mathematics

Article

Predictive Maintenance with Linguistic Text Mining

Alberto Postiglione [1,*,†] and Mario Monteleone [2,†]

[1] Department of Business Science and Management & Innovation Systems, University of Salerno, Via San Giovanni Paolo II, 132, 84084 Fisciano, Italy
[2] Department of Political and Communication Sciences, University of Salerno, Via San Giovanni Paolo II, 132, 84084 Fisciano, Italy; mmonteleone@unisa.it
* Correspondence: ap@unisa.it
† These authors contributed equally to this work.

Abstract: The escalating intricacy of industrial systems necessitates strategies for augmenting the reliability and efficiency of industrial machinery to curtail downtime. In such a context, predictive maintenance (PdM) has surfaced as a pivotal strategy. The amalgamation of cyber-physical systems, IoT devices, and real-time data analytics, emblematic of Industry 4.0, proffers novel avenues to refine maintenance of production equipment from both technical and managerial standpoints, serving as a supportive technology to enhance the precision and efficacy of predictive maintenance. This paper presents an innovative approach that melds text mining techniques with the cyber-physical infrastructure of a manufacturing sector. The aim is to improve the precision and promptness of predictive maintenance within industrial settings. The text mining framework is designed to sift through extensive log files containing data on the status of operational parameters. These datasets encompass information generated by sensors or computed by the control system throughout the production process execution. The algorithm aids in forecasting potential equipment failures, thereby curtailing maintenance costs and fortifying overall system resilience. Furthermore, we substantiate the efficacy of our approach through a case study involving a real-world industrial machine. This research contributes to the progression of predictive maintenance strategies by leveraging the wealth of textual information available within industrial environments, ultimately bolstering equipment reliability and operational efficiency.

Keywords: predictive maintenance; text mining; finite automata; natural language processing; log message; cyber-physical systems

MSC: 68T50; 90B25

Citation: Postiglione, A.; Monteleone, M. Predictive Maintenance with Linguistic Text Mining. *Mathematics* **2024**, *12*, 1089. https://doi.org/10.3390/math12071089

Academic Editors: Qing Li, Bing Si and Renjie Hu

Received: 20 February 2024
Revised: 30 March 2024
Accepted: 2 April 2024
Published: 4 April 2024

1. Introduction

Industrial machinery forms the cornerstone of modern manufacturing processes, facilitating large-scale production and promoting economic development in several sectors. However, the reliability and efficiency of industrial machinery face ongoing challenges due to factors such as wear and tear, unexpected breakdowns, and the need for routine maintenance. In response, the field of industrial machine maintenance has undergone substantial evolution, with a shift towards proactive, data-driven strategies aimed at minimizing downtime, optimizing performance, and containing maintenance costs.

The issue of maintaining production equipment is fundamental for effectively managing the production processes of many manufacturing industries, prompting the development of various maintenance approaches within the manufacturing sector.

Conventional maintenance strategies, such as reactive maintenance and preventive maintenance based on fixed schedules, often fall short in addressing the dynamic and intricate nature of modern industrial systems. Reactive maintenance, entailing responses to equipment failures as they occur, can result in costly unplanned downtime and production losses. Conversely, preventive maintenance strategies, while effective in reducing the likelihood of failures, may lead to unnecessary maintenance activities and increased operational expenses.

In recent years, predictive maintenance (PdM) has emerged as a promising alternative, harnessing advanced analytics, sensor technologies, and machine-learning algorithms to forecast equipment failures before they happen. By continuously monitoring machine health in real time and analyzing historical performance data, predictive maintenance enables proactive interventions, such as timely repairs and component replacements, thereby mitigating the risk of unplanned downtime and optimizing asset utilization.

As Industry 4.0 unfolds, traditional maintenance methodologies are being reexamined to harness the potential of emerging technologies, thereby improving their effectiveness. These new technologies afford intervention opportunities unattainable with traditional approaches. As Industry 4.0 emerges, conventional maintenance practices are being reevaluated to leverage advancements in technology, thereby boosting their effectiveness. These advancements present new opportunities for intervention that were previously inaccessible through traditional methods.

Despite the potential of predictive maintenance, implementing effective PdM programs in industrial settings remains a complex and multifaceted challenge. Key considerations encompass the selection of appropriate sensor technologies, the development of robust predictive models, and the seamless integration of predictive maintenance into existing maintenance workflows and organizational processes.

Building upon these principles, this study aims to utilize text mining methodologies for predictive maintenance within the framework of a cyber-physical production system (CPPS). During the manufacturing process, the CPPS gathers industrial data from various sources such as machine tools and other equipment. These datasets can be examined using computational linguistics techniques, particularly those related to tagged terminological multiword units. The proposed system begins with the analysis of extensive textual data generated by a machine tool, subsequently furnishing insights into the machine's health status and the potential implications of evaluated conditions on its future health state.

The paper is organized as follows:

Section 2 reviews relevant literature about maintenance and maintenance approaches adopted in manufacturing industries, predictive maintenance approaches, text mining technology, and the scientific foundations of our research.

In Section 3, the text mining approach that we propose for predictive maintenance in Industry 4.0 scenarios is described, while in Section 4 we analyze its performances and architecture.

In Section 5, we compare our proposal with other approaches for automatic predictive maintenance of industrial machines, considering both efficiency and computational performance.

In Section 6, we present the case study, based on real data from an Italian industry operating in the automotive sector. In Section 7 are reported some future research directions, while in Section 8 there are some conclusions.

2. Literature Review

2.1. Maintenance

Maintenance is defined as

"The combination of all technical and administrative actions, including supervision, which ensure that a system is in its required functioning state" [1].

It encompasses a variety of tasks directed towards restoring an asset to a condition where it can efficiently fulfill its intended functions. The maintenance of production equipment holds paramount importance for managing the production processes of manufacturing industries effectively. The primary objective of maintenance is to minimize production downtimes, ultimately striving for zero breakdowns. The economic significance of maintenance has spurred scholarly interest in investigating various aspects related to maintaining physical resources, encompassing techniques and management methods [2].

The emergence of Industry 4.0, grounded in a suite of technologies such as cyber-physical systems, Industrial Internet of Things (IIoT), and big data analytics [3], has spurred the evolution of maintenance strategies. These technologies enable seamless integration of data from diverse sources, including sensors, actuators, and operational databases, facilitating comprehensive condition monitoring and predictive analytics. Recent research in industrial machinery maintenance has delved into the utilization of Industry 4.0 technologies [4,5], particularly concerning predictive maintenance approaches ([6–8]). Leveraging cyber-physical systems as the technological backbone, predictive approaches based on text mining [9] and machine learning [10] have been explored to enhance the performance of the maintenance process.

This paper is a heavily revised and expanded version of [9].

2.2. Maintenance Approaches

The process of equipment maintenance is typically initiated either as a planned intervention or in response to an emergency request stemming from equipment failure. To aid practitioners in understanding the complexity of the maintenance system, conceptual models have been proposed based on **unplanned maintenance** (UM) and **planned maintenance** (PM) activities ([11,12]). The model represented in Figure 1 illustrates the classification scheme adopted in this paper; see [13] for an updated classification scheme, which presents, in broad terms but with a greater level of detail, the main distinctions reported in Figure 1 between unplanned maintenance and planned maintenance and, within the planned maintenance category, between preventive and predictive maintenance.

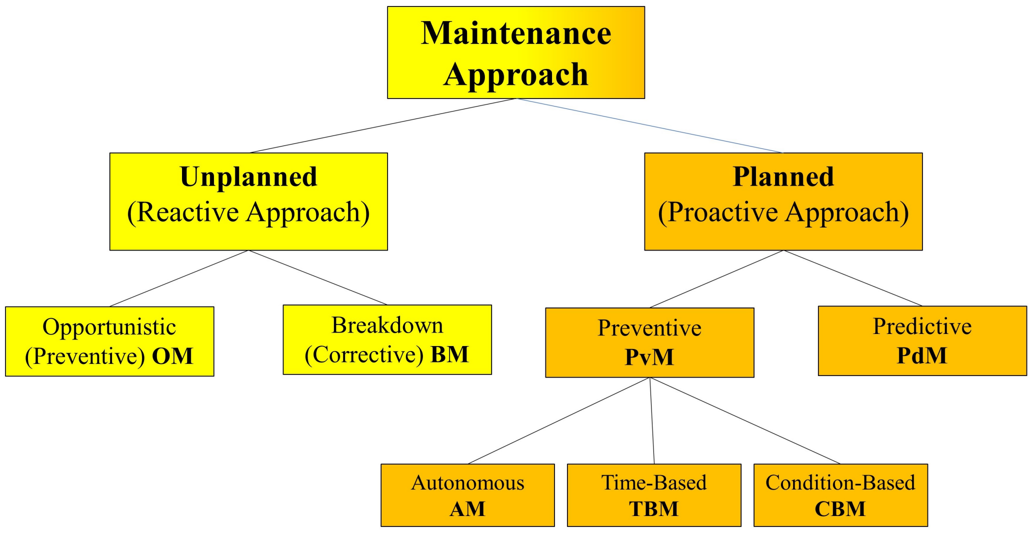

Maintenance Approach
Unplanned (Reactive Approach)
Planned (Proactive Approach)
Opportunistic (Preventive) OM
Breakdown (Corrective) BM
Preventive PvM
Predictive PdM
Autonomous AM
Time-Based TBM
Condition-Based CBM

Figure 1. Classification of approaches to machine tool maintenance.

The most basic method of equipment maintenance is **unplanned maintenance** (UM) (see Section 2.2.1 and refer to the yellow left branch of Figure 1). **Planned maintenance** (PM) (see the right orange branch of Figure 1) includes various aspects of planned maintenance interventions, divided into two main strategies: ***preventive maintenance*** (PvM) (see Section 2.2.2) and ***predictive maintenance*** (PdM) (see Section 2.2.3).

2.2.1. Unplanned Maintenance

UM has a reactive behavior and encompasses two main strategies:

- ***Opportunistic maintenance*** (OM), which strives to turn a failure event into an opportunity for improvement; for instance, if a production line stops because of a machine failure, the maintenance team can conduct inspections on other machines in the line while the repair is underway. A classification of opportunistic maintenance models can be found in Di Dio et al. ([14]).

- ***Breakdown maintenance*** (BM), also known as corrective maintenance (CM) or run-to-failure or reactive maintenance, refers to maintenance activities performed on industrial machinery or equipment to identify and rectify the causes of failures in a failed system (Wang et al., 2014). Unlike preventive maintenance, which involves scheduled inspections and servicing to prevent failures, corrective maintenance occurs after a breakdown has occurred and aims to restore the equipment to its normal operating condition. In Wang et al. ([15]), a comprehensive corrective maintenance scheme for engineering equipment is proposed.

2.2.2. Planned Maintenance: Preventive Maintenance (PvM)

A primary type of PM is preventive maintenance (PvM) [1], which aims to reduce the probability of occurrence of specific failure modes or to detect hidden failures. PvM activities are geared towards repairing equipment, such as replacing key components, to prevent breakdowns. PvM involves monitoring the machine state to identify unsatisfactory conditions necessitating maintenance intervention. Three specialized maintenance techniques are commonly employed within PvM: autonomous, time-based, and condition-based, all aimed at predicting equipment failures to apply corrective maintenance promptly.

- The simplest form of PvM is ***autonomous maintenance*** (AM), defined by regular tasks involving equipment monitoring, adjustments, and basic maintenance by machine operators, and the process is commonly referred to as routine activity. A simplified approach to autonomous maintenance is outlined in a study by Gajdzik et al. [16]. Additionally, preventive maintenance (PvM) can utilize time-based methods.
- ***Time-based maintenance*** (TBM) is a maintenance strategy based on fixed time intervals, where predefined maintenance tasks are performed regularly according to predetermined schedules. Maintenance decisions in TBM are triggered solely by time, with preventive repairs determined through failure time analysis ([17]). TBM strives to mitigate system degradation by executing preventive maintenance tasks while the system remains operational ([18]).
- Another preventive technique is ***condition-based maintenance*** (CBM) [19], which monitors the actual condition of assets to inform maintenance decisions. CBM prescribes maintenance based on physical variables indicating decreasing performance or imminent failure. Introduced as an alternative to TBM, CBM addresses its limitations and serves as a decision-making method covering equipment condition evaluation and maintenance decision processes ([20]).

2.2.3. Planned Maintenance: Predictive Maintenance (PdM)

The second type of PM is predictive maintenance (PdM), which employs specialized techniques to analyze equipment states during operation, predicting when equipment will fail. This predictive capability allows for timely maintenance interventions to prevent breakdowns. PdM can be viewed as a specific form of CBM ([21]). Both predictive maintenance (PdM) and condition-based maintenance (CBM) share a proactive approach to ensuring machine health, employing innovative maintenance methodologies, like the Internet of Things, data analysis, and machine learning. However, they diverge significantly in their methodologies. CBM operates on the basis of real-time machine condition monitoring: a single observation can suffice to identify substantial deviations from expected operating modes, guiding maintenance decisions promptly. Conversely, PdM techniques center around repeated observations of the overall machine health, accumulating a historical record of past behaviors. These observations are then leveraged to derive predictions regarding future equipment failures, facilitating a proactive maintenance approach.

This distinct perspective has profound implications for PdM, especially concerning the application of Industry 4.0 technologies to address maintenance challenges. In Figure 1, PdM occupies a different level compared with AM, TBM, and CBM, because PdM focuses on predicting future failures and requires maintaining a history of past equipment behavior.

2.3. Predictive Maintenance Approaches

Predictive maintenance (PdM) is rooted in the recognition that the majority of failures that occur following a degradation process do not occur suddenly or instantaneously. The goal of predictive maintenance is to reduce uncertainty in maintenance activities by identifying issues before potential damage occurs. Choosing the appropriate method for predictive maintenance in industrial machinery depends on several factors, including machinery requirements, industry type, available resources, and desired sophistication level. Predictive maintenance systems vary in complexity, capability, and application, but share the common goal of minimizing downtime, reducing maintenance costs, and optimizing equipment performance in industrial settings. Efficient methods for predictive maintenance include:

Machine learning-based (or data-driven) (ML) systems [22–25], which utilize historical data, sensor readings, and other parameters to predict equipment failures and recommend maintenance actions, optimizing maintenance schedules. ML systems, being machine learning-based, may require a pre-training phase but do not analyze textual semantics; and *condition monitoring systems (CMSs)* [26–28], which continuously monitor machinery condition by collecting and analyzing real-time sensor data.

Further methods for predictive maintenance are much more hardware driven, as they base their analysis predominantly on the acquisition of some parameters related to the machine's hardware, analyzing the cumulative damage and degradation of the components. These methods include *vibration analysis systems* [29–31], *infrared thermography systems* [32], *oil analysis systems* [33], *ultrasonic monitoring systems* [34], and *prognostics and health management (PHM) systems* [35–37], which integrate data analysis with physics-based models to predict equipment health and estimate the remaining useful life. These methods detect machine deterioration when it is already occurring, albeit at an early stage. Therefore, they may not be suitable for detecting potential problems before they arise. Combining these technologies with CMS and ML methods, which offer predictive and early detection capabilities, can provide the most comprehensive approach to predictive maintenance.

2.4. Text Mining

Text mining [38–44], also referred to as text analytics or text data mining, constitutes an interdisciplinary domain amalgamating various technologies such as natural language processing (NLP), statistical methods, computer science, and machine learning. Its primary objective is to automatically extract knowledge and meaningful information from unstructured or semi-structured natural language textual documents. Unlike conventional web searches where users typically pursue known information, text mining delves into the discovery of new, obscure data that may not be readily apparent through individual perusal of existing text documents. Positioned as a specialized offshoot of data mining [42,45,46], text mining narrows its focus specifically on the analysis of textual documents, distinguishing itself from the broader spectrum of techniques employed in data mining across diverse data types.

Its application extends to critical domains including social media analytics, search engines, email and document filtering, product recommendation analysis, fraud detection, and customer relationship management systems. Within these domains, text mining serves various purposes such as feature extraction and document classification, summarization, topic modeling, trend analysis, named entity recognition, opinion mining, and sentiment analysis.

In the field of natural language processing (NLP), *text annotation* denotes the procedure of appending pertinent information or labels to textual data. The objective is to augment the comprehension of the text by furnishing supplementary context, structure, or significance. The quality of text annotation is critically important. The creation of annotated datasets can be accomplished manually by human annotators, given that it frequently demands industry expertise to guarantee precise and meaningful labeling. However, automation utilizing specific tools or pre-existing datasets is also feasible.

Text mining plays a fundamental role in research [47,48] and medicine [49–54], and has important applications in several other fields, such as psychiatry [55], risk management [56], financial domains [57], finance [58], service management [59], social networks [60], social media [61], education and training [62], policy-making [63], and agriculture [64], among others.

In [43], a text mining system is classified into pre-processing, text representation, and four operational phases:

1. Pre-processing;
2. Text representation;
3. Dimensionality reduction;
4. Features extraction;
5. Document classification;
6. Evaluation.

In the first two phases, the central aim is to transform unstructured text into structured features, with the goal of standardizing the input text into a uniform format to make it suitable for analysis. This entails cleaning and preparing raw text data, removing irrelevant or unnecessary characters and words, converting text to lowercase (or uppercase), handling special characters, and tokenizing the document, which involves subdivision of the text into single words or sentences. Figure 2 in [41] visually illustrates the interactions among the four operational phases (phases 3–6), providing a comprehensive depiction of the subsequent stages of the text mining process.

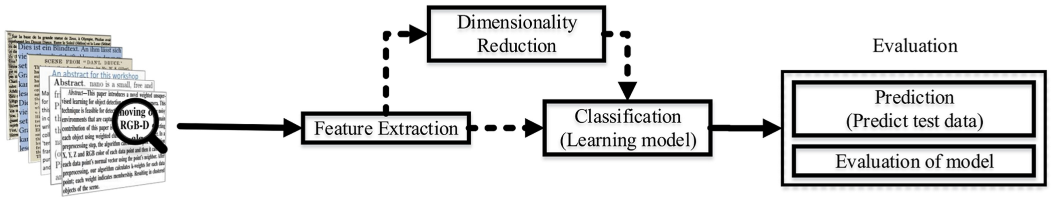


Figure 2. Text mining system phases (from [41]).

2.5. Scientific Foundations of the Research

To efficiently analyze log files emitted by industrial machines and predict potential future malfunctions, we utilize AUTOMETA, a text mining system introduced in prior works ([65,66]). AUTOMETA employs finite automata and a coherent formalization of natural language, focusing on the universal concept of *units of meaning*.

AUTOMETA identifies multiword units within natural language texts, such as log messages, associating an alert level with the message cluster. It operates by mapping digitized text to an ontology covering all possible log message types. The algorithm, customized for processing lengthy log messages comprised of varying word counts, operates by reading input text character by character at runtime. Simultaneously, it identifies all multiword units, even if they partially or fully overlap.

The system's foundation stems from decades of joint research in computer science and computational linguistics, marking a significant evolution from its predecessor, CATALOGA [67]. Beginning in 2021, a comprehensive process of redefinition, redesign, and rewriting of CATALOGA culminated in the development of the AUTOMETA system prototype. While retaining the general approach and some ontologies (subject to review), every other aspect underwent thorough redesign, including technology, algorithms, data structures, source code, and interface. The scientific underpinnings of AUTOMETA trace back to studies on Lexicon-Grammar (LG) ([68–72]), mainly those dealing with the creation and use of terminological electronic dictionaries, and finite automata-based string matching ([73–79]).

3. System Description

The proposed methodology focuses on condition monitoring systems (CMSs) and entails the ongoing analysis of log messages generated continuously by a data source connected to industrial machinery.

The entire process is schematized in Figure 3.

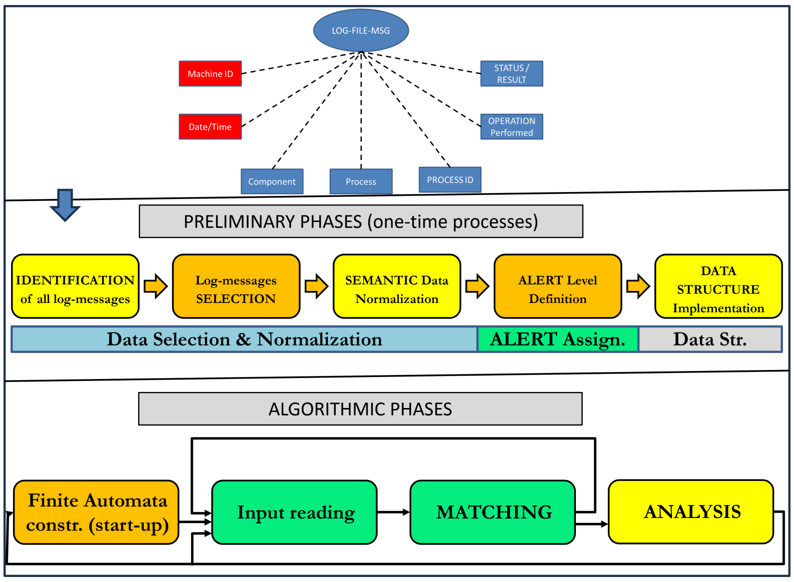


Figure 3. System process diagram.

3.1. LOG Messages

Log messages originating from industrial machines represent a sequential compilation of contextual data integral to the machining process, offering invaluable real-time or near-real-time insights for monitoring, analyzing, and optimizing machine performance and operational health.

Generated by the machine's control system and pertinent auxiliary data sources such as sensors and auxiliary equipment, this information primarily pertains to routine machine operations, encompassing operational status updates, performance metrics, and occasional asynchronous event notifications. These events may span from critical machine failures to minor anomalies or slight deviations from the expected operational parameters.

By amalgamating and correlating these log messages, a comprehensive overview of the machine's functionality can be constructed. Such a holistic approach fosters a deeper comprehension of the intricate interplays and dependencies within the industrial ecosystem. Moreover, log messages play a central role in predictive maintenance strategies, empowering proactive intervention to prevent potential downtimes and enhance operational efficiency.

Log messages are stored in text files, with each line representing a single message entry. The content of these log messages may vary depending on the context and objectives of the logging system. Nonetheless, even with potential variations in format or the arrangement of elementary data, a typical log message from an industrial machine comprises several common types of information:

Machine ID:	A unique identifier denotes the machine from which the message originated.
Date/time:	The date and time at which the event or message was generated.
Component:	Text strings are employed to logically group messages, where components may represent various subsystems or components of the machine, such as "Controller", "Database Server", or "Motor Drive".

Process:	Identification of the running process that generated the message, typically including the full path and filename, and sometimes the module name.
Process ID (PID):	An integer that uniquely identifies the running process.
Operation performed:	Details about the specific operation or event that occurred. This could include commands executed, sensor readings, status updates, or any other relevant information about the machine's activities.
Result or status:	The outcome or status of the operation performed. This could indicate success, failure, warnings, errors, or other relevant states. The result may be presented as a free-form text string or a predefined code indicating the status.

Each message is uniquely identified by its date/time stamp, considering that the same message can be issued multiple times by the same machine, albeit never concurrently. To integrate messages from multiple data sources related to the same machine, assigning a unique ID to each data source could facilitate correlation.

3.2. System Preliminary Phase

The preliminary phase comprises three distinct one-time sub-phases for each of the data sources associated with the industrial machine. These sub-phases are conducted either during the initial installation process or following significant modifications to the data source infrastructure. Some of the sub-steps of the preliminary phase must be conducted in close collaboration with the company's machinery experts.

In the first sub-step, the task involves identifying, selecting, and normalizing all relevant log messages from the individual data source. Subsequently, each identified message is associated with its corresponding alert level. This association aids the algorithm in proactively identifying potential machine malfunctions. Finally, the third sub-step entails constructing the computer data structure utilized by the algorithm.

The preliminary process is schematized in the central part of Figure 3; in orange are the steps that must be carried out in close collaboration with machine industry experts.

3.2.1. Preliminary Sub-Step 1: Data Selection and Normalization

The process of identifying and normalizing all relevant log messages emitted by a data source associated with an industrial machine comprises the following stages:

1. Identification of all message types: this initial stage involves identifying every possible type of message that can be generated.
2. Relevance-based log messages selection. Each selected message type should contain only relevant information and will constitute the initial segment of every element within the ontology. This stage is carried out in strict collaboration with the machine's domain experts.
3. Semantic data normalization: in this stage, the focus is on aligning the data with standardized semantics and formats.

3.2.2. Preliminary Sub-Step 2: Assigning Alert Level to Each Log Message

This stage involves developing a linguistic model that assigns an alert level, based on a chromatic scale, with each message obtained from the previous step. This stage is carried out in strict collaboration with the machine's domain experts. The defined levels are as follows:

White Level (All Clear):	Signifies that everything is functioning properly.
Yellow Level (Minor—Caution):	Indicates occasional anomalies that have occurred, none of which are critical, and the system can continue operating without issues.

Orange Level (Moderate—Act Promptly):	Indicates significant anomalies or a grouping of basic anomalies that, if recurring, could impact production continuity.
Red Level (Severe—Immediate Action Required):	Signifies the presence of severe anomalies within the system that could significantly disrupt production activities.
Black Level (Emergency Shutdown):	Indicates that the system is at risk of irreversible equipment or product failures and requires an immediate halt.

3.2.3. Preliminary Sub-Step 3: Data Structure Establishment

In this stage, a digital data structure is established to encompass all potential "composite messages" structured as a dictionary. This dictionary plays a pivotal role in the algorithms' processing of source messages, where each entry comprises a "message" in the first segment and its corresponding general alert level in the second segment.

In this context, the term "dictionary" refers to the conventional computer science data structure used to host a collection of objects ([76]). Essentially, a dictionary constitutes a collection of pairs (entries) structured as (k, v), where k represents the (typically unique) key and v denotes the value associated with k. Upon presentation of a key, the dictionary retrieves the associated value. This dynamic data structure allows entries to be inserted, deleted, and modified as needed.

3.3. System Algorithmic Steps

The algorithm operates on a set $L = w_1, w_2, \ldots, w_m$ of m log messages ($m \geq 1$), where each $w_j = w_{j_1}, w_{j_2}, \ldots, w_{j_n}$ ($1 \leq n \leq Log\_Max\_length$) is associated with its respective alert level, A_{w_j}. In practical terms, w_j comprises a log message along with its alert level. The algorithmic process consists of a one-time pre-processing phase and a two-stage processing phase: matching and analysis.

The algorithmic process is schematized in the lower part of Figure 3.

3.3.1. Pre-Processing

The system builds a data structure based on the finite automaton and initializes the analysis system. This phase occurs only once, during the startup of the data source builder, and its duration is linearly proportional to the sum of the lengths of all dictionary entries, measured as the total number of characters in the messages.

3.3.2. Matching

The automaton continuously processes input log messages character by character, with each character of the input text mapping to a state of the finite automaton. It advances through the automaton based on the characters of the input text, without revisiting past characters. Upon reaching a final state corresponding to the end of a message, the automaton outputs the associated alert level. The number of state transitions executed by the automaton is equivalent to the number of characters in the input text. Consequently, the algorithm operates in linear time relative to the number of characters in the input text, irrespective of the dictionary size (i.e., the set of all possible log messages). This phase is repeated for each generated log message.

3.3.3. Analysis

Upon receiving a request from the matching algorithm, the system initiates the analysis of the acquired alert levels. After parsing the log file or a segment of it and determining the alert levels for each message, the system can assess the machine's health status based on the frequency and severity of the extracted alerts. It is capable of promptly responding to highly critical alerts, evaluating a cluster of alerts over a period to anticipate potential future critical situations, or analyzing an entire historical sequence of alerts in batches.

Following the analysis, the system may trigger automated safeguard protocols, prompt human expert intervention, or recommend minor adjustments to be implemented during the next scheduled maintenance interval.

4. System Performances

4.1. Algorithms' Performances

The system employs a finite automaton to traverse input text in search of log messages. Upon reaching a state corresponding to the end of a log message, it generates the associated semantic domain. The algorithms exhibit high efficiency, speed, and precision, enabling them to detect all occurrences of log messages, even in cases of partial or complete overlap.

Pre-processing: Initially, the system constructs a finite automaton in the computer's memory for the entire set of log messages. This process occurs once per session and whenever the reference ontology changes.
The time complexity of the pre-processing phase, denoted by the total number of elementary algorithmic steps (and hence the number of state transitions in the finite automaton), is $O(m)$ [73], where m is linearly proportional to the sum of the lengths of all entries in the dictionary with n elements, given by $m = \sum_{i=1}^{n} \text{length}(a_i)$. This implies that the time required is linearly proportional to the number of characters in the dictionary.

Processing: The automaton continuously processes lines of input text, character by character, without revisiting previously read characters. Each character guides the automaton through its state transitions, and, upon reaching a state corresponding to the end of a log message, it emits the associated alert. Consequently, analyzing a text of k characters requires k state transitions.
All log messages are simultaneously recognized in a single pass, even if they partially or fully overlap, regardless of their length. Thus, the algorithm operates in linear time relative to the number of characters in the input text [73], and its performance remains unaffected by the size of the ontologies (i.e., the set of all possible log messages). Furthermore, it operates entirely within the computer's main memory.

Analysis: At the conclusion of the matching phase, once the input text has been fully processed, the system tallies the frequency of each alert detected and suggests a classification for the input text.

4.2. System Architecture

The system runs on a standard architecture, using an INSPIRON 16 (by Dell Technologies, Round Rock, TX, USA) laptop, with 11th Generation Intel(R) Core(TM) (by INTEL, Santa Clara, CA, USA) i7-11800H CPU at 2.30GHz, 32.0GB of RAM, and runs Microsoft Windows 11 Home 64-bit ver. 22H2. It does not require specialized hardware such as a dedicated video card or large mass-memory, nor does it have specific software requirements.

The software was developed using the latest version (12) of Embarcadero RAD Studio (https://www.embarcadero.com/products/rad-studio, accessed on 19 February 2024), a robust RAD visual software development tool based on an object-oriented programming language. It includes 1711 lines of code organized into 27 operational modules, with nine methods dedicated to managing the interface, and uses 26 predefined libraries.

5. Performances Comparison

We first compare the efficiency of our method, which is part of the CMS family and is based on dictionaries created by human experts rather than automatic pre-training, with other CMS/ML-type methods (Section 2.3). Next, we compare the computational performance of our approach, based on the use of finite automata, with database-based approaches.

5.1. Efficiency

The proficiency of our proposal in detecting patterns, trends, and even subtle anomalies within the log data facilitates early issue detection, enabling timely proactive maintenance planning and minimizing downtime. The system can trigger immediate activation of safeguard procedures, either automatically or through human intervention.

Sometimes, a machine failure can be preceded by a series of apparently *regular but anomalous behaviors*, all recorded in these log messages. These "weak" signals are easily picked up by our system, but rarely by other systems.

A notable strength of our approach lies in its *hardware independence*. Once associated with a comprehensive dictionary encompassing all possible messages, the predictive maintenance system becomes adaptable across various types of machinery and equipment, enhancing its versatility and scalability.

Moreover, our method transcends traditional maintenance by offering *performance optimization capabilities*. Through refined programming, it can analyze data and pinpoint areas for enhancement, facilitating proactive adjustments to improve efficiency and productivity.

By leveraging dictionaries built by human experts, our approach may offer several advantages over automatic pre-training methods (ML), such as:

Domain expertise: human experts can provide deep domain knowledge and insights into the specific characteristics and nuances of the machinery and processes involved. This can result in more accurate and contextually relevant dictionaries tailored to the unique requirements of the industry.

Customization: manual construction of dictionaries allows for customization and fine-tuning based on specific use cases, machine types, and operational environments. This flexibility enables the system to adapt and perform effectively in diverse industrial settings.

Interpretability: dictionaries built by human experts are typically more interpretable, allowing users to understand and interpret the reasoning behind the system's decisions. This transparency can enhance trust and acceptance of the system among operators and maintenance personnel.

Robustness: human-curated dictionaries can potentially capture subtle nuances and variations in machine behavior that may be missed by automatic pre-training methods. This can improve the system's robustness and reliability in detecting anomalies and predicting failures.

5.2. Computational Performance

We propose a linguistic-based text mining system that leverages a finite automaton to effectively identify log messages from a dictionary of potential log messages within an input text. Manually building dictionaries can be time- and labor-intensive, particularly for large-scale deployments across multiple machines or industrial sites, while automated pre-training methods could offer faster scalability in such scenarios. In reality, however, the number of different log messages for a machine is often limited, so constructing the ontology could be relatively quick.

In ML-based methods, a pre-training phase requires a *high quantity* of *unfiltered data*. The system's statistical engine enhances its performance as it processes more data, although the semi-structured texts utilized for training may contain errors.

The main computational drawback of a dictionary-based method lies in managing the dictionaries. They could require a high *maintenance overhead*: manual maintenance and updates of dictionaries may be required periodically to ensure relevance and accuracy over time. This can introduce additional overhead and resource requirements compared with automatic pre-training methods.

Let us delve into the functionality of a CMS system that analyzes log messages emitted by an industrial machine. Log messages are MWUs, which are "units of meaning" with high semantic value, making their processing not straightforward. Usually, a system does not possess explicit knowledge of multiword units (MWUs) or phrases as atomic entities;

rather, it treats an MWU as a sequence of individual tokens. Each token is independently considered during both model training and inference.

A multiword unit (MWU) can be extensive, comprising any number of basic words. Additionally, two distinct MWUs might partially or entirely overlap. In contrast to single words, an MWU lacks an "end unit" symbol, such as a space or punctuation.

From an algorithmic standpoint, a traditional database-like approach relying on word matching (e.g., log messages) against a dictionary proves to be highly time-consuming and is contingent upon the size of the dictionary. As the dictionary grows larger, the waiting time for a response increases proportionally.

In a log file input text X containing n words, identifying a single log message from a dictionary, D, with m items requires $O(\log m)$ accesses to D to verify if the first word in X is an item in D. If it is not, we have to verify if the combination of the first and second words in X forms an item in D, which also requires $O(\log m)$ accesses to D, and so on. Thus, the identification of a log message in a text with n words requires $O(n^2)$ searches in the ontology because, for each word, we have to try the chains with all its successive words, and each search in an ontology with m items costs $O(\log m)$ accesses. In the worst case, we have to access the dictionary for all the words in X, i.e., n times. Since X contains n words, in the worst case, the total number of accesses to D is $O(n^2 \log m)$ to identify all the log messages within X.

Contrarily, in our method, the processing time for a text of k characters only necessitates $O(k)$ state transitions on a finite automaton. Consequently, identifying all the log messages within X entails linear time relative to the number of characters in the text. This is facilitated by our method's ability to concurrently identify all partially or fully overlapping log messages, without additional effort.

Let us now normalize the concept of word and character, and say that a word is composed, on average, of c characters. Thus, our approach requires $O(k)$ elementary operations, where $k = nc$, so our method requires $O(nc)$ elementary operations, while the other methods require $O(n^2 \log m)$ elementary operations and, since c is a small integer (the average length of a word), our method is linear and the others are quadratic.

Moreover, in the case of deletion, insertion, or modification of an entry in an ontology, maintaining order within the ontology is essential. Such operations necessitate at least $O(\log m)$ accesses to the ontology, potentially involving physical movement of entries from one memory area to another.

In contrast, our proposal eliminates the need for sorting the dictionary. Therefore, adding, modifying, or deleting one or more entries incurs virtually no computational cost. We simply insert a new word at the end of the ontology or modify/delete a word without affecting others. Additionally, in our method, the entire ontology resides within a finite automaton, typically in the central memory of a computer. Conversely, in other approaches, elementary operations access a computer's secondary memory.

Our method operates independently of the length and number of elementary words in a log message. It can effectively manage large dictionary sizes without encountering issues. Additionally, it seamlessly identifies all partially or fully overlapping log messages without requiring any additional computational effort. Moreover, our method directly recognizes log messages while reading the input text, without providing any form of feedback.

6. Case Study

In this paper, we present an analysis of a log file obtained from a real industrial machine. The case study focuses on an actual company operating within the national territory, specializing in the production of metal molds for other companies.

6.1. Source Data Description

During the preliminary phase, we examined 1200 log files, each comprising 1000 messages. The reference period extends from 6:05:01 on 13 February 2023, to 21:59:40 on 10 March 2023. The log file consists of four blocks of pertinent data and a fifth block, which is not relevant for analysis. Each field within these blocks is delimited by the semicolon symbol (;). The fundamental details contained within the first four blocks are as follows:

Date/time	of event recording (e.g., "13/02/2023 14:25:00").
Process ID	of the running process that generated the message (e.g., "MSG_SYS").
Operation	performed (e.g., "Scrive").
Execution status/result	of the event (e.g., "Fine corsa asse. . . , Y+").

Out of the total 1,200,000 messages (excluding empty ones, which amount to 1,199,784), 1714 contained different "semantic" content. To achieve this, we filtered out messages without useful information and truncated the first 20 characters of each remaining line (date and time). Subsequently, we identified all completely distinct messages and calculated their frequencies.

In the subsequent step, we assigned an ALERT level to each of the 1714 different messages. A "normalized" record consists of a line containing the following information, separated by a hashtag:

Date/Time # Process ID # Operation # Execution status/result # "Alert Level"

6.2. Simulation Results

6.2.1. Preliminary Phase

During sub-step 1 of the preliminary phase (see Section 3.2.1), we identified 1714 different types of log messages. In sub-step 2 (see Section 3.2.2), each message was assigned an alert level. In sub-step 3 (see Section 3.2.3), each log message along with its corresponding alert level was stored in the dictionary. The total number of characters in the dictionary is 154,928.

6.2.2. Algorithmic Phase 1: Pre-Processing

During the pre-processing phase, the system generated a finite automaton with 27,357 states. The time required to construct the finite automaton is practically instantaneous (8–10 s), irrespective of system overhead. The system necessitates 1176 MB of random access memory (RAM) to store the entire finite automaton. Note that both the memory required by the finite automaton and the total number of characters in the dictionary remain stable, since they depend on the total number of possible types of log messages.

6.2.3. Algorithmic Phase 2: Processing

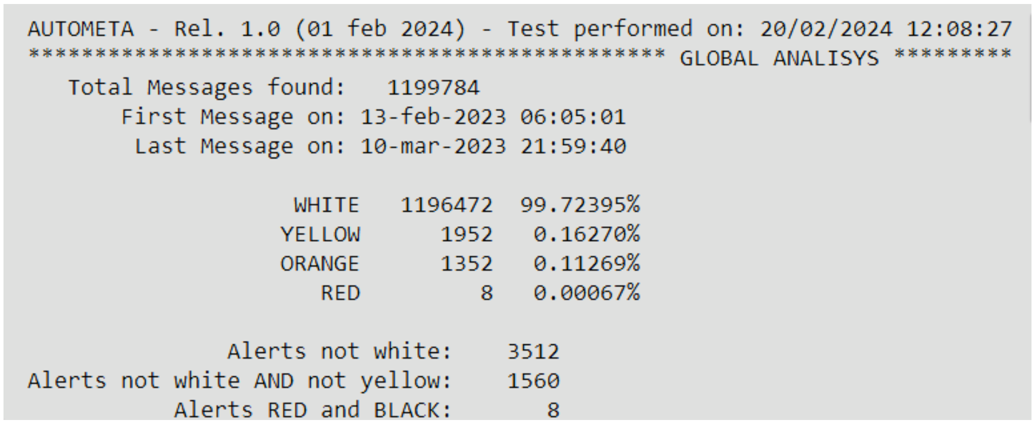
In Phase 2 of the simulation, we employed a sample log message file obtained from a real industrial machine. We discovered a total of 1,199,984 log messages, amounting to 59,637,872 characters in total (including spaces and punctuation), resulting in an average length of 49.70 characters per log message. The automaton made a total of 119,279,005 forward transitions, with an additional 1,212,637 transitions back to the root. The processing time for our test data was approximately 5 s; this step does not require additional RAM.

6.2.4. Algorithmic Phase 3: Analysis

In Phase 3, during the analysis, we found the following distribution among the 1,199,784 alerts:

- 1,196,472 are white (99.72%) and 3312 are not white (0.28%);
- 1952 are yellow (0.17%) and 1360 (0.13%) are neither white nor yellow;
- 1352 are orange (0.11%);
- 8 are red (0.00067%).

In Figure 4, we report the general output with some of the log messages found.



```
AUTOMETA - Rel. 1.0 (01 feb 2024) - Test performed on: 20/02/2024 12:08:27
************************************************ GLOBAL ANALISYS *********
   Total Messages found:   1199784
       First Message on: 13-feb-2023 06:05:01
        Last Message on: 10-mar-2023 21:59:40

                   WHITE   1196472  99.72395%
                  YELLOW      1952   0.16270%
                  ORANGE      1352   0.11269%
                     RED         8   0.00067%

              Alerts not white:    3512
Alerts not white AND not yellow:    1560
           Alerts RED and BLACK:       8
```

Figure 4. Case study results (4 weeks of logs).

7. Future Research Directions

As a first research direction, one could analyze log messages coming from a very long period of time, such as one or more years of machine activity.

The current analysis focuses on the percentage of alert levels identified, but, as future research directions, clusters, even small ones, could be identified in which the percentage presence and/or relative disposition of some alerts could indicate potential future industrial machine malfunctions.

Another line of research could be dictated by the analysis of messages coming from multiple data sources connected to an industrial machine.

8. Conclusions

We present an innovative linguistic-based text mining approach that utilizes a finite automaton to precisely identify log messages, drawn from an ontology of log messages, continuously emitted by a data source connected to industrial machinery. Through the integration of machinery and physical devices with text mining technologies, our proposal facilitates the algorithmic prediction of anomalous behavior of industrial machines, even at a very preliminary stage, aiming to prevent malfunctions or machine downtime before they become manifest. The algorithms employed exhibit linear execution time on the number of input characters, operate on a data structure entirely in RAM, and remain unaffected by the data structure size, facilitating effortless modification without incurring additional computational costs.

The system conducts continuous analysis without requiring the end of a shift or machine stoppage. Upon identifying potential anomalies, it can trigger automatic safeguard procedures, notify human experts, or schedule minor tuning operations.

Overall, our methodology promises enhancing predictive maintenance strategies in industrial settings through the integration of industrial standard technologies and linguistic (or rule-based) text mining.

Author Contributions: Methodology, A.P. and M.M.; Software, A.P. and M.M.; Writing—original draft, A.P. and M.M.; Writing—review & editing, A.P. and M.M. All authors have read and agreed to the published version of the manuscript.

Funding: Research FARB 2022 funded by University of Salerno (300639FRB22POSTI).

Data Availability Statement: Restrictions apply to the availability of these data. Data were obtained from third party and are available from the authors with the permission of third party.

Conflicts of Interest: The authors declare no conflicts of interest.

References

1. Basri, E.I.; Razak, I.H.A.; Ab-Samat, H.; Kamaruddin, S. Preventive Maintenance (PM) Planning: A Review. *J. Qual. Maint. Eng.* **2017**, *23*, 114–143. [CrossRef]
2. Trojan, F.; Marçal, R.F. Proposal of Maintenance-Types Classification to Clarify Maintenance Concepts in Production and Operations Management. *J. Bus. Econ.* **2017**, *8*, 560–572.
3. Rüßmann, M.; Lorenz, M.; Gerbert, P.; Waldner, M.; Justus, J.; Engel, P.; Harnisch, M. Industry 4.0: The Future of Productivity and Growth in Manufacturing Industries. *Boston Consult. Group* **2015**, *9*, 54–89.
4. Silvestri, L.; Forcina, A.; Introna, V.; Santolamazza, A.; Cesarotti, V. Maintenance Transformation through Industry 4.0 Technologies: A Systematic Literature Review. *Comput. Ind.* **2020**, *123*, 103335. [CrossRef]
5. Silva Neto, A.V.; Silva, H.L.; Camargo, J.B.; Almeida, J.R.; Cugnasca, P.S. Design and Assurance of Safety-Critical Systems with Artificial Intelligence in FPGAs: The Safety ArtISt Method and a Case Study of an FPGA-Based Autonomous Vehicle Braking Control System. *Electronics* **2023**, *12*, 4903. [CrossRef]
6. Zonta, T.; da Costa, C.A.; da Rosa Righi, R.; de Lima, M.J.; da Trindade, E.S.; Li, G.P. Predictive Maintenance in the Industry 4.0: A Systematic Literature Review. *Comput. Ind. Eng.* **2020**, *150*, 106889. [CrossRef]
7. Shcherbakov, M.V.; Glotov, A.V.; Cheremisinov, S.V. Proactive and Predictive Maintenance of Cyber-Physical Systems. *Stud. Syst. Decis. Control* **2020**, *259*, 263–278. [CrossRef]
8. Bampoula, X.; Siaterlis, G.; Nikolakis, N.; Alexopoulos, K. A Deep Learning Model for Predictive Maintenance in Cyber-Physical Production Systems Using LSTM Autoencoders. *Sensors* **2021**, *21*, 972. [CrossRef] [PubMed]
9. Nota, G.; Postiglione, A.; Carvello, R. Text Mining Techniques for the Management of Predictive Maintenance. In Proceedings of the 3rd International Conference on Industry 4.0 and Smart Manufacturing, ISM 2021, Linz, Austria, 19–21 November 2021; Longo, F., Affenzeller, M.P.A., Eds.; Elsevier B.V.: Amsterdam, The Netherlands, 2022. [CrossRef]
10. Sharanya, S. A Cyber Physical System Framework for Industrial Predictive Maintenance Using Machine Learning. In *Real-Time Applications of Machine Learning in Cyber-Physical Systems*; IGI Global: Hershey, PA, USA, 2022; pp. 241–269.
11. Duffuaa, S.; Ben-Daya, M.; Al-Sultan, K.; Andijani, A. A Generic Conceptual Simulation Model for Maintenance Systems. *J. Qual. Maint. Eng.* **2001**, *7*, 207–219. [CrossRef]
12. Nyman, D.; Levitt, J. *Maintenance Planning, Scheduling, and Coordination*; Industrial Press Inc.: New York, NY, USA, 2001.
13. Patil, A.; Soni, G.; Prakash, A.; Karwasra, K. Maintenance strategy selection: A comprehensive review of current paradigms and solution approaches. *Int. J. Qual. Reliab. Manag.* **2022**, *39*, 675–703. [CrossRef]
14. Di Dio, M.; Iannone, R.; Miranda, S.; Riemma, S. A Framework for the Choice of the Opportunistic Maintenance Policy in Industrial Contexts. In Proceedings of the IEEE International Conference on Industrial Engineering and Engineering Management, Selangor Darul Ehsan, Malaysia, 9–12 December 2014; IEEE Computer Society: Washington, DC, USA, 2014; pp. 1716–1720. [CrossRef]
15. Wang, Y.; Deng, C.; Wu, J.; Wang, Y.; Xiong, Y. A Corrective Maintenance Scheme for Engineering Equipment. *Eng. Fail. Anal.* **2014**, *36*, 269–283. [CrossRef]
16. Gajdzik, B. Autonomous and Professional Maintenance in Metallurgical Enterprise as Activities within Total Productive Maintenance. *Metalurgija* **2014**, *53*, 269–272.
17. Ahmad, R.; Kamaruddin, S. An Overview of Time-Based and Condition-Based Maintenance in Industrial Application. *Comput. Ind. Eng.* **2012**, *63*, 135–149. [CrossRef]
18. Kim, J.; Ahn, Y.; Yeo, H. A Comparative Study of Time-Based Maintenance and Condition-Based Maintenance for Optimal Choice of Maintenance Policy. *Struct. Infrastruct. Eng.* **2016**, *12*, 1525–1536. [CrossRef]
19. Shin, J.H.; Jun, H.B. On Condition Based Maintenance Policy. *J. Comput. Des. Eng.* **2015**, *2*, 119–127. [CrossRef]
20. Ahmad, R.; Kamaruddin, S. A Review of Condition-Based Maintenance Decision-Making. *Eur. J. Ind. Eng.* **2012**, *6*, 519–541. [CrossRef]
21. Mobley, R.K. *An Introduction to Predictive Maintenance*; Elsevier: Amsterdam, The Netherlands, 2002.
22. Frankó, A.; Varga, P. A Survey on Machine Learning Based Smart Maintenance and Quality Control Solutions. *Infocommunications J.* **2021**, *13*, 28–35. [CrossRef]
23. Florian, E.; Sgarbossa, F.; Zennaro, I. Machine Learning-Based Predictive Maintenance: A Cost-Oriented Model for Implementation. *Int. J. Prod. Econ.* **2021**, *236*, 108114. [CrossRef]
24. Drakaki, M.; Karnavas, Y.L.; Tziafettas, I.A.; Linardos, V.; Tzionas, P. Machine Learning and Deep Learning Based Methods toward Industry 4.0 Predictive Maintenance in Induction Motors: A State of the Art Survey. *J. Ind. Eng. Manag.* **2022**, *15*, 31–57. [CrossRef]
25. Rosati, R.; Romeo, L.; Cecchini, G.; Tonetto, F.; Viti, P.; Mancini, A.; Frontoni, E. From Knowledge-Based to Big Data Analytic Model: A Novel IoT and Machine Learning Based Decision Support System for Predictive Maintenance in Industry 4.0. *J. Intell. Manuf.* **2023**, *34*, 107–121. [CrossRef]
26. Higgs, P.A.; Parkin, R.; Jackson, M.; Al-Habaibeh, A.; Zorriassatine, F.; Coy, J. A Survey on Condition Monitoring Systems in Industry. In Proceedings of the 7th Biennial Conference on Engineering Systems Design and Analysis, ESDA, Manchester, UK, 19–22 July 2004; American Society of Mechanical Engineers: New York, NY, USA, 2004; Volume 3, pp. 163–178. [CrossRef]

27. Zhu, J.; Nostrand, T.; Spiegel, C.; Morton, B. Survey of Condition Indicators for Condition Monitoring Systems. In Proceedings of the PHM 2014—The Annual Conference of the Prognostics and Health Management Society 2014, Spokane, WA, USA, 29 September–2 October 2014; pp. 635–647.
28. Surucu, O.; Gadsden, S.A.; Yawney, J. Condition Monitoring Using Machine Learning: A Review of Theory, Applications, and Recent Advances. *Expert Syst. Appl.* **2023**, *221*, 119738. [CrossRef]
29. Jung, D.; Zhang, Z.; Winslett, M. Vibration Analysis for Iot Enabled Predictive Maintenance. In Proceedings of the International Conference on Data Engineering, San Diego, CA, USA, 19–22 April 2017; IEEE Computer Society: Washington, DC, USA, 2017; pp. 1271–1282. [CrossRef]
30. Popescu, T.D.; Aiordachioaie, D.; Culea-Florescu, A. Basic Tools for Vibration Analysis with Applications to Predictive Maintenance of Rotating Machines: An Overview. *Int. J. Adv. Manuf. Technol.* **2022**, *118*, 2883–2899. [CrossRef]
31. Boughardini, I.E.; Mechkouri, M.H.; Reklaoui, K. A Predictive Maintenance System Based on Vibration Analysis for Rotating Machinery Using Wireless Sensor Network (WSN). *Lect. Notes Netw. Syst.* **2023**, *712 LNNS*, 93–106. [CrossRef]
32. Ortega, M.; Ivorra, E.; Juan, A.; Venegas, P.; Martínez, J.; Alcañiz, M. Mantra: An Effective System Based on Augmented Reality and Infrared Thermography for Industrial Maintenance. *Appl. Sci.* **2021**, *11*, 385. [CrossRef]
33. Keartland, S.; Van Zyl, T.L. Automating Predictive Maintenance Using Oil Analysis and Machine Learning. In Proceedings of the 2020 International SAUPEC/RobMech/PRASA Conference, Cape Town, South Africa, 29–31 January 2020; Institute of Electrical and Electronics Engineers Inc.: Piscataway, NJ, USA, 2020. [CrossRef]
34. Johansen, R.; Dupont, S.; Christiansen, U.N.; Horning, J.; Catalano, J.; Sorensen, J.L.; Bentien, A. An In-Line, High-Flowrate, and Maintenance Free Ultrasonic Sensor with a High Dynamic Range for Particle Monitoring in Fluids. *IEEE Sens. J.* **2018**, *18*, 2299–2304. [CrossRef]
35. Gouriveau, R.; Medjaher, K.; Zerhouni, N. *From Prognostics and Health Systems Management to Predictive Maintenance 1: Monitoring and Prognostics*; John Wiley & Sons: Hoboken, NJ, USA, 2016; Volume 4, pp. 1–163. [CrossRef]
36. Guillén, A.; Crespo, A.; Macchi, M.; Gómez, J. On the Role of Prognostics and Health Management in Advanced Maintenance Systems. *Prod. Plan. Control* **2016**, *27*, 991–1004. [CrossRef]
37. Mancuso, A.; Compare, M.; Salo, A.; Zio, E. Optimal Prognostics and Health Management-driven Inspection and Maintenance Strategies for Industrial Systems. *Reliab. Eng. Syst. Saf.* **2021**, *210*, 107536. [CrossRef]
38. Aggarwal, C.C.; Zhai, C. *Mining Text Data*; Springer: Boston, MA, USA, 2012; pp. 1–526.
39. Aggarwal, C.C.; Zhai, C. A Survey of Text Classification Algorithms. In *Mining Text Data*; Aggarwal, C.C., Zhai, C., Eds.; Springer: Boston, MA, USA, 2012; pp. 163–222. [CrossRef]
40. Usai, A.; Pironti, M.; Mital, M.; Aouina Mejri, C. Knowledge Discovery out of Text Data: A Systematic Review via Text Mining. *J. Knowl. Manag.* **2018**, *22*, 1471–1488. [CrossRef]
41. Kowsari, K.; Meimandi, K.J.; Heidarysafa, M.; Mendu, S.; Barnes, L.; Brown, D. Text Classification Algorithms: A Survey. *Information* **2019**, *10*, 150. [CrossRef]
42. Tandel, S.S.; Jamadar, A.; Dudugu, S. A Survey on Text Mining Techniques. In Proceedings of the 2019 5th International Conference on Advanced Computing and Communication Systems, ICACCS, Coimbatore, India, 15–16 March 2019; Institute of Electrical and Electronics Engineers Inc.: Piscataway, NJ, USA, 2019; pp. 1022–1026. . [CrossRef]
43. Zong, C.; Xia, R.; Zhang, J. *Text Data Mining*; Springer: Singapore, 2021; pp. 1–351. [CrossRef]
44. Kumar, M.; Kumar, S.; Yadav, S.L. *Data Mining for the Internet of Things: A Survey*; Apple Academic Press: Palm Bay, FL, USA, 2023; pp. 93–109.
45. Navathe, S.B.; Ramez, E. Data Warehousing and Data Mining. In *Fundamentals of Database Systems*; Pearson Education: Singapore, 2000; pp. 841–872.
46. Gupta, V.; Lehal, G.S. A Survey of Text Mining Techniques and Applications. *J. Emerg. Technol. Web Intell.* **2009**, *1*, 60–76. [CrossRef]
47. Kusakin, I.; Fedorets, O.; Romanov, A. Classification of Short Scientific Texts. *Sci. Tech. Inf. Process.* **2023**, *50*, 176–183. [CrossRef]
48. Danilov, G.; Ishankulov, T.; Kotik, K.; Orlov, Y.; Shifrin, M.; Potapov, A. *The Classification of Short Scientific Texts Using Pretrained BERT Model*; IOS Press: Amsterdam, The Netherlands, 2021; pp. 83–87. [CrossRef]
49. Ongenaert, M.; Van Neste, L.; De Meyer, T.; Menschaert, G.; Bekaert, S.; Van Criekinge, W. PubMeth: A Cancer Methylation Database Combining Text-Mining and Expert Annotation. *Nucleic Acids Res.* **2008**, *36*, D842–D846. [CrossRef] [PubMed]
50. Cejuela, J.M.; McQuilton, P.; Ponting, L.; Marygold, S.; Stefancsik, R.; Millburn, G.H.; Rost, B. Tagtog: Interactive and Text-Mining-Assisted Annotation of Gene Mentions in PLOS Full-Text Articles. *Database Mag. Electron. Database Rev.* **2014**, *2014*, bau033. [CrossRef] [PubMed]
51. Cheerkoot-Jalim, S.; Khedo, K.K. A Systematic Review of Text Mining Approaches Applied to Various Application Areas in the Biomedical Domain. *J. Knowl. Manag.* **2020**, *25*, 642–668. [CrossRef]
52. Rodríguez-Rodríguez, I.; Rodríguez, J.V.; Shirvanizadeh, N.; Ortiz, A.; Pardo-Quiles, D.J. Applications of Artificial Intelligence, Machine Learning, Big Data and the Internet of Things to the COVID-19 Pandemic: A Scientometric Review Using Text Mining. *Int. J. Environ. Res. Public Health* **2021**, *18*, 8578. [CrossRef] [PubMed]
53. Baltoumas, F.A.; Zafeiropoulou, S.; Karatzas, E.; Paragkamian, S.; Thanati, F.; Iliopoulos, I.; Eliopoulos, A.G.; Schneider, R.; Jensen, L.J.; Pafilis, E.; et al. OnTheFly2.0: A Text-Mining Web Application for Automated Biomedical Entity Recognition, Document Annotation, Network and Functional Enrichment Analysis. *Nar Genom. Bioinform.* **2021**, *3*, lqab090. [CrossRef] [PubMed]

54. Fenza, G.; Orciuoli, F.; Peduto, A.; Postiglione, A. Healthcare Conversational Agents: Chatbot for Improving Patient-Reported Outcomes. *Lect. Notes Netw. Syst.* **2023**, *661 LNNS*, 137–148. [CrossRef]
55. Abbe, A.; Grouin, C.; Zweigenbaum, P.; Falissard, B. Text Mining Applications in Psychiatry: A Systematic Literature Review. *Int. J. Methods Psychiatr. Res.* **2016**, *25*, 86–100. [CrossRef] [PubMed]
56. Chu, C.Y.; Park, K.; Kremer, G.E. A Global Supply Chain Risk Management Framework: An Application of Text-Mining to Identify Region-Specific Supply Chain Risks. *Adv. Eng. Inform.* **2020**, *45*, 101053. [CrossRef]
57. Kumar, B.S.; Ravi, V. A Survey of the Applications of Text Mining in Financial Domain. *Knowl.-Based Syst.* **2016**, *114*, 128–147. [CrossRef]
58. Gupta, A.; Dengre, V.; Kheruwala, H.A.; Shah, M. Comprehensive Review of Text-Mining Applications in Finance. *Financ. Innov.* **2020**, *6*, 1–25. [CrossRef]
59. Kumar, S.; Kar, A.K.; Ilavarasan, P.V. Applications of Text Mining in Services Management: A Systematic Literature Review. *Int. J. Inf. Manag. Data Insights* **2021**, *1*, 100008. [CrossRef]
60. Irfan, R.; King, C.K.; Grages, D.; Ewen, S.; Khan, S.U.; Madani, S.A.; Kolodziej, J.; Wang, L.; Chen, D.; Rayes, A.; et al. A Survey on Text Mining in Social Networks. *Knowl. Eng. Rev.* **2015**, *30*, 157–170. [CrossRef]
61. Salloum, S.A.; Al-Emran, M.; Monem, A.A.; Shaalan, K. A Survey of Text Mining in Social Media: Facebook and Twitter Perspectives. *Adv. Sci. Technol. Eng. Syst.* **2017**, *2*, 127–133. [CrossRef]
62. Ferreira-Mello, R.; André, M.; Pinheiro, A.; Costa, E.; Romero, C. Text Mining in Education. *Wiley Interdiscip. Rev. Data Min. Knowl. Discov.* **2019**, *9*, e1332. [CrossRef]
63. Ngai, E.; Lee, P. A Review of the Literature on Applications of Text Mining in Policy Making. In Proceedings of the Pacific Asia Conference on Information Systems, PACIS, Chiayi, Taiwan, 27 June–1 July 2016.
64. Drury, B.; Roche, M. A Survey of the Applications of Text Mining for Agriculture. *Comput. Electron. Agric.* **2019**, *163*, 104864. [CrossRef]
65. Postiglione, A. Text Mining with Finite State Automata via Compound Words Ontologies. *Lect. Notes Data Eng. Commun. Technol.* **2024**, *193*, 194–205. [CrossRef]
66. Postiglione, A. Finite State Automata on Multi-Word Units for Efficient Text-Mining. *Mathematics* **2024**, *12*, 506. [CrossRef]
67. Elia, A.; Monteleone, M.; Postiglione, A. Cataloga: A Software for Semantic-Based Terminological Data Mining. In Proceedings of the 1st International Conference on Data Compression, Communication and Processing, Palinuro, SA, USA, 21–24 June 2011; IEEE Computer Society: Washington, DC, USA, 2011; pp. 153–156. [CrossRef]
68. Gross, M. Lexicon-Grammar and the Syntactic Analysis of French. In Proceedings of the 10th International Conference on Computational Linguistics, COLING, 1984 and 22nd Annual Meeting of the Association for Computational Linguistics, ACL, Stanford, CA, USA, 2–6 July 1984; pp. 275–282.
69. Gross, M. The construction of electronic dictionaries; [La construction de dictionnaires électroniques]. *Ann. Télécommun.* **1989**, *44*, 4–19. [CrossRef]
70. Gross, M. The Use of Finite Automata in the Lexical Representation of Natural Language. *Lect. Notes Comput. Sci. (Subser. Lect. Notes Artif. Intell. Lect. Notes Bioinform.)* **1989**, *377 LNCS*, 34–50. [CrossRef]
71. Monteleone, M. NooJ for Artificial Intelligence: An Anthropic Approach. *Commun. Comput. Inf. Sci.* **2021**, *1389*, 173–184. [CrossRef]
72. Monteleone, M.; Mirto, I.M. NooJ Grammars for Italian Negation System and Sentiment Analysis. *Commun. Comput. Inf. Sci.* **2021**, *1520 CCIS*, 39–50. [CrossRef]
73. Aho, A.V.; Corasick, M.J. Efficient String Matching: An Aid to Bibliographic Search. *Commun. ACM* **1975**, *18*, 333–340. [CrossRef]
74. Boyer, R.S.; Moore, J.S. A Fast String Searching Algorithm. *Commun. ACM* **1977**, *20*, 762–772. [CrossRef]
75. Crochemore, M.; Hancart, C.; Lecroq, T. *Algorithms on Strings*; Cambridge University Press: Cambridge, UK, 2007; Volume 9780521848992, pp. 1–383. [CrossRef]
76. Cormen, T.H.; Leiserson, C.E.; Rivest, R.L.; Stein, C. *Introduction to Algorithms*, 3rd ed.; The MIT Press: Cambridge, MA, USA, 2009.
77. Harrag, F.; El-Qawasmeh, E.; Salman Al-Salman, A.M. Extracting Named Entities from Prophetic Narration Texts (Hadith). *Commun. Comput. Inf. Sci.* **2011**, *180 CCIS*, 289–297. [CrossRef]
78. Singh, R.; Ghorbani, A.A. EfficientPMM: Finite Automata Based Efficient Pattern Matching Machine. *Procedia Comput. Sci.* **2017**, *108*, 1060–1070. [CrossRef]
79. Hakak, S.I.; Kamsin, A.; Shivakumara, P.; Gilkar, G.A.; Khan, W.Z.; Imran, M. Exact String Matching Algorithms: Survey, Issues, and Future Research Directions. *IEEE Access Pract. Innov. Open Solut.* **2019**, *7*, 69614–69637. [CrossRef]