive maintenance strategies perform a time-driven schedule, e.g., yearly maintenance checks [2].

However, both approaches contradict optimized decision-making strategies that minimize costs and increase quality, safety, and productivity [10]. Therefore, the PM approach relies on an infrastructure's physical condition to predict its future condition and potential failures [2]. To determine the physical condition, traditional PM algorithms use sensor information like the machine's temperature, voltage, or current [18]. Typical PM approaches detect faults, i.e., low-frequency but high-impact events [3], or calculate an infrastructure's lifetime and metrics like the mean time to failure [19]. However, the definition of PM is ambiguous, but its main aim is to improve an infrastructure's operation using data [2]. This paper adopts the definition of Carvalho and colleagues, who define PM as a method that "uses predictive tools to determine when maintenance actions are necessary" by monitoring the specific infrastructure (like a machine) and utilizing historical data [20] (p. 20).

Since most PM approaches rely on sensor data, applying PM for facility management is not trivial. Most corporate information is in the form of texts, especially in the construction sector [10,14]. Such texts can be document files, sheets, or semi-structured forms like Extensible Markup Language or Hypertext Markup Language that require Natural Language Processing (NLP) to extract the information from human language texts [9]. Maintenance requests are a valuable source of information to assess the condition of specific infrastructure in the context of facility management [15]. The work of [21] is among the first to utilize these requests. In this study, the authors create a prediction model that automatically assigns staff to specific requests containing unstructured text. Another field in this area is a relation analysis of such maintenance requests [10]. Wu and colleagues [10], in an extensive literature review, present that co-occurrence analysis, ARM, heuristic rules, or supervised learning are suitable approaches for relation analysis. While staff assignment is also a form of supervised machine learning, it requires explicit labels to learn a classification task. The authors of this literature review outline that most papers build on a supervised problem, while manually labeling datasets is often cumbersome and impractical.

In contrast, using unstructured text data is cheaper to collect and more accessible [10]. Only a few works use unsupervised techniques on textual data for predictive maintenance. Akhbardeh and colleagues utilize clustering on maintenance logs to find similar maintenance logs [22]. Bhardwaj and colleagues apply hand-crafted lexicographic sentiment analysis on maintenance reports to identify infrastructure health status [23]. A last unsupervised PM approach utilizing maintenance logs extracts machine components and their associated failures [24]. However, these approaches mainly help to estimate the status quo from maintenance requests but do not predict helpful information. Other works that integrate unsupervised machine learning for predictive maintenance, like anomaly detection, do not rely on textual data but on sensor data [25–27]. Therefore, this paper utilizes ARM as an unsupervised method that is easy to implement for unlabeled data, as is mostly the case for facility management.

2.2. Association Rule Mining

ARM is a suitable method for PM tasks [7]. In their seminal work, Agrawal and colleagues [28] introduce ARM by utilizing shopping basket data and mine rules that present what items customers have purchased together. These rules are in the form of A $\rightarrow$ B, where A is the antecedent; B is the consequent, and A and B are a set of items/articles ($n \geq 1$) in the basket. To determine relevant rules and their quality, ARM algorithms use different metrics, i.e., especially support and confidence, which the literature defines as follows [7]:

$$support(A) = \frac{|A|}{m}, \tag{1}$$

$$support(A \rightarrow B) = \frac{|A \cup B|}{m}, \tag{2}$$

$$confidence(A \rightarrow B) = \frac{support(A \rightarrow B)}{support(A)}. \tag{3}$$

Support is the probability of finding a transaction containing a respective item set [7]. The expression $|A|$ represents the number of transactions containing A; $|A \cup B|$ represents the number of transactions containing both A and B, and m represents the number of transactions. In addition, confidence is the conditional probability of finding the item set of the consequence given the occurrence of the antecedent [7]. These definitions are consistent with a wide range of works in the field of ARM [28–31]. Another popular metric is the lift, which calculates whether the two item sets A and B of rule $A \rightarrow B$ are dependent or independent of each other [29]:

$$lift(A \rightarrow B) = \frac{support(A \rightarrow B)}{support(A)*support(B)}. \quad (4)$$

A lift value of one means that there is no relation between both item sets; a degree greater than one indicates a positive dependence and makes them interesting for further mining [29].

The most prominent algorithm for finding association rules is Apriori, which was introduced by [32]. It was one of the first algorithms for ARM that worked efficiently by maintaining minimum support and confidence in the association rules [32]. In their paper, the authors also present two other algorithms, AprioriTid and Apriori Hybrid. In [33], the authors give an overview of different ARM algorithms like the aforementioned Apriori derivatives and newer algorithms like frequent pattern-growth (FP-growth) and evaluate their performance regarding data support, speed, and accuracy. However, all ARM algorithms have in common that they rely on support and confidence.

Indeed, practice vastly uses ARM algorithms. One significant advantage is their ability to deal with large amounts of unstructured data and that they provide very interpretable rules suitable to enhance decision-making [30]. For instance, [34] applies the FP-growth algorithm for PM sensor data of Internet of Things hardware to extract association rules. ARM can also be used on text data, even if one cannot apply it on raw text but on different representations like a bag of words or term frequency [30]. Bag of words and term frequency are typical text representations using NLP. The concept of the bag of words considers the frequency of each word in a text while ignoring their position in the text [35]. The term frequency originates from [36] and describes how often a specific term (e.g., a word) occurs in a document. If one considers multiple documents, one can calculate the document frequency, i.e., the number of documents a term occurs [35]. Word embeddings are feature representations of a word (e.g., in the form of a vector) where each dimension aims to capture its syntactic and semantic meaning [37]. Typical tasks using ARM on text are summarization, topic and event detection, forecasting, and collaborative social systems [30]. For instance, the work of [38] applies Apriori to customer reviews. The work of [39] follows the Apriori algorithm, combining text analysis with knowledge bases to form semantic rules. In [40], the authors mine association rules from medical records by extracting medical features like symptoms, diseases, and medicine. The work of [29] employs ARM to find associations between words in the Azerbaijani language using a bag-of-words approach. Finally, the authors of [7] demonstrate the suitability of ARM for industrial use cases. They first generate association rules and then apply linear programming to select components to repair to improve their infrastructure's overall robustness.

However, there is one major problem when applying ARM on text using simple representations like bag-of-words or term frequency–inverse document frequency (TF–IDF). Unstructured free texts use heterogeneous language with little consistency, i.e., they contain typos, acronyms, abbreviations, and jargon [41,42]. Applying ARM on large text can result in data dispersion and binary representations that might lead to sparse matrices [30]. Since facility management is a domain where one can expect such an inconsistency, there is a need to extend the typical ARM definitions. Transfer learning and large language models like BERT or GPT can overcome such limitations by capturing contextual information [9]. Bidirectional Encoder Representations from Transformers (BERT) is a language representation model that is already pre-trained, meaning that one

only needs to fine-tune a model for a specific use case, making it handy to apply for various problems [43]. A transformer is an ML model that typically uses an encoder component to transform text into a computer-understandable format (i.e., encodes an input sequence of symbol representations like words into a sequence of continuous representations) [44] and adds the concept of attention. The attention mechanism allows a neural network to focus on specific, relevant parts of the input sequence [42]. The Generative Pre-trained Transformer (GPT) is a transformer model that can handle text and images as inputs and generates text as output by adding an additional decoder component to recast the continuous representation into the original symbol representation [45]. The model became famous for its application in the software ChatGPT. However, since its current version, GPT-4, is not open-source, this paper focuses on BERT-based open-source alternatives like RoBERTa, an optimized version of BERT [46], and MiniLM, a compressed version of a pre-trained transformer model that has fewer capacity constraints compared to larger models [47].

2.3. Extensions of Association Rule Mining

Most ARM research consists of applying ARM algorithms to various domains or focuses on improving the performance of different mining techniques. However, this paper wants to emphasize two crucial extensions to ARM that are necessary when applying it to facility management: (1) the integration of similarity; and (2) temporal information.

The main idea of integrating similarity into ARM is that a specific term or phrase can be very similar or even identical to another term while not being classified as such by ARM due to its separate representation. For instance, the work of [48] gives the example of Ceylon as the former name of Sri Lanka. Both terms reflect the same nation. However, they have different textual representations. In their work, the authors of [48] use Apriori to detect semantically identical but temporally different concepts and utilize Jaccard's coefficient as a similarity measure.

Another strain of the literature integrates the concept of similarity by mining rules from similar texts. The work of [14] applies ARM to text previously clustered by similarity. However, one can also integrate the concept of similarity directly after rule creation. In [49], the authors present a way to apply cosine similarity to rules generated by Apriori to unify similar rules. In [50], the authors present a similar approach where they create rules by running Apriori and extend their rules based on similarity by re-calculating the support of extended rules. They also use cosine similarity as a general-purpose metric but emphasize that a domain-specific similarity metric could be helpful. However, their approach is only suitable for generating rules with a unitary length of the antecedent [50]. Therefore, this paper's approach will integrate the concept of similarity by utilizing LLMs and integrate the similarity measure directly into the definition of support as proposed in [50].

Another essential extension when using ARM for facility management is integrating temporal information. In a literature review about temporal ARM performed in [31], the authors construct a taxonomy of how to incorporate time into ARM. They suggest integrating time as an implied component or an integral component. When integrating time as an implied component, the time variable provides information about the order, including temporal constraints and sequences [31]. Furthermore, the authors describe time as an integral component in that the time variable becomes an attribute within the learning process and that time indicates potentially periodical or time interval-based patterns. Both approaches are helpful for ARM and PM. For instance, the work of [51] applies temporal ARM for train maintenance. It uses time information as a criterion that ARM can predict target events like repairs early enough to allow for logistic and maintenance actions. On the contrary, it utilizes time information as a limit that the potential items for ARM are also recent and relevant. This procedure allows for splitting the prediction time into warning and monitoring times [51]. The work of [52] adds a recency weight to transactions to limit items being temporally close to each other. In addition, it uses a time decay function to avoid making wrong decisions with out-of-date rules and adopt the corresponding definition of the support function.

3. Methodology

Based on the previous literature, a research gap exists in easily integrating an unsupervised PM methodology for facility management that can handle heterogeneous language. This paper proposes using ARM on the maintenance request level. Figure 1 summarizes this paper's methodology. The basic idea is to find meaningful maintenance requests that represent an early warning indicator for other repairs (in the form of further maintenance requests) and to reveal interrelationships in the data. This knowledge in the form of association rules helps facility management decision-making to optimize maintenance processes and warn them of upcoming failures. Thus, this paper considers each maintenance request as a unity. In the wording of ARM, a maintenance request corresponds to an item, and all maintenance requests of a specific infrastructure (e.g., a power supply, as shown in the first step of Figure 1) correspond to a transaction. The goal of the following approach is to transfer repairs manifested through maintenance requests from one infrastructure (e.g., a machine, an air conditioner, or a lift) to another of the same kind.

This paper proposes two extensions to apply ARM on textual maintenance requests. The first one considers semantic similarity. Heterogeneous language makes finding duplicate maintenance requests almost impossible when relying on the classical forms of text representations used for ARM, like bag-of-words or TF–IDF, that work on a word level. The second one integrates temporal information into the proposed algorithm. The following subsections dive more into detail about the specific methodology.

3.1. ARM with Semantic Similarity

The literature shows that integrating semantic similarity is vital when dealing with textual data and that various ways exist to measure the semantic similarity between words or texts. Especially when comparing short texts, the word co-occurrence between both texts might be very low or even nonexistent [53]. One more advanced method is to use word embeddings that operationalize the semantic meaning of a word into a vector and allow one to compare it with other words [9]. A widely used similarity measure that can handle such word vectors is the cosine similarity [49,50], ranging from -1 to 1, where -1 means that two vectors are the opposite; 0 means that they are dissimilar or orthogonally, and 1 means that they are the same. One perk of using this kind of word embedding is that a specific word can have a different meaning depending on the context [42]. Therefore, this work uses embeddings from large language model (LLMs) that can capture the context [9].

Furthermore, these language models are pre-trained on large text corpora, enabling them to be applied for various applications [42]. Since this paper relies on maintenance requests that consist of a short description, including a few words or sentences, it is beneficial to use sentence transformers that operate on a sentence level instead of a word level. Sentence transformers can calculate the semantic similarity between two sentences using the cosine similarity of the vector representation and are computationally faster and more precise compared to word-level similarity calculations [54].

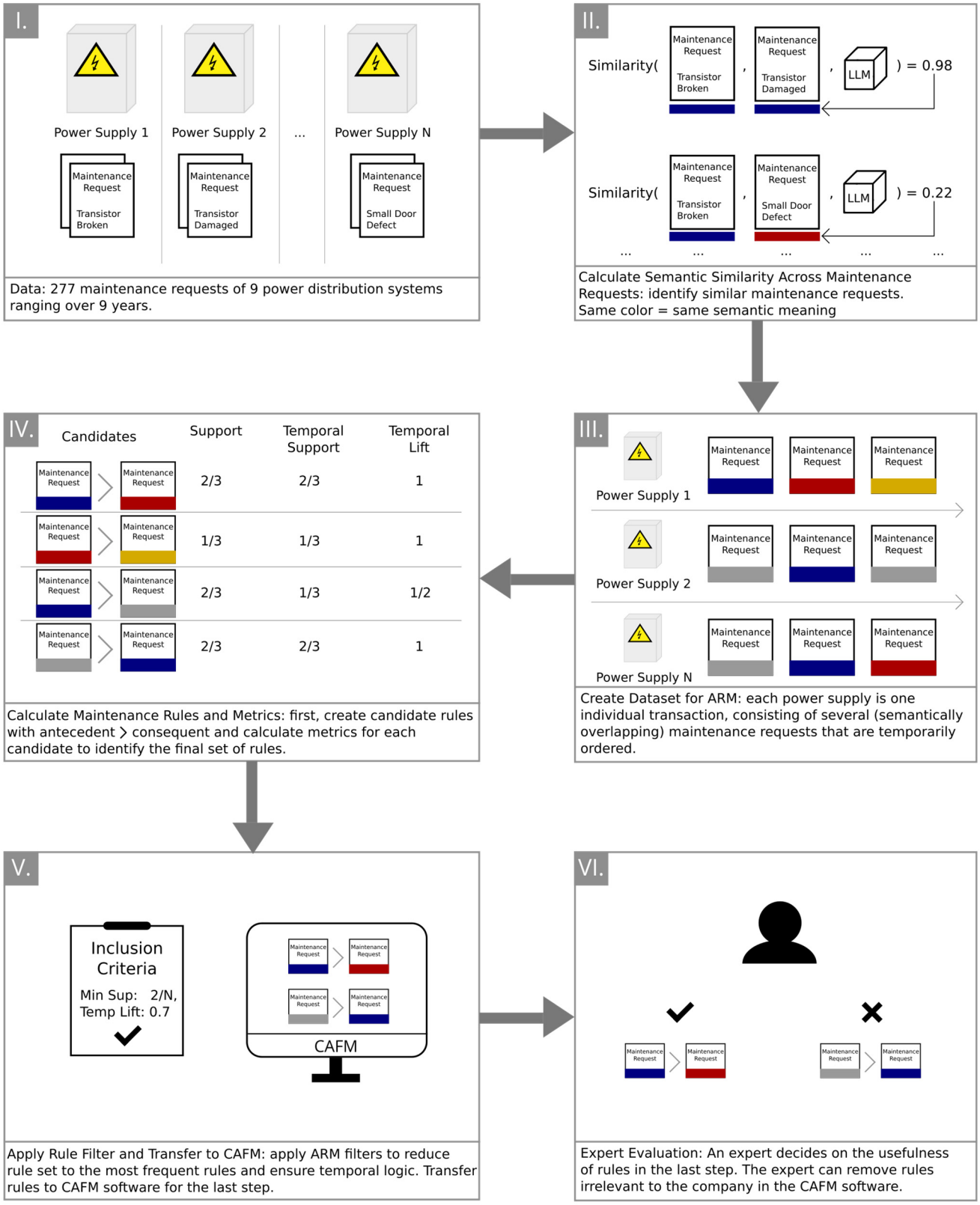


Figure 1. Overview of the Methodology.

Thus, this paper proposes calculating the semantic similarity across all maintenance requests to identify which requests are similar to each other (step II of Figure 1). This pre-calculation allows the ARM algorithm to handle the dataset as structured information. Instead of the textual information of the maintenance requests, the ARM algorithm interprets coded items for each infrastructure where the same code equals the same semantic meaning. Figure 2 shows this code in the form of colors (see step III). This extension to ARM improves the creation of the candidate itemsets (e.g., via Apriori-Gen) in step IV, on which the ARM algorithms calculate the support function to find frequent itemsets. However, this paper proposes a different approach than [49] or [50] to integrate semantic similarity into ARM. Instead of first applying an ARM algorithm like Apriori on the raw data and applying the similarity measure to rules created, this paper proposes to directly alter the support function as an input to the ARM algorithms. The work of [50] suggests a similar approach in its outlook section without delving deeper into the topic. The proposed modification utilizes the pre-calculated similarity scores from step II and allows for identifying semantic similar maintenance requests as one unity with more occurrences.

```
function support_similarity(items_for_support, transactions, min_similarity):
    support_nominator = 0
    support_denominator = length(transactions)
    found_items = Empty Set, e.g. HashSet

    for each items_of_transactions in transactions:
        for each item_of_transaction in items_of_transactions:
            for each item_to_compare in items_for_support:
                if similarity(item_to_compare, item_of_transaction) >= min_similarity:
                    add item_of_transaction to found_items

                if length(found_items) == length(items_for_support):
                    increase support_nominator by 1
                    break loop, go to next items_of_transactions

    return support_nominator / support_denominator
```

Figure 2. Pseudo-Code of Modified Support Function, Including Similarity.

In order to do so, the support function needs an additional parameter to account for the minimum similarity that ARM can treat one item to be a similar maintenance request. Therefore, Figure 2 shows a pseudo-code of a suggested implementation of a modified version of the support function that incorporates item similarity.

The main idea is that the support function checks for each transaction whether all items for which it should compute the support value (variable items_for_support, equivalent to variables A and B in Equations (1) and (2) and originating from the candidate set; see Figure 1) are part of this transaction under having a certain similarity to the original item. If a transaction contains all items or similar versions of it, it increases the nominator of the support by one (and repeats this check for each transaction). The function uses a similarity function that relies on the cosine similarity of the vectors created by a sentence transformer. Please note that this pseudo code is only an exemplary implementation and that its computational complexity can be further decreased (but harming legibility for this paper).

The modified version of the support function directly applies to all ARM algorithms that rely on the support of item sets. This paper uses the support function with a default implementation of Apriori to demonstrate its suitability and utilizes the pre-calculated similarity scores from step II. One can also replace the support function of computationally less complex algorithms like AprioriHybrid or FP-growth [33]. Besides including semantic similarity in ARM, extending a temporal component into the algorithm is also beneficial when applying ARM to facility management.

3.2. Temporal Extension

Since ARM algorithms can integrate temporal information as implied or integrated components [31], facility management supports both views of the time dimension. First, all maintenance requests reach the CAFM software in a temporal order. Therefore, it is essential to include this implied view of temporal information. In addition, for PM, it might influence the timing of when a maintenance request arrives. PM algorithms can use this integrated information directly in their reasoning. However, to reduce the complexity, this paper only concentrates on the implied component view.

Therefore, this paper introduces the concept of temporal lift. While the original lift of ARM is a metric that manifests the meaningfulness of a rule [52], the temporal lift evaluates whether the consequent of a rule succeeds temporally after the antecedent. First, it is necessary to define a temporal support function that counts the support for all transactions where the consequent comes temporally after the antecedent. Figure 3 shows the pseudo-code of the temporal support function that only increases the support of a transaction if the consequent comes after the antecedent(s). Furthermore, it assumes that all transaction items are in a temporal order.

```
function support_similarity_temp(items_antec, conseq, transactions, min_similarity):
    support_nominator = 0
    support_denominator = length(transactions)
    found_items = Empty Set, e.g. HashSet

    for each items_of_transactions in transactions:
        found_antecedent = Empty Set, e.g. HashSet
        found_consequent = Empty Set, e.g. HashSet
        for each item_of_transaction in items_of_transactions:
            if length(found_antecedent) == length(items_for_support):
                for each item_to_compare in conseq:
                    if similarity(item_to_compare, item_of_transaction) >=
                        min_similarity:
                            add item_to_compare in found_consequent
                if length(found_consequent) == length(conseq):
                    increase support_nominator by 1
                    break loop, go to next items_of_transaction
            for each item_to_compare in items_antec:
                if similarity(item_to_compare, item_of_transaction) >=
min_similarity:
                    add item_of_transaction to found_antecedent

    return support_nominator / support_denominator
```

Figure 3. Pseudo-Code of Temporal Support Function.

The temporal lift utilizes the temporal support function. It calculates the share of transactions where a rule A → B, indeed, appears in the right temporal order compared to an unordered set of items:

$$Lift_{Temporal}(A \rightarrow B) = \frac{Support_{Temporal}(A \rightarrow B)}{Support(A \rightarrow B)}. \quad (5)$$

Instead of the original lift definition, which ranges from 0 to ∞, the temporal lift ranges from 0 to 1. In this case, 0 means that none of the items occurring in the rules are in temporal order. Conversely, a value of 1 means that all items that are part of the rule occur in a temporal order in the respective transactions.

After the ARM algorithm extracted all potential association rules using the semantic similarity extension, this paper proposes applying the temporal lift to remove all rules that violate the temporal logic. Thus, a minimum support and a minimum temporal lift

parameter can be dealt with as inclusion criteria, as shown in step V of Figure 1. The CAFM system can record the included rules for the next steps.

3.3. Expert Evaluation

Finally, a domain expert needs to evaluate the suitability of the semantic similarity model and the quality of the created rules depending on their eligibility for the specific context of the facility management company and its decision-making. First, a human domain expert needs to assess the quality of the semantic similarity model, as human judgment is the gold standard for evaluating semantic text similarity [55]. Using a human evaluation of automated outputs is also common for other unsupervised NLP techniques, such as topic modeling, to guarantee the appropriateness of the model outputs for a particular setting [56]. Human domain experts can determine the semantic similarity using an ordinal scale, e.g., a five-point scale ranging from "highly unrelated" to "highly related" [57]. This paper adopts the semantic similarity scale from the work of Agirre and colleagues [58], who propose a six-point Likert scale. In their definition, a label of 0 means that the two texts are entirely dissimilar. A label of 1 indicates that two texts are not equivalent but topically related. A label of 2 indicates texts that are not equivalent but agree on some details. A label of 3 indicates that the two texts are approximately equal, but some important details differ. Likewise, a label of 4 indicates that the two texts are roughly equivalent and only unimportant details differ. Last. A label of 5 indicates that the two texts are semantically identical. This scale allows human annotators to intuitively label sentence pairs without any training in formal semantics [58].

Second, this paper acknowledges that also the ARM process is not entirely automatic and needs human intervention to adapt the association rules to fit the company's context. Since rules extracted by ARM are not causal per definition, an expert needs to evaluate their causality and impact. In addition, a few rules might mistakenly obtain a high support value because of the semantic similarity classification and a threshold that a company puts too low. This paper proposes adding an expert evaluation step of the created rules at the end of the ARM process by integrating a dedicated user interface into the CAFM software. Such a user interface should allow for the viewing of the created rules and the related maintenance requests that led to the respective rules. Experts must be able to alter the rules, remove them, or adjust the ARM thresholds. In addition, they should be able to weigh the rules according to their impact. This evaluation process helps the company extract the most relevant rules and enables decision-makers to optimize maintenance processes and prevent infrastructural breakdowns. Finally, there might be a temporal gap between two maintenance events that a domain expert must refer to and needs to consider. Also, the expert must view contextual information like process changes that make a rule obsolete. Although these manual steps are necessary, the proposed approach helps extract crucial knowledge from large textual databases that single experts cannot quickly overview without such a support tool. It allows for finding associations that are not straightforward for a human expert, and the result of the process should be that only meaningful rules remain. These can be rules that help optimize maintenance processes, help identify high-impact events in a timely manner, support decision-makers by uncovering hidden connections, or manifest associations in an easily understandable form. The proposed approach's ultimate goal is to support company decision-making by enhancing its knowledge base.

3.4. Technical Case Study

This paper uses a real-world dataset that contains actual maintenance requests from a German industrial company to evaluate the suggested ARM modifications. The company collects maintenance requests in CAFM software. It collects information like the date, description text, and respective infrastructure of a request. In particular, since comparing maintenance requests within the same type of infrastructure is especially interesting, this paper uses only maintenance requests for power distribution systems. In total, there are 277 maintenance requests ranging over a period of 11 years for nine different power

distribution systems. The description text is in German and has a mean and standard deviation of 78 ± 61 characters.

While the implementation of ARM algorithms is language-independent, the semantic similarity extension in the case of this specific dataset necessitates that the similarity function works in the German language. Although most sentence transformer models work only for English, there are several options for working in German. First, NLP models work well for machine translation [38]. Therefore, one can first use an NLP translation model to translate German texts into English. This paper uses the DeepL API in version 1.14.0 for translation and a sentence transformer for the English language, e.g., based on the MiniLM model [59]. Second, several sentence transformers that can understand multiple languages are available. One example is the Cross English–German RoBERTa for Sentence Embeddings [60]. Third, a few models solely work on the German language, e.g., [61]. This paper uses four different language models and compares their usefulness for ARM. Table 1 gives an overview of the four models.

Moreover, this paper compares the similarity score and ARM performance using four additional similarity metrics as a baseline: BLEU; Rouge-L; METEOR; and BERTScore. BLEU is a metric initially developed to compare the quality of machine translations versus a corpus of human reference translations [62]. It is *n*-gram-based and, therefore, computationally less complex than LLMs. Similarly, Rouge-L also evaluated *n*-gram co-occurrence and was developed to determine the quality of text summaries [63]. METEOR applied unigram matching and was also developed to assess the quality of machine translations [64]. Finally, BERTScore calculates the token similarity between two texts using the word embeddings [65]. All four scores calculate the similarity between two (or more) texts and are suitable to integrate them into the ARM process, like the similarity scores obtained from the LLM sentence transformers. This paper uses BLEU, Rouge-L, and METEOR as a baseline comparison to the LLM sentence transformer-based similarity score metrics due to their lower computational complexity. In addition, it uses the BERTScore as a different LLM-based approach.

Table 1. LLMs Used for Similarity Measure.

Model Identifier	Type of Model	Languages	Huggingface Handle	Publisher
German RoBERTa	RoBERTa	German	T-Systems-onsite/ german-roberta-sentence-transformer-v2	T-Systems on site services GmbH, Berlin, Germany
Cross RoBERTa	RoBERTa	German, English	T-Systems-onsite/cross-en-de-roberta-sentence-transformer	T-Systems on site services GmbH, Berlin, Germany
English RoBERTa	RoBERTa	English	sentence-transformers/ all-roberta-large-v1	Nils Reimers, Ubiquitous Knowledge Processing (UKP) Lab, Technical University of Darmstadt, Darmstadt, Germany
English MimiLM	MiniLM	English	sentence-transformers/ all-MiniLM-L6-v2	Nils Reimers, Ubiquitous Knowledge Processing (UKP) Lab, Technical University of Darmstadt, Darmstadt, Germany

The following section first compares the quality of the semantic similarity extracted via the four LLM-based sentence transformers and the four baseline models. It first creates rules iteratively and evaluates the number of rules found and the number of hits based on a simulation approach that considers the temporal order of the maintenance requests. Then, it applies a human expert evaluation to validate the meaningfulness of the created rules.

This paper defines a hit as a match between the maintenance request of a specific infrastructure and a rule created by ARM, i.e., its antecedents(s) and consequence(s). A

hit is only present if the consequent(s) proceeds after the antecedent(s) in the data to ensure temporal logic. A rule can have multiple hits if it matches other infrastructure. The simulation is iterative, meaning there will be $n = 5$ additional maintenance requests in sequential order in each iteration. The simulation consists of the following steps:

1. Create rules based on all maintenance requests that are available at the time of the current iteration;
2. Check for all rules, whether they are new rules or whether they have been found in a previous iteration (by applying a similarity measure also to identify very similar rules);
3. Optional: check for each rule whether a human domain expert estimates a contextual connection between the antecedent and the consequence of a rule. Only keep rules with a possible connection;
4. Calculate the hits of the new rules on the current data (i.e., the hits that the ARM algorithm used to create the new rules);
5. Calculate the hits of the new rules on all available data (also future maintenance requests);
6. Calculate the future hits by subtracting the hits on the current data from those on all available data;
7. Keep the new rules in the backlog for the next iteration.

The main idea behind this procedure is to ensure a realistic view of ARM that creates association rules near the arrival of the maintenance requests. The differentiation between steps 4 and 5 ensures that it is possible to calculate the future hits in step 6. This measure serves as a proxy for a rule's applicability and to measure an ARM algorithm's accuracy. Step 3 consists of the domain expert evaluation of the created rules. In this technical case study, the domain expert evaluation takes place retrospectively to minimize the burden of the domain experts. This paper first calculates all rules, excluding step 3. Then, the human domain experts need to evaluate the eligibility of all rules. Finally, this paper repeats the rule creation process (steps 1–7) and uses the expert evaluation as an additional filter in step 3. This paper utilizes two domain experts, one from the practice partner who is familiar with the facility of the dataset and one electrical expert who is familiar with the technical background. This paper further considers a rule if an expert estimates the rule as meaningful.

The simulation uses a minimum support threshold of 0.2, i.e., a maintenance request must be present in at least two power distribution systems, a minimum confidence of 0.1., and a minimum temporal lift of 0.7. The support and confidence thresholds are relatively low due to the nature of PM, and most defects are rare events [3]. Furthermore, similar works like [49] also use a minimum support threshold of 0.2. Other works, like [7], do not explicitly state their thresholds but also present rules with confidence close to 0.1. Since the similarity of the maintenance requests heavily depends on the utilized similarity measure, the results section gives more insights into the minimum similarity threshold.

4. Results

4.1. Semantic Similarity Comparison

This paper first calculates the semantic similarity between all maintenance requests. This pre-calculation helps to obtain an overview of the classification performance of the utilized sentence transformer LLMs and reduces the simulation's complexity. In addition, it helps execute the function similarity(item1, item2) repeatedly quickly. In addition, pre-calculating similarity scores allows for comparing the correlation across the different language models. Table 2 shows an excerpt of some maintenance requests and their semantic similarity calculations using the German RoBERTa sentence transformer.

Table 2. Examples of Semantic Similarity Using German RoBERTa.

Item 1	Item 2	Semantic Similarity
Room [room number]—Fire bulkhead defective (defect from test [test number])	Room [room number]—The fire bulkhead on the ceiling is damaged (defect from test [test number])	0.78
Meter Reading November 2020	Meter Reading December 2020	0.91
[room number]; no electricity in the entire area	No electricity in the [company name] warehouse in the [room number]	0.76
Cable break at plug	Replacement circuit breaker UV outdoor lighting [room number]	0.19
Power failure at pillar [pillar number]	[room number], the blue cover is missing on a socket, socket still OK.	0.17

Table 3 shows Pearson's correlation coefficient across all four sentence-transformer LLMs used in the analysis, as well as their descriptive statistics (i.e., mean, median, and standard deviation). The table allows the estimation of how related the LLM-based similarity scores are when interchanging the respective LLM. A higher dissimilarity (i.e., lower correlation and higher difference between the descriptive statistics) would indicate that association rules obtained from these dissimilar LLMs will create different rules (ceteris paribus).

Table 3. Pearson's Correlation Coefficient of Similarity Measures and Descriptive Statistics.

		German RoBERTa	Cross RoBERTa	English RoBERTa	English MiniLM
Pearson's Correlation Coefficient	German RoBERTa	1			
	Cross RoBERTa	0.95	1		
	English RoBERTa	0.59	0.57	1	
	English MiniLM	0.62	0.60	0.80	1
Mean		0.25	0.25	0.22	0.19
Median		0.24	0.25	0.20	0.17
Standard Deviation		0.14	0.15	0.14	0.13

Table 3 shows a high positive correlation between the German RoBERTa and the Cross RoBERTa. Both models use the original text of the maintenance requests written in German. In contrast, English RoBERTa and English MiniLM use the translated text in English as input. Accordingly, there is also a high correlation between the similarity scores the English language models calculated. In addition, the similarity scores obtained from the English texts also show a moderate correlation to the other two metrics calculated on the raw German text. Thus, the rules from the translated English text might slightly differ from those created from the plain German text.

Looking into the descriptive statistics of the four LLM-based similarity scores (i.e., German RoBERTa, Cross RoBERTa, English RoBERTa, and EnglishMiniLM) shown in Table 3 affirms that the LLM-based similarity scores obtained from English text are slightly different compared to the LLMs working on the German texts. While this difference is slight, each ARM based on another LLM potentially has a different optimal threshold for the minimum similarity.

The similarity scores from all four LLM sentence transformers have more than 99% of their values below 0.65. Therefore, this paper tried various minimum similarity thresholds (i.e., 0.65, 0.7, 0.75, 0.8, and 0.85). None of the algorithms could construct a rule with a threshold of 0.85, except for most smaller thresholds. Additionally, all models behave similarly.

A human judgment serves as a quality assurance that the various semantic similarity models are helpful in the context of the technical case study. This paper provides a subset of maintenance request pairs to two human judges who have to rate the semantic similarity using the six-point Likert scale proposed by Agirre and colleagues. This scale allows human judges to label sentence pairs without any training in formal semantics [58]. One of the human judges originates from the partnering company (domain judge), and the other one is an independent researcher with a technical background (technical judge). This paper draws a subset of 100 maintenance request pairs from the dataset. Since drawing random sentence pairs would result in most pairs being semantically unrelated [66], this paper draws the request pairs in a structured way. It utilizes one LLM-based semantic similarity model (i.e., German RoBERTa), separates the semantic similarity pairs into ten bins ([0,0.1), [0.1,0.2), ...), and draws ten random pairs from each of these bins. This paper first uses the Pearson correlation between the human judgment obtained from the Likert score and the semantic similarity calculated from the various models to evaluate the semantic similarity calculation. Second, since the Pearson correlation can be sensitive to non-linear relations and outliers, this paper uses the F1-score [55]. For the F1-score calculation, this paper again uses binning, as suggested in [55]. It defines a request pair as semantically similar if the human judge evaluates the semantic text similarity as four or five, meaning that, at most, only unimportant details differ between the texts [58]. Likewise, this paper considers maintenance request pairs as similar if the semantic similarity score of the semantic text similarity model is at least 65%. Table 4 reveals the performance metrics between the two human judges and the eight semantic text similarity models. The table reveals that most models show a moderate to high correlation to the human judges. Some relatively low F1-Scores indicate that fine-tuning thresholds might increase performance. The four LLM-based sentence transformers show the highest performance metrics among both judges compared to the baseline models, suggesting they are better suited for the case study. The English MiniLM model shows the most considerable difference between the human domain judge and the technical judge. This finding might indicate that the MiniLM model can extract semantic similarities in general but not perfectly for the respective facility management domain.

Table 4. Performance Metrics between Semantic Text Similarity Models and Human Judges.

Semantic Text Similarity Model	Human Domain Judge		Human Technical Judge	
	Pearson Correlation	F1-Score	Pearson Correlation	F1-Score
German RoBERTa	0.65	0.66	0.54	1.0
Cross RoBERTa	0.64	0.66	0.56	1.0
English RoBERTa	0.63	0.5	0.69	0.67
English MiniLM	0.57	0.5	0.74	0.8
BLEU	0.34	0.0	0.39	0.0
ROUGE-L	0.50	0.5	0.65	0.8
METEOR	0.46	0.29	0.63	0.5
BERTScore	0.49	0.5	0.65	0.08

4.2. Comparison of Rules without Human Expert Evaluation

The ARM algorithm creates rules as antecedent → consequent, while antecedent and consequent are one or more maintenance requests. For instance, the German RoBERTa model generates four rules using a minimum similarity threshold of 0.75 and temporal lift filtering. Likewise, the Cross RoBERTa model generates six rules with the same parameters. Table 5 shows these rules to understand what ARM-generated maintenance request rules look like. As one can expect from Table 3, the rules look pretty similar when comparing the different semantic similarity bases. However, the Cross RoBERTa model finds two

additional rules not part of the German RoBERTa model. Section 4.3 and 5 discuss the quality of the created rules in more detail.

Table 6 gives more insight into the interplay between the similarity threshold and the number of rules created by the ARM, including semantic similarity, before applying the temporal lift filtering. All ARM approaches using LLM sentence transformers produce a reasonable number of rules on the input data and a decent number of hits on future, unseen data. A higher similarity threshold decreases the number of rules and—associated with this—the number of hits (please note that the number of hits can contain several hits of the same rule but on other infrastructure. i.e., having two hits on four rules can mean that one rule has two hits on different infrastructures). Since Table 3 shows that the similarity score obtained from English MiniLM has, on average, the lowest cosine similarity, the ARM algorithm cannot find matching rules with a similarity threshold higher or equal to 0.7. In contrast, even if the mean and median for English RoBERTa are lower than the similarity scores obtained from the German text, the ARM algorithm performed on English RoBERTa also creates some rules for higher thresholds. However, all similarity scores fail to find rules with a threshold higher than 0.8.

Table 5. Rules Found with Cross RoBERTa and German RoBERTa Using a Minimum Similarity Threshold of 0.75 and a Temporal Lift Filtering.

Model	Antecedent		Consequent
German RoBERTa	Room X: power socket problem; voltage is only 137 V	→	Room Z: fire bulkhead in ceiling defect
	Room Y: PEN error; voltage is only 138 V	→	Room Z: fire bulkhead in ceiling defect
	Room Y: PEN error; voltage is only 138 V	→	External system labeling is missing
	Room X: power socket problem; voltage is only 137 V	→	External system labeling is missing
Cross RoBERTa	Room W: no electricity	→	Room V: fire bulkhead in wall defect
	Room W: no electricity	→	Room Z: fire bulkhead in ceiling defect
	Room Y: PEN error; voltage is only 138 V	→	External system labeling is missing
	Room X: power socket problem; voltage is only 137 V	→	External system labeling is missing
	Room W: no electricity	→	Room T: Coffee machine not working
	Room U: no electricity	→	Room T: Coffee machine not working

Table 7 shows the same metrics when filtering the rules with the temporal lift. While the number of rules reduces by more than 50% for all models and thresholds, the reduction in the number of hits is considerably lower. Thus, comparing Tables 6 and 7 suggests that the temporal lift can potentially reduce the number of irrelevant rules.

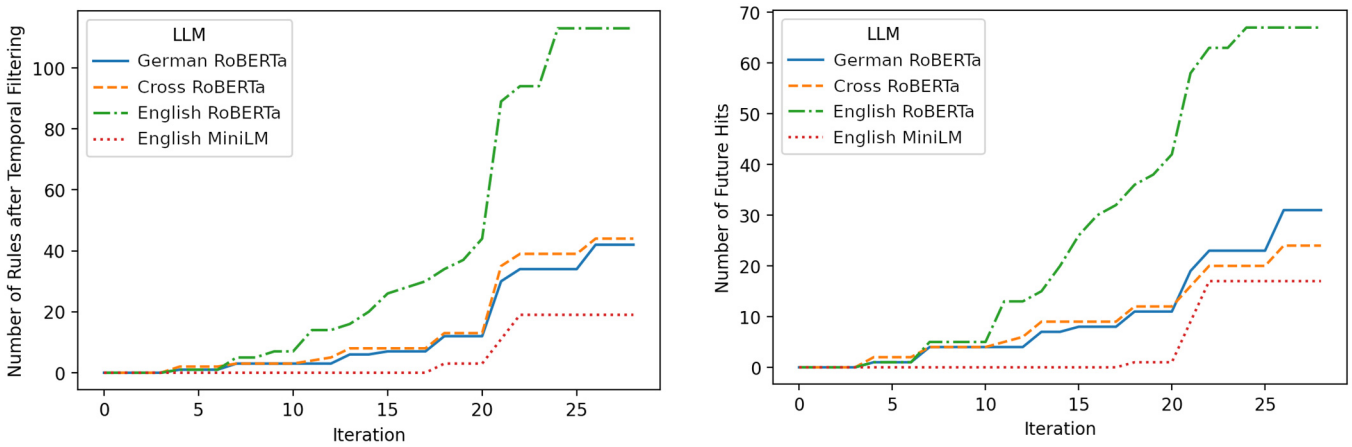
Table 6. Number of Rules and Hits for German RoBERTa, Cross RoBERTa, English RoBERTa, and EnglishMiniLM Using ARM with Semantic Similarity.

Minimum Similarity Threshold	German RoBERTa		Cross RoBERTa		English RoBERTa		English MiniLM	
	Rules	Hits	Rules	Hits	Rules	Hits	Rules	Hits
0.65	104	46	120	48	228	80	54	23
0.7	38	13	22	9	46	21	0	0
0.75	14	5	16	4	4	2	0	0
0.8	0	0	0	0	4	2	0	0

Table 7. Number of Rules and Hits for German RoBERTa, Cross RoBERTa, English RoBERTa, and English MiniLM using ARM with Semantic Similarity and Temporal Lift.

Minimum Similarity Threshold	German RoBERTa		Cross RoBERTa		English RoBERTa		English MiniLM	
	Rules	Hits	Rules	Hits	Rules	Hits	Rules	Hits
0.65	42	31	44	24	113	67	19	17
0.7	15	11	9	7	13	12	0	0
0.75	4	4	6	4	0	0	0	0

The two diagrams in Figure 4a,b shed more light on the interplay between the iteration time (i.e., iteration = the number of maintenance requests divided by n = 5) and the number of rules and hits until this iteration. All rules shown have a minimum similarity of 0.65. Figure 4a shows that almost all ARM runs create their first rules after a few iterations. There is one exception, namely, English MiniLM, which creates its first rule in iteration 17. Figure 4b shows the future hits of a newly created rule in the respective iteration. Since both diagrams run nearly parallel and the right diagram shows the first hits after iteration 3, it seems that the proposed ARM extension can find relevant rules already with little data.



(**a**) (**b**)

Figure 4. Number of (**a**) Rules and (**b**) Future Hits after Temporal Filtering.

As a last analysis considering the rules created using the four LLM sentence transformer-based similarity scores, Table 8 gives an overview of the overlap of rules conducted using the similarity score of the four different LLMs. A rule is an overlap if it is present in both rule sets of two different sentence transformers. Overlaps between the same transformer models are omitted (marked with a dash) and overlaps with empty rule sets are marked as not available (N/A). The comparison utilizes the German RoBERTa similarity score since the wording of the two rules might be similar but not identical (using another LLM for the similarity score results in similar values that differ slightly, mostly ± two percentage points). In addition, it uses a minimum similarity threshold of 0.65 to identify a similar written rule as an overlap. Table 8 confirms that the German RoBERTa and the Cross RoBERTa lead to similar results, as there is a high overlap between both rule sets. Table 5, showing exemplary rules created from both LLMs, already suggested this high overlap. Contrarily, the English RoBERTa produces mostly wholly different rules. The English MiniLM is a mixture of all rules created with the RoBERTa-based LLMs.

Table 8. Overlap of Rules.

LLM	Minimum Similarity	German RoBERTa	Cross RoBERTa	English RoBERTa	English MiniLM
German RoBERTa	0.65	-	0.77	0.05	0.52
Cross RoBERTa	0.65	0.79	-	0.04	0.48
English RoBERTa	0.65	0.13	0.07	-	0.56
English MiniLM	0.65	0.21	0.12	0.13	-
German RoBERTa	0.7	-	1.0	0.04	N/A
Cross RoBERTa	0.7	0.68	-	0.04	N/A
English RoBERTa	0.7	0.16	0.09	-	N/A
English MiniLM	0.7	0.0	0.0	0.0	N/A
German RoBERTa	0.75	-	0.5	0.0	N/A
Cross RoBERTa	0.75	0.57	-	0.0	N/A
English RoBERTa	0.75	0.0	0.0	-	N/A
English MiniLM	0.75	0.0	0.0	0.0	N/A

Besides creating rules with the LLM-based sentence transformer similarity scores, this paper also applies the ARM process with the other four baseline similarity scores obtained from BLEU, METEOR, Rouge-L, and BERTScore. Starting with BLEU, the correlation between the similarity scores obtained from the first baseline model and the LLM-based similarity scores is positive but low (between 30% and 40%). Furthermore, the absolute BLEU values are lower than the other similarity scores. Since setting the minimum similarity threshold of the ARM to a value higher or equal to 0.4 results in no association rules, it necessitates setting it to a lower value (e.g., 0.3 or 0.35). However, this change would put maintenance requests that do not belong together semantically on the same level. For instance, a threshold of 0.35 would mark similarly worded maintenance requests like "Deficiency item [item number] from [external company] report: [maintenance request description]—continuation of maintenance order [order number]" (where only the middle part marked as "[maintenance request description]" changes) as identical. Therefore, basing ARM on BLEU does not seem to produce reliable association rules with high support as intended. Table 4 already indicates the inappropriateness of BLEU for the context of this case study.

Furthermore, the similarity scores obtained from Rouge-L and METEOR are very similar, manifesting in a correlation coefficient of 93% between both. In addition, their correlation with the LLM-based similarity scores is also higher than the ones with BLEU, ranging between 45% and 55%. Since the similarity values have a higher absolute value than BLEU, the ARM results in association rules with a minimum similarity score of up to 60%. However, it seems that the similarity measurement is still problematic for shorter texts. For instance, both mark the maintenance requests "Area [Room X] Defective socket" and "Area [Room Y] socket loose" as similar, while the LLM-based similarity scores marked them as dissimilar. Last, utilizing the BERTScore similarity measure allows the comparison between the previous four LLM sentence transformer-based similarity scores with a second type of LLM-based similarity scores. Indeed, the correlations between both groups are highest compared to the other baseline models, ranging between 50% and 60%. However, one can still observe similar problems of a high similarity score between maintenance requests with similar wording but different meanings. It did not show such flaws when manually screening all similar maintenance requests marked by the four LLM-based sentence transformers. Thus, the sentence-based consideration of the four selected LLM-based sentence transformers fits better in the case study context than a word/tensor-based semantics consideration, ARM using BERTScore with a minimum similarity threshold of

0.85, ARM using Rouge-L and METEOR with a minimum similarity threshold of 0.6, and ARM using German RoBERTa with a minimum similarity threshold of 0.75 results in the same four rules when using the temporal lift filter.

4.3. Comparison of Rules with Human Expert Evaluation

Finally, two human domain experts evaluate the quality of the rules. In this use case, a rule is meaningful if a domain expert estimates a connection between the antecedent and the consequence in a technical manner. The domain experts evaluated 200 pairs of maintenance requests that originate from the different association rules generated by the four LLM-based sentence transformers and a minimum similarity threshold of 65%. This selection allows a retrospective evaluation of all rules created using the sentence transformers.

Table 9 looks into the accepted rate of the created rules after the human domain expert evaluation. This paper defines the acceptance rate as the number of rules the domain experts evaluate as meaningful divided by the number of rules generated using the described ARM process and the four LLM-based sentence transformers. Please refer to Tables 6 and 7 to compare the number of rules per LLM and similarity threshold.

Table 9 reveals that a higher minimum similarity threshold generally increases the acceptance rate. There are only two exceptions where the acceptance rate is 0 with a higher minimum similarity threshold. Both acceptance rates rely on a subset of only four rules to evaluate from (see Tables 6 and 7). This finding indicates that a higher minimum similarity threshold can help generate more meaningful rules. Moreover, the English RoBERTa model obtains the highest acceptance rates for the minimum similarity thresholds of 0.65 and 0.7.

Table 9. Acceptance Rate of Rules after Human Expert Evaluation.

Minimum Similarity Threshold		German RoBERTa		Cross RoBERTa		English RoBERTa		English MiniLM	
	Temporal Lift Filter	No	Yes	No	Yes	No	Yes	No	Yes
0.65		0.15	0.15	0.12	0.14	0.17	0.18	0.04	0.05
0.7		0.26	0.20	0.27	0.33	0.26	0.38		
0.75		0.29	0.00	0.50	0.33	0.00			

Contrary to intuition, the temporal lift filter does not significantly increase the acceptance rate of the rules. However, the temporal lift filter does not negatively harm the number of hits when only considering rules with a positive human domain expert evaluation. Table 10 shows the number of rules and hits for all four LLM-based sentence transformers using a minimum similarity threshold of 0.65.

Table 10. Number of Rules and Hits after Human Expert Evaluation.

		German RoBERTa		Cross RoBERTa		English RoBERTa		English MiniLM	
	Temporal Lift Filter	No	Yes	No	Yes	No	Yes	No	Yes
Rules		18	7	18	7	40	20	2	1
Hits		7	6	10	8	11	9	1	1

As in Table 6, the number of hits is comparable between association rules obtained with and without a temporal filter. However, the temporal lift filter results in only half the number of rules compared to the same ARM process without that filter. Hence, the temporal lift filter further benefits from the human domain expert evaluation and mainly provides rules useful for future maintenance works. This filter minimizes the burden on human domain experts since they need to evaluate fewer rules manually.

5. Discussion

The proposed ARM extension utilizing semantic similarity and temporal lift demonstrates that it is possible to apply ARM on maintenance requests for facility management. One benefit of this approach is that companies can quickly adopt the algorithm on their use cases and facility management software without needing labeled data or sharing business secrets. In addition, relying on pre-trained LLMs helps to overcome the data availability problem in specific sectors like construction [9]. This paper utilizes four different sentence transformer LLMs to quantify the semantic similarity between maintenance requests. The results demonstrate that all LLMs are suitable for ARM. However, even if the similarity function that compares the similarity between two maintenance requests utilizes the cosine similarity metric between two sentence vectors, the different LLMs require different thresholds for the minimum similarity. Furthermore, they also produce partly different rules. Therefore, decision-makers should test which model works well for their use case by manually evaluating a subset of rules.

Since all maintenance requests of this paper's dataset are in German, it serves as a good use case to evaluate the applicability of LLMs to languages other than English. While most LLMs work well in the English language, only a few models support the German language. Conversely, the results show that LLMs considered in this paper (i.e., German RoBERTa and Cross RoBERTa) are suitable for similarity measures. In addition, it is possible to use NLP for machine translation first, i.e., from German to English, as [38] suggests. However, not all similarity models work well in this context. An early test with another language model (i.e., symanto/xlm-roberta-base-snli-mnli-anli-xnli) applied to the English translation produced high similarity scores between most maintenance requests. While increasing the minimum similarity threshold is one possibility to counteract an excessive number of rules, it is questionable whether it is helpful to apply an unsupervised algorithm where changing a threshold to an extreme value like 0.95. However, comparing the four selected LLMs suggests that using the proposed methodology also applies to other languages.

Another finding when comparing the number of rules and hit rates of the four models is that the English MiniLM model only finds rules with a minimum similarity threshold of 0.65, while the two models working on the German language still find rules with a minimum similarity of 0.75. One explanation can be that the MiniLM model cannot quantify the similarity as well as the other models due to its smaller size. The human domain judgment has the lowest correlation to the semantic similarity scores obtained from the MiniLM model, too, suggesting that it is less suitable for this case study than the other models. However, since the presented approach does not fine-tune the LLM, one can easily use larger language models in this context without having performance problems.

Comparing the four sentence transformer LLMs with four other baseline models that operate on a word, i.e., token or n-gram, level shows that the sentence transformers produce more reliable similarity scores. While the sentence transformer models have a higher computational complexity than the n-gram-based scores, their complexity helps outperform the simpler models. In addition, applying the sentence transformers on this paper's dataset did not require excessive computational resources as the transformer models were already pre-trained.

One peculiarity of this paper's dataset is the relatively short text length. Some maintenance requests contain less than five words (see Tables 2 and 5). The short length is a particular challenge for semantic similarity algorithms since short texts are frequently grammatically incorrect [67]. A transformer model assuming two grammatically correct sentences might struggle to extract the suitable domain-specific similarity. Furthermore, a slight variation in the sentence can significantly impact the metric. The second example of Table 2 ("Meter Reading November 2020" and "Meter Reading December 2020") shows that a slight variation (changing "November" to "December") impacts the semantic similarity. In the context of maintenance works, the activity of a meter reading should be similar, independent of the time of the reading. While the transformer model considers complete sentences, the semantic similarity is still high (0.91). The baseline models would mainly

detect that three out of four words are similar, resulting in a similarity of 0.75. Even if the transformer model can handle this example quite well, it might struggle with shorter texts that differ in unimportant details (e.g., room numbers in the same building, abbreviations, names). A reason for this behavior is that comparing the semantic similarity using sentence transformers and cosine similarity treats all dimensions of the sentence vector equally [54], resulting in a higher impact of a slight change. Replacing such named entities or dates with placeholders might work well for particular use cases. However, numbers can have a specific meaning depending on the context, and LLMs might need help to utilize them effectively as features [68]. Since the transformer model is not pre-trained on the respective use case, it cannot always distinguish between important and unimportant details. Thus, companies should carefully monitor the semantic similarity calculation when applying the proposed approach to short texts. Creating explicit features via information extraction and adding them into a decision algorithm, as shown in [68], can be beneficial in particular domains.

This paper's second extension includes a temporal lift as a rule filter. While the original ARM idea is to work on basket data as a transaction where all items belong to one entity, this paper considers a transaction as all maintenance requests from one infrastructure. In contrast to the original setting, in this paper, all requests arrive in a temporal order and not within a short time. The consequent must come temporally after the antecedent to find suitable rules for PM. Otherwise, a rule in the form of A $\rightarrow$ B would be valid even if B breaks before A, as long as it breaks in the same infrastructure. The temporal lift ensures that this temporal logic is the case for all rules to a certain extent. Indeed, the results suggest that the temporal lift helps to filter out irrelevant rules with low hit rates. Fine-tuning the minimum temporal lift threshold might further increase the rules' appropriateness. One concern when considering the temporal nature of maintenance requests is that it might take time between the antecedent(s) and the consequent(s). This time gap might make rules less relevant, especially when some processes change in the meantime. One might use a temporal decay factor for this problem, as we further detail in the future work section of this paper's conclusion.

Nonetheless, to decide the quality of the resulting association rules, one should not simply rely on the hit rate of a rule but present the rule to a domain expert, who can judge whether a rule makes sense in the context of the predictive maintenance company. For example, one rule the algorithm found on the dataset was "No electricity in X' warehouse" $\rightarrow$ "Power sockets in office Y do not work". This rule might be helpful if there is a connection between the warehouse's electricity and the office's electricity (which one might assume since the maintenance request originate from the same power distribution system). However, if both infrastructures are not connected, e.g., in a different city, decision-makers should neglect such a rule by reviewing the rules by a domain expert.

Even though the maximum acceptance rate of the domain expert evaluation in this paper's use case is only 50%, meaning that at least half of the generated rules are meaningless, the practice partner ensured that a few meaningful rules already help optimize the maintenance process and are valuable support. Although some rules sound trivial in the first place, double arrivals by maintenance workers are an essential problem and a substantial cost driver. According to our practice partner, every rule with a hit saves additionally at least 10 to 15 min per request due to the manual need to create a new maintenance request. The CAFM tool can support this and automatically provide helpful information, e.g., the infrastructure manufacturer, possible warranties, service contracts, or available repair parts.

To support decision-makers, an ARM solution embedded in a CAFM tool might not only present the rule and potential further upcoming maintenance requests but also link to recent other, similar maintenance requests. This functionality would also increase the algorithm's transparency to the end users. Even though the algorithm does not entirely automate the maintenance decision-making process, it can help optimize the maintenance processes. In addition, when dealing with a higher number of infrastructures, a company

creates thousands of maintenance requests that a single decision-maker cannot review manually. The presented tool helps to extract the most common associations.

Another concern of applying ARM on maintenance requests is that specific maintenance events are rare. This paper uses a minimum support value of 0.2, meaning that a maintenance event has to occur in at least two power supply systems. However, for even more rare events, one might set the threshold to 0.1 in this paper's technical case study. This lower threshold allows finding all associations independent of whether they occur multiple times. While this low threshold would create an increased number of rules, adding a weighting term considering the respective impact of such a rare event might help only integrate the rules with the highest impact. The limitation section in this paper's conclusions sheds more light on such a weighting term.

One advantage of the ARM approach presented is that it only alters the definition of the support function and adds the semantic similarity logic. While this paper uses a default Apriori implementation to evaluate the approach, researchers and practitioners can easily change the ARM algorithm by simply replacing the default support calculation. This flexibility is an advantage compared to other works that utilize ARM. For instance, [50] presents an approach that integrates semantic similarity into ARM and relies on Apriori to mine association rules. However, their modification only creates rules where the antecedents have a unitary length. In the results presented, this paper mined several rules with multiple maintenance requests in the antecedent and consequent.

Similarly, calculating the temporal lift is also independently possible from the actual ARM methodology, as it solely consists of the modified support function and a candidate list of rules.

An alternative to ARM can be topic modeling, a class of unsupervised algorithms capable of identifying similar maintenance requests belonging to the same topic. Since topic modeling can output a vector for each maintenance request representing a distribution of different topics [56], one can use it as an input for clustering, nearest neighbor analysis, or an alternative for semantic similarity search for rule mining [69]. However, classical topic modeling algorithms do not work well for short texts due to a lack of word co-occurrence within short texts and its neglect of a word's context [70]. There are less prominent topic modeling algorithms that can handle short texts; some use word embeddings, but their performance is dataset-dependent [70]. Topic modeling further requires specifying the number of topics in a corpus beforehand [69], which can be tricky for maintenance decision-makers due to the dynamic nature of maintenance requests. Interpretation of topics is also non-trivial, even for researchers [69]. Finally, the proposed rule-based approach allows finding coherent maintenance requests even if they are topically not connected (antecedent and consequent do not have to be semantically similar or from the same topic). Thus, the proposed ARM method is less restricted and more accessible to apply in a maintenance setting.

6. Conclusions

While more and more data become available for facility management, the sector does not yet exploit all opportunities that data serve. For example, PM is a field that facility management companies rarely consider here [15]. Potential reasons are the lack of structured data and explicit labels. Most PM algorithms are supervised ML and require such input data. However, most corporate data's available information is mainly in text form [9]. Therefore, this paper investigated the opportunity of applying association rule mining as an unsupervised ML method on textual data from maintenance requests. By integrating temporal constraints and the concept of semantic similarity obtained from LLMs into the process of ARM, this paper demonstrated that it is possible to create handy association rules suitable for facility management decision-making.

This paper has several theoretical and practical implications. First, on a theoretical side, this paper contributes to integrating semantic similarity into ARM. Former research on integrating semantic similarity is scarce. While there are a few works, for instance, [50]

and [49], this paper's approach differs in integrating the semantic similarity to be independent of the concrete ARM algorithm. Second, this paper is one of the first works that apply ARM on maintenance requests for facility management and proposes an easy-to-use approach that one can adopt according to individual needs. Third, managers and decision-makers can rely on PM even if no specific labels are available in a corporate context. Leveraging their existing data can be beneficial. This paper's results suggest that a couple of maintenance requests can already help to mine the first meaningful rules. In case structured information becomes available (e.g., cost estimates or priorities), these data can be either used to sharpen the rules (e.g., via priority weighting) or might allow running (semi) supervised ML. Finally, this paper's approach enables facility managers to understand the interdependencies between their infrastructure better. This support helps especially for complex facilities or facility managers who are new to a specific facility. The rules created by ARM are easily interpretable and vivid due to the possibility of comparing similarity scores between maintenance requests.

One limitation of this work is that it demonstrates the methodology only with the default Apriori algorithm. While the literature shows that a range of algorithms outperforms Apriori, e.g., in turn of performance or computation time [33], future work can study the computational performance of this paper's approach, especially on large databases. Another limitation is the focus on only one type of infrastructure on a limited number of maintenance requests. However, future works can apply the proposed approach to different data, e.g., another type of infrastructure. An interesting extension is the transferability of rules from one type of infrastructure to another.

Additionally, this paper only integrates the temporal concept as a temporal lift that can filter out rules that do not match a specific temporal pattern. It does not incorporate longer temporal gaps between the antecedent and the consequences of rules or structural changes that make rules obsolete over time. Future work can improve the mining process so that it might only mine temporal rules or handle out-of-date rules, e.g., by using time decay functions, as in [52]. Finally, this paper only integrates the concept of time as an implied component and not as an integral part [31]. Thus, future research might shed more light on this research stream for ARM in PM.

In addition, even if ARM produces a variety of reasonable rules, facility management cannot proactively maintain every infrastructure that might break according to the constructed rules. Since what can be maintained depends on the available time and repair costs [7], prioritizing what should be maintained might be helpful. The work of [7] proposes a two-stage process. It first mines association rules from the data and then formulates a mathematical optimization problem to create a constraint maintenance policy. Another approach used in the literature is to weigh the association rules, either manually by experts or automatically [51]. In the case of facility management and maintenance requests, such a weight might be potential repair costs or the severity of a failure. However, this information was unavailable for the data used in the simulation. Thus, this paper omits the rule weighting and leaves it open for further research. Consistent with the work of [41], the data for this future evaluation might also originate from manual inputs or automatically. Manual inputs might stem from real-time estimates when writing a maintenance request. Automatic inputs, for instance, can arise from an NLP algorithm that estimates the costs based on the maintenance text and previous repair work where the cost information is available. In addition, this paper conducted the expert evaluation of the rules only retrospectively, i.e., on a complete dataset of maintenance requests. Future work can analyze the entire ARM process iteratively to evaluate it in actual facility maintenance processes. This iterative approach might help find the optimal values for the ARM parameters (i.e., minimum support and temporal lift). Future work might also apply the approach for various use cases as the value might change depending on the respective data.

The proposed ARM approach serves as an input for a range of applications for facility management. Due to its adaptability, future research can utilize the concepts from the presented approach and adapt them according to the specific needs.

Funding: This work was funded by the German Federal Ministry for Economic Affairs and Climate Action (BMWK) as part of the ForeSight project (Grant: 01MK20004J).

Data Availability Statement: Restrictions apply to the availability of these data. Data were obtained from a third party and are confidential due to business secrets.

Acknowledgments: The author would like to thank Keßler Real Estate Solutions GmbH for the inspiring knowledge exchange and the insights from practice.

Conflicts of Interest: The author declares no conflicts of interest. The funders had no role in the design of this study, in the collection, analyses, or interpretation of data, in the writing of the manuscript, or in the decision to publish the results.

References

1. Widodo, A.; Yang, B.-S. Support Vector Machine in Machine Condition Monitoring and Fault Diagnosis. *Mech. Syst. Signal Process.* **2007**, *21*, 2560–2574. [CrossRef]
2. Mobley, R.K. *An Introduction to Predictive Maintenance*; Elsevier: Amsterdam, The Netherlands, 2002; ISBN 978-0-7506-7531-4.
3. Alestra, S.; Bordry, C.; Brand, C.; Burnaev, E.; Erofeev, P.; Papanov, A.; Silveira-Freixo, C. Rare Event Anticipation and Degradation Trending for Aircraft Predictive Maintenance. In Proceedings of the 11th World Congress on Computational Mechanics, WCCM, Barcelona, Spain, 20–25 July 2014; Volume 5, p. 6571.
4. Zhu, H.; Gao, J.; Li, D.; Tang, D. A Web-Based Product Service System for Aerospace Maintenance, Repair and Overhaul Services. *Comput. Ind.* **2012**, *63*, 338–348. [CrossRef]
5. Adu-Amankwa, K.; Attia, A.K.; Janardhanan, M.N.; Patel, I. A Predictive Maintenance Cost Model for CNC SMEs in the Era of Industry 4.0. *Int. J. Adv. Manuf. Technol.* **2019**, *104*, 3567–3587. [CrossRef]
6. Hrnjica, B.; Softic, S. Explainable AI in Manufacturing: A Predictive Maintenance Case Study. In *Advances in Production Management Systems. Towards Smart and Digital Manufacturing*; Lalic, B., Majstorovic, V., Marjanovic, U., von Cieminski, G., Romero, D., Eds.; Springer International Publishing: Cham, Switzerland, 2020; pp. 66–73.
7. Antomarioni, S.; Pisacane, O.; Potena, D.; Bevilacqua, M.; Ciarapica, F.E.; Diamantini, C. A Predictive Association Rule-Based Maintenance Policy to Minimize the Probability of Breakages: Application to an Oil Refinery. *Int. J. Adv. Manuf. Technol.* **2019**, *105*, 3661–3675. [CrossRef]
8. Nadj, M.; Jegadeesan, H.; Maedche, A.; Hoffmann, D.; Erdmann, P. A Situation Awareness Driven Design for Predictive Maintenance Systems: The Case of Oil and Gas Pipeline Operations. In Proceedings of the 24th European Conference on Information Systems, Istanbul, Turkey, 12–15 June 2016; pp. 1–10.
9. Ding, Y.; Ma, J.; Luo, X. Applications of Natural Language Processing in Construction. *Autom. Constr.* **2022**, *136*, 104169. [CrossRef]
10. Wu, C.; Li, X.; Guo, Y.; Wang, J.; Ren, Z.; Wang, M.; Yang, Z. Natural Language Processing for Smart Construction: Current Status and Future Directions. *Autom. Constr.* **2022**, *134*, 104059. [CrossRef]
11. Ghofrani, A.; Nazemi, S.D.; Jafari, M.A. HVAC Load Synchronization in Smart Building Communities. *Sustain. Cities Soc.* **2019**, *51*, 101741. [CrossRef]
12. West, S.R.; Guo, Y.; Wang, X.R.; Wall, J. Automated Fault Detection and Diagnosis of HVAC Subsystems Using Statistical Machine Learning. In Proceedings of the Building Simulation, Sydney, Australia, 14–16 November 2011.
13. Cheng, J.C.; Chen, W.; Tan, Y.; Wang, M. A BIM-Based Decision Support System Framework for Predictive Maintenance Management of Building Facilities. In Proceedings of the 16th International Conference on Computing in Civil and Building Engineering (ICCCBE2016), Osaka, Japan, 6–8 July 2016; pp. 711–718.
14. Ur-Rahman, N.; Harding, J.A. Textual Data Mining for Industrial Knowledge Management and Text Classification: A Business Oriented Approach. *Expert Syst. Appl.* **2012**, *39*, 4729–4739. [CrossRef]
15. Bortolini, R.; Forcada, N. Analysis of Building Maintenance Requests Using a Text Mining Approach: Building Services Evaluation. *Build. Res. Inf.* **2020**, *48*, 207–217. [CrossRef]
16. Folino, F.; Folino, G.; Guarascio, M.; Pontieri, L. Semi-Supervised Discovery of DNN-Based Outcome Predictors from Scarcely-Labeled Process Logs. *Bus. Inf. Syst. Eng.* **2022**, *64*, 729–749. [CrossRef]
17. Chandrasekaran, D.; Mago, V. Evolution of Semantic Similarity—A Survey. *ACM Comput. Surv.* **2021**, *54*, 41:1–41:37. [CrossRef]
18. Hashemian, H.M. State-of-the-Art Predictive Maintenance Techniques. *IEEE Trans. Instrum. Meas.* **2010**, *60*, 226–236. [CrossRef]
19. Virk, S.M.; Muhammad, A.; Martinez-Enriquez, A. Fault Prediction Using Artificial Neural Network and Fuzzy Logic. In Proceedings of the 2008 Seventh Mexican International Conference on Artificial Intelligence, Atizapan de Zaragoza, Mexico, 27 October 2008; pp. 149–154.
20. Carvalho, T.P.; Soares, F.A.A.M.N.; Vita, R.; Francisco, R.d.P.; Basto, J.P.; Alcalá, S.G.S. A Systematic Literature Review of Machine Learning Methods Applied to Predictive Maintenance. *Comput. Ind. Eng.* **2019**, *137*, 106024. [CrossRef]
21. Mo, Y.; Zhao, D.; Du, J.; Syal, M.; Aziz, A.; Li, H. Automated Staff Assignment for Building Maintenance Using Natural Language Processing. *Autom. Constr.* **2020**, *113*, 103150. [CrossRef]

22. Akhbardeh, F.; Desell, T.; Zampieri, M. NLP Tools for Predictive Maintenance Records in MaintNet. In Proceedings of the 1st Conference of the Asia-Pacific Chapter of the Association for Computational Linguistics and the 10th International Joint Conference on Natural Language Processing: System Demonstrations; Wong, D., Kiela, D., Eds.; Association for Computational Linguistics: Suzhou, China, 2020; pp. 26–32.
23. Bhardwaj, A.S.; Veeramani, D.; Zhou, S. Identifying Equipment Health Status from Maintenance Records Using Lexicon Based Unsupervised Sentiment Analysis Adjusted for Negation (LUSAA-N). *Comput. Ind. Eng.* **2023**, *186*, 109693. [CrossRef]
24. Devaney, M.; Ram, A.; Qiu, H.; Lee, J. Preventing Failures by Mining Maintenance Logs with Case-Based Reasoning. In Proceedings of the 59th Meeting of the Society for Machinery Failure Prevention Technology (MFPT-59), Virginia Beach, VA, USA, 18–21 April 2005.
25. Carrasco, J.; López, D.; Aguilera-Martos, I.; García-Gil, D.; Markova, I.; García-Barzana, M.; Arias-Rodil, M.; Luengo, J.; Herrera, F. Anomaly Detection in Predictive Maintenance: A New Evaluation Framework for Temporal Unsupervised Anomaly Detection Algorithms. *Neurocomputing* **2021**, *462*, 440–452. [CrossRef]
26. da Silva Arantes, J.; da Silva Arantes, M.; Fröhlich, H.B.; Siret, L.; Bonnard, R. A Novel Unsupervised Method for Anomaly Detection in Time Series Based on Statistical Features for Industrial Predictive Maintenance. *Int. J. Data Sci. Anal.* **2021**, *12*, 383–404. [CrossRef]
27. Graß, A.; Beecks, C.; Soto, J.A.C. Unsupervised Anomaly Detection in Production Lines. In *Machine Learning for Cyber Physical Systems*; Beyerer, J., Kühnert, C., Niggemann, O., Eds.; Springer: Berlin/Heidelberg, Germany, 2019; pp. 18–25.
28. Agrawal, R.; Imieliński, T.; Swami, A. Mining Association Rules Between Sets of Items in Large Databases. In Proceedings of the 1993 ACM SIGMOD International Conference on Management of Data, Washington, DC, USA, 26–28 May 1993; pp. 207–216.
29. Adamov, A.Z. Mining Term Association Rules from Unstructured Text in Azerbaijani Language. In Proceedings of the 2018 IEEE 12th International Conference on Application of Information and Communication Technologies (AICT), Almaty, Kazakhstan, 17–19 October 2018; pp. 1–4.
30. Diaz-Garcia, J.A.; Ruiz, M.D.; Martin-Bautista, M.J. A Survey on the Use of Association Rules Mining Techniques in Textual Social Media. *Artif. Intell. Rev.* **2023**, *56*, 1175–1200. [CrossRef]
31. Segura-Delgado, A.; Gacto, M.J.; Alcalá, R.; Alcalá-Fdez, J. Temporal Association Rule Mining: An Overview Considering the Time Variable as an Integral or Implied Component. *WIREs Data Min. Knowl. Discov.* **2020**, *10*, e1367. [CrossRef]
32. Agrawal, R.; Srikant, R. Fast Elgorithms for Mining Association Rules. In Proceedings of the 20th International Conference on Very Large Data Bases (VLDB), Santiago, Chile, 12–15 September 1994; Volume 1215, pp. 487–499.
33. Kumbhare, T.A.; Chobe, S.V. An Overview of Association Rule Mining Algorithms. *Int. J. Comput. Sci. Inf. Technol.* **2014**, *5*, 927–930.
34. Kireev, V.S.; Guseva, A.I.; Bochkaryov, P.V.; Kuznetsov, I.A.; Filippov, S.A. Association Rules Mining for Predictive Analytics in IoT Cloud System. In *Biologically Inspired Cognitive Architectures 2018*; Samsonovich, A.V., Ed.; Springer International Publishing: Cham, Switzerland, 2019; pp. 107–112.
35. Jurafsky, D.; Martin, J.H. *Speech and Language Processing: An Introduction to Natural Language Processing, Computational Linguistics, and Speech Recognition*; Prentice Hall: Upper Saddle River, NJ, USA, 2008; ISBN 0-13-095069-6.
36. Luhn, H.P. A Statistical Approach to Mechanized Encoding and Searching of Literary Information. *IBM J. Res. Dev.* **1957**, *1*, 309–317. [CrossRef]
37. Turian, J.; Ratinov, L.; Bengio, Y. Word Representations: A Simple and General Method for Semi-Supervised Learning. In Proceedings of the 48th Annual Meeting of the Association for Computational Linguistics, Uppsala, Sweden, 11–16 July 2010; pp. 384–394.
38. Hemalatha, B.; Velmurugan, T. Direct-Indirect Association Rule Mining for Online Shopping Customer Data Using Natural Language Processing. *Int. J. Recent Technol. Eng.* **2019**, *8*, 2277–3878. [CrossRef]
39. Ren, S.; Li, Z.; Wang, H.; Li, Y.; Shen, K.; Cheng, S. NEARM: Natural Language Enhanced Association Rules Mining. In Proceedings of the 2018 IEEE International Conference on Data Mining Workshops (ICDMW), Singapore, 17–20 November 2018; pp. 438–445.
40. Lakshmi, K.S.; Vadivu, G. Extracting Association Rules from Medical Health Records Using Multi-Criteria Decision Analysis. *Proc. Comput. Sci.* **2017**, *115*, 290–295. [CrossRef]
41. Edwards, B.; Zatorsky, M.; Nayak, R. Clustering and Classification of Maintenance Logs Using Text Data Mining. In Proceedings of the 7th Australasian Data Mining Conference, Glenelg, SA, Australia, 27–28 November 2008; Volume 87, pp. 193–199.
42. Usuga-Cadavid, J.P.; Lamouri, S.; Grabot, B.; Fortin, A. Using Deep Learning to Value Free-Form Text Data for Predictive Maintenance. *Int. J. Prod. Res.* **2022**, *60*, 4548–4575. [CrossRef]
43. Devlin, J.; Chang, M.-W.; Lee, K.; Toutanova, K. BERT: Pre-Training of Deep Bidirectional Transformers for Language Understanding. In Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Minneapolis, MN, USA, 2–7 June 2019; Association for Computational Linguistics: Minneapolis, MN, USA, 2019; pp. 4171–4186.
44. Vaswani, A.; Shazeer, N.; Parmar, N.; Uszkoreit, J.; Jones, L.; Gomez, A.N.; Kaiser, Ł.; Polosukhin, I. Attention Is All You Need. *Adv. Neural Inf. Process. Syst.* **2017**, *30*, 5998–6008.
45. OpenAI GPT-4 Technical Report. *arXiv* **2023**, arXiv:2303.08774.

46. Liu, Y.; Ott, M.; Goyal, N.; Du, J.; Joshi, M.; Chen, D.; Levy, O.; Lewis, M.; Zettlemoyer, L.; Stoyanov, V. RoBERTa: A Robustly Optimized BERT Pretraining Approach. *arXiv* **2019**, arXiv:1907.11692.
47. Wang, W.; Wei, F.; Dong, L.; Bao, H.; Yang, N.; Zhou, M. MiniLM: Deep Self-Attention Distillation for Task-Agnostic Compression of Pre-Trained Transformers. In Proceedings of the Advances in Neural Information Processing Systems 34: Annual Conference on Neural Information Processing Systems 2020, NeurIPS 2020, Vancouver, BC, Canada, 6–12 December 2020.
48. Kaluarachchi, A.C.; Varde, A.S.; Bedathur, S.; Weikum, G.; Peng, J.; Feldman, A. Incorporating Terminology Evolution for Query Translation in Text Retrieval with Association Rules. In Proceedings of the 9th ACM International Conference on Information and Knowledge Management, New York, NY, USA, 6–11 November 2010; Association for Computing Machinery: New York, NY, USA, 2010; pp. 1789–1792.
49. Zeng, A.P.; Liu, D.; Chen, H.M. An Improved Apriori Algorithm Based on Similarity. *Adv. Mater. Res.* **2012**, *532–533*, 1825–1829.
50. Keith Norambuena, B.; Villegas, C. An Extension to Association Rules Using a Similarity-Based Approach in Semantic Cector Spaces. *Intell. Data Anal.* **2019**, *23*, 587–607. [CrossRef]
51. Sammouri, W.; Côme, E.; Oukhellou, L.; Aknin, P. Mining Floating Train Data Sequences for Temporal Association Rules within a Predictive Maintenance Framework. In *Advances in Data Mining. Applications and Theoretical Aspects*; Perner, P., Ed.; Springer: Berlin/Heidelberg, Germany, 2013; pp. 112–126.
52. Han, Y.; Yu, D.; Yin, C.; Zhao, Q. Temporal Association Rule Mining and Updating and Their Application to Blast Furnace in the Steel Industry. *Comput. Intell. Neurosci.* **2020**, *2020*, 7467213. [CrossRef]
53. Li, Y.; McLean, D.; Bandar, Z.A.; O'Shea, J.D.; Crockett, K. Sentence Similarity Based on Semantic Nets and Corpus Statistics. *IEEE Trans. Knowl. Data Eng.* **2006**, *18*, 1138–1150. [CrossRef]
54. Reimers, N.; Gurevych, I. Sentence-BERT: Sentence Embeddings Using Siamese BERT-Networks. *arXiv* **2019**, arXiv:1908.10084.
55. Reimers, N.; Beyer, P.; Gurevych, I. Task-Oriented Intrinsic Evaluation of Semantic Textual Similarity. In Proceedings of the 26th International Conference on Computational Linguistics: Technical Papers, Osaka, Japan, 11–16 December 2016; pp. 87–96.
56. Spasic, I.; Button, K. Patient Triage by Topic Modeling of Referral Letters: Feasibility Study. *JMIR Med. Inform.* **2020**, *8*, e21252. [CrossRef] [PubMed]
57. Lee, M.D.; Pincombe, B.; Welsh, M. An Empirical Evaluation of Models of Text Document Similarity. In Proceedings of the Annual Meeting of the Cognitive Science Society, Stresa, Italy, 21–23 July 2005; Volume 27.
58. Agirre, E.; Banea, C.; Cer, D.; Diab, M.; Gonzalez Agirre, A.; Mihalcea, R.; Rigau Claramunt, G.; Wiebe, J. Semeval-2016 Task 1: Semantic Textual Similarity, Monolingual and Cross-Lingual Evaluation. In Proceedings of the 10th International Workshop on Semantic Evaluation, San Diego, CA, USA, 16–17 June 2016; ACL (Association for Computational Linguistics): San Diego, CA, USA, 2016; pp. 497–511.
59. Reimers, N.; Freire, P.; Becquin, G.; Espejel, O.; Gante, J. Sentence-Transformers/All-MiniLM-L6-v2 Hugging Face. Available online: https://huggingface.co/sentence-transformers/all-MiniLM-L6-v2 (accessed on 2 June 2023).
60. May, P. T-Systems-Onsite/Cross-En-de-Roberta-Sentence-Transformer Hugging Face. Available online: https://huggingface.co/T-Systems-onsite/cross-en-de-roberta-sentence-transformer (accessed on 2 June 2023).
61. May, P. T-Systems-Onsite/German-Roberta-Sentence-Transformer-v2 Hugging Face. Available online: https://huggingface.co/T-Systems-onsite/german-roberta-sentence-transformer-v2 (accessed on 2 June 2023).
62. Papineni, K.; Roukos, S.; Ward, T.; Zhu, W.-J. Bleu: A Method for Automatic Evaluation of Machine Translation. In Proceedings of the 40th Annual Meeting of the Association for Computational Linguistics, Philadelphia, PA, USA, 7–12 July 2002; Isabelle, P., Charniak, E., Lin, D., Eds.; Association for Computational Linguistics: Philadelphia, PA, USA, 2002; pp. 311–318.
63. Lin, C.-Y. ROUGE: A Package for Automatic Evaluation of Summaries. In *Text Summarization Branches Out*; Association for Computational Linguistics: Barcelona, Spain, 2004; pp. 74–81.
64. Banerjee, S.; Lavie, A. METEOR: An Automatic Metric for MT Evaluation with Improved Correlation with Human Judgments. In Proceedings of the ACL Workshop on Intrinsic and Extrinsic Evaluation Measures for Machine Translation and/or Summarization, Ann Arbor, MI, USA, 25–30 June 2005; Goldstein, J., Lavie, A., Lin, C.-Y., Voss, C., Eds.; Association for Computational Linguistics: Ann Arbor, MI, USA, 2005; pp. 65–72.
65. Zhang, T.; Kishore, V.; Wu, F.; Weinberger, K.Q.; Artzi, Y. BERTScore: Evaluating Text Generation with BERT. *arXiv* **2020**, arXiv:1904.09675.
66. Agirre, E.; Cer, D.; Diab, M.; Gonzalez-Agirre, A. *Semeval-2012 Task 6: A Pilot on Semantic Textual Similarity, Proceedings of the SEM 2012: The First Joint Conference on Lexical and Computational Semantics—Volume 1: Proceedings of the Main Conference and the Shared Task, and Volume 2: Proceedings of the Sixth International Workshop on Semantic Evaluation (SemEval 2012), Montréal, QC, Canada, 7–8 June 2012*; Association for Computational Linguistics: Kerrville, TX, USA, 2012; pp. 385–393.
67. O'Shea, J.; Bandar, Z.; Crockett, K.; McLean, D. A Comparative Study of Two Short Text Semantic Similarity Measures. In *Lecture Notes in Computer Science*; Nguyen, N.T., Jo, G.S., Howlett, R.J., Jain, L.C., Eds.; Springer: Berlin/Heidelberg, Germany, 2008; Volume 4953, pp. 172–181.
68. Miok, K.; Corcoran, P.; Spasić, I. The Value of Numbers in Clinical Text Classification. *Mach. Learn. Knowl. Extr.* **2023**, *5*, 746–762. [CrossRef]

69. Laureate, C.D.P.; Buntine, W.; Linger, H. A Systematic Review of the Use of Topic Models for Short Text Social Media Analysis. *Artif. Intell. Rev.* **2023**, *56*, 14223–14255. [CrossRef] [PubMed]
70. Qiang, J.; Qian, Z.; Li, Y.; Yuan, Y.; Wu, X. Short Text Topic Modeling Techniques, Applications, and Performance: A Survey. *IEEE Trans. Knowl. Data Eng.* **2022**, *34*, 1427–1445. [CrossRef]